



Isling: A Tool for Detecting Integration of Wild-Type Viruses and Clinical Vectors

Suzanne Scott^{1,2,3}, Claus V. Hallwirth^{2,3}, Felix Hartkopf⁴, Susanna Grigson⁵, Yatish Jain¹, Ian E. Alexander^{2,3,6}, Denis C. Bauer^{1,6,7*} and Laurence O. W. Wilson^{1,8*}

1 - Australian e-Health Research Centre, Commonwealth Scientific and Industrial Research Organisation, North Ryde, Australia

2 - Gene Therapy Research Unit, Children's Medical Research Institute, Westmead, Australia

3 - The Sydney Children's Hospitals Network, Faculty of Medicine and Health, The University of Sydney, Westmead, Australia

4 - Department of Mathematics and Computer Science, Freie Universität Berlin, Berlin, Germany

5 - College of Science and Engineering, Flinders University, Adelaide, Australia

6 - Discipline of Child and Adolescent Health, Faculty of Medicine and Health, The University of Sydney, Sydney, New South Wales, Australia

7 - Macquarie University, Department of Biomedical Sciences, Faculty of Medicine and Health Science, Macquarie Park, Australia

8 - Macquarie University, Applied BioSciences, Faculty of Science and Engineering, Macquarie Park, Australia

Correspondence to Denis C. Bauer and Laurence O.W. Wilson: Building 53, 11 Julius Ave, North Ryde, NSW 2113, Australia. denis.bauer@csiro.au (D.C. Bauer), Laurence.wilson@csiro.au (L.O.W. Wilson), [@garbled](https://twitter.com/garbled) (S. Scott), [@allpowerde](https://twitter.com/allpowerde) (D.C. Bauer), [@dr_lwilson](https://twitter.com/dr_lwilson) (L.O.W. Wilson)

<https://doi.org/10.1016/j.jmb.2021.167408>

Edited by David Mathews

Abstract

Detecting viral and vector integration events is a key step when investigating interactions between viral and host genomes. This is relevant in several fields, including virology, cancer research and gene therapy. For example, investigating integrations of wild-type viruses such as human papillomavirus and hepatitis B virus has proven to be crucial for understanding the role of these integrations in cancer. Furthermore, identifying the extent of vector integration is vital for determining the potential for genotoxicity in gene therapies. To address these questions, we developed isling, the first tool specifically designed for identifying viral integrations in both wild-type and vector from next-generation sequencing data. Isling addresses complexities in integration behaviour including integration of fragmented genomes and integration junctions with ambiguous locations in a host or vector genome, and can also flag possible vector recombinations. We show that isling is up to 1.6-fold faster and up to 170% more accurate than other viral integration tools, and performs well on both simulated and real datasets. Isling is therefore an efficient and application-agnostic tool that will enable a broad range of investigations into viral and vector integration. These include comparisons between integrations of wild-type viruses and gene therapy vectors, as well as assessing the genotoxicity of vectors and understanding the role of viruses in cancer.

Crown Copyright © 2021 Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Introduction

Genetic material from a viral genome or the therapeutic cassette of a gene therapy vector can be integrated into the host genome. Some viruses,

such as retroviruses, integrate into a host's genome in a controlled and regulated manner, typically as an obligate step in their life cycle, while others, such as Adeno-associated viruses (AAV) and human papillomavirus (HPV), integrate

in a more random and stochastic manner. Sporadic integration events such as these have been linked to oncogenesis.^{1–3}

Integration events from both wild-type viruses and gene therapy vectors pose potential health risks. Characterising integrated viruses and vectors is therefore key when elucidating the role of integration events in cancer,^{3–6} evaluating genotoxicity of gene therapy vectors,^{7–9} and discovering novel viral genomes and genomic fragments, including endogenous viral elements.¹⁰ Since wild-type viruses and their vectorized counterparts may have different mechanisms of integration,¹¹ comparing their patterns of integration across the genome is also an important step in investigating their different modes of integration.

The goal of integration site analysis is to identify junctions between integrated viral DNA, and a host genome. There are several challenges in this process, some of which are specific to identifying gene therapy vector integrations, and others that are specific to wild-type viral integrations. For example in gene therapy, integrations of AAV vectors tend to be distributed and heterogenous,^{8,12–14} and every integration event is potentially of interest. Rearrangement and fragmentation can occur for certain vectors, such as recombinant AAV vectors,¹⁵ during production, and the episomal presence of these non-standard genomes can confound results if not explicitly considered. Conversely, wild-type viruses are typically investigated in the context of cancer-causing integrations, where the focus is on clonally expanded integrations. The challenge here is that integrated viruses may differ from previously described isolates.⁵

Several bioinformatic tools have been developed to address the challenge of identifying integration junction locations (reviewed by Chen *et al.*¹⁶ These tools are often designed to detect either certain types of integration events or require specific input data. Tools such as VIRUSBreakend,¹⁷ Polyidus,⁴ ViFi¹⁸ and VERSE¹⁹ were designed to detect wild-type viral integration events, but cannot deal with vector-specific issues such as vector rearrangement and low coverage of rare integration sites. Other tools are designed to specifically identify viral vector integrations, such as INSPIRED,^{20,21} Ub-ISAP,²² and VISA²³ but require NGS data prepared by ligation-mediated PCR (LM-PCR) based approaches.^{24,25} These approaches also assume the entire viral genome is integrated, which may not be the case for viruses for which integration is not a part of their life cycle. Other tools such as GENE-IS²⁶ and VSeq-Toolkit²⁷ are more flexible by detecting vector fragments and fusions using data from hybridization capture or LM-PCR amplicon libraries,^{28,29} however lack the ability to tailor analyses to wild-type or vector analyses for example by automatically selecting high-clonality junctions in wild-type datasets. The most application-agnostic approaches are structural variation tools,

such as Seeksv,³⁰ however, these are inefficient for whole-genome or RNA-seq datasets because they identify all structural variations, not only integrations, resulting in excess computation. These excess structural variations may be numerous, especially in the case of RNA-seq data where each splice junction may appear to be a structural variation.

Many of the existing tools also suffer from usability issues, including excessive runtime or memory usage, as well as reliance on multiple third-party dependencies, making installation difficult. Some tools only work with particular reference genomes (most commonly human) making them inapplicable in safety and efficacy studies of gene therapy vectors, which often use model organisms.

To address these issues, we developed *isling*, a tool for identifying integration junctions for both wild-type viruses as well as clinical vectors. *Isling* handles issues arising during vector integration, such as rearrangement and fragmentation, and is able to detect clonally expanded events for wild-type viruses. *Isling* is also agnostic with regards to host organism and integrated virus/vector, and fully scalable to large NGS datasets.

Here, we demonstrate *isling*'s utility for analysing wild-type virus and vector integration datasets. To benchmark *isling*'s performance, we generated synthetic test datasets for vector and wild-type viral integrations, and used them to compare the performance of *isling* against other tools. We further compared the performance of *isling* and other tools on previously published whole-genome sequencing and RNA-seq datasets, containing both wild-type and vector integrations.

Results

Comparison of viral integration tools on simulated data

We developed *isling* (Figure 1) to identify integrations of both wild-type and vector genomes into a host genome (see Methods for details), and compared it against several other viral integration tools. We first compared the performance of *isling* with four other viral integration detection tools, *seeksv*,³⁰ *Polyidus*,⁴ *ViFi*¹⁸ and *VSeq-Toolkit*,²⁷ on simulated data (Figure 2, Supplementary table 2). Viral and vector integrations were simulated by integrating either AAV2 (Figure 2(A)–(D)) or a pre-clinical recombinant AAV vector encoding human ornithine transcarbamylase (rAAV(OTC) vector; Figure 2(E)–(H)) into human chromosome 1.

Isling had the highest overlap between identified and simulated integration junctions as measured by the Jaccard statistic, which is a value between 0 (no overlap between simulated and output integration sites), and 1 (perfect overlap; Figure 2 (A), (E); Supplementary table 2); for *isling* this statistic was 150% higher on the AAV2 dataset

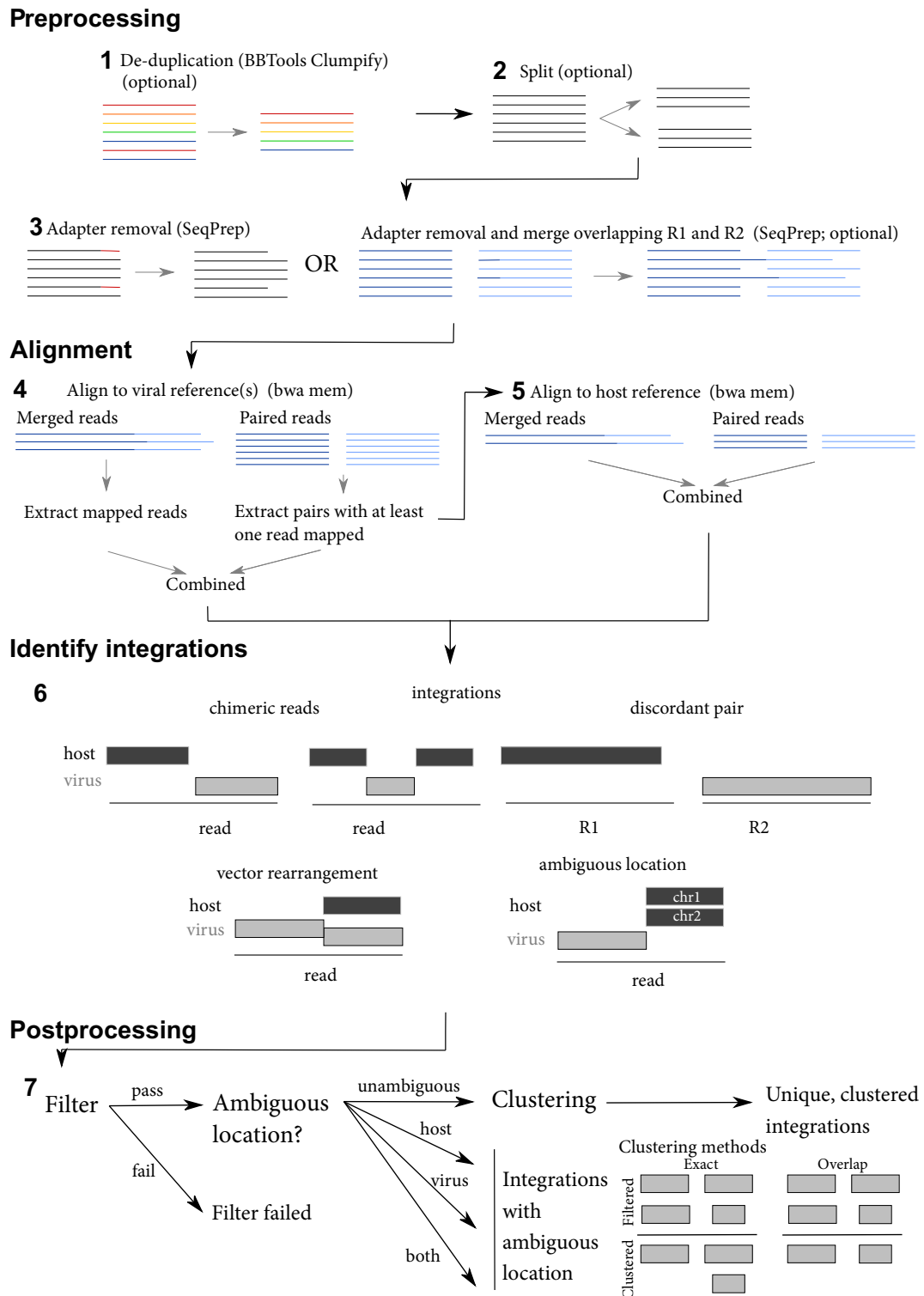


Figure 1. Steps taken by isling to identify integrations. In the preprocessing phase, reads are de-duplicated, split into multiple fastq files for parallel processing, stripped of adapters and merged if R1 and R2 are overlapping. All preprocessing steps are optional. In the alignment phase, reads are first aligned to the viral reference, and then reads of interest are extracted and aligned to the host reference. In the integration detection phase, integrations are detected based on the host and viral alignments. Putative integrations are also examined for vector rearrangements and location ambiguity. In the postprocessing phase, integrations are filtered based on user-defined criteria and separated into pools based on their location ambiguity (unambiguous, ambiguous in host, ambiguous in virus, or both ambiguous). Integrations with an unambiguous location are then clustered based on host and viral coordinates. The two different clustering methods are illustrated bottom right.

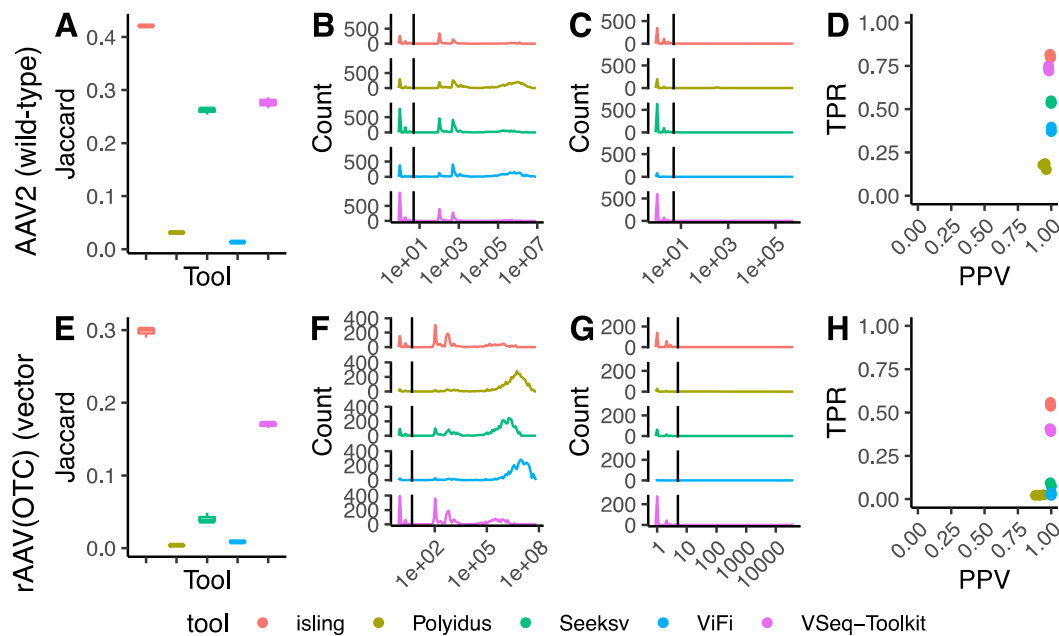


Figure 2. Benchmarking viral integration tool performance on two simulated integration datasets. A-D: Performance of isling, Polyidus, seeksv, ViFi and VSeq-Toolkit on a dataset of wild-type AAV2 integrations into human chr1 ($n = 3$ replicates, 1000 integrations each). Tool performance was assessed by Jaccard statistic (A), distance from each simulated integration to the nearest output integration (B), distance from each output integration to the nearest simulated integration (C), as well as true positive rate (TPR) and positive predictive value (PPV) calculated by applying a threshold of 5 bp to these distances (D; vertical line in B, C). E-H: Performance of the same tools on an AAV vector dataset, comparing using Jaccard statistic (E), distance from each simulated integration to the nearest output integration (F), distance from each output integration to the nearest simulated integration (G), TPR and PPV (H). A, E are boxplots showing the Jaccard statistic from each of three replicates, B, C, F and G are frequency polynomials aggregating all the distances observed amongst all replicates, and in D, H each point represents the PPV and TPR from one replicate.

and 170% higher on the rAAV(OTC) dataset compared to the next best tool (VSeq-Toolkit). The Jaccard statistic for isling was significantly higher than Polyidus (Dunn's test, p -value adjusted using a Benjamini-Hochberg correction; AAV2: $z = -2.46$, $n = 3$ each group, $p = 0.045$; OTC: $z = -3.29$, $n = 3$ each group, $p = 0.01$) and ViFi (AAV2: $z = -3.29$, $n = 3$ each group, $p = 0.001$; OTC: $z = -2.46$, $n = 3$ each group, $p = 0.046$), but the increases over seeksv (AAV2: $z = -1.64$, $n = 3$ each group, $p = 0.17$; OTC: $z = -1.64$, $n = 3$ each group, $p = 0.17$), and VSeq-Toolkit (AAV2: $z = -0.82$, $n = 3$ each group, $p = 0.41$; AAV2: $z = -0.82$, $n = 3$ each group, $p = 0.41$) were not significant.

Isling was also very precise, as most of the distances between simulated integration breakpoints and junctions identified by isling were small (≤ 5 bp; Figure 2(B) and (F)). Conversely, the distribution of distances between each identified junction to its nearest simulated breakpoint indicated that isling had also few false positives, since as the only major peak was at 0–3 bp (Figure 2(C) and (G)). Figure 2(D) and (H) summarize these observations, showing that while all tools had a high positive predictive value

(PPV), indicating hardly any false positives, isling also had the highest true positive rate (TPR) and therefore lowest false negative rate of all tools. Isling achieved this on both datasets, confirming that it is application agnostic.

Seeksv, which is a tool aimed at analysis of wild-type integrations, performed well at detecting AAV2 integrations but suffered a performance decrease on the rAAV(OTC) dataset as measured by the Jaccard statistic (Figure 2(A) versus (E)) and TPR (Figure 2(D) versus (H)). This difference was also visible in the distribution of distances from each simulated integration to the nearest output integration, with a visibly larger peak at large distances (1×10^6 – 1×10^7 bp). This peak indicated many more missed integrations in the rAAV(OTC) condition (Figure 2(B) and (F)).

VSeq-Toolkit performed well on both datasets despite being designed for vector data, and had the second highest Jaccard statistic (Figure 2(A) and (E)), and the second highest TPR (Figure 2(D) and (H)) for both datasets. The gap in performance between isling and VSeq-Toolkit was larger in the Jaccard statistic compared to the TPR, indicating that while VSeq-Toolkit correctly identified some integrations (TPR in Figure 2(D)

and (H)), it reported their junctions differently than isling, resulting in a lower Jaccard statistic.

Conversely, ViFi and Polyidus, both suitable for wild-type virus data analysis, had the lowest Jaccard statistics and TPR of all tools. This was true, even for the wild-type AAV2 dataset, where they identified just under half (ViFi) and just under a quarter (Polyidus) of simulated integrations (Figure 2(D)). Their performance was worse on the rAAV(OTC) dataset, likely because this dataset had low coverage and both tools require multiple supporting reads to call an integration junction. ViFi and Polyidus also had very low Jaccard statistics for both datasets (Figure 2(A) and (E)). For the AAV2 dataset, this a result of the larger intervals that ViFi reports for integration junctions, which are much longer than the true simulated junction lengths (average window of 170 ± 130 bp versus 1.2 ± 1.4 bp for isling on the AAV2 dataset; Supplementary table 3). These larger intervals resulted in a larger union between simulated and reported junctions, and therefore a lower Jaccard statistic. For Polyidus, the lower Jaccard statistic was the result of fewer identified integration events compared to the other tools, and therefore a lower intersection (96 bp mean intersection compared to 1625 bp for isling on the AAV2 dataset; Supplementary table 3). For the rAAV(OTC) dataset, the low Jaccard for both tools was caused by a high number of false negatives, consistent with the low TPRs observed (Figure 2 (D) and (H)).

Both isling and VSeq-Toolkit also had a second peak in the distance from simulated integration junction to the nearest output integration junction at around 1×10^2 bp (Figure 2(B) and (F)). This peak is the result of integration events with an associated deletion in the host genome at the integration site, which are difficult to detect and missed altogether by the other tools. Isling and VSeq-Toolkit detected at least one junction from these events, resulting in distances from the missed junction to the closest output junction of roughly 500 bp, hence the peak around 1×10^2 bp.

Consistent with the lack of false positives seen for all tools, there were no significant differences in the median PPV of isling compared to any of the other tools (Dunn's test, *p*-values adjusted using the Benjamini-Hochberg method, *n* = 3 each group; *p* = 0.20, 0.45, 0.21, 0.45 for isling versus Polyidus, seeksv, ViFi and VSeq-Toolkit, respectively, for the AAV2 data). On the other hand, the number of missed events varied between tools; the TPR for isling was significantly higher than Polyidus and ViFi for the AAV2 data, although the difference between isling and seeksv and VSeq-Toolkit was not significant (Dunn's test, *p*-values adjusted using the Benjamini-Hochberg method, *n* = 3 each group; *p* = 0.010, 0.045, 0.17 and 0.41 for isling versus Polyidus, ViFi, seeksv and VSeq-Toolkit, respectively). Further details of

the statistical tests can be found in [Supplementary File 1](#).

We next examined the rAAV(OTC) vector integration condition more closely, investigating how performance of these tools varied depending on the host chromosome used and the fold-coverage (Figure 3). Fold-coverage can affect accuracy since integrations are harder to detect when there are fewer reads that cross their host/virus junctions, while using different host chromosomes can lead to different alignments of the reads from the rAAV(OTC) vector, which contains portions of human chr14 (hAAT promoter) and chrX (OTC enhancer).³¹

Figure 3 shows the Jaccard statistic, mean positive predictive value (PPV) and true positive rate (TPR) for the five tools when varying fold-coverage and host reference chromosome, respectively. Isling had the highest Jaccard statistic over a range of fold-coverage values (Figure 3(A)). Indeed, the differences in Jaccard statistic between the tools were significant for all chromosomes and levels of fold-coverage, as indicated by the results of multiple Kruskal-Wallis tests (*H* = 10.38–13.5, *df* = 4, *p* = 0.014–0.016, *p*-values adjusted using the Benjamini-Hochberg method; see [Supplementary File 1](#) for the full details of each test). For all tools, the PPV was constant across different fold-coverage values, with the exception of a slight drop-off at high coverage for isling, a larger drop-off for VSeq-Toolkit, and an increase for Polyidus from low to high fold-coverage (Figure 3(B)). There is a clear rise in TPR with fold-coverage increase (Figure 3 (C)), consistent with more supporting reads for each junction at higher coverage resulting in more detected junctions. The difference between tools was again statistically significant for most chromosomes and levels of fold-coverage (see [Supplementary File 1](#) for the full details of each test). For each level of fold-coverage, the host chromosome did not appear to influence on the performance of most tools.

Comparison with other tools on real data

Next, isling was tested on four publicly available datasets in which integration of either wild-type viruses or vectors has been identified, and compared isling against the other tools which were suitable for each dataset. For this comparison, we used whole genome sequencing data and RNA-seq data from two studies that identified wild-type hepatitis B virus (HBV) integrations,^{32,33} as well as amplicon datasets from an AAV therapy for haemophilia³⁴ and a study of a dual-AAV gene therapy for Duchenne's muscular dystrophy (DMD).³⁵

First, we analysed whole genome sequencing data from a study of HBV integration in patients with hepatocellular carcinoma (HCC; SRA PRJEB2869).³² We downloaded reads for 13 sam-

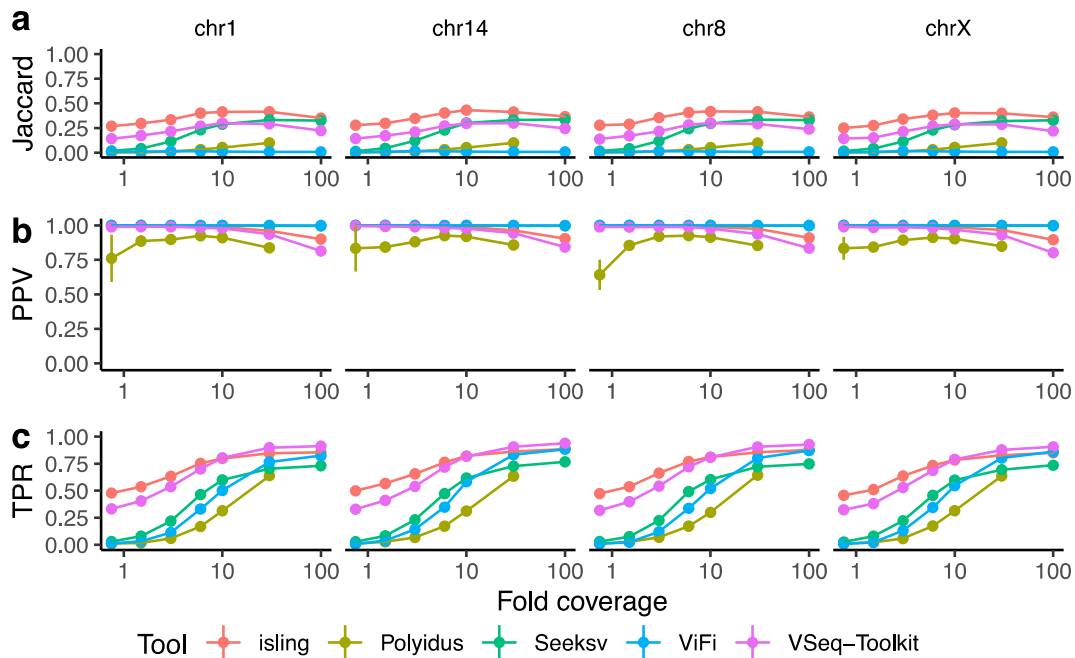


Figure 3. Performance of viral integration tools on AAV vector integration datasets with varying fold-coverage and host chromosome, assessed by Jaccard statistic (A), positive predictive value (B; PPV) and true positive rate (C; TPR). Points indicate the mean value from three replicates, and vertical lines at each point indicate the standard error of the mean. Missing points (for Polyidus, 100X coverage) indicate that analysis could not be completed within 7 days.

ples available on the SRA, with a total of 22 validated integration junctions. This was a large dataset, with an average of 667 million read pairs per sample. For each validated integration junction in this dataset, we identified the closest output integration breakpoint from each of the tools, and considered a junction to be found if the distance between validated and nearest output breakpoint was 5 bp or less (Figure 4(A), Supplementary tables 4 and 5). Isling identified the most validated integration breakpoints with 18 of 22 junctions found (81%), followed by ViFi with 14 (64%), Polyidus with 8 (36%) and seeksv with 6 (27%) (Figure 4(A)). Since VSeq-Toolkit is aimed at the analysis of vector data, we did not test this tool on this dataset.

Next, we investigated an RNA-seq dataset from a study of HBV-positive HCC cell lines, which identified chimeric transcripts containing HBV genomes.³³ This study identified 11 integration junctions from four cell lines. One of these junctions was complex, involving a structural variation in the host and appearing as virus/host (68 bp of *CDHR3* exon 19)/host (*TRRAP* intron 7). This complex integration could be output by the tools as two separate integration junctions, and therefore we treated this event as two possible integration junctions (Supplementary table 6), resulting in 12 junctions altogether. This dataset contained an average of 67 million read pairs per sample for the four cell lines.

Of the 12 validated integration junctions from the original study, isling identified 10 (83%), ViFi identified 12 (100%), Polyidus identified 6 (50%)

and seeksv identified 5 (42%) (Figure 4(B)). Some of the integrations fell within regions that were difficult to map (because the host alignments had low mapping qualities), and as isling subsets integrations based on the degree of confidence in their location, we were able to find all integration junctions by including junctions with ambiguous locations and increasing the distance threshold to 8 bp (Supplementary table 6). Importantly, these changes had no effect on the results for the other tools.

We next analysed a vector dataset from dogs with Haemophilia A, which were treated with an AAV-based gene therapy and followed over a number of years (SRA PRJNA606282).³⁴ This study recovered integration sites from liver samples by nested PCR, and validated 13 of these integration sites. We analysed these data using isling, VSeq-Toolkit, Polyidus and seeksv, but left out ViFi, which processes only human data (Figure 4(C)). Isling correctly identified 13 (100%) of these junctions, and Polyidus identified 11 (85%). Seeksv identified some integration sites but none of these were within 5 bp of a validated site, although two of these were within 100 bp of a validated site (Supplementary tables 4 and 7). VSeq-Toolkit did not output any integration sites for this dataset.

Finally, we tested isling on a second vector dataset from a mouse study of a dual AAV-CRISPR therapy for Duchenne's muscular dystrophy (SRA PRJNA485509).³⁵ Here, we compared isling against the other tools, again leaving

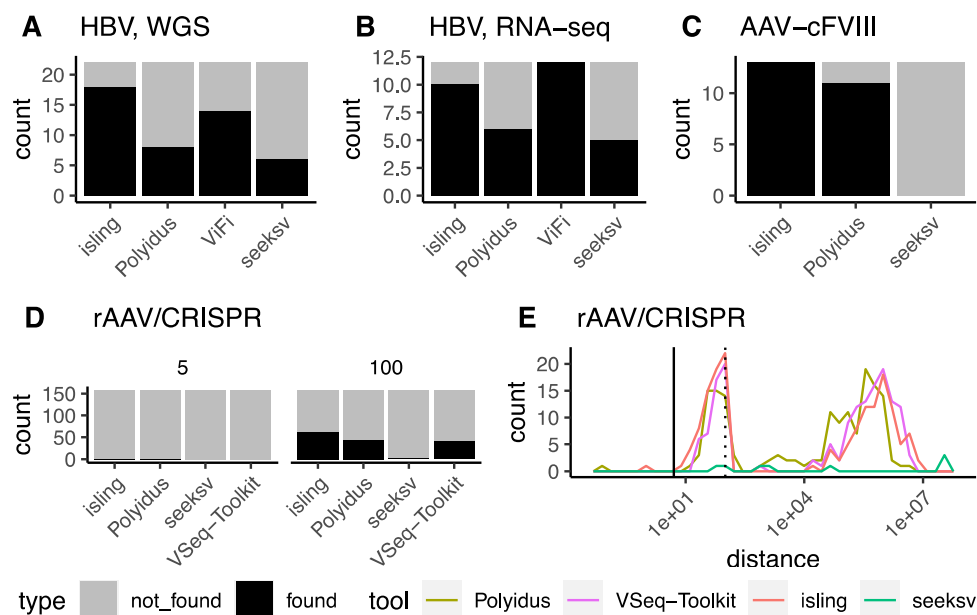


Figure 4. Performance on real datasets. A: Number of validated integration junctions from a whole genome sequencing (WGS) dataset of human liver tissue with Hepatitis B (HBV) integrations³² identified by four integration junction tools. B: Number of validated junctions identified by each tool in a RNA-seq data from cell lines with HBV integrations and expressing chimeric transcripts.³³ In a and b, output junctions are considered found if they are within 5 bp of a validated integration junction from the original study. C: Number of validated integration junctions found by each tool in a dataset of an AAV-cFVIII gene therapy administered to dogs.³⁴ D: Number of integration junctions in an amplicon dataset of integrations of an AAV/CRISPR therapy in mice³⁵ using a threshold of 5 bp (left) or 100 bp (right) of an integration junction from the original study. For a, b and d, black bars indicate number of junctions identified, grey fill indicates total number of integration junctions in the study. E: Frequency polynomial of the distances from each output integration junction to the nearest integration junction output for an amplicon sequencing dataset of rAAV vector integrations in various mouse tissues. Solid line indicates a distance of 5 bp, and dotted line indicates a distance of 100 bp.

out ViFi as it does not handle non-human data. This amplicon study used a Nextera-based library preparation method in which Tn5 transposons are first integrated into extracted DNA, then AAV/host junction fragments are amplified using one primer that binds in AAV and a second primer that binds in the integrated transposon. This results in enrichment of host/virus junctions, but only for junctions in which the primer-binding part of the AAV genomes are integrated. The authors identified 158 integration junctions in a total of 17 million reads, although these were not independently validated.

Using the same 5 bp threshold as for the previous datasets, all tools found only a small fraction of the reported potential integration junctions; isling and Polyidus identified one junction, while seeksv and VSeq-Toolkit identified none (Figure 4(D)). Examining the distances between validated and output junctions revealed that a majority of these distances were less than 100 bp (Figure 4(E), Supplementary table 8). We therefore relaxed the threshold for all tools to 100 bp, after which isling identified 63 (40%) junctions, Polyidus 44 (28%), VSeq-Toolkit 40 (25%), and seeksv 2 (1%) (Figure 4(D)).

Runtime

We next investigated the runtime (wall-clock time) of all tools on simulated data, varying either fold-coverage or viral load (number of integrations and episomes) as these have the largest influence on runtime.

First, 1000 integrations of the rAAV(OTC) vector into human chromosome 1 were simulated, and then reads were generated at a fold-coverage between 0.2 and 100x (Figure 5(A)). Of the tools tested, isling was the fastest at higher fold-coverage (1.6 times faster than the next fastest tool, ViFi, at 100x coverage), while VSeq-Toolkit and Polyidus were the slowest. This is due to isling's ability to split input reads for parallel processing, allowing the user to take advantage of all available cores. In general, as the fold-coverage increased, all tools had increased runtime, since each read must be examined to identify if it harbors an integration. Isling was slightly slower than other tools for low coverage conditions, likely due to the overheads from snakemake (for example, building the directed acyclic graph of jobs). The difference between the median runtimes of each tool was highly significant at each level of fold-coverage (Kruskal-Wallis tests,

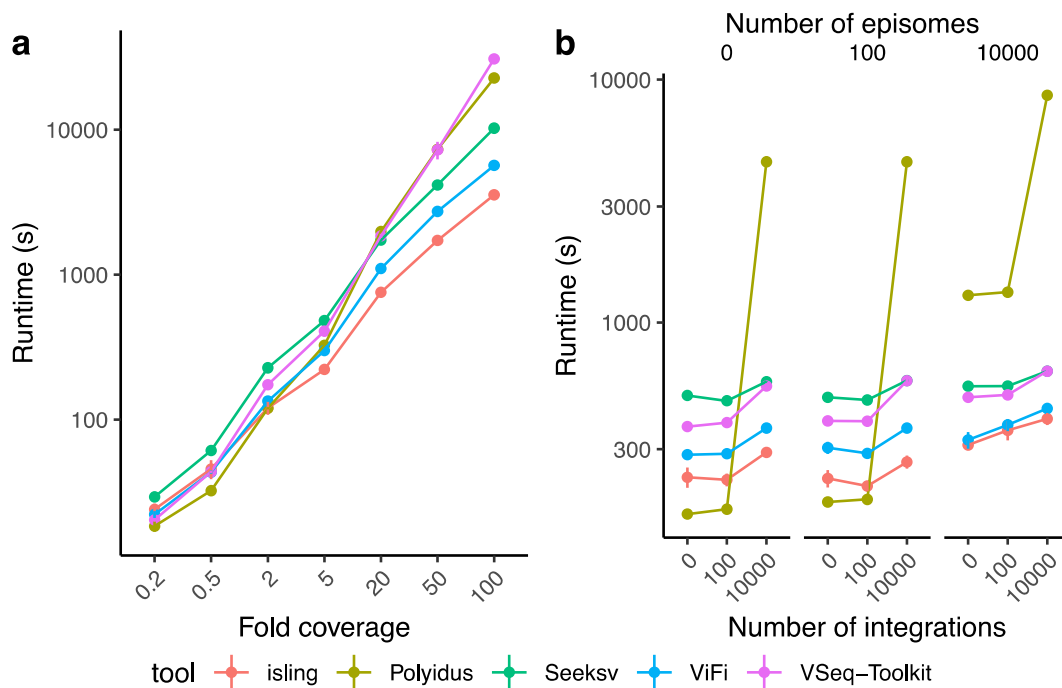


Figure 5. Effect of fold-coverage and viral load on viral integration tool runtime. A: Increase in runtime for increasing levels of fold-coverage in simulated data consisting of 1000 integrations of an AAV vector into human chromosome 1. B: Effect of both number of integrations (bottom axis) and number of episomal sequences (top labels) on reads simulated after integration of an AAV vector into human chromosome 1. Points indicate the mean of nine replicates, where each simulation (using a different random seed) was conducted three times, and runtime measured three times on each simulated set of reads. Vertical lines at each point indicate the standard error of the mean.

p-values adjusted using the Benjamini-Hochberg method, $p = 45$, $df = 4$, $H = 21.6\text{--}42.2$, $p = 2 \times 10^4\text{--}2 \times 10^8$, see [Supplementary File 1](#) for the full details of each test).

Next, we investigated the effect of increasing viral load on runtime by increasing both number of integration junctions and number of unintegrated episomal sequences, while keeping the fold-coverage constant at 5x (Figure 5(B)). At higher viral loads, isling was faster than the other tools tested, followed by ViFi, VSeq-Toolkit, seeksv and Polyidus. All tools had increased runtime as the viral load increased, with a pronounced increase for Polyidus at high numbers of episomes. The increase in runtime was likely due to the time taken to calculate the properties of each integration, and for clustering integrations, but the reason for the sharp increase for Polyidus is unclear. The difference between the median runtimes of each tool was also highly significant at each level of viral load (Kruskal-Wallis tests, p-values adjusted using the Benjamini-Hochberg method, $p = 45$, $df = 4$, $H = 33.6\text{--}41.2$, $p = 8.6 \times 10^4\text{--}9.7 \times 10^8$, see [Supplementary File 1](#) for the full details of each test).

Discussion

We developed isling to detect viral integration events in a general and highly flexible way,

providing an improvement over currently published tools. Specifically, isling can identify integration junctions that are supported by only one read or have ambiguous locations in the host or vector genome, and flags integrations with vector rearrangements, making it suitable for detecting wild-type viral and vector integration events alike. This will be useful for identifying differences in integration behaviour between wild-type viruses and vectors using the same tool. For example, wild-type AAV is known to undergo rep-mediated integration,³⁶ whereas AAV vector genomes do not contain the *rep* so become integrated through a different mechanism.^{11,37}

To demonstrate the utility of isling, we compared it against several other viral integration tools on simulated and real-world data (pipeline for generating datasets with the aim of assessing the characteristic differences between wild-type viral and vector integration scenarios).

In vector datasets, integration junctions of interest are sometimes supported by only one read. By detecting these low-coverage junctions, isling supports the requirement in these kinds of data for high sensitivity, having the highest true positive rate and Jaccard statistic on the simulated vector dataset, rAAV(OTC). It also identified the most integration junctions for the real datasets (dual-AAV CRISPR therapy for DMD,³⁵ AAV-cFIII.³⁴ Tools aimed at wild-type viral integration detection

(ViFi, Polyidus and seeksv) generally performed poorly on these vector datasets, likely because they identify only integration junctions supported by multiple reads. The ability to detect integration junctions that are supported by a single read are critical for some, but not all, vector integration datasets. For those datasets where clonally expanded events are primarily of interest, isling performs clustering to combine reads originating from the same integration event. This gives users the flexibility to filtering out single-read integration junctions when only clonally expanded integration events are of interest, or retain all single-read breakpoints if they are required.

While the clustering of reads enable isling to cater for the properties of different datasets, it also poses a trade-off the user should be aware of. When manipulating the fold-coverage of simulated reads, we observed that isling and VSeq-Toolkit both had a drop in positive predictive value at very high coverage (>30X). This drop is caused by integrations identified in discordant pairs: where possible, isling outputs a 'best guess' location based on mean fragment size, and can cluster these with overlapping chimeric reads resulting in a more accurate integration junction location. However, in larger datasets, discordant pairs in fragment sizes divergent from the mean are more common, but for these integrations the 'best guess' coordinates may not overlap with chimeric reads from the same junction. These therefore can be scored as false positives, despite originating from true integrations. When clustering, there is a trade-off between running the risk of combining independent integration events that are close to one another, and failing to combine reads originating from the same integration event. Isling is relatively conservative in this regard, aiming to avoid false clustering by combining either events that overlap, or have exactly the same genomic coordinates (two options that the user can decide between). However, this conservatism can have the downside of failing to combine discordant pairs in some cases. Isling indicates the types of reads for each integration junction, so if integrations with clearly defined locations are of interest, the user may ignore any integrations resulting only from discordant pairs in downstream analysis.

Another issue that was apparent, particularly in the RNA-seq HBV integration dataset³³ was the challenge presented by integration junctions present in reads with low mapping qualities or multiple equivalent alignments. These are correctly identified as integrations by isling, but cannot be uniquely localised in the host and/or viral genome. For example, isling correctly identified all integrations in the RNA-seq HBV integration dataset, but some of these mapped with low mapping qualities and were therefore not in the 'unambiguous' subset of the data. While the strategy of using single reads to

identify integration junctions is required for vector datasets and therefore necessary for an application-agnostic tool, one of the disadvantages of this approach is that reads that map to either host or vector with low mapping quality or in an ambiguous manner cannot be localised. Other approaches, such as VERSE¹⁹ and VIRUSbreakend,¹⁷ use localised assembly at putative integration junctions to solve this issue, creating a longer contig which is more likely to be mapped unambiguously. However, such approaches are unsuited for vector data reads spanning the host/vector junction may be few. Yet other tools, such as ViFi and Polyidus, attempt to remove ambiguity from results by removing reads with low mapping qualities or from regions of the genome that have low mappability. However, in vector data, all putative integrations are of interest, even those with ambiguous locations. Therefore, isling takes the approach of flagging all possible ambiguity in integration junction locations, in a similar manner to the retroviral-specific pipeline INSPIRED.²¹ The user can then decide which of the classes of output integrations are of interest.

Finally, isling is configurable to various kinds of analyses. For example, it does not impose restrictions on the host or vector reference that can be used, and these references can be supplied in FASTA format or as a pre-built *bwa mem* index. Isling also provides a flexible interface for filtering integrations, which allows the user to easily subset integrations of interest based on their properties. Developed using snakemake,³⁸ isling can automatically supply dependencies using either conda or singularity,³⁹ or alternatively can be run in a pre-built docker⁴⁰ container. Isling is also fast, due to the combination of parallelization provided by snakemake, and the 'split' parameter which divides the workload for input files into smaller chunks. Another benefit of snakemake is ease of deployment and parallelization of isling in a range of environments, including locally, in high-performance computing environments, and in the cloud.

Materials and Methods

Terminology

We use the following terms to refer to integrations of a viral genome into a host genome. An *integration event* occurs when a viral genome, whole or in part, is integrated into a host genome. The *integration site* refers to the place in the host genome at which integration occurs, and at each integration site there are two *integration junctions*, which refer to the breakpoints created between host and viral genomes by the integration event. An integration junction is specified by the host and viral coordinates at the breakpoint between host and virus. If there is a deletion from the host genome during the integration event, the integration site may not be a single location. We use the term

chimeric read to indicate a single read which contains portions that originate from more than one reference (for example, part host and part virus), and *discordant pair* to indicate a pair of reads in which one read originates from the host and the other to the virus.

Integration junction detection

We developed *isling*, a pipeline for viral integration detection from paired-end next generation sequencing (NGS) data. *Isling*'s overall goal is to detect integration events using host and viral alignments. It does this by identifying chimeric reads, consisting of either one host/virus breakpoint that falls within a single read, or a short integration event where a read harbors both host/virus junctions from a single integration event. It also identifies integrations which originate from discordant pairs, in which an integration breakpoint falls between the two reads of a pair. *Isling* was developed using *snakemake*,³⁸ which allows the parallelization of jobs, the automated deployment of dependencies using *singularity* or *conda*, and the ability to run locally, on a cluster or in the cloud. *Isling* is also available as a Docker image at Docker Hub (<https://hub.docker.com/r/szsctt/isling>).

Isling has four main steps: read pre-processing, alignment, integration identification, and post-processing (Figure 1). In the pre-processing step, reads may be de-duplicated, trimmed of adapters. Overlapping read-pairs may be merged at this stage. The de-duplication step uses *Clumpify* from the *BBTools* package,⁴¹ and merging and trimming are performed by *SeqPrep*.⁴² All are optional. In the alignment step, reads are first aligned to a viral reference, and then reads of interest are extracted using *SAMtools*.⁴³ For potential chimeric reads, the reads of interest all mapped reads. For potential discordant pairs, reads of interest are all paired reads with at least one read in the pair mapped. Extracted reads are then aligned to a host reference. Conducting alignments in this order typically reduces the computational requirements due to the smaller genome size of the virus(es) compared to the host, and the smaller number of reads originating from the virus compared to the host. Alignments are conducted using *bwa mem*.⁴⁴

Next, integrations are detected using the host and viral alignments by identifying chimeric reads and discordant pairs. Chimeric reads may contain 'ambiguous bases', which either map to both host and viral genomes (an overlap), or are unmapped in both (a gap). Each chimeric read is assigned an 'edit distance', which is the sum of base pairs that differ from the host and viral genomes to which the chimeric read maps (the Levenshtein distance reported by *bwa mem*), as well as the number of unmapped bases. Since the edit distance is intended to enumerate bases in the read that are unaccounted for, we do not include overlapped

bases (mapped to both host and virus) in this metric.

For each putative integration, *isling* checks if they are likely 'rearrangements' of the host or viral genome (Figure 1). These rearrangements are identified by evaluating how much of the read is covered by additional alignments, i.e. alignments to the appropriate genome with a lower score than the primary alignment (that is, supplementary and secondary alignments from *bwa mem*). To decide if a read could be a rearrangement, we compare a 'rearrangement edit distance' with the 'integration edit distance' described above. The rearrangement edit distance is calculated in the same way as for integration edit distance, except that the primary, secondary and supplementary alignments from a single reference are used. If there are multiple alignments that cover the same part of the read, only the longest of these overlapping alignments are retained for the edit distance calculation. Reads are indicated to be a rearrangement, rather than an integration junction, if the rearrangement edit distance is smaller than the integration edit distance.

In some cases, one or more secondary alignments are equivalent to the primary alignment in either host or vector, in that they cover the same part of the read. In this case the integration junction cannot be uniquely localised in the reference genome and is considered to have multiple possible locations (an 'ambiguous location'). To identify all possible junction locations, each primary, secondary and supplementary alignment from the host and viral reference are considered, and if they constitute a valid integration all possible locations are output. For each alternate location we again compute an edit distance (calculated for integrations as explained above); if this is not similar (within a user-defined threshold) to the edit distance for the primary integration, it is discarded. Additionally, an integration junction is considered to have an ambiguous location if the host or viral alignments have a low mapping quality (below a user-defined threshold).

After detection, integration junctions can be filtered according to user-defined criteria (such as total edit distance, number of ambiguous bases, and vector rearrangement status). After filtering, the remaining integration junctions are then sorted into four categories, based on the ambiguity of the junction location. Each junction has a location in host and viral genomes that are either uniquely localised or not, as outlined above. Integration junctions are categorised into four groups: unambiguous location in both host and virus or vector, ambiguous location in host genome, ambiguous location in virus or vector genome, and ambiguous locations in both host and virus or vector genome.

Lastly, redundancy that occurs when multiple reads originate from the same integration event is

Table 1 Parameters used for AAV2 and rAAV(OTC) integration simulation

Parameter	AAV2 integrations	OTC vector integrations	Description
host	GRCh38 chr1	GRCh38 chr1	Host sequence into which viral fragments are integrated
virus	AAV2 (Genbank NC_001401.2)	OTC vector	Viral sequence to be integrated
int_num	1000	1000	Number of integration events
epi_num	1000	100	Number of episomes (non-integrated viral genomes)
p(delete)	0.1	0	Probability of an integrated virus harbouring a deletion of a randomly chosen fragment
p(rearrange)	0.1	0	Probability of an integrated virus being rearranged by the creation of a breakpoint, and swapping the two fragments either side of that breakpoint
fcov	1.5	5	The fold-coverage of the simulated reads

removed by clustering integration junctions based on both host and vector genome coordinates. During clustering, a balance must be struck between correctly combining reads from the same integration event (due to clonal expansion), and avoiding clustering reads from independent integration events that occur close to each other. There are therefore two options for clustering integration junctions available in *isling*. In ‘overlap’ clustering, two or more integration junctions are combined if their coordinates *overlap* in both host and virus or vector genome. The output coordinates reflect the coordinates that are common to all combined integrations, which has the effect of combining ‘best guess’ coordinates from a discordant pair with more precise coordinates from a chimeric read. In ‘exact’ clustering, two or more integration junctions are combined if their coordinates are the *same* in both host and virus or vector genome. This option may be useful for vector datasets where clonality is expected to be low and therefore combining overlapping integration junctions may result in combining distinct integration events. These two clustering methods are illustrated in [Figure 1](#).

Isling also produces an Rmarkdown⁴⁵ html report containing a general summary of the detected integration junctions, including an overview of the parameters used, the number of detected junctions and their location in the host and viral genomes.⁴⁶

Comparing viral integration tool performance: simulated data

To benchmark the performance of *isling* against other tools, viral integration events were simulated by inserting a viral sequence into randomly chosen positions in the host sequence, yielding a uniform distribution of integration events. Two tested scenarios simulate either wild-type AAV2 (GenBank NC_001401.2) or recombinant AAV vector integrations into human (GRCh38) chromosome 1. The AAV vector reference used was a pre-clinical recombinant AAV vector cassette encoding human Ornithine

Transcarbamylase, henceforth referred to as rAAV (OTC) vector.³¹ This vector contains homology with the OTC locus on chrX, and the SERPINA1 locus on chr14, due to the presence of an OTC enhancer element and the SERPINA1 promoter.

During simulation, we aimed to reproduce previously described elements of AAV integration behaviour ([Table 1](#); supplementary table 1), which include the propensity for sub-genomic fragments of wild-type and vectorized AAV genomes to be integrated, rather than the whole genome.⁷ The propensity for AAV vector genomes to have structural variation introduced during vector production was also simulated by including a probability of rearrangement and deletion for each integrated viral fragment. The possibility of deletions from the host chromosome at the integration junction³⁷ was included in the form of a probability of this occurring at each integration junction. To mimic the fact that most AAV or vector genomes are not integrated but are present as episomes,¹¹ additional virus or vector sequences were included in the output FASTA used for read simulation. [Table 1](#) shows the choices made for simulation parameters. Three replicates ($n = 3$) were performed for each condition, each with a different random seed.

Next, reads were simulated from the respective integration scenarios outlined above using *art_illumina*.⁴⁷ For vector integration, a lower coverage was used to reflect the distributed, heterogeneous nature of vector integration, whereas for wild-type integrations a higher fold-coverage was used to mimic clonal expansion of integrations. After simulation, reads that cross each integration junction were annotated. Integration junctions without any associated reads, which can occur if the fold-coverage is low, were discarded.

The performance of *isling* was compared against *seeksv*,³⁰ *Polyidus*,⁴ *ViFi*¹⁸ and *VSeq-Toolkit*.²⁷ In order to run these tools, dependencies and code were included in Docker⁴⁰ images and run with Singularity.³⁹ Dockerfiles for these containers are available at GitHub⁴⁸ and Docker images can be downloaded from Docker Hub.^{49–52} We also tested

GENE-IS,²⁶ BATVI,⁵³ HGT-ID⁵⁴ and Virus-Clip,⁵⁵ which did not run successfully in our hands.

The host genome locations of the integration junctions identified by these tools were compared against the locations of the simulated integrations using BEDtools 'closest'.⁵⁶ We calculated the distance between each simulated integration junction and the nearest output integration junction from the tested tools, as well as the distance between each output integration breakpoint and the nearest simulated junction. The start and stop coordinates of the simulated integrations were the coordinates of the ambiguous bases (gap or overlap) at each junction, and accounted for deletions from the host, which result in a displacement of the right junction coordinates from the left junction in the host genome, where necessary. For tools that only output a single coordinate for integration junction location, this location was used for the start coordinate, and one base added to this value for the stop coordinate.

To calculate the positive predictive value and true positive rate, integrations were classified into three categories: *true positives*, which are simulated integrations that are close to an output integration; *false negatives*, which are simulated integrations that are far from an output integration; and *false positives*, which are output integrations that are far from a simulated integration. Typically, the threshold used for distinguishing true positives from false positives and false negatives was 5 bp. We did not enumerate true negatives, since these would be all positions in the host reference at which an integration was not simulated or lacking an output integration junction. These true negatives would greatly outnumber the other classes and would skew the calculated metrics. The positive predictive value (PPV) and true positive rate (TPR) were calculated using the below equations:

$$PPV = \frac{tp}{tp + fp}$$

$$TPR = \frac{tp}{tp + fn}$$

Additionally, the Jaccard statistic was used to compare simulated integration locations in the host genome with the output of each tool. To calculate this metric, output files from each tool were converted to bed format and compared against the simulated junctions using BEDtools 'jaccard'.⁵⁶ The Jaccard statistic calculates the relative difference between the intersection and union without the intersection (symmetric difference) of two given sets of genomic regions.⁵⁷ For isling, we compared only the output integrations that had unambiguous locations in the host genome (combining both the sets of integrations that had unambiguous locations in both references with those that had unambiguous locations only in the host genome). For VSeq-Toolkit, we similarly used for

analysis those integration junctions that were unique for analysis.

Comparing viral integration tool performance: published datasets

SRA datasets PRJEB2869, PRJNA298941, PRJNA485509 and PRJNA606282 were downloaded using the SRA toolkit,⁵⁸ or directly from the European Nucleotide Archive. Viral integration tools were run and compared as outlined above, and their outputs converted to BED file format. The output integration locations were compared against the published integration site locations for each tool using BEDtools 'closest' as outlined above.

Comparing viral integration tool performance: runtime

To compare runtime, Docker images containing each tool, including isling, were converted to the Singularity 'sif' format. All tools were run using these Singularity images on a Dell PowerEdge M630. Each run was given access to 20 cores, 128 Gb of memory and a maximum wall time of 24 hours, and was repeated three times on each input pair of FASTQ files. Elapsed time was measured with GNU time.

Figure preparation and statistics

Data were analysed using R (version 4.1.1),⁵⁹ including the tidyverse (1.3.1),⁶⁰ cowplot (1.1.1)⁶¹ and rstatix (0.7.0)⁶² packages. [Supplementary File 1](#) was additionally created using the knitr (1.35)⁶³ and rmarkdown (2.11)⁶⁴ packages.

Data Availability

Code for simulation and analysis available on GitHub at <https://github.com/aeherc/isling>, and at the CSIRO data access portal (<https://doi.org/10.25919/gmpm-w041>).

Acknowledgements

The authors gratefully acknowledge assistance from the CSIRO Scientific Computing team.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary data to this article can be found online at <https://doi.org/10.1016/j.jmb.2021.167408>.

Received 30 September 2021;

Accepted 13 December 2021;

Available online 17 December 2021

Keywords:

virology;
gene therapy;
bioinformatics;
genotoxicity;
cancer

References

- White, M.K., Pagano, J.S., Khalili, K., (2014). Viruses and human cancers: a long road of discovery of molecular paradigms. *Clin. Microbiol. Rev.* **27**, 463–481.
- Martin, D., Gutkind, J.S., (2008). Human tumor-associated viruses and new insights into the molecular mechanisms of cancer. *Oncogene* **27** (Suppl 2), S31–S42.
- Nault, J.-C., Datta, S., Imbeaud, S., Franconi, A., Mallet, M., Couchy, G., Letouzé, E., Pilati, C., et al., (2015). Recurrent AAV2-related insertional mutagenesis in human hepatocellular carcinomas. *Nature Genet.* **47**, 1187–1193.
- Karimzadeh, M., Arlidge, C., Rostami, A., Lupien, M., Bratman, S.V., Hoffman, M.M., (2020). Viral integration transforms chromatin to drive oncogenesis. *BioRxiv* <https://www.biorxiv.org/content/early/2020/08/18/2020.02.12.942755>.
- La Bella, T., Imbeaud, S., Peneau, C., Mami, I., Datta, S., Bayard, Q., Caruso, S., Hirsch, T.Z., et al., (2019). Adeno-associated virus in the liver: natural history and consequences in tumour development. *Gut. gutjnl-318281*.
- Zapatka, M., Borozan, I., Brewer, D.S., Iskar, M., Grundhoff, A., Alawi, M., Desai, N., Sülthmann, H., et al., (2020). The landscape of viral associations in human cancers. *Nature Genet.* **52**, 320–330. <https://doi.org/10.1038/s41588-019-0558-9>.
- Nguyen, G.N., Everett, J.K., Kafle, S., Roche, A.M., Raymond, H.E., Leiby, J., Wood, C., Assenmacher, C.-A., et al., (2020). A long-term study of AAV gene therapy in dogs with hemophilia A identifies clonal expansions of transduced liver cells. *Nature Biotechnol.*, 1–9. <https://doi.org/10.1038/s41587-020-0741-7>.
- Chandler, R.J., LaFave, M.C., Varshney, G.K., Trivedi, N. S., Carrillo-Carrasco, N., Senac, J.S., Wu, W., Hoffmann, V., et al., (2015). Vector design influences hepatic genotoxicity after adeno-associated virus gene therapy. *J. Clin. Investig.* **125**, 870–880.
- Kohn, D.B., Sadelain, M., Glorioso, J.C., (2003). Occurrence of leukaemia following gene therapy of X-linked SCID. *Nature Rev. Cancer* **3**, 477–488. <https://doi.org/10.1038/nrc1122>.
- Smith, R.H., Hallwirth, C.V., Westerman, M., Hetherington, N.A., Tseng, Y.-S., Cecchini, S., Virag, T., Ziegler, M.-L., et al., (2016). Germline viral “fossils” guide in silico reconstruction of a mid-Cenozoic era marsupial adeno-associated virus. *Sci. Rep.* **6** 28965.
- Smith, R.H., (2008). Adeno-associated virus integration: virus versus vector. *Gene Ther.* **15**, 817–822.
- Donsante, A., Miller, D.G., Li, Y., Vogler, C., Brunt, E.M., Russell, D.W., Sands, M.S., (2007). AAV vector integration sites in mouse hepatocellular carcinoma. *Science* **317**, 477. <https://doi.org/10.1126/science.1142658>.
- Kaepfel, C., Beattie, S.G., Fronza, R., van Logtenstein, R., Salmon, F., Schmidt, S., Wolf, S., Nowrouzi, A., et al., (2013). A largely random AAV integration profile after LPD gene therapy. *Nature Med.* **19**, 889–891. <https://doi.org/10.1038/nm.3230>.
- Nguyen, G.N., Everett, J.K., Raymond, H., Kafle, S., Merricks, E.P., Kazazian, H.H., Nichols, T.C., Bushman, F.D., et al., (2019). Long-term AAV-mediated factor VIII expression in nine Hemophilia A dogs: a 10 year follow-up analysis on durability, safety and vector integration. *Blood* **134**, 611. <https://doi.org/10.1182/blood-2019-126007>.
- Zhang, J., Guo, P., Yu, X., Pouchnik, D., Firman, J., Wei, H., Sang, N., Li, D., et al., (2020). Subgenomic satellite particle generation in recombinant AAV vectors results from DNA lesion/breakage and non-homologous end joining. *BioRxiv*. 2020.08.01.230755-2020.08.01.230755.
- Chen, X., Kost, J., Li, D., (2019). Comprehensive comparative analysis of methods and software for identifying viral integrations. *Brief Bioinform.* **20**, 2088–2097. <https://doi.org/10.1093/bib/bby070>.
- Cameron, D.L., Jacobs, N., Roepman, P., Priestley, P., Cuppen, E., Papenfuss, A.T., (2021). VIRUSBREKEND: viral integration recognition using single breakends. *Bioinformatics* **37**, 3115–3119. <https://doi.org/10.1093/bioinformatics/btab343>.
- Nguyen, N.P.D., Deshpande, V., Luebeck, J., Mischel, P. S., Bafna, V., (2018). ViFi: Accurate detection of viral integration and mRNA fusion reveals indiscriminate and unregulated transcription in proximal genomic regions in cervical cancer. *Nucleic Acids Res.* **46**, 3309–3325. <https://doi.org/10.1093/nar/gky180>.
- Wang, Q., Jia, P., Zhao, Z., (2015). VERSE: a novel approach to detect virus integration in host genomes through reference genome customization. *Genome Med.* **7** <https://doi.org/10.1186/s13073-015-0126-6>.
- Sherman, E., Nobles, C., Berry, C.C., Six, E., Wu, Y., Dryga, A., Malani, N., Male, F., et al., (2017). INSPIRED: a pipeline for quantitative analysis of sites of new DNA integration in cellular genomes. *Mol. Therapy. Methods Clin. Develop.* **4**, 39–49.
- Berry, C.C., Nobles, C., Six, E., Wu, Y., Malani, N., Sherman, E., Dryga, A., Everett, J.K., et al., (2017). INSPIRED: quantification and visualization tools for analyzing integration site distributions. *Mol. Therapy. Methods Clin. Develop.* **4**, 17–26. <https://doi.org/10.1016/j.omtm.2016.11.003>.
- Kamboj, A., Hallwirth, C.V., Alexander, I.E., McCowage, G. B., Kramer, B., (2017). Ub-ISAP: A streamlined UNIX pipeline for mining unique viral vector integration sites from next generation sequencing data. *BMC Bioinf.* **18**, 305. <https://doi.org/10.1186/s12859-017-1719-4>.
- Hocum, J.D., Battrell, L.R., Maynard, R., Adair, J.E., Beard, B.C., Rawlings, D.J., Kiem, H.P., Miller, D.G., et al., (2015). VISA - Vector Integration Site Analysis server: a web-based server to rapidly identify retroviral integration sites

- from next-generation sequencing. *BMC Bioinf.* **16** <https://doi.org/10.1186/s12859-015-0653-6>.
24. Jones, D.H., Winistorfer, S.C., (1992). Sequence specific generation of a DNA panhandle permits PCR amplification of unknown flanking DNA. *Nucleic Acids Res.* **20**, 595–600. <https://doi.org/10.1093/nar/20.3.595>.
 25. Rosenthal, A., Jones, D.S., (1990). Genomic walking and sequencing by oligo-cassette mediated polymerase chain reaction. *Nucleic Acids Res.* **18**, 3095–3096. <https://doi.org/10.1093/nar/18.10.3095>.
 26. Afzal, S., Wilkening, S., von Kalle, C., Schmidt, M., Fronza, R., (2017). GENE-IS: time-efficient and accurate analysis of viral integration events in large-scale gene therapy data. *Mol. Ther. Nucleic Acids* **6**, 133–139. <https://doi.org/10.1016/j.omtn.2016.12.001>.
 27. Afzal, S., Fronza, R., Schmidt, M., (2020). VSeq-toolkit: comprehensive computational analysis of viral vectors in gene therapy. *Mol. Ther. Methods Clin. Dev.* **17**, 752–757. <https://doi.org/10.1016/j.omtm.2020.03.024>.
 28. Schmidt, M., Schwarzwaelder, K., Bartholomae, C., Zaoui, K., Ball, C., Pilz, I., Braun, S., Glimm, H., et al., (2007). High-resolution insertion-site analysis by linear amplification-mediated PCR (LAM-PCR). *Nature Methods* **4**, 1051–1057.
 29. Schmidt, M., Hoffmann, G., Wissler, M., Lemke, N., Müssig, A., Glimm, H., Williams, D.A., Ragg, S., et al., (2001). Detection and direct genomic sequencing of multiple rare unknown flanking DNA in highly complex samples. *Hum. Gene Ther.* **12**, 743–749. <https://doi.org/10.1089/104303401750148649>.
 30. Liang, Y., Qiu, K., Liao, B., Zhu, W., Huang, X., Li, L., Chen, X., Li, K., (2017). Seeksv: an accurate tool for somatic structural variation and virus integration detection. *Bioinformatics* **33**, 184–191. <https://doi.org/10.1093/bioinformatics/btw591>.
 31. Baruteau, J., Cunningham, S.C., Yilmaz, B.S., Perocheau, D.P., Eaglestone, S., Burke, D., Thrasher, A.J., Waddington, S.N., et al., (2021). Safety and efficacy of an engineered hepatotropic AAV gene therapy for ornithine transcarbamylase deficiency in cynomolgus monkeys. *Mol. Ther. Methods Clin. Dev.* **23**, 135–146. <https://doi.org/10.1016/j.omtm.2021.09.005>.
 32. Sung, W.-K., Zheng, H., Li, S., Chen, R., Liu, X., Li, Y., Lee, N.P., Lee, W.H., et al., (2012). Genome-wide survey of recurrent HBV integration in hepatocellular carcinoma. *Nature Genet.* **44**, 765–769.
 33. Lau, C.-C., Sun, T., Ching, A.K.K., He, M., Li, J.-W., Wong, A.M., Co, N.N., Chan, A.W.H., et al., (2014). Viral-human chimeric transcript predisposes risk to liver cancer development and progression. *Cancer Cell* **25**, 335–349.
 34. Nguyen, G.N., Everett, J.K., Kafle, S., Roche, A.M., Raymond, H.E., Leiby, J., Wood, C., Assenmacher, C.-A., et al., (2021). A long-term study of AAV gene therapy in dogs with hemophilia A identifies clonal expansions of transduced liver cells. *Nature Biotechnol.* **39**, 47–55. <https://doi.org/10.1038/s41587-020-0741-7>.
 35. Nelson, C.E., Wu, Y., Gemberling, M.P., Oliver, M.L., Waller, M.A., Bohning, J.D., Robinson-Hamm, J.N., Bulaklak, K., et al., (2019). Long-term evaluation of AAV-CRISPR genome editing for Duchenne muscular dystrophy. *Nature Med.* **25**, 427–432.
 36. Hüser, D., Gogol-Döring, A., Lutter, T., Weger, S., Winter, K., Hammer, E.-M., Cathomen, T., Reinert, K., et al., (2010). Integration Preferences of Wildtype AAV-2 for Consensus Rep-Binding Sites at Numerous Loci in the Human Genome. *PLoS Pathog.* **6** e1000985.
 37. Deyle, D.R., Russell, D.W., (2009). Adeno-associated virus vector integration. *Curr. Opin. Mol. Ther.* **11**, 442–447.
 38. Köster, J., Rahmann, S., (2012). Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics* **28**, 2520–2522. <https://doi.org/10.1093/bioinformatics/bts480>.
 39. Kurtzer, G.M., Sochat, V., Bauer, M.W., (2017). Singularity: Scientific containers for mobility of compute. *PLoS ONE* **12** <https://doi.org/10.1371/journal.pone.0177459>.
 40. Merkel, D., (2014). Docker: lightweight Linux containers for consistent development and deployment. *Linux J.* **2014** 2:2.
 41. BBTools (n.d.) <https://sourceforge.net/projects/bbmap/> (accessed February 12, 2021).
 42. SeqPrep (n.d.) <https://github.com/jstjohn/SeqPrep> (accessed February 3, 2021).
 43. Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., et al., (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**, 2078–2079. <https://doi.org/10.1093/bioinformatics/btp352>.
 44. Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. <http://arxiv.org/abs/1303.3997>.
 45. Allaire, J., Yihui, X., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H. (2020). rmarkdown: Dynamic Documents for R. <https://rmarkdown.rstudio.com/> (accessed July 19, 2021).
 46. Gel, B., Serra, E., (2017). karyoploteR: an R/Bioconductor package to plot customizable genomes displaying arbitrary data. *Bioinformatics* **33**, 3088–3090. <https://doi.org/10.1093/bioinformatics/btx346>.
 47. Huang, W., Li, L., Myers, J.R., Marth, G.T., (2012). ART: a next-generation sequencing read simulator. *Bioinformatics* **28**, 593–594. <https://doi.org/10.1093/bioinformatics/btr708>.
 48. Viral integration tool dockerfiles (n.d.) https://github.com/szsctt/intvi_other-tools (accessed August 15, 2021).
 49. VSeq-Toolkit docker hub page (n.d.) https://hub.docker.com/r/szsctt/vseq_toolkit (accessed August 15, 2021).
 50. ViFi docker hub page (n.d.) <https://hub.docker.com/r/szsctt/vifi> (accessed August 15, 2021).
 51. Seeksv docker hub page (n.d.) <https://hub.docker.com/r/szsctt/seeksv> (accessed August 15, 2021).
 52. Polyidus docker hub page (n.d.) <https://hub.docker.com/r/szsctt/polyidus> (accessed August 15, 2021).
 53. Tennakoon, C., Sung, W.-K., (2017). BATVI: Fast, sensitive and accurate detection of virus integrations. *BMC Bioinf.* **18**, 71–111.
 54. Baheti, S., Tang, X., O'Brien, D.R., Chia, N., Roberts, L.R., Nelson, H., Boughey, J.C., Wang, L., et al., (2018). HGT-ID: An efficient and sensitive workflow to detect human-viral insertion sites using next-generation sequencing data. *BMC Bioinf.* **19** <https://doi.org/10.1186/s12859-018-2260-9>.
 55. Ho, D.W.H., Sze, K.M.F., Ng, I.O.L., (2015). Virus-Clip: a fast and memory-efficient viral integration site detection tool at single-base resolution with annotation capability. *Oncotarget* **6**, 20959–20963.
 56. Quinlan, A.R., Hall, I.M., (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26**, 841–842. <https://doi.org/10.1093/bioinformatics/btq033>.

57. Favorov, A., Mularoni, L., Cope, L.M., Medvedeva, Y., Mironov, A.A., Makeev, V.J., Wheelan, S.J., (2012). Exploring Massive Genome Scale Datasets with the GenometriCorr Package. *PLOS Comput. Biol.* **8**, <https://doi.org/10.1371/journal.pcbi.1002529> e1002529.
58. SRA Toolkit (n.d.) <http://ncbi.github.io/sra-tools/> (accessed August 2, 2021).
59. R Core Team (2021). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. <https://www.R-project.org/>.
60. Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Grolemund, G., Hayes, A., et al., (2019). Welcome to the tidyverse. *J. Open Source Software* **4**, 1686. <https://doi.org/10.21105/joss.01686>.
61. C.O. Wilke, cowplot: Streamlined Plot Theme and Plot Annotations for “ggplot2,” 202AD. <https://CRAN.R-project.org/package=cowplot>.
62. The Rstatix team, Rstatix, n.d. <https://cran.r-project.org/web/packages/rstatix/index.html>.
63. Xie, Y. (2014). knitr: A Comprehensive Tool for Reproducible Research in R. In: *Implementing Reproducible Computational Research* (Stodden, V., Leisch, F., & Peng, R.D. eds), Chapman and Hall/CRC, 2014. <http://www.crcpress.com/product/isbn/9781466561595>.
64. rmarkdown: Dynamic Documents for R (2021). <https://github.com/rstudio/rmarkdown>.