Scalable and efficient DNA sequencing analysis on different compute infrastructures aiding variant discovery

Friederike Hanssen[1,2,3], Maxime U. Garcia[4,5,6], Lasse Folkersen[7], Anders Sune Pedersen[8], Francesco Lescai[9], Susanne Jodoin[1,3], Edmund Miller[10], Oskar Wacker[1,3], Nicholas Smith[11], nf-core community, Gisela Gabernet[1,12], Sven Nahnsen[1,2,3*]

[1]Quantitative Biology Center, University of Tübingen, Otfried-Müler Str. 37, Tübingen, 72076, Baden-Württemberg, Germany.
[2]Department of Computer Science, University of Tübingen, , , 72076, Baden-Württemberg, Germany.
[3]M3 Research Center, University Hospital, Otfried-Müler Str. 37, Tübingen, 72076, Baden-Württemberg, Germany.
[4]Seqera Labs, Carrer de Marià Aguiló, 28, Barcelona, 08005, Spain.
[5]Barntumörbanken, Department of Oncology-Pathology, Karolinska Institutet, BioClinicum, Visionsgatan 4, Solna, 17164, Sweden.
[6]National Genomics Infrastructure, SciLifeLab, SciLifeLab, Tomtebodavägen 23, Solna, 17165, Sweden.
[7]Nucleus Genomics, 584 Broadway, New York, 10012, NY, USA.
[8]National Genome Center Denmark, Ørestads Boulevard 5, Copenhagen, 2300, Denmark.
[9]Department of Biology and Biotechnology "L. Spallanzani", University of Pavia, via Ferrata, 9, Pavia, 27100, PV, Italy.
[10]Department of Biological Sciences and Center for Systems Biology, University of Texas at Dallas, 800 W Campbell Rd, Richardson, 75080, TX, USA.
[11]Department of Informatics, Technical University of Munich, Boltzmannstr. 3, Garching, 85748, Bavaria, Germany.
[12]Department of Pathology, Yale School of Medicine, 300 George, New Haven, 06510, CT, USA.

*Corresponding author(s). E-mail(s): sven.nahnsen@uni-tuebingen.de;
Contributing authors: friederike.hanssen@qbic.uni-tuebingen.de;
maxime.garcia@seqera.io; lasse@mynucleus.com; asp@ngc.dk;
francesco.lescai@unipv.it; susanne.jodoin@qbic.uni-tuebingen.de;
edmund.miller@utdallas.edu; oskar.wacker@qbic.uni-tuebingen.de;
smithnickh@gmail.com; core@nf-co.re; gisela.gabernet@yale.edu;

Abstract

DNA variation analysis has become indispensable in many aspects of modern biomedicine, most prominently in the comparison of normal and tumor samples. Thousands of samples are collected in local sequencing efforts and public databases requiring highly scalable, portable, and automated workflows for streamlined processing. Here, we present nf-core/sarek 3, a well-established, comprehensive variant calling and annotation pipeline for germline and somatic samples. It is suitable for any genome with a known reference. We present a full rewrite of the original pipeline showing a significant reduction of storage requirements by using the CRAM format and runtime by increasing intra-sample parallelization. Both are leading to a 70% cost reduction in commercial clouds enabling users to do large-scale and cross-platform data analysis while keeping costs and CO_2 emissions low. The code is available at https://nf-co.re/sarek.

Keywords: workflows, cloud computing, variant calling, best practices, cost efficiency

1 Introduction

Genomic variation analysis of short-read data has become a key step for modern personalized medicine as well as for fundamental biomedical research. In particular, for biomedical assessment, it is used for characterizing genomes of samples taken from both healthy or tumor tissue. In clinical applications, the resulting information can be used to classify tumors and support treatment decisions [1–3] or research questions, such as drug development [4] or identify variations of interest in larger cohorts for further studies [5, 6]. The technologies and protocols for generating DNA sequencing data vary a lot. Each of the technologies comes with different specialties ranging from targeted gene panels and whole exomes (WES) to whole genomes (WGS) resulting in raw data files from a few to hundreds of gigabytes (GB). Various project-specific factors play a role in choosing the appropriate sequencing technologies, such as the particular type of genomic variations of interest, the cost for sequencing, analysis, and data storage or turn-around times [7]. Panel and exome sequencing is cheaper than WGS [8]. Targeting defined regions allows for having high coverage in these regions. Hence, single nucleotide variants (SNVs) and small insertions and deletions (Indels) can be determined with high confidence. WGS, on the other hand, can be used to additionally investigate more complex alterations such as non-coding variants, large structural variants (SV), and copy-number variations (CNV). Another aspect

is ethical considerations on how to handle 'accidentally detected' genomic variation in non-targeted genes which had been identified during the whole genome or whole exome sequencing [9, 10].

Examples of large-scale genomics collection projects are TCGA/ICGC or the 100,000 Genomes Project. Some 6,800 whole-genome samples from the former were uniformly processed for the 'Pan Cancer Analysis of Whole Genomes' study to obtain a consistent set of somatic mutation calls [11]. More than 12,000 whole-genome samples from the latter were analyzed with respect to their mutational signatures to gain insights into tissue-specific markers [12]. There are several national and international initiatives that aim at gathering more and more sequenced genomes, such as the Estonia Genome Project, the German Human Genome-Phenome Archive, the Iceland Genome Project [13], or the European '1+ Million Genomes' Initiative. Such studies often encompass many patients and their samples are often collected over longer periods of time at multiple sites. This requires stable, multi-infrastructure supported, and reproducible pipelines for distributed and homogeneous data processing [14].

Several pipelines [11, 15–18] have been published in different workflow languages to automatically process reads from FastQ files to called (and annotated) variants accompanied by countless in-house workflows. With a certain variety of tools, the workflows usually encompass: quality control steps, read trimming, mapping, duplicate marking, base quality score recalibration, variant calling, and possibly annotation (see Fig. 1).

								Preprocessing	Quality Control	Variant calling				Annotation
Workflow language	Paired	Tumor-only	Germline	Panel	WES	WGS	Code available?			SNP/Indel	CNV	SV	MSI	
NF	✓	✓	✓	✓	✓	✓	✓	nf-core/sarek		✓	✓	✓	✓	
SM	-	-	✓	✓	✓	✓	✓	OvarFlow		✓	-	-	-	
SM	-	-	✓	-	-	✓	✓	SequanaVC		✓	-	-	-	
SM	-	✓	-	✓	✓	-	✓	TOSCA		✓	✓	-	-	
Roddy	✓	✓	-	-	✓	✓	✓	One-touch-pipelines		✓	✓	✓	-	

Fig. 1 There are many published and countless unpublished variant calling pipelines written in dedicated workflow languages like Nextflow (NF), SnakeMake (SM) [19], and Roddy. All pipelines align the reads and duplicate mark them. nf-core/sarek, Ovarflow, and TOSCA have additional option for base quality score recalibration. All pipelines allow variant calling and annotation. For the One-touch pipelines, separate workflows have to be triggered for each variant calling type with build-in annotation. nf-core/sarek stands out by covering germline, tumor-only, and paired variant calling followed by annotation across whole genome, whole-exome, and panel sequencing data. The code is available online and well-documented, implemented in Nextflow to enable portability to various infrastructures, and supported by an active community.

While there are many workflows available, nf-core/sarek [15] stands out with its ability to process germline, tumor-only, and paired samples in one run. It can perform SNV/Indel, SV, and CNV calling, as well as micro-satellite instability(MSI) analysis of WGS, WES, and panel data with currently 12 different tools. Since the pipeline is written in Nextflow [20], it benefits from the portability to any supported infrastructure, in particular several cloud vendors and common HPC schedulers enabling cross-platform homogeneous data processing. Furthermore, the pipeline allows the processing of non-model organisms. While the reference genomes and databases are most comprehensively provided for human and mouse genomes as well as subsets for many other organisms, they can be generated and saved for future runs for non-supported species. nf-core/sarek is part of the nf-core community project [21] and has a growing user base with now 242 stars on GitHub and 47.47K unique repository visitors since July 2019 (as of 1st June 2023) who additionally contributes with supporting and improving the code base either by direct contributions, suggesting features, or raising issues.

The pipeline has been used within the field of cancer research [22–27] and beyond, such as the identification of rare variants in tinnitus patients [28], finding SNPs in driver genes related to stress-response in cowpeas [29], the genomic profiling of wild and commercial bumble bee populations [30], or the Personal Genome Project-UK [31].

Here, we present a re-implementation of the nf-core/sarek pipeline using the Nextflow DSL2 framework which increases user-based customization to maintain a modern pipeline with a focus on reducing required compute resources for efficient runs on different infrastructures. Minimizing required computing resources has always been of large interest. In particular, in the genomics space, more users run their calculations on several commercial and non-commercial cloud platforms [14]. Commercial platforms usually come with a pay-per-use model, thus there is a high interest to reduce costs due to finite funding. For non-commercial platforms or local clusters, direct costs are possibly of lower interest, however, reducing required resources allows for processing more samples in a shorter time frame. Furthermore, all used tools have been updated to their latest version upon release. For various steps, new tool options have been added, i.e. mapping with DragMap or variant calling with DeepVariant [32], or fastP [33] for adapter trimming.

Using this re-implementation, we show for the first time that population-scale homogeneous recomputing of WGS on commercial clouds is possible. Our results show that when using the nf-core/sarek 3.1.1 pipeline we can reduce compute costs to 34% compared to nf-core/sarek 2.7.2 to approximately 20$ for germline short and structural variant calling and annotation.

2 Results

2.1 Pipeline overview & summary of new tools and features

An overview of nf-core/sarek v3.1.1 is shown in Fig. 2. The input data is an nf-core community standardized samplesheet in comma-separated value (CSV) format, that provides all relevant metadata needed for the analysis as well as the paths to the FastQ files. The pipeline has multiple entry points to facilitate (re-)computation of specific

steps (e.g. recalibration, variant calling, annotation) by providing a samplesheet with paths to the intermediary (recalibrated) BAM/CRAM files. The pipeline processes input sequencing data in FastQ file format based on GATK best-practice recommendations [34], [35]. It consists of four major processing units: pre-processing, variant calling, variant annotation, and quality control (QC) reporting.

Pre-processing

Enabling homogeneous processing of global genomic resources requires flexibility on the genomic "raw" input data. To cope with the fact that different data repositories provide their "primary" data in different formats, nf-core/sarek support both BAM and FastQ as input. When BAMs are provided as starting input, they are converted to FastQ via SAMtools [36] which allows for fully homogenous processing independent of the provided input format.

The FastQ files are then split into shards with fastP including optional adapter trimming allowing the subsequent alignment step to be run on smaller machines. FastP has been introduced with the release v3.0 and is advantageous over other splitting and adapter removal tools as it combines FastQ sharding and adapter removal into one step, speeding up the computation. With this new implementation, we no longer need to rely on Trim Galore! and Nextflow's native `splitFastq()` function.

Version 3.1.1 of nf-core/sarek can handle UMI barcodes, which are used in some protocols to detect low allele frequency variants [37]. The user can opt for using Fulcrumgenomics' fgbio[1] tool, which generates a consensus read among the ones carrying the same UMI. It will then use these reads as input for the remaining pre-processing steps.

The split FastQ files are aligned with one of the available mappers, which include BWA-MEM [38], BWA-MEM2 [39], or DragMap,[2] and name- or coordinate-sorted with SAMtools. By adding DragMap support, we comprehensively cover the community's needs. We added the missing pre-computed reference indices for BWA-MEM2 and DragMap for GRCh38 and GRCh37 to speed up the computation. As recommended by GATK guidelines, we use the entire genome during mapping. Off-target reads for WES and panel analysis are removed according to the provided BED file during the base quality score recalibration (BQSR) step.

By default, the aligned BAMs are then merged, duplicates are marked with GATK4 Markduplicates, and converted to CRAM format in one process to reduce runtime and storage needs. The duplicate marking step was improved by providing name-sorted alignment files to GATK4 MarkDuplicatesSpark. If duplicate marking is skipped, SAMtools is used for merging and conversion to CRAM format. BQSR on the resulting CRAM files is run with GATK4 BaseRecalibrator and GATK4 ApplyBQSR. For both, the GATK Spark implementation is available. Both steps can be skipped, in which case the mapped BAMs are converted to CRAMs using SAMtools.

In order to speed up the computation, genomic regions are processed in parallel following the duplicate marking step. Small regions are grouped and processed together to reduce the number of jobs spun up. By default, interval lists for the complete

[1] https://github.com/fulcrumgenomics/fgbio
[2] https://github.com/Illumina/DRAGMAP)

reference genome provided by GATK are used, enabling scattering by chromosome, and removing unresolved and difficult regions. For targeted sequencing data, we have added support to use the respective target bed files for parallelization as recommended by the GATK guidelines.[3] Previously, BQSR was always run on the intervals provided for WGS, which led to recalibrating off-target reads increasing computational resources needed. We have added further support to allow users to control group size not just for custom interval files, but also for the ones generated from the genomic regions, allowing a more tailored setup.

Variant calling

nf-core/sarek includes a comprehensive set of variant callers to obtain SNPs/Indels, SV, MSI, and/or CNV values using a total of 12 tools (Fig. 2). The variant calling tools to be run can be selected by the user and are executed in parallel. Newly included tools in the v3.1.1 release are Deepvariant, CNVKit [40], and Tiddit [41]. Furthermore, Haplotypecaller supports both single sample or joint-germline calling [42]. When both Strelka2 [43] and Manta [44] are selected, the candidate Indels from Manta are used for SNP/Indel calling according to the Strelka2 best-practices.[4] We added a new parameter that allows skipping germline-only variant calling for paired samples to further reduce time, costs, and compute resources for somatic variant calling. Furthermore, scatter-gathering is now supported for all applicable variant calling tools across intervals (see Supplementary Fig. A). The sharded VCF files are then merged with the GATK4 MergeVCF tool. In this way, we reduce computing demands, by avoiding repeated cycles of (de)compressing the files.

Variant annotation

The resulting VCF files can be annotated with VEP [45], snpEff [46], or both either separately or by merging the output annotations. The annotation tool VEP has been extended with new plugins allowing more comprehensive annotation: the previously used plugin CADD [47] has been superseded by dbNSFP [48] providing 36 additional prediction algorithms. Furthermore, the plugins LOFTee [49], spliceAI [50], and spliceRegion[5] have been added.

QC reporting

Throughout the pipeline, quality control tools are run, including FastQC before alignment, mosdepth [51], and SAMtools post-duplicate marking and BQSR, as well as vcftools [52] and bcftools on called variants. These results are collected into a MultiQC [53] report together with software version numbers of all the executed tools. The previously used Qualimap [54], which has no direct CRAM support and requires a high amount of computational resources, has been replaced with mosdepth, a fast quality

[3] https://gatk.broadinstitute.org/hc/en-us/articles/360035889551-When-should-I-restrict-my-analysis-to-specific-intervals-
[4] https://github.com/Illumina/strelka/blob/v2.9.x/docs/userGuide/README.md
[5] https://github.com/Ensembl/VEP_plugins/blob/release/109/SpliceRegion.pm

control tool for alignment files. The tool produces comprehensive output files allowing to visualize, e.g. coverage data with the Integrated Genome Viewer (IGV) [55] for easy inspection.

All tools have been updated to their latest stable release at the time of writing. For a complete overview of the most important tool changes, see Supplementary Table B.

Pipeline skeleton changes

We expanded the continuous integration testing to include all the new functionality, as well as adding md5sum checking of output files wherever possible. Additionally, we added full size tests that are automatically run on each pipeline release. All nf-core pipelines require a full-size test on realistic data upon release to ensure functionality beyond small test data and portability to cloud infrastructures. The datasets used here include the Genome in a Bottle data set HG001 (downsampled to 30X WGS) for germline variant calling testing and the tumor-normal pair SRR7890919/SRR7890918 provided by the SEQC2 effort for somatic variant calling testing. Since each of the selected datasets comes with validated VCFs to compare against, they are suited for further benchmarking to investigate the variant calling results. The results for each full size test are displayed on the website nf-co.re/sarek and are available for anyone to explore or download.

High-quality code readability is achieved by combining modules used in the same analysis context into subworkflows, e.g. variant calling with a specific tool and subsequent indexing of the resulting VCF files. In addition, new analysis steps can be added by providing such encapsulated subworkflows, and obsolete parts can quickly be removed entirely. Furthermore, dividing different analysis steps into subworkflows written in separate files simplifies development, which is often done asynchronously with developers at different institutes.

2.2 CRAM format allows for storage space reduction

The pipeline has a large data footprint due to the number of computational steps, input data size, and Nextflow's requirement for a work directory with intermediate results to facilitate resuming. In order to ease storage needs as a possible bottleneck, CRAM files are used as of nf-core/sarek 3. They are a more compressed alternative to BAM files storing only differences to the designated reference. A majority of tools post-duplicate marking support CRAM files. The pipeline can handle both BAM files and CRAM files as in- and output to accommodate various usage scenarios.

We evaluated the resource usage of the two alternatives by running five tumor-normal pairs on nf-core/sarek 3.1.1 as well as on a branch[6] based on the release in which the internal format was replaced with BAM. Pre-processing (Fig. 3 and Supplementary Fig. C2) and variant calling (Supplementary Fig. C3) were evaluated separately. For eleven processes, the CRAM-based setup resulted in a significant decrease in runtime, for ten an increase in memory, and for eleven an increase in CPU hours could be measured. The overall average CPU hours for the pre-processing benchmark on CRAM version was 3,252.37, in comparison to BAM 3,761.07. The gain in CPU hours usage,

[6] https://github.com/FriederikeHanssen/sarek/tree/bam_31

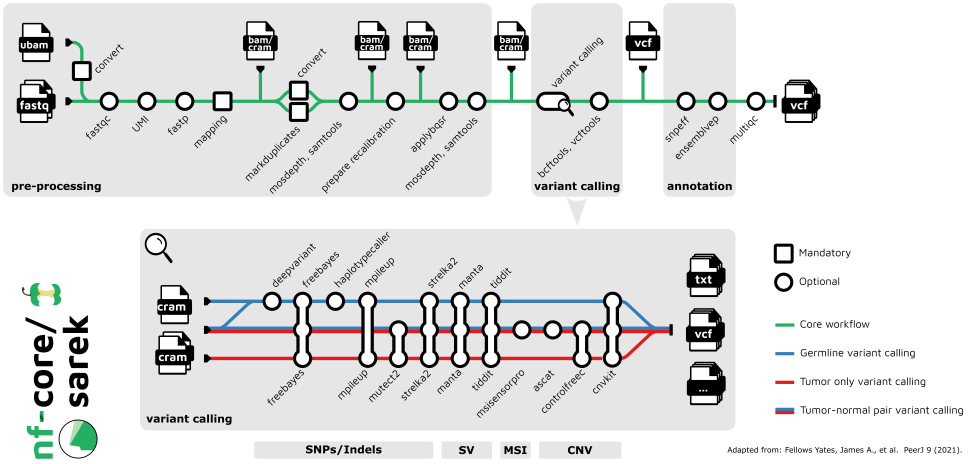

Fig. 2 Overview over nf-core/sarek. The pipeline consists of three sections: pre-processing based on GATK Best-practice recommendations (mapping, duplicate marking, and base quality score recalibration), variant calling supporting tools for SNP/Indel, SV, CNV, MSI calling, and annotation. Throughout the pipeline, various quality control tools are run and collated into a comprehensive MultiQC report. The variant calling tools can be mixed in any combination and are all run in parallel.

however, had to be compensated by a 34% increase in memory usage. The overall average total memory usage on the CRAM version was 10,346.8GB, for the BAM version it summed up to 7,739.51GB. The storage usage for the work directory for preprocessing these samples drops by 65%, from 170.4TB (BAM) to 59.7TB (CRAM). Processes outputting CRAM files reduce their storage needs by at least a third. In the case of GATK4 ApplyBQSR it was reduced by 64%. Processes operating on CRAM files outputting a different format, e.g. VCFs, show no change in storage usage.

2.3 Scatter-gather implementations reduce runtime and resource usage

Scatter/gather implementations are highly relevant for parallel processing approaches across genomic regions for BQSR and variant calling. In this release, we have further extended these options: Before mapping, the input FastQ files can now be split and mapped in parallel. For BQSR and variant calling more options to customize the amount of scattering as well as further support for all eligible variant callers are implemented.

We evaluated the impact of different degrees of FastQ file sharding on the mapping process by investigating the division step (fastP), mapping (BWA-MEM), and subsequent merging (GATK4 Markduplicates). The realtime as reported by the Nextflow trace of the longest running mapping process of any one sample was summed up with the realtime of fastP and GATK4 Markduplicates. The space of the work directories of each involved task was summed up, as well as the CPU hours (see Fig. 4A). The overall runtime for the mapping processes decreases until it reaches a plateau at 12 shards,

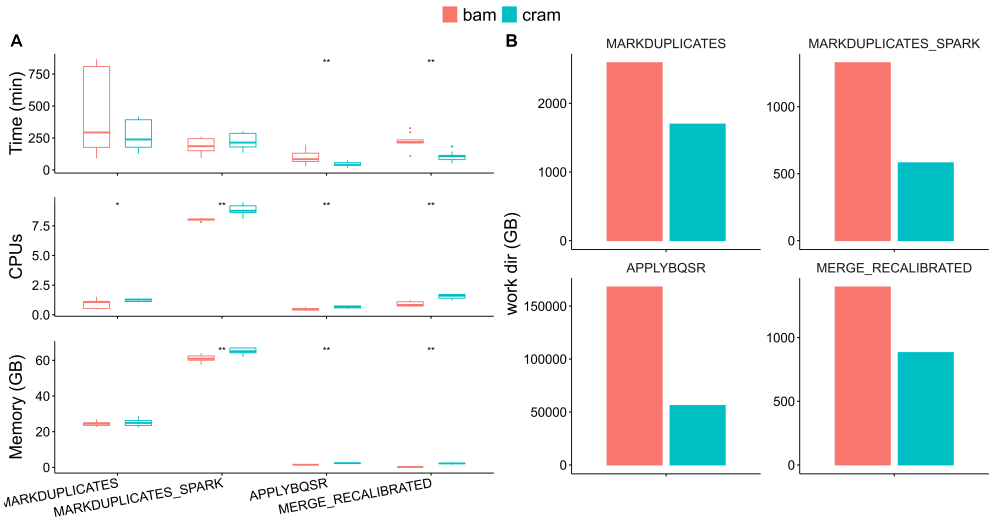

Fig. 3 Resource usage of nf-core/sarek 3.1.1 when storing intermediate data in BAM versus CRAM format. **A)** Average realtime, and maximum CPU and memory usage (peak_rss) as reported by the Nextflow trace file for the main processes. For processes split within a sample (i.e. ApplyBQSR), the task with the highest runtime per sample is shown as the process runtime. Resource usage was compared using the paired Wilcoxon test (** $p < 0.01$, * $p < 0.05$). Two out of the four shown processes are significantly faster when using CRAMs instead of BAMs at the expense of an increase in memory or CPU usage. **B)** Storage was evaluated by calculating the total size of the work directories of all tasks of the respective process. Each condition was repeated three times for samples of five tumor-normal paired patients.

achieving a reduction of the median runtime to 37%. The storage usage increased as soon as any sharding was done due to the sub-FastQs being written to the disk. The CPU hours remain approximately the same due to the long alignment time for large files (see Supplementary Fig D4).

We evaluated the impact of different degrees of scattering across genomic intervals on the recalibration and variant calling process with respect to resource usage. Similarly to the mapping processes, the most extended runtime per interval group per samples of all involved processes for BQSR (GATK4 BaseRecalibrator, GATK4 GatherBQSRReports, GATK4 ApplyBQSR, and SAMtools merge) and variant calling (calling and GATK4 MergeVCFs) was summed up respectively. Storage usage and CPU hours results for all tasks were added up. The runtime decreased the most for all measured tools (see Fig. 4B and Supplementary E) when the number of interval groups was set to 21. Raising the number of interval groups did not decrease runtime further. For GATK-based tools, storage usage increased with each further splitting of interval groups. For BQSR storage requirements between 21 intervals groups and 124, the default value, increased by a factor of five. For Deepvariant less storage space was required when applying scattering (Supplementary E6), however, for all other variant callers the storage needs remain on a stable level. The required CPU hours remain stable across various amounts of scattering for all tools.

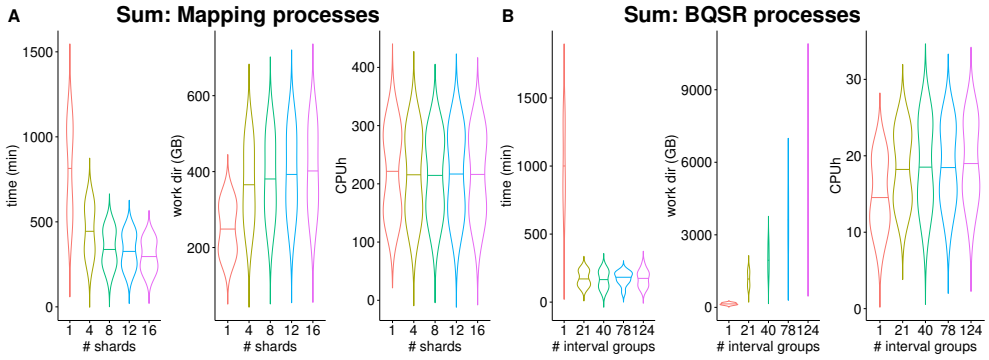

Fig. 4 Sharding the input FastQ files and parallelizing computation on interval groups reduces the overall runtime of the nf-core/sarek pipeline. **A)**: Effect of sharding the input files on the mapping processes, including fastP, BWA-MEM, and Markduplicates. The input FastQ files were split into smaller pieces increasing the amounts of shards and the runtime, work directory size, and CPU hours were evaluated for each split size. FastP was run with a different number of CPUs corresponding to the desired number of shards. **B)**: Effect of parallelizing computations across interval groups on BQSR processes, which include the BaseRecalibrator, GatherBQSRReports, ApplyBQSR, and SAMtools merge process. When all intervals were processed together as one group the memory requests for ApplyBQSR had to be increased. The violin plots show computations on tumor-normal paired samples of five patients. The time was evaluated by summing up the highest realtime per task per sample as reported by the Nextflow trace report. The work directory size and CPU hours are the sums of all involved tasks.

2.4 Benchmarking of short variants against truth VCFs

The pipeline's SNP and indel variant callers were evaluated for both the germline and paired somatic analysis tracks, the former on three WGS datasets from Genome in a Bottle (HG002-4) [56], the latter on three tumor-normal paired WES datasets from the SEQ2 Consortium [57]. The results were compared to the respective "gold standard" VCFs for high-confidence calls.

We evaluated the precision, recall, and F1 score over all samples. For the germline calls of Deepvariant, only sample HG003 was used since its model was trained on the remaining datasets. The tools' precision, recall, and F1 scores are in accordance with the previously reported FDA precision challenge [58] results for Genome in a Bottle(GiaB) samples (see Figure 5A, B for SNPs, Supplementary Figure F8 for Indels). BWA-MEM and BWA-MEM2 lead to higher recall values than DragMap. Strelka2 together with Manta and DeepVariant perform best in all three evaluated metrics. In addition, we investigated one sample sequenced with MGI and BGISEQ respectively with similar results for all variant callers (Supplementary F9, F10).

Similarly, we evaluated the precision, recall, and F1-score for the somatic calls. Filtered Mutect2 calls have the highest precision calls for all samples, and FreeBayes ones the highest recall values (Fig. 5C,D for SNVs). The highest F1-Score is measured for Mutect2, followed by Strelka2. For Indels, Strelka2 outperforms all other tools (Supplementary Figure F8). The results are in-line with what has been previously reported [59].

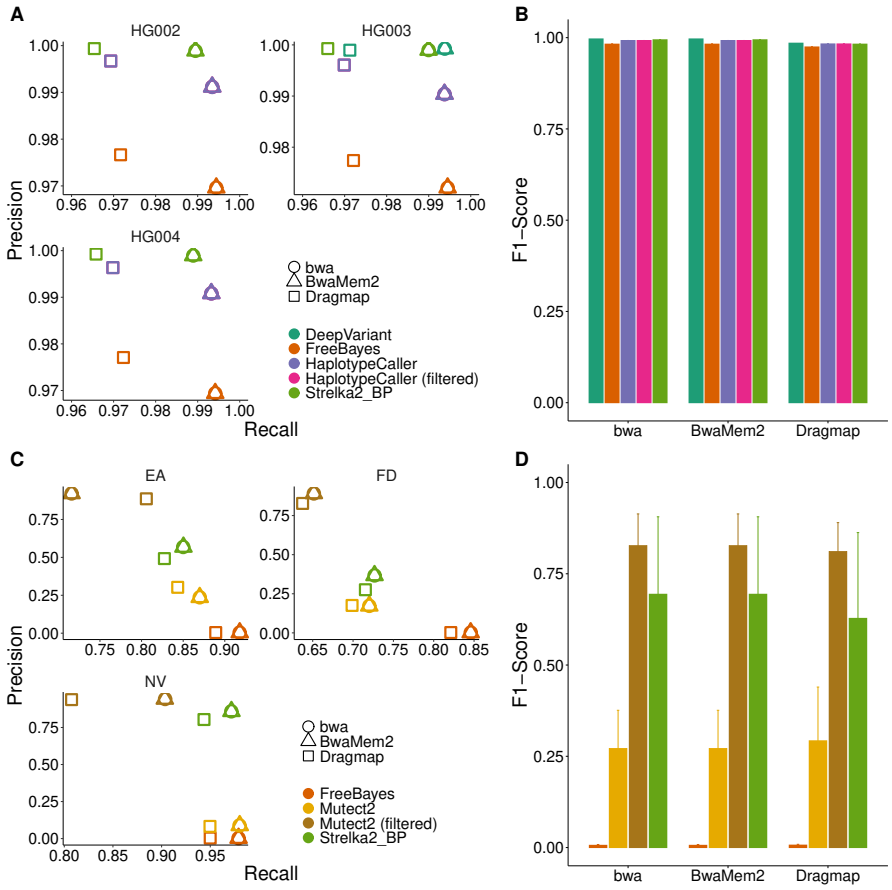

Fig. 5 Germline and somatic variant calling evaluation of high-confidence calls using ground-truth benchmarking data with respect to SNPs. **A,B**: The germline variant calling track of the pipeline was evaluated using 3 WGS GiaB datasets (HG002-HG004). The average precision, recall, and F1-score values across all the samples are plotted, respectively. **C,D**: The somatic paired variant calling track was evaluated using three tumor-normal WES pairs (EA, FD, NV) from SEQ2C.

2.5 Portability to AWS and computing costs

The sheer amount of existing genomics data and the unavoidable need for even more data for the detection of disease-causing genotype-phenotype correlations or population-scale analyses will test the limits of on-premise computing sooner or later. Consequently, more and more data analysis is shifted or supplemented by computation in the cloud. The most recent Nextflow Community survey 2023 indicates that 43% of users use cloud services, which represents an increase of 20% compared to the previous year[7] with AWS still being the most popular among the respondents. Therefore, we evaluated the cost development of nf-core/sarek between 2.7.1 and 3.1.1 on AWS Batch.

[7] https://seqera.io/blog/the-state-of-the-workflow-2023-community-survey-results/

We were able to reduce cloud computing costs to approximately 30%(see Table 1). Furthermore, we could reduce the overall runtime and CPU hours. For a single sample the needed CPU hours are reduced by approximately 70%, and the runtime by 84%. Here, we used spot instances - unused instances auctioned off at a percentage of their on-demand price - whose prices fluctuate constantly. The business models of the cloud providers result in varying spot price percentages and spot prices are frequently subject to change depending on the overall demand.

Table 1 Average costs per patient on AWS batch for nf-core/sarek version 2.7.2 and 3.1.1. All pipeline runs were performed with the tools, Strelka2, Manta, and VEP. Each analysis run is repeated three times and run on data from the donor DO50970 with either their normal or tumor-normal paired sample.

Version	Samples	Avg. costs [$]	Runtime	CPU hours
2.7.2	1 normal	68.04	46h8m	1118.4
3.1.1	1 normal	20.82	12h4m	342.5
3.1.1	1 paired	66.83	31h47m	1324.3

3 Discussion

An essential aspect of high-throughput processing is seamless scalability, one of the main advantages of using a dedicated workflow language such as Nextflow. Combining adapter trimming, quality control, and sharding of FastQ files in a single step, as well as more tailored splitting into intervals for variant calling, reduces the needed CPU hours by 66%. Replacing Nextflow's native `splitFastq()` function with a dedicated process, allows us to make use of all advantages of regular job submission, including assigning resources to the jobs, automatic retries on job failure, and resume functionality. Previous pipeline versions have typically not been able to split the FastQ files and have, thus, missed out on scalability options. Our experiments have shown clear limits for parallelization into interval groups. There is no further benefit to reducing runtime beyond 21 interval groups. However, storage usage increases. This is respected in future releases by setting the default number of interval groups to 21 further reducing the storage footprint of the pipeline.

The switch to using CRAM files results in reduced storage space usage of 65%, at the expense of higher memory requirements. The additional memory needs are distributed over the thousands of tasks run for the benchmark. Due to this, in practice, memory is not a limiting factor. The additional needs are only required for the task's run time, usually comprised of a couple of GB at most, and can subsequently be reused. The storage, however, accumulates over the entire pipeline run and can therefore pose a bottleneck for usage scenarios with large input data sets relative to the available storage on the respective compute system.

Both changes, switching to CRAM format reducing storage requirements by two-thirds and reducing the amount of scattering - further reducing storage requirements by a factor of 5, will enable users to run the pipeline on smaller systems more efficiently. While there are many benefits for the scientists running the pipeline, we would like to emphasize also the ecological need to reduce the carbon footprint in computational research. This is relevant in particular considering the ever-growing number of available samples in national and international genome repositories, which aim at facilitating truly comprehensive population-based analysis and understanding underlying genome variations. Our results provide solutions to reduce CO_2 emissions.

One of the main objectives of this work is to enable scientists to run the pipeline in cloud environments at low costs per sample. Cost-efficient cloud computing is increasingly important for data-driven science. Intransparent and unpredictable cost models discourage a scientist. Using the cloud setup as described in this work, we reduced the costs to 50% in comparison to previous releases. Recently, Seqera Labs posted a blog article[8] showing further cost reductions when using NVMEs and a new fusion file system promising even cheaper runs in the future. In addition, users can further reduce their cloud costs by selecting compute instances in cheaper AWS regions and fixing the spot price percentage they are willing to pay more tightly by enforcing an upper bound for costs per sample at the expense of possible waiting time for such machines to become available.

nf-core/sarek is an established, comprehensive variant calling pipeline in the genomics field, which can be applied to any organism for which a reference genome exists. Future releases further simplify analyses with custom references by enabling pre-computation of all needed indices, an interesting feature when multiple users work with organisms on shared systems for which reference files are not provided by default. On request, such references and their corresponding indices and database files can be added to the central resource AWS-iGenomes and made available to the community.

The nf-core/sarek rewrite to DSL2 makes the code base more maintainable and easier to read, a factor that is crucial to allow new developers to join the effort with a reasonable learning curve. All pipeline processes are specified in separate files in the form of modules a majority of which are maintained by the nf-core community. Tools used in the same context are combined into subworkflows. They will be added to the nf-core subworkflows collection in the near future allowing further collaboration and shared maintenance across pipelines and beyond the nf-core community. Modularising all tools will enable us to simply do a drop-in replacement when tools should be exchanged for a different one or new ones added as they emerge. nf-core/tools installs them in the appropriate directory, they just need to be called at the appropriate position in a (sub)workflow. Furthermore, the use of modules allows users to customize the released pipeline version at runtime. Before this change, it was necessary to change the underlying code if arguments of a tool were not exposed to the pipeline. This

[8] https://seqera.io/blog/breakthrough-performance-and-cost-efficiency-with-the-new-fusion-file-system/

was limiting for users since they had to wait for the feature to be implemented and released before using it. With the new modular config files, arguments can be modified by providing a user-based custom configuration setting the exact command line arguments without changing the underlying code or the release tag. This allows a higher degree of flexibility for the analysis whilst simultaneously using a released version. Reproducibility can then be ensured as before by providing the exact release version, pipeline parameters, and the respective custom config(s).

4 Methods

4.1 Implementation

nf-core/sarek is a Nextflow-based pipeline that has been part of the nf-core project since release 2.5. Within nf-core, all pipelines must adhere to the nf-core template, which provides a common skeleton structure to ensure current best practices. The community provides a Python-based helper package nf-core/tools which enforces these guidelines through code linting and continuous integration testing. In 2020, the Nextflow language evolved from DSL1 to DSL2. DSL2 allows the modularization of each pipeline step. For this, the nf-core community has established a shared repository, nf-core/modules, implementing Nextflow wrappers around ideally individual tools, which are made available to anyone implementing Nextflow-based pipelines. The tools are typically accessible through (bio)conda [60] and have a corresponding docker and singularity container provided by the Biocontainers [61] community enabling portability and reproducibility for each such 'module'. This single-tool-per-process approach ensures that previously occurring dependency conflicts are mitigated. On each module change, the code is run through continuous integration testing on a minimal data set before making it available to the community using Github Actions with docker, singularity, and conda as compute environments. nf-core/tools allows easy creation, installation, and re-use of these modules. In order to benefit from these changes, a re-implementation was necessitated. In fact, since nf-core/tools version 2.0, all pipelines must be (re-)written to adhere to the new Nextflow DSL2 syntax. In nf-core/sarek, 80 different modules are used of which 78 are made available from the shared repository. This allows the pipeline to heavily benefit from and contribute to the nf-core framework. In addition, future additions or replacements of tools with successors are greatly simplified.

4.2 Data sets & compute environments

In order to evaluate the computational requirements of the pipeline, five tumor-normal paired samples from the ICGC LICA-FR [62] cohort are used (see Table 2). The unaligned BAM files are downloaded and converted to paired-end FastQ files using nf-core/bamtofastq v1.0.0 (formerly qbic-pipelines/bamtofastq) with Nextflow version 20.10.0 and singularity.

Unless otherwise indicated, evaluations are done with Nextflow version 22.10.2 build 5832 and Singularity 3.8.7-1.el8 on a shared HPC cluster with a parallel BeeGFS Filesystem (400TB) using Slurm with 24 nodes with 32 cores and 64 threads (2* AMD

Table 2 Datasets used for benchmarking are part of the LICA-FR cohort. The BAM files are downloaded and converted to FastQ files. The respective donor IDs and file sizes of the converted FastQ files are listed below.

Donor ID	File size (normal) [GB]	File size (tumor)[GB]
DO50970	88	152
DO50974	101	173
DO50933	92	154
DO50935	116	166
DO50936	97	174

EPYC 7343), with 512GB RAM and 2TB NVME as disk and four nodes with 64 cores and 128 threads (2* AMD EPYC 7513), with 2048GB and 4TB NVME as disk. The cluster is shared and resources per user are allocated by a Fair-share policy. At any time 100 tasks can be run at most by a user in parallel.

All jobs are submitted using `-profile cfc` providing a cluster-specific configuration, which is stored in the GitHub repository nf-core/configs.[9]

Resource usage for all experiments was evaluated by supplying:

Listing 1 trace.config

```
trace {
    fields = 'task_id,hash,native_id,process,tag,name,
    status,exit,module,container,cpus,time,disk,memory,
    attempt,submit, start,complete,duration,realtime,
    queue,%cpu,%mem,rss,vmem,peak_rss,peak_vmem,
    rchar,wchar,syscr,syscw,read_bytes,write_bytes,
    vol_ctxt,inv_ctxt,workdir,scratch,error_action'
    raw = true
}
```

in a custom configuration file and collecting the file sizes of the work directory with:

Listing 2 storage.sh

```
du -hb --all --max-depth=4 <absolute/path/to>/work/
    > folder_sizes.tsv
```

4.3 Reducing storage requirements

In this release, the internal file format following duplicate marking is changed to using CRAM files. To evaluate the required compute resources and the actual data footprint for both file formats, five paired ICGC genomes are run through all tools part of nf-core/sarek 3.1.1 and an altered version of nf-core/sarek 3.1.1 that uses the BAM format instead. For each process, singularity containers from the Biocontainers registry are used. Each run configuration is repeated three times. Unless otherwise specified, the default parameters for nf-core/sarek 3.1.1 are used. We evaluate the pre-processing and variant calling steps independently.

[9] https://github.com/nf-core/configs/blob/c709be3b599d463fcfa82196fd4c9c5fa1e99513/conf/cfc.config

We evaluate all processes corresponding to FastQ quality control, aligning the reads to the reference genome, duplicate marking, base quality score recalibration (BQSR), and quality control of aligned reads. The command for running the pre-processing steps is the following:

Listing 3 Pre-processing

```
nextflow run nf-core/sarek -r 3.1.1 -profile cfc \
--input ./input.csv
```

The memory requirements for BWA-MEM are increased to 60GB, as well as the runtime for GATK4 Markduplicates and SAMtools merge to 16h and 8h respectively from the provided defaults.

Secondly, we evaluate variant calling with all tools for all germline and paired somatic variant calling for all samples:

Listing 4 Variant calling

```
nextflow run nf-core/sarek -r 3.1.1 -profile cfc \
--input recalibrated.csv \
--tools deepvariant,haplotypecaller,mutect2,strelka, \
freebayes,ascat,controlfreec, \
cnvkit,manta,tiddit,msisensorpro \
--step variant_calling
```

The requested time for all processes is increased to 144h to mitigate interruptions due to runtime time-outs by providing a custom configuration file.

4.4 Evaluation of pipeline runtime and resource usage

In order to evaluate the impact on runtime and storage requirements, ten samples are run with different sizes of scattered groups: 1, 21, 40, 78, and 124 (default). In order to generate the respective interval group sizes the parameter `--nucleotides_per_second` is set to 5000000, 200000, 70000, 10001, 1000 (default). For read splitting, fastP is run with various numbers of CPUs specified (respectively, 0, 4, 8, 12 (default), and 16) since the tools generates chunks firstly by number of CPUs and secondly by the maximum number of entries per file as defined by a parameter. Here, it is set to 500,000,000 to prevent any further subdivision.

Example command for 40 interval groups and 8 FastQ file chunks:

Listing 5 Intra-sample parallelization

```
nextflow run nf-core/sarek -r 3.1.1 -profile cfc \
--input input.csv \
--tools deepvariant,haplotypecaller,mutect2,strelka,
freebayes,ascat,controlfreec,
cnvkit,manta,tiddit,msisensorpro \
--nucleotides_per_second 70000 \
--split_fastq 500000000
-c ressource.config
```

The memory requirements for BWA-MEM are increased to 60GB for all tests. The memory for FreeBayes is increased to 24GB, for one interval group it is reduced

again to 18GB. For GATK4 ApplyBQSR the memory is increased to 16GB when all intervals were processed in one group. For a full list of configs, see https://github.com/qbic-projects/QSARK.

4.5 Benchmarking of short variants against truth VCFs

In order to benchmark the variants called by the pipeline, both the germline and paired variant calling tools are evaluated on three datasets each: for the germline callers the Genome in a Bottle(GiaB) datasets HG002-HG004 [56] are used, for the paired somatic callers three WES datasets from the Sequencing Quality Control Phase II Consortium [57]. Comparisons are made only in the high-confidence regions defined for the benchmark, i.e. omitting difficult-to-call regions.

Germline variant calling

The whole-genome germline GiaB samples from an Illumina NovaSeq are downloaded from the GiAB consortium's ftp-server. Downsampling to 40x is done using the seqtk[10] tool. nf-core/sarek is run with default parameters. The parameter `--nucleotides_per_second` is increased to 200,000. All eligible variant callers are combined with all three mappers. Comparisons are calculated using hap.py,[11] version v0.3.14:

Listing 6 Evaluation of germline calls

```
hap.py HG00{2,3,4}_GRCh38_1_22_v4.2.1_benchmark.vcf.gz \
       query.vcf.gz \
       -o results/ \
       -V --engine=vcfeval \
       --engine-vcfeval-template grch38.sdf \
       --threads 3 \
       -f \
       HG00{2,3,4}_GRCh38_1_22_v4.2.1_benchmark_noinconsistent.bed \
       --logfile results/
       --scratch-prefix .
```

Short variant calls from Haplotypecaller, Deepvariant, Freebayes, and Strelka2 mapped with Dragmap (base quality recalibration is skipped), BWA-MEM, and BWA-MEM2 are included in the analysis. Deepvariant is evaluated only on HG003.

Furthermore, reads for the sample HG002 sequenced with MGISEQ and BGISEQ500 are downloaded from the manufacturer. They are downsampled to 20X, 30X, 40X, and 50X using seqtk and subsequently processed with nf-core/sarek 3.1.1 using default parameters and all eligible variant callers. Evaluation against the truth VCF is done using hap.py.

Somatic variant calling

The somatic short variant calls are evaluated on three whole-exome sequencing datasets: SRR7890919/SRR7890918 (EA), SRR7890878/SRR7890877 (FD),

[10] https://github.com/lh3/seqtk
[11] https://github.com/illumina/hap.py

SRR7890830/SRR7890846 (NV) run on an Illumina HiSeq 1500 (EA), 4000 (FD), 2500 (NV). The data is downloaded using nf-core/fetchngs v1.10.0. nf-core/sarek is run on default parameters. In addition, trimming is enabled with `--trim_fastq`. When using DragMap `--skip_tools baserecalibrator` is set. The VCFs are PASS filtered using bcftools v1.10.2. The calls are evaluated using RTGTools [63] for a combination of each mapper with all available variant callers: Strelka2 together with Manta, Freebayes, and Mutect2.

Listing 7 Evaluation of somatic calls

```
bcftools view -f 'PASS,.' \
            results.vcf.gz \
            -o query.vcf

rtg vcfeval -c query.vcf.gz \
    -b high-confidence_sSNV_in_HC_regions_v1.2.vcf.gz \
    -o ./out/ \
    -t grch38.sdf \
    -e High-Confidence_Regions_v1.2.bed \
    --squash-ploidy --all-records --sample=ALT \
    --bed-regions \
    S07604624_Padded_Agilent_SureSelectXT_allexons_V6_UTR.bed
```

4.6 Portability to AWS cloud and computing cost

In order to evaluate the costs for running nf-core/sarek on AWS batch, the compute environments are created using Tower Forge,[12] with the following settings: spot instance, max CPUs 1,000, EBS Auto scale, and fusion mounts enabled. Instance types are chosen by using strategy 'optimal' with the allocation strategy `spot_capacity_optimized`. The computation is run in AWS region us-east-1. The pipeline runs are launched with Nextflow Tower setting the Nextflow version to 22.10.3 by adding export `NXF_VER=22.10.3` to the pre-run script and `process.afterScript = 'sleep 60'` to the config section.

The pipeline is run on the normal sample for DO50970 using default settings together with Strelka2, Manta, and VEP. For one paired sample evaluation, the pipeline was run the tumor-normal pair (DO50970/DO50970).

nf-core/sarek 3.1.1 is run with `--nucleotides_per_second 200000`. For the paired run, we set `--only_paired_variant_calling`.

The costs for nf-core/sarek 2.7.1 are evaluated on the same normal sample using Strelka2, Manta, and VEP on default parameters. Compute resources for mapping (372GB memory, 48 cpus), duplicate marking (30GB memory, 6 cpus), quality control with BamQC (372GB memory, 48 cpus), GATK4 BaseRecalibrator (4GB memory, 4 cpus), and GATK4 ApplyBQSR (4GB memory) is increased.

Supplementary information. All supplementary files are available at https://github.com/qbic-projects/qsark.

[12] https://cloud.tower.nf/

Acknowledgments. We want to thank Johannes Köster for his advice during the benchmarking. We would also like to thank the nf-core and Nextflow community for developing the nf-core infrastructure and resources for Nextflow pipelines and, in particular Mahesh Binzer-Panchal and Alexander Peltzer, for their assistance in reviewing and releasing the pipeline. A full list of nf-core community members is available at https://nf-co.re/community.

Declarations

Some journals require declarations to be submitted in a standardized format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading 'Declarations':

- **Funding**
  S.N. acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) via the projects Sonderforschungsbereich SFB/TR 209 "Liver cancer" and NFDI 1/1 "GHGA - German Human Genome-Phenome Archive" (441914366 to NS), and the DFG im Rahmen der Exzellenzstrategie des Bundes und der Länder EXC 2180 – 390900677 and EXC 2124 – 390838134.
- **Conflict of interest/Competing interests**
  None to declare.
- **Ethics approval**
  Not applicable.
- **Consent to participate**
  Not applicable.
- **Consent for publication**
  All authors read the manuscript and consent to its publication.
- **Availability of data and materials**
  The comparison pipeline using BAM files is available at https://github.com/FriederikeHanssen/sarek/tree/bam_31. All pipeline run commands used for the benchmarking, configurations, trace reports, evaluation, and visualization scripts are available at https://github.com/qbic-projects/qsark.
- **Code availability**
  The pipeline is available at https://github.com/nf-core/sarek and each release is archived on Zenodo https://doi.org/10.5281/zenodo.3476425
- **Authors' contributions**
  FH and MUG lead the project. FH, MUG, LF, ASP, FL, SJ, OW, NS wrote the pipeline. GG added the full-size tests. EM and MUG added the CI testing framework. LF and FH added the evaluation against the truth datasets. GG and FH added the AWS experiments. FH added the remaining experiments. FH, LF, and FL wrote the manuscript with comments from all authors. GG and SN supervised the project.

Appendix A Intra-sample parallelization

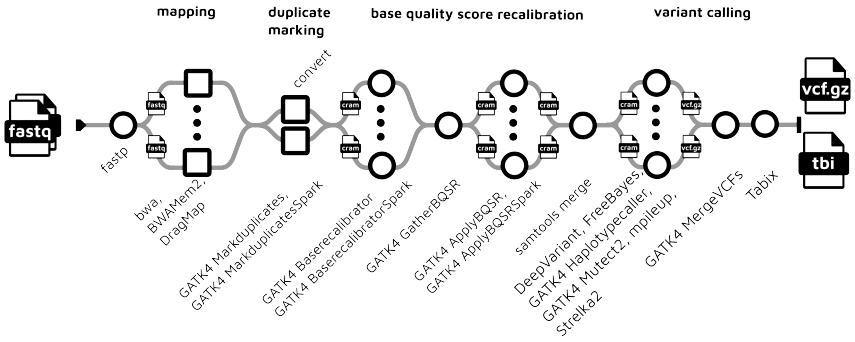

Fig. A1 The figure depicts the processing of a single sample. While Nextflow runs the analysis of each sample in parallel, intra-sample parallelization was implemented for the mapping step by splitting the FastQ files beforehand, mapping each, and merging all FastQ files belonging to one sample and duplicate marking them. Base quality score recalibration and SNP/Indel calling are run across genomic intervals, which can be either user-provided or GATK-provided interval files.

Appendix B Table with detailed changes between Sarek 2.5.2 and 3.1.1

Appendix C Reducing storage requirements with CRAM

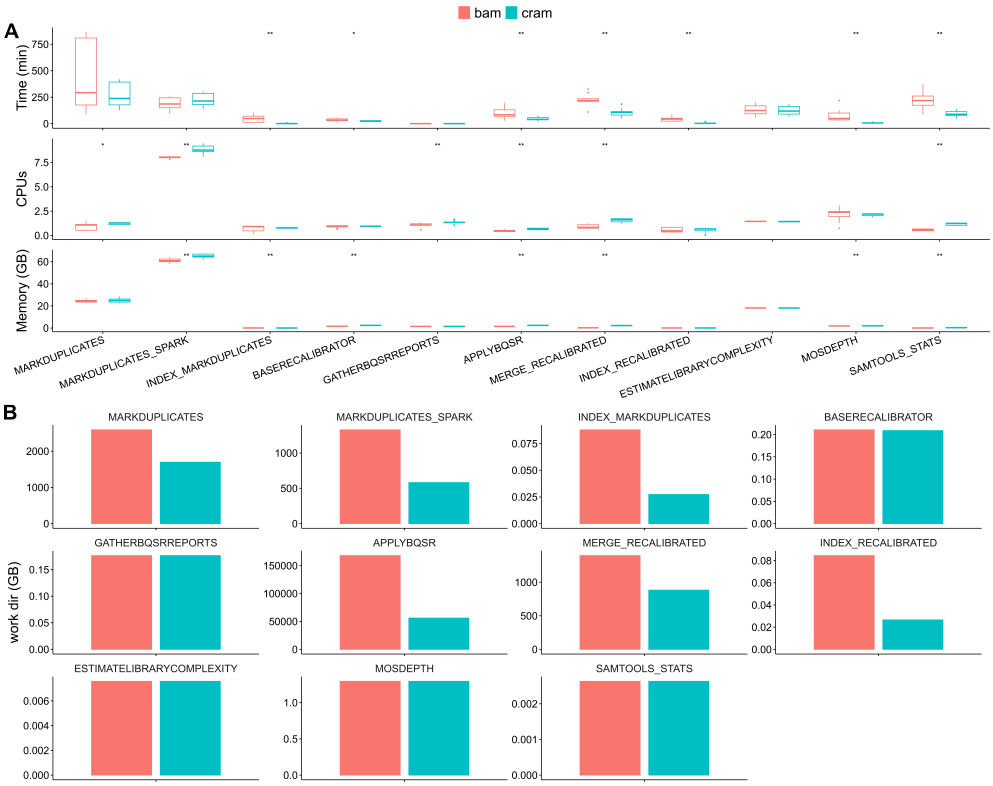

Fig. C2 Resource usage of nf-core/sarek 3.1.1 when storing intermediate data in BAM versus CRAM format for all pre-processing processes. **A)**: Average realtime, and maximum CPU and memory usage (peak_rss) as reported by the Nextflow trace file. For processes split within a sample (i.e. ApplyBQSR), the task with the highest runtime per sample is shown as the process runtime. Resource usage was compared using the paired Wilcoxon test (** $p < 0.01$, * $p < 0.05$). Seven of the eleven processes are significantly faster when using the CRAM format. Five processes have a significantly higher CPU hour usage, and three require more memory in comparison to using BAMs. **B)**: Storage was evaluated by calculating the total size of the work directories of all tasks of the respective process. The storage usage between both is identical, due to Nextflow accessing the input files via symlinks. Thus only the output is measured here for each process, which is independent of input format. Each condition was repeated three times for samples of five tumor-normal paired patients.

Fig. C3 Resource usage of all variant calling processes in nf-core/sarek 3.1.1 when starting from BAM and CRAM format. **A)**: Average realtime, and maximum CPU and memory usage (peak_rss) as reported by the Nextflow trace file. For processes split within a sample, the task with the highest runtime per sample is shown as the process runtime. Resource usage was compared using the paired Wilcoxon test (** $p < 0.01$, * $p < 0.05$). Four of the 12 processes are significantly faster when using the CRAM format. Six have a significantly higher CPU hour usage, and seven require more memory in comparison to using BAMs. **B)**: Storage was evaluated by calculating the total size of the work directories of all tasks of the respective process. Six processes have reduced storage requirements. Each condition was repeated three times for samples of five tumor-normal paired patients.

Appendix D Sharding FastQ files reduces runtime

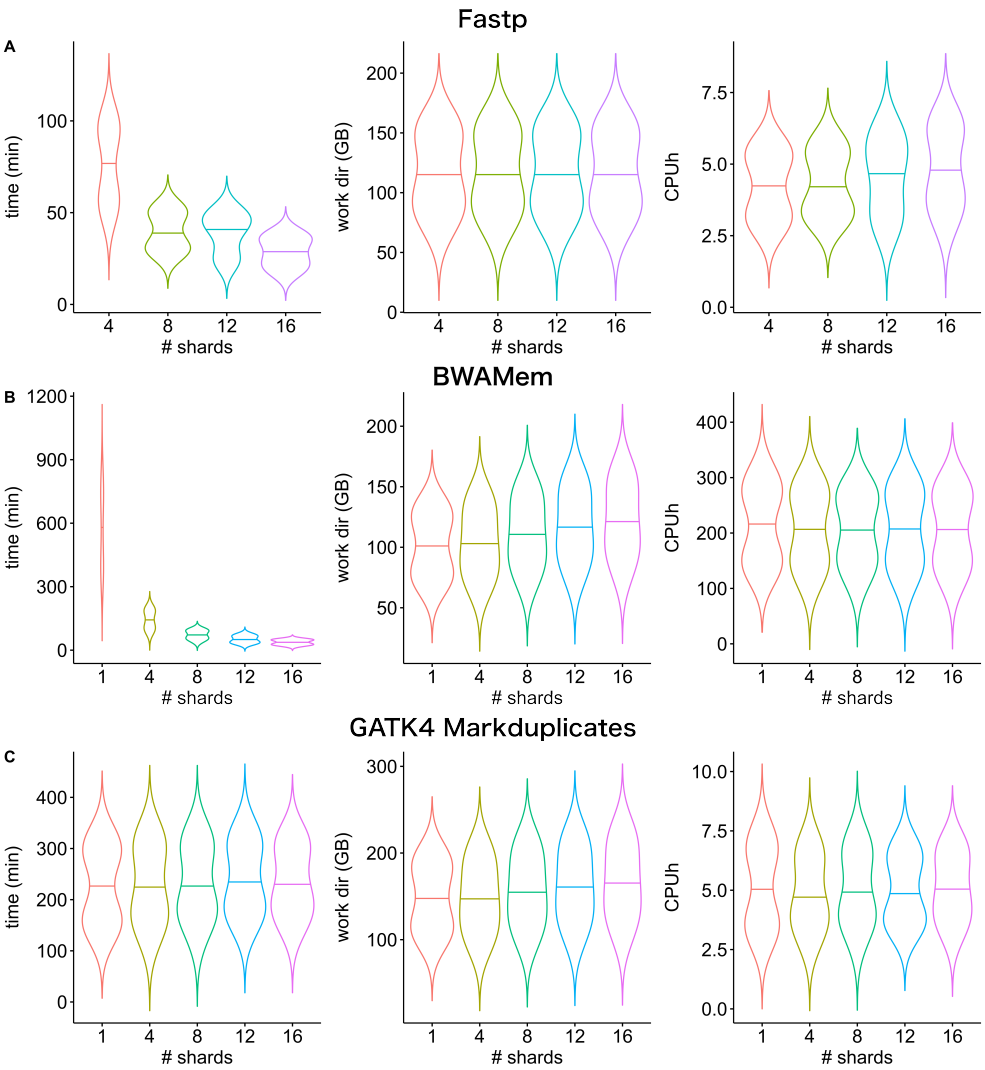

Fig. D4 Dividing the input FastQ files into increasing amounts of shards. **A)**: Resource usage of fastP during sharding of the input FastQ files. The tool was run with a different count of CPUs corresponding to the desired number of shards. **B)**: Resource usage of BWA-MEM during mapping of each shard. **C)**: Resource usage of the duplicate marking process. Merging of sharded bam files and duplicate marking is performed with GATK4 Markduplicates. CRAM conversion is done with SAMTools. The violin plots show computations on tumor-normal paired samples of five patients. The time was evaluated by summing up the highest realtime per task per sample as reported by the Nextflow trace report. The work directory size and CPU hours are the sums of all involved tasks.

Appendix E Splitting by intervals reduces runtime

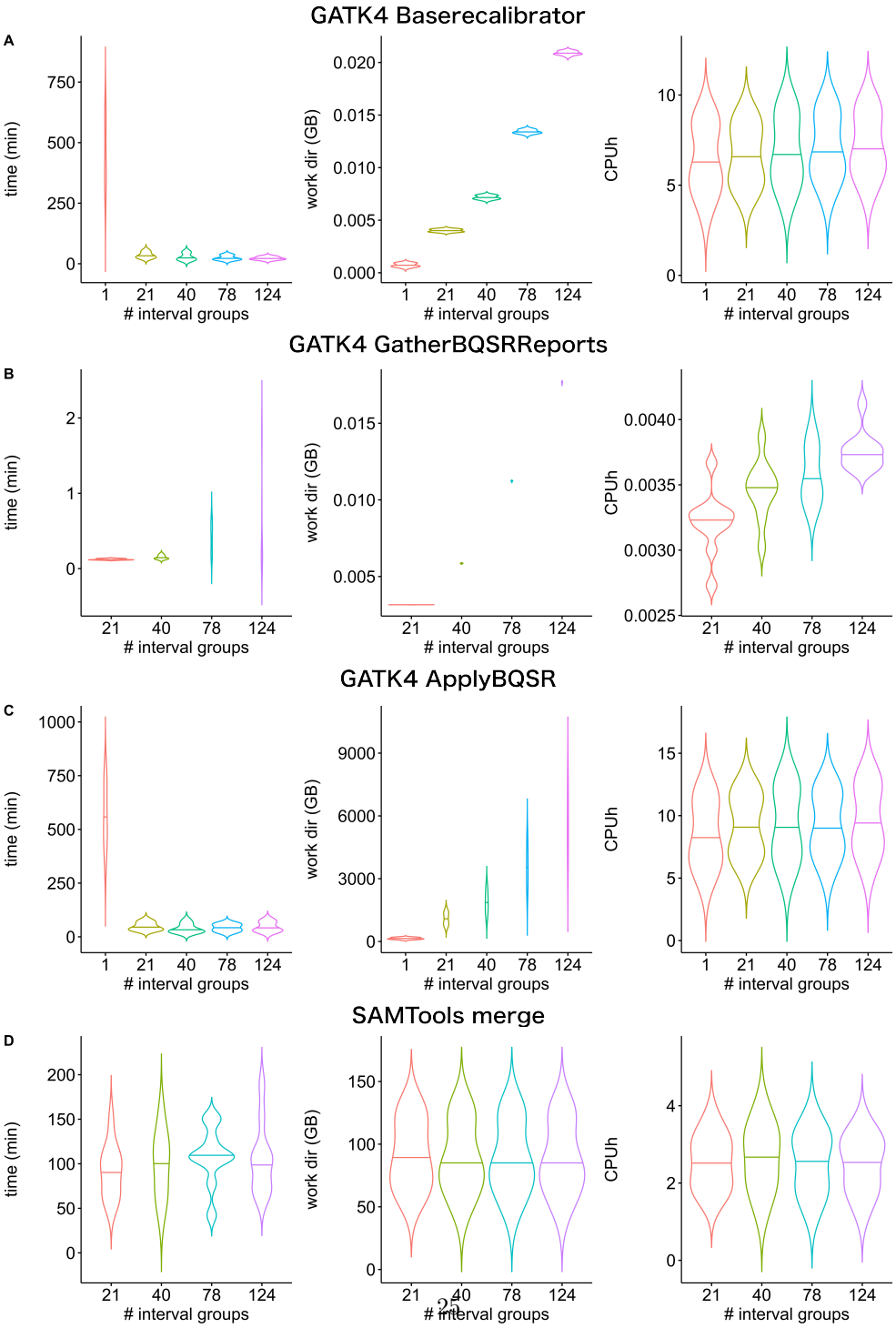

Fig. E5 Parallel processing of interval groups reduces runtime **A)**: GATK4 BaseRecalibrator runtime decreases with an increasing number of interval groups. Storage space requirements increase, while CPU hours stay consistent. **B)**: For GATK4 GatherBQSRReports all three metrics increase with an increasing number of interval groups. **C)**: For GATK4 ApplyBQSR the runtime decreases with an increasing number of interval groups. Storage space requirements increase, while CPU hours stay consistent. **D)**: For SAMTools Merge all three metrics are consistent across the number of interval groups. The violin plots show computations on tumor-normal paired samples of five patients for each tool. The time was evaluated by summing up the highest realtime per task per sample as reported by the Nextflow trace report. The work directory size and CPU hours are the sums of all involved tasks.

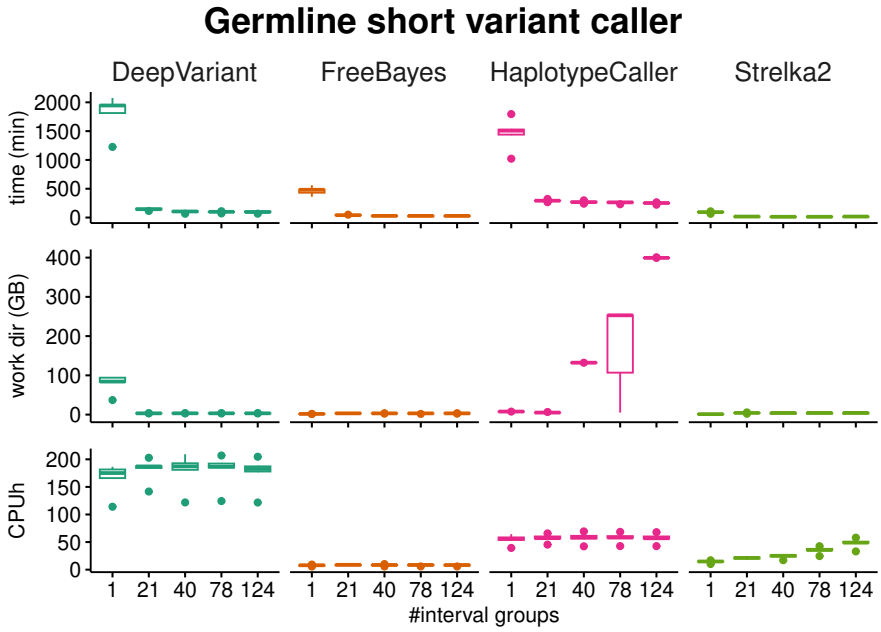

Fig. E6 Effect of parallelizing computations across interval groups on germline variant calling processes, which include the respective variant caller followed by GATK4 MergeVCFs. FreeBayes VCFs are sorted before merging. GATK4 HaplotypeCaller is followed by GATK4 CNNSCoreVariants and GATK4 FilterVariantTranches. All variant callers speed up on parallel processing across 21 interval groups. Storage usage decreases for DeepVariant for 21 interval groups and remains stable. FreeBayes and Strelka2 have similar storage usage across parallelization. For VCFs called by HaplotypeCaller storage usage increases. CPU hours are similar across degrees of parallelization with an increase measured for Strelka2. The violin plots show computations on normal samples of five patients. The time was evaluated by summing up the highest realtime per task per sample as reported by the Nextflow trace report. The work directory size and CPU hours are the sums of all involved tasks.

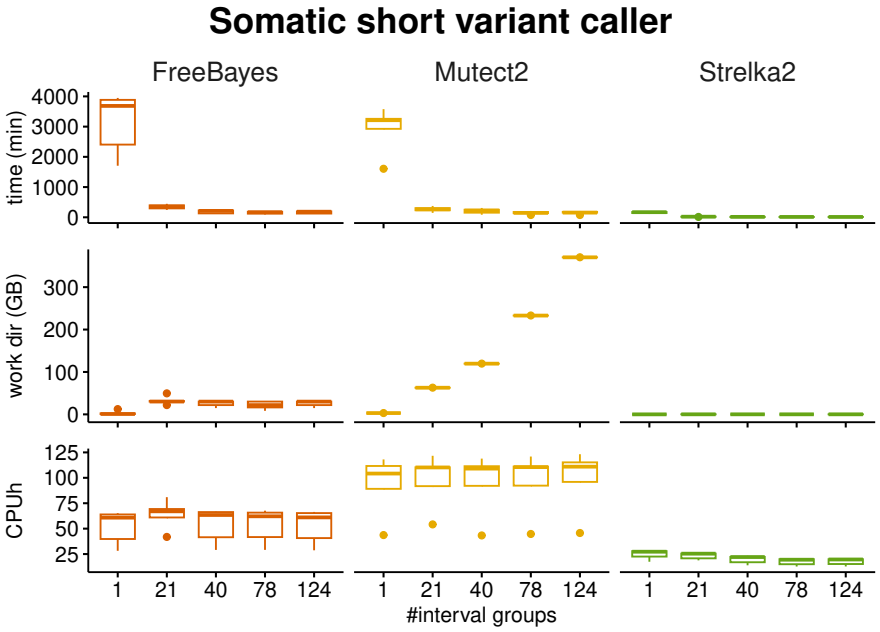

Fig. E7 Effect of parallelizing computations across interval groups on somatic variant calling processes, which include the respective variant caller followed by GATK4 MergeVCFs. FreeBayes VCFs are sorted before merging. All variant callers speed up on parallel processing across 21 interval groups. Storage usage increases for FreeBayes for 21 interval groups and remains stable. For Mutect2 storage usage increases with increasing interval groups. It remains stable for Strelka2. CPU hours are similar across degrees of parallelization with a decrease measured for Strelka2. The violin plots show computations on tumor-normal paired samples of five patients. The time was evaluated by summing up the highest realtime per task per sample as reported by the Nextflow trace report. The work directory size and CPU hours are the sums of all involved tasks.

Appendix F Benchmarking against truth datasets

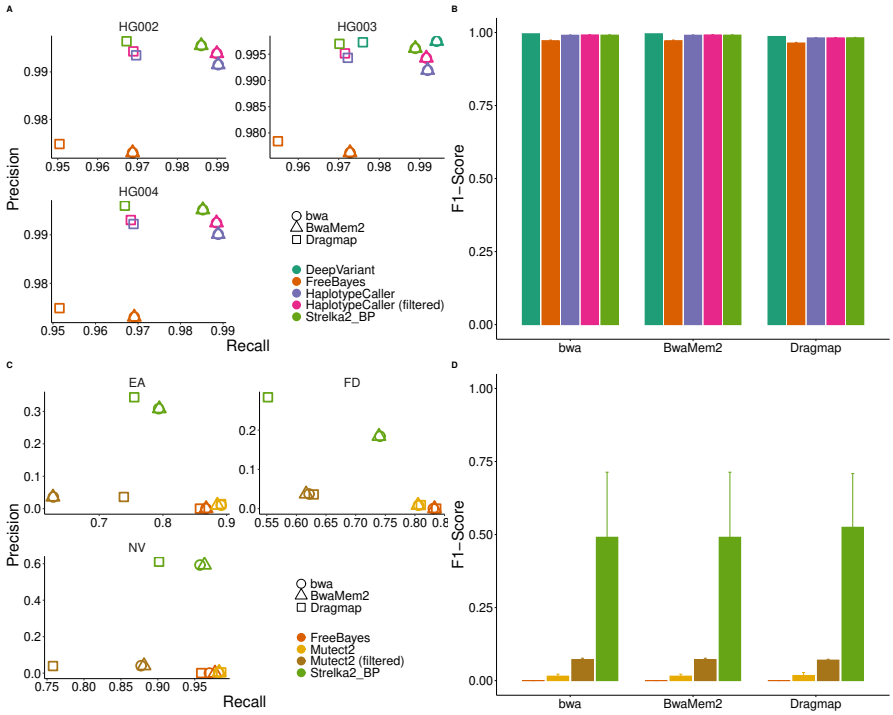

Fig. F8 Germline and somatic variant calling evaluation of high-confidence calls using ground-truth benchmarking data with respect to Indels. **A,B**: The germline track of the pipeline was evaluated using 3 WGS GiaB datasets (HG002-HG004). The average precision, recall and F1-score values across all the samples are plotted respectively. **C,D**: The paired calling track was evaluated using three tumor-normal WES pairs (EA, FD, NV) from SEQ2C.

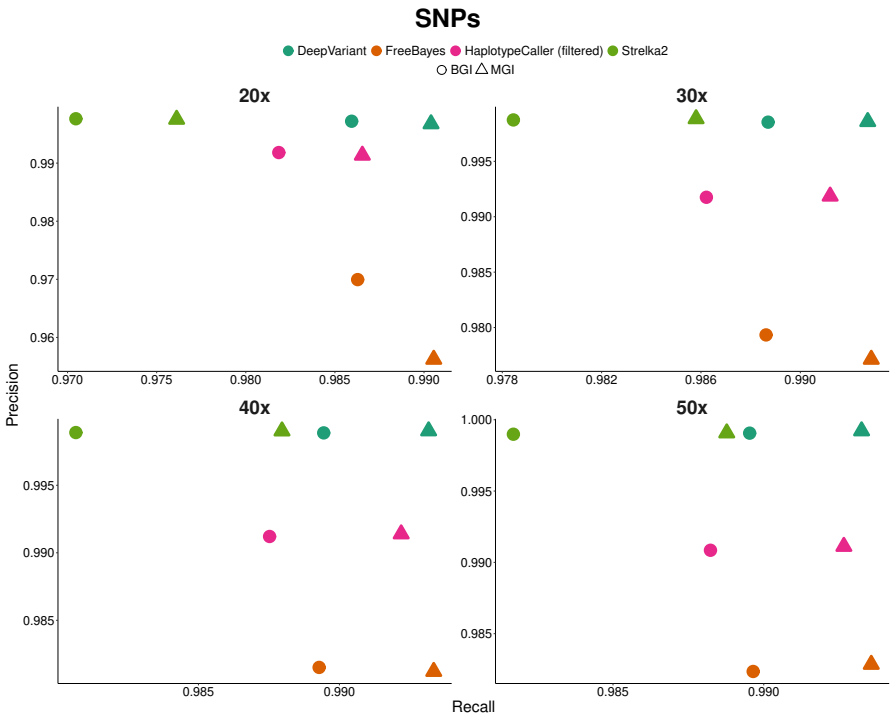

Fig. F9 Germline variant calling evaluation of high-confidence calls using ground-truth benchmarking data with respect to SNPs. Samples from MGISeq and BGISeq500 were mapped with BWA-MEM. Different coverages were used as input. For all investigated coverage values FreeBayes and DeepVariant have the highest recall. Strelka2 and DeepVariant show the highest precision values for all samples.

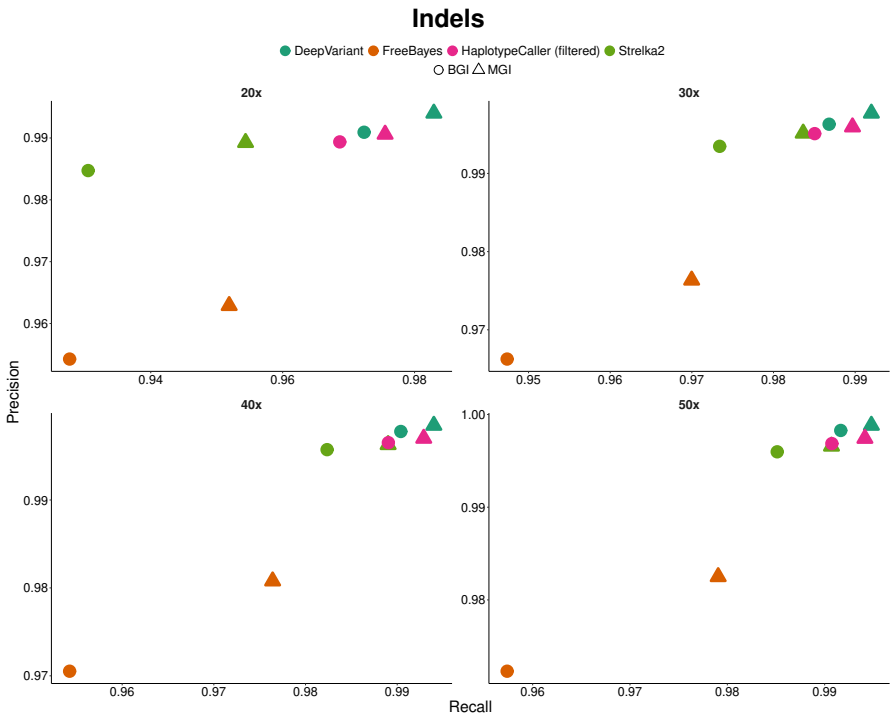

Fig. F10 Germline variant calling evaluation of high-confidence calls using ground-truth benchmarking data with respect to Indels. Samples from MGISeq and BGISeq500 were mapped with BWA-MEM. Different coverages were used as input. For all investigated coverage values HaplotypeCaller and DeepVariant have the highest recall, followed by Strelka2. MGI samples analyzed with DeepVariant had the highest precision values.

References

[1] Luchini, C., Lawlor, R. T., Milella, M. & Scarpa, A. Molecular Tumor Boards in Clinical Practice. *Trends in Cancer* **6**, 738–744 (2020). URL https://linkinghub.elsevier.com/retrieve/pii/S2405803320301631.

[2] Beaubier, N. *et al.* Integrated genomic profiling expands clinical options for patients with cancer. *Nature Biotechnology* **37**, 1351–1360 (2019). URL https://www.nature.com/articles/s41587-019-0259-z.

[3] Kato, S. *et al.* Real-world data from a molecular tumor board demonstrates improved outcomes with a precision N-of-One strategy. *Nature Communications* **11**, 4965 (2020). URL https://www.nature.com/articles/s41467-020-18613-3. Number: 1 Publisher: Nature Publishing Group.

[4] Morganti, S. *et al.* in *Role of Next-Generation Sequencing Technologies in Personalized Medicine* (eds Pravettoni, G. & Triberti, S.) *P5 eHealth: An Agenda for*

the Health Technologies of the Future 125–154 (Springer International Publishing, Cham, 2020). URL https://link.springer.com/10.1007/978-3-030-27994-3_8.

[5] Staaf, J. *et al.* Whole-genome sequencing of triple-negative breast cancers in a population-based clinical study. *Nature Medicine* **25**, 1526–1533 (2019). URL https://www.nature.com/articles/s41591-019-0582-4.

[6] Barroso-Sousa, R. *et al.* Prevalence and mutational determinants of high tumor mutation burden in breast cancer. *Annals of Oncology* **31**, 387–394 (2020). URL https://linkinghub.elsevier.com/retrieve/pii/S0923753419406984.

[7] Klein, C. J. & Foroud, T. M. Neurology Individualized Medicine: When to Use Next-Generation Sequencing Panels. *Mayo Clinic Proceedings* **92**, 292–305 (2017). URL https://linkinghub.elsevier.com/retrieve/pii/S0025619616305547.

[8] Suwinski, P. *et al.* Advancing Personalized Medicine Through the Application of Whole Exome Sequencing and Big Data Analytics. *Frontiers in Genetics* **10**, 49 (2019). URL https://www.frontiersin.org/article/10.3389/fgene.2019.00049/full.

[9] Lantos, J. D. in *Ethical and Psychosocial Issues in Whole-Genome Sequencing for Newborns* 295–300 (Elsevier, 2016). URL https://linkinghub.elsevier.com/retrieve/pii/B9780128017395000167.

[10] Martinez-Martin, N. & Magnus, D. Privacy and ethical challenges in next-generation sequencing. *Expert Review of Precision Medicine and Drug Development* **4**, 95–104 (2019). URL https://www.tandfonline.com/doi/full/10.1080/23808993.2019.1599685.

[11] The ICGC/TCGA Pan-Cancer Analysis of Whole Genomes Consortium *et al.* Pan-cancer analysis of whole genomes. *Nature* **578**, 82–93 (2020). URL https://www.nature.com/articles/s41586-020-1969-6.

[12] Degasperi, A. *et al.* Substitution mutational signatures in whole-genome–sequenced cancers in the UK population. *Science* **376**, abl9283 (2022). URL https://www.science.org/doi/10.1126/science.abl9283.

[13] Gudbjartsson, D. F. *et al.* Large-scale whole-genome sequencing of the Icelandic population. *Nature Genetics* **47**, 435–444 (2015). URL https://www.nature.com/articles/ng.3247.

[14] Tanjo, T., Kawai, Y., Tokunaga, K., Ogasawara, O. & Nagasaki, M. Practical guide for managing large-scale human genome data in research. *Journal of Human Genetics* **66**, 39–52 (2021). URL https://www.nature.com/articles/s10038-020-00862-1.

[15] Garcia, M. *et al.* Sarek: A portable workflow for whole-genome sequencing analysis of germline and somatic variants. *F1000Research* **9**, 63 (2020). URL https:

//f1000research.com/articles/9-63/v2.

[16] Bathke, J. & Lühken, G. OVarFlow: a resource optimized GATK 4 based Open source Variant calling workFlow. *BMC Bioinformatics* **22**, 402 (2021). URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-021-04317-y.

[17] Cokelaer, T., Desvillechabrol, D., Legendre, R. & Cardon, M. 'Sequana': a Set of Snakemake NGS pipelines. *The Journal of Open Source Software* **2**, 352 (2017). URL http://joss.theoj.org/papers/10.21105/joss.00352.

[18] Del Corvo, M., Mazzara, S. & Pileri, S. A. TOSCA: an automated Tumor Only Somatic CAlling workflow for somatic mutation detection without matched normal samples. *Bioinformatics Advances* **2**, vbac070 (2022). URL https://academic.oup.com/bioinformaticsadvances/article/doi/10.1093/bioadv/vbac070/6717791.

[19] Köster, J. & Rahmann, S. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics* **28**, 2520–2522 (2012). URL https://academic.oup.com/bioinformatics/article/28/19/2520/290322.

[20] Di Tommaso, P. *et al.* Nextflow enables reproducible computational workflows. *Nature Biotechnology* **35**, 316–319 (2017). URL https://www.nature.com/articles/nbt.3820.

[21] Ewels, P. A. *et al.* The nf-core framework for community-curated bioinformatics pipelines. *Nature Biotechnology* **38**, 276–278 (2020). URL https://www.nature.com/articles/s41587-020-0439-x.

[22] Røssevold, A. H. *et al.* Atezolizumab plus anthracycline-based chemotherapy in metastatic triple-negative breast cancer: the randomized, double-blind phase 2b ALICE trial. *Nature Medicine* **28**, 2573–2583 (2022). URL https://www.nature.com/articles/s41591-022-02126-1.

[23] Strand, S. H. *et al.* Molecular classification and biomarkers of clinical outcome in breast ductal carcinoma in situ: Analysis of TBCRC 038 and RAHBT cohorts. *Cancer Cell* **40**, 1521–1536.e7 (2022). URL https://linkinghub.elsevier.com/retrieve/pii/S1535610822005128.

[24] Elez, E. *et al.* RNF43 mutations predict response to anti-BRAF/EGFR combinatory therapies in BRAFV600E metastatic colorectal cancer. *Nature Medicine* **28**, 2162–2170 (2022). URL https://www.nature.com/articles/s41591-022-01976-z.

[25] Peña-Pérez, L. *et al.* Linked-read whole-genome sequencing resolves common and private structural variants in multiple myeloma. *Blood Advances* **6**, 5009–5023 (2022). URL https://ashpublications.org/bloodadvances/article/6/17/5009/485485/Linked-read-whole-genome-sequencing-resolves.

[26] Erickson, A. *et al.* Spatially resolved clonal copy number alterations in benign and malignant tissue. *Nature* **608**, 360–367 (2022). URL https://www.nature.com/articles/s41586-022-05023-2.

[27] Díaz De Ståhl, T. *et al.* The Swedish childhood tumor biobank: systematic collection and molecular characterization of all pediatric CNS and other solid tumors in Sweden. *Journal of Translational Medicine* **21**, 342 (2023). URL https://translational-medicine.biomedcentral.com/articles/10.1186/s12967-023-04178-4.

[28] Gallego-Martinez, A. *et al.* Using coding and non-coding rare variants to target candidate genes in patients with severe tinnitus. *npj Genomic Medicine* **7**, 70 (2022). URL https://www.nature.com/articles/s41525-022-00341-w.

[29] Kang, B. H. *et al.* Transcriptome Analysis of Differentially Expressed Genes Associated with Salt Stress in Cowpea (Vigna unguiculata L.) during the Early Vegetative Stage. *International Journal of Molecular Sciences* **24**, 4762 (2023). URL https://www.mdpi.com/1422-0067/24/5/4762.

[30] Kardum Hjort, C. *et al.* Genomic divergence and a lack of recent introgression between commercial and wild bumblebees (*Bombus terrestris*). *Evolutionary Applications* **15**, 365–382 (2022). URL https://onlinelibrary.wiley.com/doi/10.1111/eva.13346.

[31] Guerra-Assunção, J. A. *et al.* GenomeChronicler: The Personal Genome Project UK Genomic Report Generator Pipeline. *Frontiers in Genetics* **11**, 518644 (2020). URL https://www.frontiersin.org/article/10.3389/fgene.2020.518644/full.

[32] Poplin, R. *et al.* A universal SNP and small-indel variant caller using deep neural networks. *Nature Biotechnology* **36**, 983–987 (2018). URL https://www.nature.com/articles/nbt.4235.

[33] Chen, S. Ultrafast one-pass FASTQ data preprocessing, quality control, and deduplication using fastp. *iMeta* **2**, e107 (2023). URL https://onlinelibrary.wiley.com/doi/10.1002/imt2.107.

[34] DePristo, M. A. *et al.* A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature Genetics* **43**, 491–498 (2011). URL https://www.nature.com/articles/ng.806.

[35] Van der Auwera, G. A. & O'Connor, B. D. *Genomics in the cloud: using Docker, GATK, and WDL in Terra* (O'Reilly Media, 2020).

[36] Danecek, P. *et al.* Twelve years of SAMtools and BCFtools. *GigaScience* **10**, giab008 (2021). URL https://academic.oup.com/gigascience/article/doi/10.1093/gigascience/giab008/6137722.

[37] Kou, R. *et al.* Benefits and Challenges with Applying Unique Molecular Identifiers in Next Generation Sequencing to Detect Low Frequency Mutations. *PLOS ONE* **11**, e0146638 (2016). URL https://dx.plos.org/10.1371/journal.pone.0146638.

[38] Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM (2013). URL https://arxiv.org/abs/1303.3997. Publisher: arXiv Version Number: 2.

[39] Vasimuddin, M., Misra, S., Li, H. & Aluru, S. *Efficient Architecture-Aware Acceleration of BWA-MEM for Multicore Systems*, 314–324 (IEEE, Rio de Janeiro, Brazil, 2019). URL https://ieeexplore.ieee.org/document/8820962/.

[40] Talevich, E., Shain, A. H., Botton, T. & Bastian, B. C. CNVkit: Genome-Wide Copy Number Detection and Visualization from Targeted DNA Sequencing. *PLOS Computational Biology* **12**, e1004873 (2016). URL https://dx.plos.org/10.1371/journal.pcbi.1004873.

[41] Eisfeldt, J., Vezzi, F., Olason, P., Nilsson, D. & Lindstrand, A. TIDDIT, an efficient and comprehensive structural variant caller for massive parallel sequencing data. *F1000Research* **6**, 664 (2017). URL https://f1000research.com/articles/6-664/v2.

[42] Poplin, R. *et al.* Scaling accurate genetic variant discovery to tens of thousands of samples. preprint, Genomics (2017). URL http://biorxiv.org/lookup/doi/10.1101/201178.

[43] Kim, S. *et al.* Strelka2: fast and accurate calling of germline and somatic variants. *Nature Methods* **15**, 591–594 (2018). URL https://www.nature.com/articles/s41592-018-0051-x.

[44] Chen, X. *et al.* Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics* **32**, 1220–1222 (2016). URL https://academic.oup.com/bioinformatics/article/32/8/1220/1743909.

[45] McLaren, W. *et al.* The Ensembl Variant Effect Predictor. *Genome Biology* **17**, 122 (2016). URL http://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-0974-4.

[46] 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 w ^{1118} ; iso-2; iso-3. *Fly* **6**, 80–92 (2012). URL http://www.tandfonline.com/doi/abs/10.4161/fly.19695.

[47] Kircher, M. *et al.* A general framework for estimating the relative pathogenicity of human genetic variants. *Nature Genetics* **46**, 310–315 (2014). URL https://www.nature.com/articles/ng.2892.

[48] Liu, X., Li, C., Mou, C., Dong, Y. & Tu, Y. dbNSFP v4: a comprehensive database of transcript-specific functional predictions and annotations for human nonsynonymous and splice-site SNVs. *Genome Medicine* **12**, 103 (2020). URL https://genomemedicine.biomedcentral.com/articles/10.1186/s13073-020-00803-9.

[49] Karczewski, K. J. *et al.* The mutational constraint spectrum quantified from variation in 141,456 humans. *Nature* **581**, 434–443 (2020). URL https://www.nature.com/articles/s41586-020-2308-7.

[50] Jaganathan, K. *et al.* Predicting Splicing from Primary Sequence with Deep Learning. *Cell* **176**, 535–548.e24 (2019). URL https://linkinghub.elsevier.com/retrieve/pii/S0092867418316295.

[51] Pedersen, B. S. & Quinlan, A. R. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics* **34**, 867–868 (2018). URL https://academic.oup.com/bioinformatics/article/34/5/867/4583630.

[52] Danecek, P. *et al.* The variant call format and VCFtools. *Bioinformatics* **27**, 2156–2158 (2011). URL https://academic.oup.com/bioinformatics/article/27/15/2156/402296.

[53] Ewels, P., Magnusson, M., Lundin, S. & Käller, M. MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32**, 3047–3048 (2016). URL https://academic.oup.com/bioinformatics/article/32/19/3047/2196507.

[54] Okonechnikov, K., Conesa, A. & García-Alcalde, F. Qualimap 2: advanced multi-sample quality control for high-throughput sequencing data. *Bioinformatics* **32**, 292–294 (2016). URL https://academic.oup.com/bioinformatics/article/32/2/292/1744356.

[55] Thorvaldsdottir, H., Robinson, J. T. & Mesirov, J. P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Briefings in Bioinformatics* **14**, 178–192 (2013). URL https://academic.oup.com/bib/article-lookup/doi/10.1093/bib/bbs017.

[56] Zook, J. M. *et al.* An open resource for accurately benchmarking small variant and reference calls. *Nature Biotechnology* **37**, 561–566 (2019). URL https://www.nature.com/articles/s41587-019-0074-6.

[57] Fang, L. T. *et al.* Establishing community reference samples, data and call sets for benchmarking cancer mutation detection using whole-genome sequencing. *Nature Biotechnology* **39**, 1151–1160 (2021). URL https://www.nature.com/articles/s41587-021-00993-6.

[58] Olson, N. D. precisionFDA Truth Challenge V2: Calling variants from short- and long-reads in difficult-to-map regions (2020). URL https://data.nist.gov/od/id/

mds2-2336. Artwork Size: 401 files, 1.16 TB Pages: 401 files, 1.16 TB.

[59] Xiao, W. *et al.* Toward best practice in cancer mutation detection with whole-genome and whole-exome sequencing. *Nature Biotechnology* **39**, 1141–1150 (2021). URL https://www.nature.com/articles/s41587-021-00994-5.

[60] The Bioconda Team *et al.* Bioconda: sustainable and comprehensive software distribution for the life sciences. *Nature Methods* **15**, 475–476 (2018). URL https://www.nature.com/articles/s41592-018-0046-7.

[61] Da Veiga Leprevost, F. *et al.* BioContainers: an open-source and community-driven framework for software standardization. *Bioinformatics* **33**, 2580–2582 (2017). URL https://academic.oup.com/bioinformatics/article/33/16/2580/3096437.

[62] Schulze, K. *et al.* Exome sequencing of hepatocellular carcinomas identifies new mutational signatures and potential therapeutic targets. *Nature Genetics* **47**, 505–511 (2015). URL https://www.nature.com/articles/ng.3252.

[63] Cleary, J. G. *et al.* Comparing Variant Call Files for Performance Benchmarking of Next-Generation Sequencing Variant Calling Pipelines. preprint, Bioinformatics (2015). URL http://biorxiv.org/lookup/doi/10.1101/023754.

Table B1 The updates since nf-core/sarek 2.5.2 are summarized in this table: most notably adapter trimming and UMI support were added, and read splitting was optimized and enabled for large FastQ files by using fastP. Merging of aligned BAM files across splits and lanes was incorporated into the duplicate marking process. Furthermore, individual steps allow a wider tool selection, i.e. BWA-MEM2 and Dragmap for mapping and further tools for variant calling. Sentieon support for all steps will be added with release 3.3.

	Feature	Sarek 2.5.2	Sarek 3.1.1
pre-processing	uBAM conversion	GATK4 SamToFastq	SAMtools
	FastQ splitting	Nextflow `SplitFastq()`	fastP
	UMI support	-	fgbio
	Trimming	-	fastP
	Mapping	BWA-MEM	BWA-MEM, BWA-MEM2, Dragmap
	Merging of mapped files	SAMtools	GATK4 Markduplicates(Spark), SAMtools
	Duplicate marking	GATK4 Markduplicates	GATK4 Markduplicates(Spark)
	Quality control	FastQC, SAMtools, Qualimap	FastQC, fastP, SAMtools, mosdepth, GATK4 EstimateLibraryComplexity
Variant calling	Germline	Haplotypecaller, Manta, Mpileup, Strelka2, Tiddit	CNVKit, DeepVariant, FreeBayes, Haplotypecaller (extended single-sample filtering and join-germline), Manta, Mpileup, Strelka, Tiddit
	Tumor-only	Manta, Strelka, Tiddit	FreeBayes, Manta, Mutect2, Strelka2, Tiddit
	Paired	ASCAT, ControlFreec, FreeBayes, Manta, Mutect2, Strelka2	ASCAT, CNVKit, ControlFreec, FreeBayes, Manta, MsiSensorPro, Mutect2, Strelka2, Tiddit
	Merging	Custom script with bcftools	
Annotation		VEP CADD plugin	VEP dbnsfp, Loftee, spliceAI, spliceRegion plugin
Other	Sentieon support	yes	planned for 3.3