Human Genomics

RESEARCH Open Access

Fast and accurate DNASeq variant calling workflow composed of LUSH toolkit

Taifu Wang[1,2†], Youjin Zhang[1,2†], Haoling Wang[1,2†], Qiwen Zheng[1,2], Jiaobo Yang[1,2], Tiefeng Zhang[1,2], Geng Sun[1,2], Weicong Liu[1,2], Longhui Yin[1,2], Xinqiu He[1,2], Rui You[1,2], Chu Wang[1,2], Zhencheng Liu[1,2], Zhijian Liu[1,2], Jin'an Wang[1,2], Xiangqian Jin[1,2*] and Zengquan He[1,2*]

Abstract

Background Whole genome sequencing (WGS) is becoming increasingly prevalent for molecular diagnosis, staging and prognosis because of its declining costs and the ability to detect nearly all genes associated with a patient's disease. The currently widely accepted variant calling pipeline, GATK, is limited in terms of its computational speed and efficiency, which cannot meet the growing analysis needs.

Results Here, we propose a fast and accurate DNASeq variant calling workflow that is purely composed of tools from LUSH toolkit. The precision and recall measurements indicate that both the LUSH and GATK pipelines exhibit high levels of consistency, with precision and recall rates exceeding 99% on the 30x NA12878 dataset. In terms of processing speed, the LUSH pipeline outperforms the GATK pipeline, completing 30x WGS data analysis in just 1.6 h, which is approximately 17 times faster than GATK. Notably, the LUSH_HC tool completes the processing from BAM to VCF in just 12 min, which is around 76 times faster than GATK.

Conclusion These findings suggest that the LUSH pipeline is a highly promising alternative to the GATK pipeline for WGS data analysis, with the potential to significantly improve bedside analysis of acutely ill patients, large-scale cohort data analysis, and high-throughput variant calling in crop breeding programs. Furthermore, the LUSH pipeline is highly scalable and easily deployable, allowing it to be readily applied to various scenarios such as clinical diagnosis and genomic research.

Keywords LUSH, GATK, Whole genome sequencing, DNASeq, Variant calling

Introduction

With advances in sequencing technology and lower sequencing costs, whole-genome sequencing (WGS) is playing an increasingly important role in single-gene disease screening or diagnosis, individualized cancer therapy, and pharmacogenomic screening [1, 7, 8, 14, 17]. Because it allows for the rapid, simultaneous detection of virtually all genes in a patient that may be associated with disease, which is particularly effective for patients with very rare or novel diseases, atypical clinical presentations, or prognostic responses [2, 22, 24, 26]. However, the large volume of WGS sequencing data presents new challenges in terms of analysis time and accuracy. Delayed clinical decisions may lead to severe morbidity or mortality, especially in acutely ill patients with potentially treatable genetic disorders [23, 25]. Therefore, rapid and efficient WGS analysis tools or pipelines are essential

†Taifu Wang, Youjin Zhang and Haoling Wang have contributed equally to this work and shared the co-first authorship.

*Correspondence:
Xiangqian Jin
jinxiangqian@genomics.cn
Zengquan He
hezengquan@genomics.cn
[1] BGI Genomics, Shenzhen 518083, China
[2] Clin Lab, BGI Genomics, Shenzhen 518083, China

for timely molecular diagnosis, staging and prognosis, and pharmacogenomics-based guidance.

Currently, the GATK best practices proposed by Broad Institute are widely accepted standards for WGS variant calling pipeline [15]. It usually consists of several steps: preprocessing, alignment to genome, sort alignments, mark duplicates, base quality score recalibration, and variant calling, each corresponding to a recommended tool. However, this process requires tens of hours to perform analysis on a single set of WGS data [9], which cannot meet the current demand for urgent medical detection of patients with tumors or severe genetic diseases. To solve this problem, several ultrafast WGS analysis tools have been developed, such as Genalice [19] and Isaac [20], but the accuracy of these algorithms are not widely recognized or validated as they do not follow GATK best practices. Although sentieon DNASeq claims to follow the GATK algorithm, it requires a license to use it [10], which restricts its widespread application. Recently, some WGS pipelines based on heterogeneous computing have been proposed. The representative tools are Dynamic Read Analysis for Genomics (DRAGEN) [16] and NVIDIA Parabricks [18], which respectively adopt highly configurable field-programmable gate array (FPGA) and graphics processing unit (GPU) hardware technologies to significantly accelerate computationally intensive genomic analysis processes. However, the utilization of such tools is often constrained by the need for specific, costly hardware, such as GPUs or FPGAs, which may limit their versatility and increase the overall cost of implementation. Therefore, there is still a need to develop more tools that are fast, accurate, economical, and easy to access and deploy.

In this paper, we develop a novel, fast and accurate pipeline for DNASeq variant calling, consisting of multiple LUSH components. The LUSH pipeline reconstructs analysis tools SOAPnuke [4], BWA [13] and GATK [15] using C/C++, and employs a new parallel computing architecture. The primary focus of engineering optimization in LUSH encompasses the elimination of superfluous I/O operations, utilization of thread pools and memory pools for efficient task and memory allocation, and attainment of task load equilibrium. We confirm that the LUSH workflow presents a compelling substitute for GATK best practices as it demonstrates commensurate levels of accuracy while exhibiting substantial superiority over GATK in computational speed.

Materials and methods

WGS benchmarking datasets

NA12878 (HG001) WGS data

Raw paired-end FASTQ files of NA12878 were downloaded from NIST's Genome in a Bottle (GIAB) project at https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/NA12878/. Then, 20X, 30X, 40X, 60X, 80X,100X data sets are obtained by down-sampling the original WGS data set under a series of gradient coverage. The gold standard truth variant calls and high confidence genomic intervals (NIST v3.3.2) were downloaded from https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/release/NA12878_HG001/NISTv3.3.2/.

"CHM-synthetic-diploid" WGS data

CHM-synthetic-diploid was constructed from the PacBio assemblies of two independent CHM cell lines using procedures largely orthogonal to the methodology used for short-read variant calling, which makes it more comprehensive and less biased in comparison to existing benchmark datasets [12]. Paired-end FASTQ files were downloaded from the European Nucleotide Archive with accession number ERR1341793 (https://www.ebi.ac.uk/ena/browser/view/ERR1341793). The benchmark truth call-sets and high-confidence regions of CHM-Synthetic-diploid were downloaded were included in the CHM-eval kit [12] and available at https://github.com/lh3/CHM-eval.

Two trios WGS data

This data set includes two son/father/mother trios of Ashkenazi Jewish (HG002/NA24385, HG003/NA24149, HG004/NA24143) and Han Chinese ancestry (HG005/NA24631, HG006/NA24694, HG007/NA24695) from the Personal Genome Project. Raw paired-end FASTQ files were downloaded from NIST GIAB repositories at https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/ and https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/ChineseTrio/. The truth call-sets and high-confidence regions used for benchmark were obtained from https://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/release/ with the latest version.

Implementation of LUSH pipeline and GATK pipeline

The GATK pipeline was built according to best practices from https://gatk.broadinstitute.org/hc/en-us/sections/360007226651-Best-Practices-Workflows. Since most raw sequencing data require preprocessing operations to obtain clean data, such as removing adapters, low quality sequences and high N-base sequences, we include the SOAPnuke tool in the first step to preprocess the data, although this is not emphasized in GATK best practices.

GATK features an open-source Spark implementation, which serves as the software for executing multi-threaded tasks and represents a form of parallelization that allows computers (or clusters) to complete tasks more rapidly. It is currently in the testing phase and has been marked as unsafe for practical use cases. We have also included it in our tests for performance comparison, encompassing MarkDuplicatesSpark, BaseRecalibratorSpark, ApplyBQSRSpark, and HaplotypeCallerSpark. During practical application, we found GATK HaplotypeCallerSpark was unable to finish processing certain samples, even with the latest GATK version. We found related issues on the GATK official website (https://github.com/broadinstitute/gatk/issues/7199), with no current solution available. As a result, some GATK HaplotypeCallerSpark results in the main text are presently unavailable, and we have denoted these instances with "N/A" in our findings. Additionally, as there is no Spark implementation for GATK GenotypeGVCFs, we have also represented these results with "N/A".

The LUSH pipeline followed a similar procedure as described for GATK best practices, including preprocessing, alignment to genome, sorting alignments, marking duplicates, base quality score recalibration and variant calling, consists of three LUSH components: LUSH_Aligner, LUSH_BQSR and LUSH_HC (Fig. 1, Table 1).

The fundamental architecture of LUSH toolkit

LUSH_Aligner incorporates multiple functional modules such as SOAPnuke, Bwa MEM, Samtools sort, and GATK-MarkDuplicates (Picard) while being entirely redeveloped based on the original algorithm. The fundamental architecture is depicted in Supplemental Fig. S1. LUSH_Aligner is composed of three primary functional modules: 'FqFilterComponent', 'bwaMEMComponent', and 'SortDuplicateComponent', which manage filtering, alignment, as well as sorting-alignments and marking-duplicates tasks, respectively. Each functional module encompasses several workers to concurrently perform the associated operations, while distinct modules communicate via the intermediary conveyor belt system. Specifically, the raw sequencing reads are first loaded into the FqFilterComponent module through the FastQReader. Within this module, a cohort of FqfilterWorker threads executes filtering operations on the reads. Concurrently, the quality-controlled reads are stored in the CleanFq file while also being transmitted to the downstream 'bwaMEMComponent' module. Herein, the bwaMEMworker performs parallel alignment of the reads, which subsequently get transmitted to the 'SortDuplicateComponent'. Within this component, the SortDuplicateWorkers undertake the crucial tasks of sorting-alignments and marking-duplicates, culminating in the generation of a sorted and deduplicated bam file. The entire process adheres to the paradigm of pipeline computing (Supplemental Fig. S1). This design significantly enhances CPU utilization and diminishes redundant IO consumption, ultimately leading to an increased processing speed.

The base quality scores generated by sequencing machines are influenced by various systematic (non-random) technical errors, leading to overestimation or underestimation of the quality scores in the data. The purpose of Base Quality Score Recalibration (BQSR) is to empirically model these errors using machine learning and adjust the quality scores accordingly. It mainly consists of two steps: Base Recalibration and applying BQSR. LUSH_BQSR implements a producer–consumer parallel computing structure to optimize task parallelism and improve CPU utilization (Supplemental Fig. S2). In the Base Recalibration phase, the producer threads read the input BAM file for data distribution, and the distributed tasks enter the queue for processing by consumer threads, which involve data processing and computational activities (such as calculating different covariates and max posterior probability). Once all the tasks are completed, the recal-table is printed, and then the downstream apply BQSR phase is entered. LUSH_BQSR applies numerical corrections to each individual base based on the Base Recalibration table in the memory pool. Different consumer threads perform different batches of reads base quality correction tasks. Finally, the sorted results are collected and outputted to the final BAM file (Supplemental Fig. S2). The number of threads engaged in the producer or consumer roles can be controlled through external parameters, making it possible to adapt to different machine configurations and improve overall thread utilization.

HaplotypeCaller employs a local de-novo assembly approach within active regions for accurate detection of single nucleotide polymorphisms (SNPs) and insertion-deletion (indel). This process involves four sequential steps: identification of active regions, local assembly of active regions to infer haplotypes, estimation of likelihood values utilizing the Pair Hidden Markov Model (HMM), and determination of genotypes based on Bayesian inference. To facilitate efficient task distribution and scheduling, we have implemented a dedicated dispatcher for each step within the LUSH_HC architecture (Supplemental Fig. S3). Prior to entering the ActiveRegionDispatcher, read alignments from the Binary Alignment Map (BAM) file undergoes filtering and downsampling. ActiveRegionDispatcher further subdivides tasks based on chromosome coordinates, enabling the calculation of likelihood values for each specific candidate interval. Completed tasks are then merged in ActiveRegionReduce to determine the ActiveRegions. The resulting ActiveRegions are subsequently dispatched to the AssembleDispatcher, where tasks are queued and executed. Upon completion, the processed data proceeds to

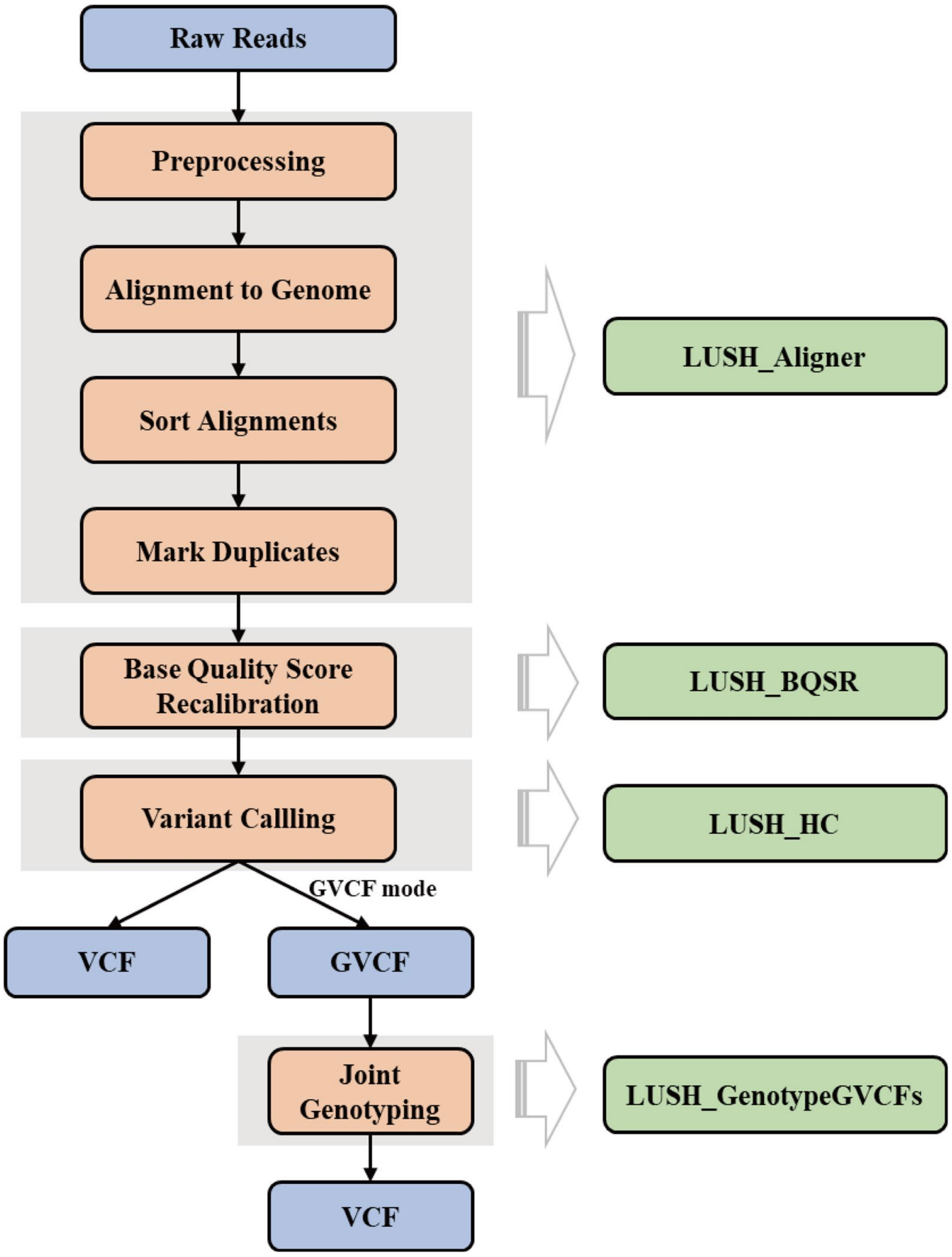


Fig. 1 Overview of LUSH variant calling workflow. The LUSH DNASeq workflow is an optimized pipeline based on GATK best practices and consists of LUSH_Aligner, LUSH_BQSR, LUSH_HC, and LUSH_GenotypeGVCFs

downstream stages, including the PairHMMDispatcher and GenotypeDispatcher. Each dispatcher promptly submits upstream tasks to the thread pool, and dynamically allocates computing resources (i.e., threads) based on the computing load of each task. Once the task is finalized, the computing resources are released back to the thread pool and marked as available for other tasks to use (Supplemental Fig. S3). Furthermore, certain active region intervals may contain a substantial number of candidate haplotypes, leading to a significant computational burden during PairHMM calculations and uneven task distribution among threads, thereby resulting in inefficient

Table 1 The composition of LUSH pipeline and GATK pipeline

Pipeline step	GATK/GATK-spark pipeline	LUSH pipeline
Preprocessing	SOAPnuke	LUSH_Aligner
Alignment to genome	Bwa mem	
Sort alignments	Samtools sort	
Mark duplicates	MarkDuplicates/MarkDuplicatesSpark	
Base quality score recalibration	BaseRecalibrator & ApplyBQSR/BaseRecalibratorSpark & Apply-BQSRSpark	LUSH_BQSR
Variant calling	HaplotypeCaller/HaplotypeCallerSpark	LUSH_HC
Joint genotyping	GenotypeGVCFs	LUSH_GenotypeGVCFs

CPU utilization. To overcome this challenge, LUSH_HC employs a further task subdivision strategy combined with dynamic resource allocation in these regions to achieve load balance and optimize resource allocation (Supplemental Fig. S3). This comprehensive approach effectively maximizes the utilization of system resources, ultimately enhancing the overall performance and efficiency of the system.

Methodology evaluation

For NA12878 and Two trios WGS data sets, we used the haplotype comparison tool hap.py (v0.3.14, default comparison engine) for the comparison of diploid genotypes at the haplotype level to calculate the performance metrics. The variant calling accuracy of CHM-Synthetic-diploid WGS dataset was evaluated using RTG in CHM-evalkit (Version 20180222) [12]. The definitions of true positive (TP), false positive (FP), and false negative (FN) were based on the types of variant matching stringencies "genotype match", and Precision, Recall, and F1-score were calculated as TP/(TP + FP), TP/(TP + FN) and 2*TP/(2*TP + FN + FP), respectively. Tools-specific SNPs and INDELs were annotated using the SNPEFF (v4.3) [5] with default parameters.

We evaluated the computational performance of all the tools on the same Linux machine, measuring both total runtime and maximum memory consumption using the "/usr/bin/time" command. All analyses were run on a Linux machine featuring an Intel(R) Xeon(R) Gold 6348 CPU 56-core processor and 500 GB memory and were performed on a shared storage disk Dell EMC Isilon H500.

Results

Overview of LUSH DNAseq workflow

The LUSH DNASeq workflow is an optimized pipeline based on GATK best practices. Its main components are LUSH_Aligner, LUSH_BQSR, LUSH_HC, and LUSH_GenotypeGVCFs (Fig. 1, Table 1).

LUSH_Aligner is a comprehensive computational framework that seamlessly integrates four distinct functional modules: preprocessing, alignment to the reference genome, sort alignments, and mark duplicates. These modules correspond to well-established software tools: SOAPnuke, Bwa mem, Samtools sort [6] and GATK-MarkDuplicates ([6], Picard), achieving seamless communication through an intermediate transmission system (Supplemental Fig. S1, details see Methods). LUSH_BQSR utilizes a parallel computing architecture with a producer–consumer pattern to implement base quality score recalibration (Supplemental Fig. S2, see Methods). LUSH_HC involves four sequential steps: identification of active regions, assembly of active regions to infer haplotypes, estimation of likelihood values utilizing Pair Hidden Markov Model (PairHMM), and determination of genotypes based on Bayesian inference. To facilitate efficient task distribution and scheduling, we have implemented a dedicated dispatcher for each step within the LUSH_HC architecture (Supplemental Fig. S3, see Methods for more details). Moreover, LUSH_HC also implements the GVCF mode algorithm to meet the demand of GVCF files in cohort studies. Correspondingly, the C/C++ re-implementation of LUSH_GenotypeGVCFs is used to perform joint genotyping on one or more samples.

Computational performance on different threads and sequencing depth

The computational performance of the pipelines may not necessarily improve with the increase of the number of cores used. Application performance can be limited due to multiple bottlenecks including contention for shared resources such as caches and memory. Thus, we specified 12, 24, 36, 48 and 56 (max) threads at a single node to test the single-node scalability of the LUSH pipeline. As shown in Fig. 2, the runtime of all LUSH tools in this pipeline decreases significantly as the threads are increased. The pipeline completed in ~ 4.89 h when running at 12 threads and ~ 1.6 h when running at 56 threads

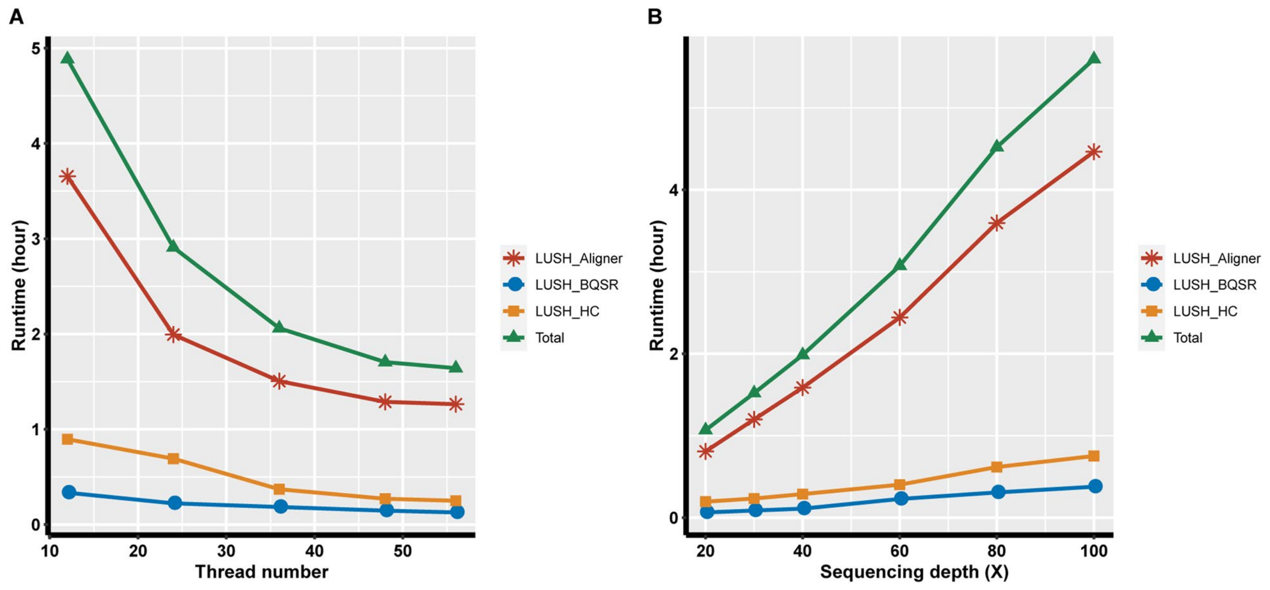


Fig. 2 Computational performance of the LUSH pipeline at different threads and sequencing depth on NA12878. **A** Running time with 12, 24, 36, 48 and 56 threads on NA12878 (30X). **B** Running time at sequencing depth 20X, 30X, 40X, 60X, 80X, 100X. Each data point is the average of two replicates

(Fig. 2A), indicating that the LUSH pipeline has great thread scalability.

We then generated NA12878 WGS datasets with 20X, 40X, 60X, 80X and 100X coverage depths by down-sampling. Each dataset was performed with the LUSH DNASeq pipeline to investigate the effect of sequencing depth on performance. Each task was run on the maximum available cores (56). The runtime of all LUSH tools and the entire pipeline increases almost linearly with increasing sequencing depth (Fig. 2B).

Speed of LUSH pipeline relative to GATK pipeline

To compare the performance of each component of the LUSH pipeline with that of the GATK and GATK-Spark pipeline, we analyzed the NA12878 30X sample on a maximum 56-core machine. Each software thread parameter was adjusted to the maximum available. LUSH_Aligner integrates four functional modules for the FASTQ to BAM process, including pre-processing, alignment to the genome, sorting alignments, and marking duplicates. LUSH_Aligner completed FASTQ to BAM on 30x NA12878 dataset in less than 1.3 h, which is more than 5 times faster than the 6.88 h of GATK pipeline (Fig. 3A). Despite the acceleration of the marking-duplicates step by MarkDuplicatesSpark, the improvement in performance was not substantial. LUSH_BQSR integrates BaseRecalibrator and ApplyBQSR of GATK pipeline to greatly improve thread utilization. On the 30x NA12878 dataset, LUSH_BQSR demonstrated remarkable efficiency by completing the task in approximately 5 min. This represents a 60-fold increase in performance compared to the 5.22 h required by the GATK pipeline and a 4.5-fold improvement over the 0.39 h taken by the GATK-Spark pipeline (Fig. 3B). To produce a VCF from a BAM file, GATK-HaplotypeCaller was widely recognized as the most time-consuming step in the GATK best-practice pipeline. It took ~ 15 h to complete the 30x NA12878 dataset, while LUSH_HC took only about 12 min (Fig. 3C). During the application of GATK-Spark in VCF-mode, a prevalent issue was observed for which no resolution is presently available (see methods). This finding implies that GATK-Spark may not be sufficiently mature for implementation in production environments. The GVCF mode was commonly used in the cohort-wide analysis, which can then be used for joint genotyping of multiple samples in a very efficient way. This enables rapid incremental processing of samples as they roll off the sequencer, as well as scaling to a very large cohort size. Thus, we also implanted GVCF mode in LUSH_HC. In GVCF mode, LUSH_HC used only 0.28 h, while the GATK pipeline took 18.35 h to process the same dataset (Fig. 3D). GATK-Spark required 0.76 h to complete the task, demonstrating a performance approximately 2.7 times less efficient than that of LUSH. Regarding the performance of single sample joint genotyping, LUSH_GenotypeGVCFs (0.21 h) was 5X faster than GATK-GenotypeGVCFs (1.13 h) (Fig. 3E).

For the whole pipeline from FASTQ to VCF, the LUSH pipeline greatly reduced the runtime in both non-GVCF and GVCF modes, taking less than 2 h for 30X WGS

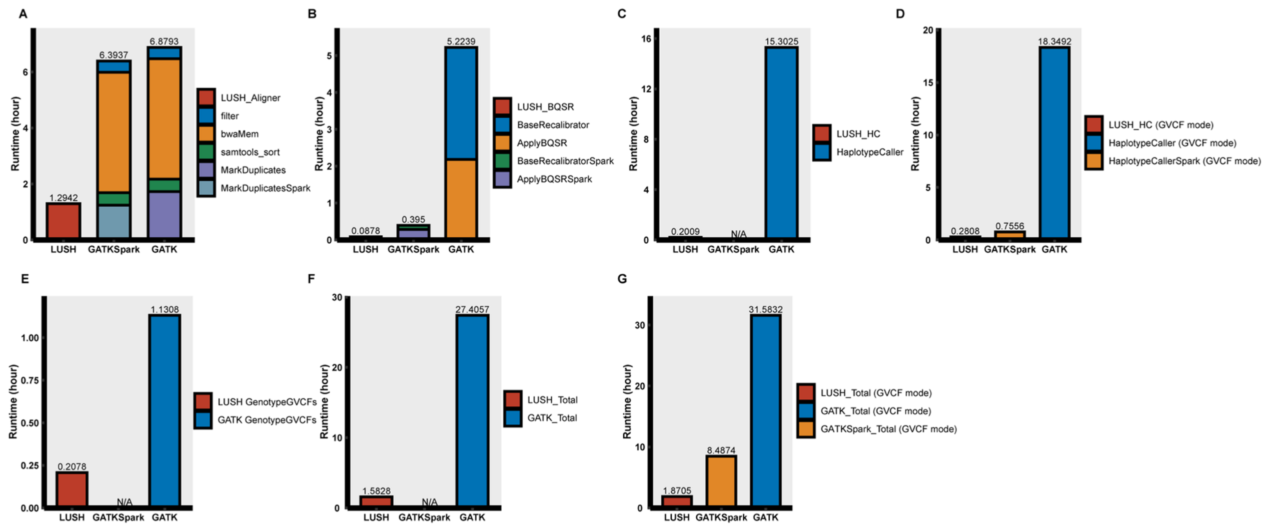


Fig. 3 Runtime of LUSH, GATK and GATK-Spark Variant Calling pipelines on NA12878. **A** Runtime of each component from FASTQ to BAM. **B** Runtime of base quality score recalibration. **C** Runtime of variant calling in non-GVCF mode. **D** Runtime of variant calling in GVCF mode. **E** Runtime of joint genotyping on one sample. **F** Total elapsed time in non-GVCF mode. **G** Total elapsed time in GVCF mode. N/A: Not applicable

data, about 17 times faster than the GATK pipeline and 4.5 times faster than the GATK-Spark pipeline (Fig. 3F,G). Likewise, the LUSH pipeline had a similar performance on CHM-Synthetic-diploid (Supplemental Fig. S5) and two-trios WGS datasets (Supplemental Table S2).

Variant calling accuracy of the LUSH pipeline

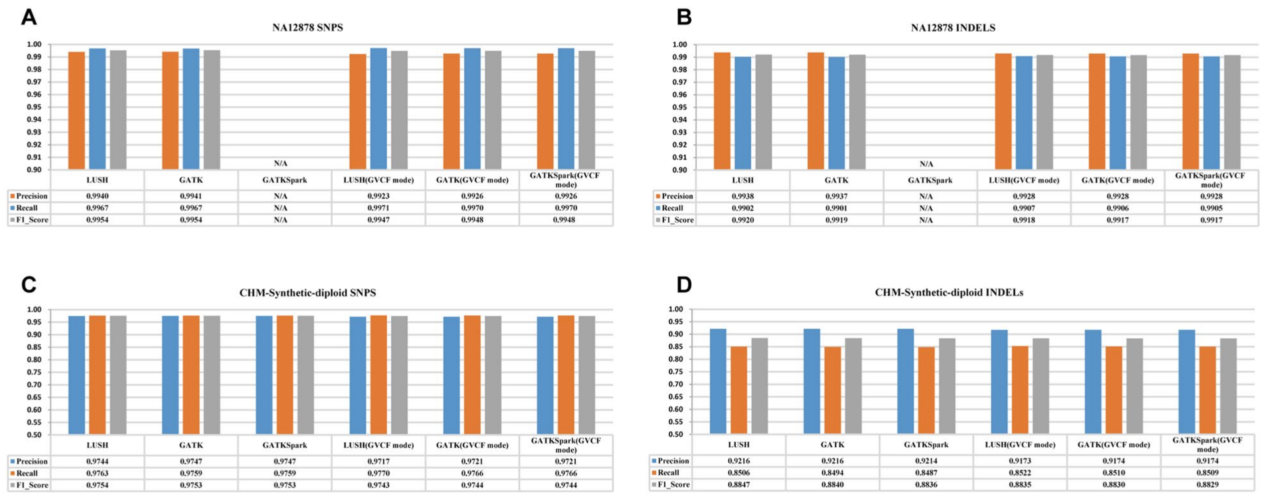
We then compared the accuracy of the LUSH pipeline with that of the GATK and GATK-Spark pipeline. The underlying algorithm of LUSH is roughly the same as that of GATK or GATK-Spark, so they were expected to produce identical results. We ran each of the three datasets mentioned above using two pipelines. The generated VCFs were compared with their respective truth sets using the haplotype comparison tool hap.py or CHM-evalkit. The comparison was limited to the high-confidence regions of each dataset (see Methods). Due to the common use of GVCF mode in cohort studies, we also add GVCF mode to the comparison. As expected, LUSH, GATK and GATK-Spark demonstrated almost the same precision, recall and F1-score on the NA12878 (Fig. 4A, B) and CHM-Synthetic-diploid datasets (Fig. 4C, D), both for the SNP and INDEL. Interestingly, the comparison of the results obtained from the non-GVCF mode and the GVCF mode for both the LUSH and GATK pipelines showed that the former exhibited higher precision and slightly lower recall. Specifically, the F1 scores revealed that the non-GVCF mode exhibited better performance in terms of accuracy for both pipelines (Fig. 4A–D). The findings are also in full agreement with the Two trios WGS data (Supplemental Table S3).

We then analyzed the intersection of the LUSH pipeline and the GATK pipeline. All variants detected by both pipelines were used for the analysis. As expected, approximately 99.11–99.14% of SNPs and 98.92–99.08% of INDELs were co-reported by the two pipelines, both in the non-GVCF mode (Fig. 5A, B) and GVCF mode (Fig. 5C, D), indicating a high consistency of the LUSH and GATK pipelines. Among these LUSH-only and GATK-only variants, the observed TP rates for SNP were 1.54% and 0.74%, respectively. For INDEL, the TP rates were 5.33% and 2.51%, respectively (Fig. 5A, B). The TP rates in GVCF mode were consistent with these results (Fig. 5C, D). We then annotated genomic regions for LUSH or GATK-specific SNPs and INDELs using SNPEFF. Among these few pipeline-specific variants, LUSH-only and GATK-only variants showed consistent distribution across the genome, and more than 97% of the variants were located in non-functional regions (sum of INTERGENIC, INTRON, DOWNSTREAM, UPSTREAM).

Moreover, we also explored the intersection of variants in non-GVCF mode and GVCF mode. The results were highly consistent in both modes, with 99.13% (GATK) and 99.25% (LUSH) of co-detected SNPs and 99.11% (GATK) and 99.18% (LUSH) of co-detected INDELs (Supplemental Fig. S6). GVCF mode significantly detected more specific SNPs and INDELs.

Computer resource utilization of the LUSH toolkit

We have recorded the average CPU load and maximum memory consumption for each LUSH component during



A — NA12878 SNPS

	LUSH	GATK	GATKSpark	LUSH(GVCF mode)	GATK(GVCF mode)	GATKSpark(GVCF mode)
Precision	0.9940	0.9941	N/A	0.9923	0.9926	0.9926
Recall	0.9967	0.9967	N/A	0.9971	0.9970	0.9970
F1_Score	0.9954	0.9954	N/A	0.9947	0.9948	0.9948

B — NA12878 INDELS

	LUSH	GATK	GATKSpark	LUSH(GVCF mode)	GATK(GVCF mode)	GATKSpark(GVCF mode)
Precision	0.9938	0.9937	N/A	0.9928	0.9928	0.9928
Recall	0.9902	0.9901	N/A	0.9907	0.9906	0.9905
F1_Score	0.9920	0.9919	N/A	0.9918	0.9917	0.9917

C — CHM-Synthetic-diploid SNPS

	LUSH	GATK	GATKSpark	LUSH(GVCF mode)	GATK(GVCF mode)	GATKSpark(GVCF mode)
Precision	0.9744	0.9747	0.9747	0.9717	0.9721	0.9721
Recall	0.9763	0.9759	0.9759	0.9770	0.9766	0.9766
F1_Score	0.9754	0.9753	0.9753	0.9743	0.9744	0.9744

D — CHM-Synthetic-diploid INDELs

	LUSH	GATK	GATKSpark	LUSH(GVCF mode)	GATK(GVCF mode)	GATKSpark(GVCF mode)
Precision	0.9216	0.9216	0.9214	0.9173	0.9174	0.9174
Recall	0.8506	0.8494	0.8487	0.8522	0.8510	0.8509
F1_Score	0.8847	0.8840	0.8836	0.8835	0.8830	0.8829

Fig. 4 Accuracy of LUSH, GATK and GATK-Spark Variant Calling pipelines. Based on two benchmark datasets, the precision and recall, as well as the F1 scores, were utilized to measure the accuracy of detected SNPs (Left panel) and INDELs (Right panel). **A**, **B** On NA12878 dataset. **C**, **D** On CHM-synthetic-diploid dataset

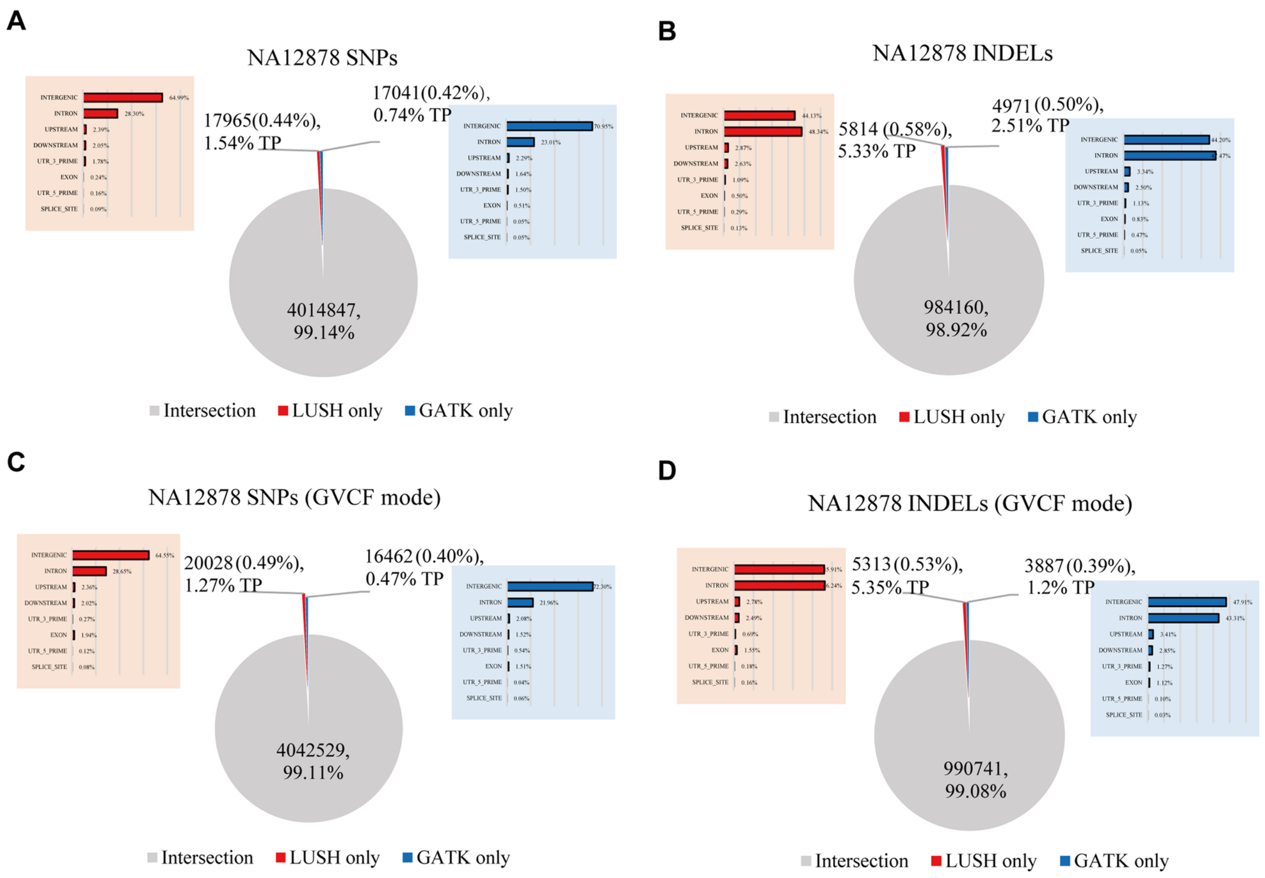


Fig. 5 The intersection of variants called by different pipelines on NA12878. The pie chart shows the overlap of SNPs (left panel) and INDELs (right panel) identified by the LUSH and GATK pipelines in two different modes. The co-detected variants are depicted in grey, while the LUSH and GATK specific variants are shown in red and blue, respectively. Additionally, the adjacent red and blue bars illustrate the genomic distribution of LUSH-only and GATK-only variants. The modes of analysis are categorized as **A**, **B** non-GVCF mode. **C**, **D** GVCF mode

the examination of the NA12878 dataset at 30x coverage, and juxtaposed these findings with the tools of the GATK and GATK-Spark workflows. Consistent with prior analyses, the LUSH, GATK, and GATK-Spark workflows were executed on a 56-core machine, with the parameters of each tool configured to the maximum of 56 threads, where applicable. Our findings reveal that the CPU utilization of the LUSH workflow components is notably elevated, akin to that of GATK-Spark, while GATK, with the exception of the preceding alignment, sorting, and marking-duplicates stages, predominantly employs 1–2 threads (Supplemental Table. S4). Regarding memory usage, the peak memory consumption of the LUSH workflow reaches 42G during the LUSH_Aligner step, slightly lower for other stages, akin to GATK-Spark, and surpassing that of GATK, which hovers around 10G.

Discussion

Genome sequencing has been commonly used for molecular diagnosis, staging, and prognosis, and the massive sequencing data presents a challenge in terms of analysis time. We have developed a LUSH pipeline consisting of LUSH toolkit enabling rapid and precise results. It takes only 1.6 h to process 30X WGS data from FASTQ to VCF and ~ 12 min from BAM to VCF, with accuracy comparable to GATK, which is extremely critical for acutely ill patients, such as infants in the Pediatric Intensive Care Unit (PICU) and Neonatal Intensive Care Unit (NICU).

We tested the performance of the LUSH pipeline at different thread scales and showed that the LUSH pipeline has remarkable thread scalability. The LUSH component is based on a parallel computing architecture of producer and consumer, which allows it to achieve optimal performance on any machine with a reasonable configuration of parameters. The LUSH pipeline also scales well linearly at different sequencing depths.

LUSH is based on the original WGS "best practices" with a C/C++ implementation that follows the underlying algorithms of the original pipeline. In terms of speed, the optimization of the underlying language, multithreaded architecture, and algorithmic framework gives the LUSH pipeline an absolute advantage over GATK. Each step in the LUSH pipeline is at least 5 times faster for the same work, and the step LUSH_HC even reaches a speed increase of 76 times. In terms of accuracy, the results of the LUSH pipeline and GATK are equally accurate and highly consistent. The annotation results for specific variants show no meaningful differences in reliability between them. We also demonstrated both high accuracy and hyper speed for the LUSH pipeline on multiple datasets of standards. Moreover, it is crucial to highlight that LUSH can effectively accelerate the analysis of all types of DNAseq data (such as WGS, WES, PANEL, etc.), even though the present study's focus on comparing LUSH's performance solely in the context of WGS data, as WGS has the extensive volume and prolonged analysis duration. The BAM files generated by LUSH can also be used for subsequent detection of structural variants (SV) and copy number variations (CNV), serving as input for common SV callers such as Manta [3], Delly [21], and Lumpy [11] etc., because it follows the original BWA algorithm.

Conclusions

In summary, the LUSH pipeline displays considerable potential as a viable alternative to the GATK pipeline for DNASeq Variant Calling, with the added advantage of being easily deployable. Consequently, its implementation is expected to enhance the bedside analysis of critically ill patients, facilitate analysis of large-scale cohort data, and expedite high-throughput variant calling in crop breeding programs.

Abbreviations

WGS	Whole genome sequencing
DRAGEN	Dynamic read analysis for genomics
FPGA	Field-programmable gate array
GPU	Graphics processing unit
GIAB	Genome in a bottle
SNPs	Single nucleotide polymorphisms
Indel	Insertion and deletion
BQSR	Base quality score recalibration
HMM	Hidden Markov model
TP	True positive
FP	False positive
FN	False negative
PICU	Pediatric intensive care unit
NICU	Neonatal intensive care unit

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40246-024-00666-w.

Additional file 1.

Additional file 2.

Additional file 3.

Additional file 4.

Additional file 5.

Acknowledgements
The authors are grateful to the people in their research group for support and valuable suggestions.

Author contributions
TW, YZ and HW devised the study, analyzed the data, interpreted the results and drafted the manuscript. TW, JY and QZ conducted the experiments and revised the manuscript. YZ, HW, QZ, TZ, GS, WL, LY, XH and RY implemented the algorithm. CW, ZL and ZL performed data analyses. JW provided critical intellectual comments. XJ and ZH supervised the study and reviewed the manuscript. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials

All data generated or analyzed during this study are included in this published article and its supplementary information files. Benchmarking scripts and commands for LUSH pipeline are available at https://github.com/Bgi-LUSH/LUSH-DNASeq-pipeline.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 19 November 2023 Accepted: 22 August 2024
Published online: 10 October 2024

References

1. Ashley EA, et al. Clinical assessment incorporating a personal genome. Lancet. 2010;375(9725):1525–35.
2. Bick D, et al. Case for genome sequencing in infants and children with rare, undiagnosed or genetic diseases. J Med Genet. 2019;56(12):783–91.
3. Chen X, et al. Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. Bioinformatics. 2016;32(8):1220–2.
4. Chen Y, et al. SOAPnuke: a MapReduce acceleration-supported software for integrated quality control and preprocessing of high-throughput sequencing data. Gigascience. 2018;7(1):1–6.
5. Cingolani P, et al. A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. Fly (Austin). 2012;6(2):80–92.
6. Danecek P, et al. Twelve years of SAMtools and BCFtools. Gigascience. 2021;10(2):giab008.
7. Dewey FE, et al. Clinical interpretation and implications of whole-genome sequencing. JAMA. 2014;311(10):1035–45.
8. Green ED, Guyer MS, National Human Genome Research, I. Charting a course for genomic medicine from base pairs to bedside. Nature. 2011;470(7333):204–13.
9. Heldenbrand JR, et al. Performance benchmarking of GATK3. 8 and GATK4. BioRxiv. 2018;11:348565.
10. Kendig KI, et al. Sentieon DNASeq variant calling workflow demonstrates strong computational performance and accuracy. Front Genet. 2019;10:736.
11. Layer RM, et al. LUMPY: a probabilistic framework for structural variant discovery. Genome Biol. 2014;15(6):R84.
12. Li H, et al. A synthetic-diploid benchmark for accurate variant-calling evaluation. Nat Methods. 2018;15(8):595–7.
13. Li H, Durbin R. Fast and accurate short read alignment with Burrows–Wheeler transform. Bioinformatics. 2009;25(14):1754–60.
14. Lupski JR, et al. Whole-genome sequencing in a patient with Charcot–Marie–Tooth neuropathy. N Engl J Med. 2010;362(13):1181–91.
15. McKenna A, et al. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res. 2010;20(9):1297–303.
16. Miller NA, et al. A 26-hour system of highly sensitive whole genome sequencing for emergency management of genetic diseases. Genome Med. 2015;7:100.
17. Nakagawa H, Fujita M. Whole genome sequencing analysis for cancer genomics and precision medicine. Cancer Sci. 2018;109(3):513–22.
18. O'Connell KA, et al. Accelerating genomic workflows using NVIDIA Parabricks. BMC Bioinform. 2023;24(1):221.
19. Pluss M, et al. Need for speed in accurate whole-genome data analysis: GENALICE MAP challenges BWA/GATK more than PEMapper/PECaller and Isaac. Proc Natl Acad Sci U S A. 2017;114(40):E8320–2.
20. Raczy C, et al. Isaac: ultra-fast whole-genome secondary analysis on Illumina sequencing platforms. Bioinformatics. 2013;29(16):2041–3.
21. Rausch T, et al. DELLY: structural variant discovery by integrated paired-end and split-read analysis. Bioinformatics. 2012;28(18):i333–9.
22. Sanford E, et al. Clinical utility of ultra-rapid whole-genome sequencing in an infant with atypical presentation of WT1-associated nephrotic syndrome type 4. Cold Spring Harb Mol Case Stud. 2020;6(4):a005470.
23. Saunders CJ, et al. Rapid whole-genome sequencing for genetic disease diagnosis in neonatal intensive care units. Sci Transl Med. 2012;4(154):154ra135.
24. Shigemizu D, et al. Whole-genome sequencing reveals novel ethnicity-specific rare variants associated with Alzheimer's disease. Mol Psychiatry. 2022;27(5):2554–62.
25. Willig LK, et al. Whole-genome sequencing for identification of Mendelian disorders in critically ill infants: a retrospective analysis of diagnostic and clinical findings. Lancet Respir Med. 2015;3(5):377–87.
26. Xing R, et al. Whole-genome sequencing reveals novel tandem-duplication hotspots and a prognostic mutational signature in gastric cancer. Nat Commun. 2019;10(1):2037.