G3, 2024, **14(1)**, jkad246
https://doi.org/10.1093/g3journal/jkad246
Advance Access Publication Date: 17 November 2023
Software and Data Resources

kGWASflow: a modular, flexible, and reproducible Snakemake workflow for k-mers-based GWAS

Adnan Kivanc Corut [1,*], Jason G. Wallace[1,2,3]

[1]Institute of Bioinformatics, University of Georgia, Athens, GA 30602, USA
[2]Institute of Plant Breeding, Genetics, and Genomics, University of Georgia, Athens, GA 30602, USA
[3]Department of Crop and Soil Sciences, University of Georgia, Athens, GA 30602, USA

*Corresponding author: Institute of Bioinformatics, University of Georgia, Athens, GA 30602, USA. Email: kivanc.corut@uga.edu

Genome-wide association studies (GWAS) have been widely used to identify genetic variation associated with complex traits. Despite its success and popularity, the traditional GWAS approach comes with a variety of limitations. For this reason, newer methods for GWAS have been developed, including the use of pan-genomes instead of a reference genome and the utilization of markers beyond single-nucleotide polymorphisms, such as structural variations and k-mers. The k-mers-based GWAS approach has especially gained attention from researchers in recent years. However, these new methodologies can be complicated and challenging to implement. Here, we present kGWASflow, a modular, user-friendly, and scalable workflow to perform GWAS using k-mers. We adopted an existing kmersGWAS method into an easier and more accessible workflow using management tools like Snakemake and Conda and eliminated the challenges caused by missing dependencies and version conflicts. kGWASflow increases the reproducibility of the kmersGWAS method by automating each step with Snakemake and using containerization tools like Docker. The workflow encompasses supplemental components such as quality control, read-trimming procedures, and generating summary statistics. kGWASflow also offers post-GWAS analysis options to identify the genomic location and context of trait-associated k-mers. kGWASflow can be applied to any organism and requires minimal programming skills. kGWASflow is freely available on GitHub (https://github.com/akcorut/kGWASflow) and Bioconda (https://anaconda.org/bioconda/kgwasflow).

Keywords: GWAS; k-mers; snakemake; bioinformatics tool; pipeline

Introduction

Identifying genotype–phenotype associations is fundamental to understanding the genetic architecture of complex traits. Genome-wide association studies (GWAS) have been the method of choice to detect associations between genetic variants and phenotypes for over 15 years (Visscher *et al.* 2017). Through GWAS, thousands of traits have been surveyed, and numerous statistically significant associations have been reported (MacArthur *et al.* 2017; Uffelmann *et al.* 2021). These findings resulted in a better understanding of complex human traits and diseases (Cano-Gamez and Trynka 2020), helped improve plant breeding (Tibbs Cortes *et al.* 2021) and animal health (Tian *et al.* 2020), and have otherwise significantly impacted our understanding of genetics.

The classical GWAS approach uses genome-wide single-nucleotide polymorphisms (SNPs) as the genotype data. During a standard GWAS, SNP markers are tested for statistically significant association with the phenotypic trait using statistical models. GWAS utilizes linkage disequilibrium (LD) information between markers and causal variants to identify trait-associated loci. However, despite its power and success, this method comes with various limitations (Gupta *et al.* 2019; Tam *et al.* 2019; Sun *et al.* 2021). It has been previously shown that GWAS often fails to pinpoint causal variants due to linkage disequilibrium (Faye *et al.* 2013; Boyle *et al.* 2017; LaPierre *et al.* 2021). Additionally, these studies primarily rely on SNPs as markers of genetic variants and often ignore other variants, such as structural variations, and therefore can sometimes explain only a fraction of heritability, particularly in cases involving highly complex traits (Manolio *et al.* 2009; Nolte *et al.* 2017). GWAS can also identify spurious associations (Sul *et al.* 2018) and fail to capture associations caused by rare variants (Wray *et al.* 2011; Young 2019).

The quality of GWAS also depends on the availability and the quality of reference genomes. Traditional GWAS relies on mapping sequencing reads to a reference genome and then calling variants. This mapping step can potentially cause biases during variant calling because reference genomes are frequently incomplete and may not represent the full spectrum of genetic variation within a population. In addition, the misalignments can result in incorrect variant calling, especially in complex genomes and/or around repetitive regions.

Due to these limitations, newer GWAS methods have been developed (Coletta *et al.* 2021; Gupta 2021). These newer methods include but are not limited to using pan-genomes instead of a single reference genome (Manuweera *et al.* 2019; Song *et al.* 2020; Zhou *et al.* 2022) and the usage of new markers beyond SNPs, such as structural variations (Prinsen *et al.* 2017; Zhou *et al.* 2018; Yang *et al.* 2019; Li *et al.* 2020; Göktay *et al.* 2021; Qin *et al.* 2021; Wei *et al.* 2021) and *k*-mers (Rahman *et al.* 2018; Voichek and Weigel 2020; He *et al.* 2021; Mehrab *et al.* 2021; Lemane *et al.* 2022). Using

Received on 07 July 2023; accepted on 15 October 2023

structural variations may capture some of the missing heritability (Génin 2020; Theunissen *et al.* 2020; Zhou *et al.* 2022). Furthermore, utilizing k-mers as genetic markers offers significant advantages compared to the traditional SNP-based approach. *k*-mers, substrings of length *k* in sequencing reads, can mark a broader range of genomic variants, including structural variations. k-mers-based GWAS also allows a reference-free association mapping and can identify trait-associated markers even in regions missing in the reference genome (Gupta 2021).

Voichek and Weigel recently developed a k-mers-based GWAS approach for both categorical and quantitative phenotypes (Voichek and Weigel 2020). In this approach, the authors use the presence or absence of k-mers in sequencing reads as genotypic variants and then apply traditional GWAS methods. Even though the k-mers-based GWAS method recently gained increased attention (Colque-Little *et al.* 2021; Tripodi *et al.* 2021; Kale *et al.* 2022; Li *et al.* 2022; Onetto *et al.* 2022; Schulthess *et al.* 2022), it has not reached its potential due to the difficulties in implementing it. These difficulties include the need for bioinformatics expertise and lack of user-friendly tools. Underlying software and library dependencies also have the potential to cause reproducibility issues. Moreover, the existing implementation of this method lacks the essential downstream analyses necessary for interpreting results from the k-mers-based GWAS approach.

In an effort to tackle these challenges, here we present kGWASflow, a modular, user-friendly, and scalable workflow to perform k-mers-based GWAS. kGWASflow adapts the approach by Voichek et al. (2020), creating a more user-friendly workflow while addressing software dependency and version conflict issues. Our workflow is highly deployable in high-performance and cloud computing environments. It also enhances the interpretation of k-mer-based GWAS findings by providing supplementary downstream analysis options. kGWASflow boasts extensive customization, providing users with a variety of options tailored to their specific requirements.

Methods

Overview

The overall workflow of kGWASflow is described in Fig. 1. With the default settings, kGWASflow comprises three main phases: preprocessing, k-mers-based GWAS (Voichek and Weigel 2020), and post-GWAS analysis. In short, the preprocessing phase conducts quality control analysis, offers optional read trimming, and organizes the input files for downstream analysis. The workflow's second and main phase performs k-mers-based GWAS by implementing the kmersGWAS method from Voichek et al. (2020) (Voichek and Weigel 2020). The post-GWAS phase generates results tables and provides multiple options to identify genomic locations and context of trait-associated *k*-mers. Lastly, kGWASflow generates an HTML report that includes QC and summary statistics, diagnostic plots, and kmersGWAS results. kGWASflow is highly customizable, as multiple steps of the workflow are optional and can be easily deactivated. kGWASflow is written in Snakemake (Mölder *et al.* 2021), a commonly used Python-based workflow engine.

Snakemake allows the workflow to be highly modular, scalable, and reproducible. Utilizing Snakemake, the workflow is constructed through a set of rules that link input sets with their corresponding outputs. Snakemake establishes the execution order by discerning the optimal combination of rules necessary to produce the target output while ensuring that each rule is triggered only upon the availability of its input files. Snakemake enhances the parallelization of independent jobs, considering computational resources and the thread requirements for each job's execution. This feature enables automatic scalability, allowing users to effectively run the workflow on local machines and high-performance and cloud-based computing platforms. Another aspect of Snakemake is that it recognizes which output files are missing if the execution of the pipeline is halted due to an error and allows users to recover the execution of failed jobs and continue after the issues are resolved. By combining Snakemake with the Conda software manager, kGWASflow automatically installs all software and library dependencies for each rule separately in a Conda environment. In doing so, kGWASflow effectively addresses potential complications arising from software dependency conflicts across various stages of the pipeline. kGWASflow also takes advantage of Conda for the initial deployment of the workflow environment. The latest version of the workflow and its dependencies can be easily installed and activated by running the command

```
conda create -c bioconda \
  --name kgwasflow kgwasflow
conda activate kgwasflow
```

Input

The workflow has two main inputs: paired-end FASTQ files and single or multiple phenotype files. FASTQ files contain the sequencing reads of each individual/sample, and each individual/sample can have multiple units of FASTQ files obtained from different sequencing runs of the same sample. A separate phenotype file for each phenotype tested needs to be provided by the user. This phenotype file consists of two columns where the first column represents the name of the individual/sample and the second column represents the phenotypic value of that corresponding sample, as described in Voichek et al. (2020) (Voichek and Weigel 2020).

By configuring a single samples sheet (samples.tsv, Supplementary Data S5), users can easily supply all necessary sample information for the pipeline, including sample names, FASTQ file paths, or SRA accessions. Users have the option to either specify the local file path for each FASTQ file or supply sequencing read archive (SRA) accessions for each sample, considering individual sequencing runs for the same sample separately. When provided with only SRA accessions, the pipeline automatically retrieves the relevant FASTQ files for each sample and its associated sequencing run using fasterq-dump. Phenotype file information can be provided by using the phenotype sheet (phenos.tsv, Supplementary Data S6). In the phenotype sheet, users specify the phenotype name in the first column and the file path of the corresponding phenotype file in the second column.

Workflow configuration and execution

KGWASflow streamlines the workflow configuration process and provides users with easy customization. Once the workflow is installed, a new kGWASflow working directory with default configuration files can be initialized with a single command.

```
kgwasflow init --work-dir working_dir/
```

This command will generate the configuration file "config.yaml" (Supplementary Data S3) alongside tab-separated sample and phenotype sheets (explained in the Input section) inside of the config directory. The command will also generate the "test" directory containing all the essential files required for executing a test

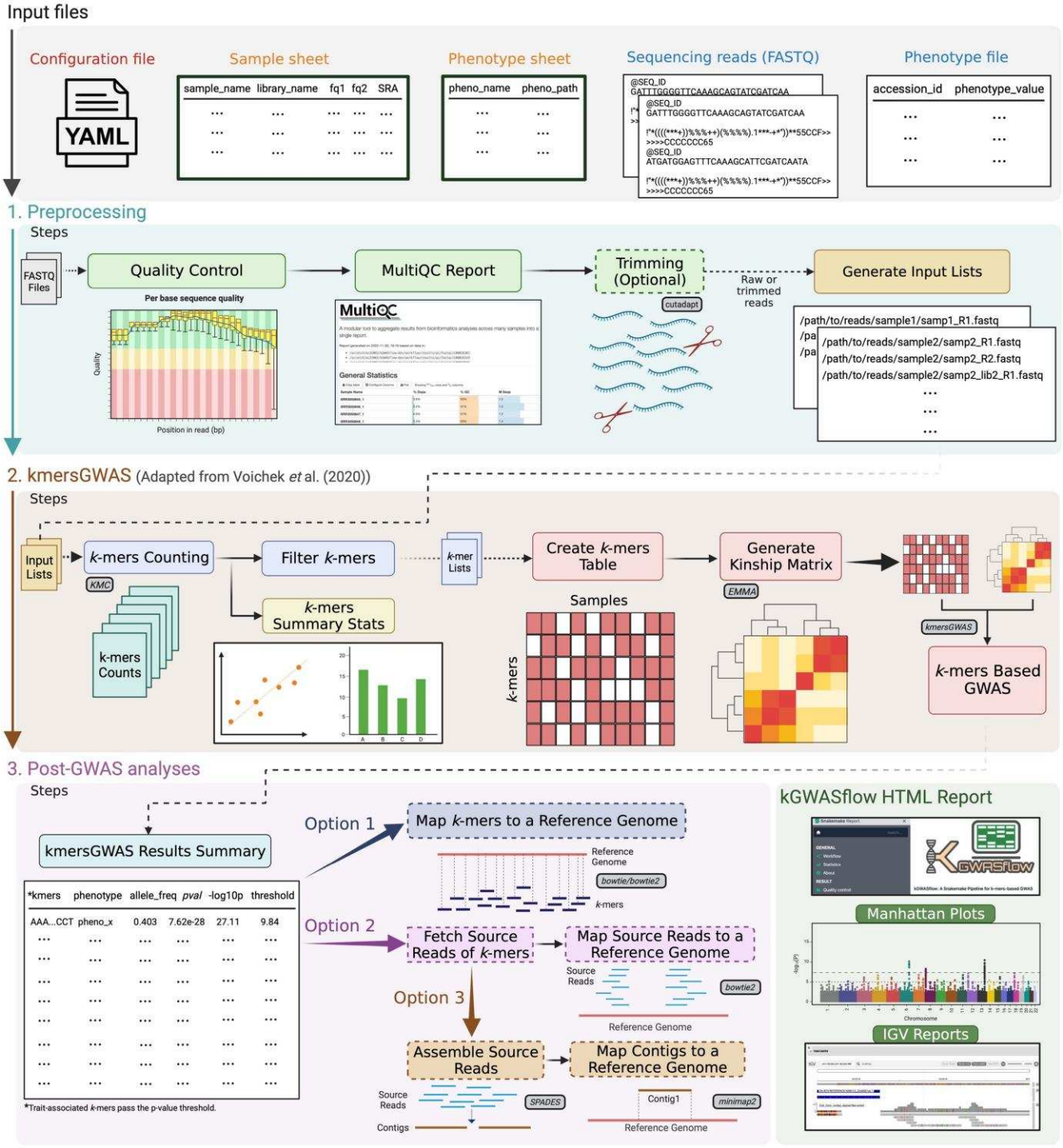

Fig. 1. Overview of the kGWASflow workflow. The default kGWASflow workflow consists of three main phases: Preprocessing (1st step), k-mers-based GWAS (2nd step), and post-GWAS analyses (3rd step). The configuration and input files required by the workflow are indicated at the top. The final outputs and report of the workflow are outlined in the bottom right corner. Small boxes with solid outlines and shaded fill denote the publicly available tools employed in the workflow. The workflow steps are customizable, with multiple optional steps, such as read trimming and post-GWAS analysis options (options 1, 2, and 3). These optional steps are in dashed boxes.

run of kGWASflow. An example directory structure initialized by `init` command can be found in Fig. 2.

To configure the workflow, the user simply needs to modify the "config.yaml" configuration file, which is composed of three main sections: input information, workflow settings, and tool parameters. In the first section, the user defines the paths for the sample and phenotype sheets and provides details regarding the reference genome, if required. The workflow settings section allows users to control optional pipeline steps, such as read trimming, by changing them to "True" or "False" to activate or deactivate. In this section, users can also specify their preferred post-GWAS analysis options (Fig. 1). Finally, in the tool parameters section, the users can define the parameters/settings for each tool used in the workflow.

```
kgwasflow-workdir/
├── config/
│   ├── config.yaml
│   ├── phenos.tsv
│   └── samples.tsv
└── test/
    ├── config_ecoli/
    │   ├── config.yaml
    │   ├── phenos.tsv
    │   └── samples.tsv
    ├── config_test/
    │   ├── ...
    └── data/
        ├── ecoli_phenos/
        │   └── resistence.pheno
        ├── ecoli_ref/
        │   └── CP026474.1.fa
        ├── test_reads/
        │   ├── ...
        ├── test_phenos/
        │   ├── ...
        └── test_ref/
```

Fig. 2. Example working directory structure generated by kGWASflow initialization command: `kgwasflow init`. This working directory contains the default kGWASflow configuration files and the test/ directory with all the essential files required for a test workflow run.

After the configuration, kGWASfow can be run with a single command

```
kgwasflow run --threads 16 \
  --work-dir working_dir
```

This command first initiates the installation of all necessary software and library dependencies for the workflow via Conda before executing the workflow steps. The execution order is determined by Snakemake, according to the user-defined configuration file and the dependencies between the workflow steps.

Alternatively, the workflow can be implemented by cloning the GitHub repository and executing `snakemake` command after the configuration of the workflow. The configuration can be done by manually configuring the configuration files within the cloned repository. A detailed guide on how to install, configure, and use the pipeline via Conda or GitHub can be found on the kGWASflow wiki page (https://github.com/akcorut/kGWASflow/wiki).

Workflow steps

Preprocessing

The preprocessing phase of kGWASflow starts with the quality control (QC) analysis of raw sequencing reads. kGWASflow first uses FastQC (Andrews 2010) to generate basic QC metrics, such as quality scores, GC content, duplication levels, and adapter content, for each FASTQ file. The pipeline then uses MultiQC (Ewels *et al.* 2016) to summarize and visualize each FastQC report in a single HTML file. If the trimming setting is activated, kGWASflow performs read trimming for each FASTQ file using cutadapt (Martin 2011). Users can define the parameters for read trimming in the configuration file. If the trimming setting is not activated, the pipeline skips this step and uses the raw reads for the rest of the workflow. During the final stage of the preprocessing phase, kGWASflow sorts the FASTQ files (trimmed or raw) into individual folders based on the sample or individual name, with one folder designated for each. Subsequently, within each folder, a text file is generated, listing the file paths of all FASTQ files associated with that particular sample. This last step is required to run *k*-mers-based GWAS.

k-mers-based GWAS

Once the preprocessing phase is complete, kGWASflow starts the *k*-mers counting step. KMC (Kokot *et al.* 2017) is used to count *k*-mers from sequencing reads (trimmed or raw) from each individual/sample. Users can specify the desired *k*-mer length and the read count threshold parameters in the configuration file. Initially, KMC is run in default mode to count canonical *k*-mers, followed by a second run using the "-b" option to count noncanonical *k*-mers. Canonization in this context means that KMC, in its default mode, considers a *k*-mer and its reverse complement as equivalent and assigns the combined count of the two to the alphabetically smaller *k*-mer (Deorowicz *et al.* 2015). Conversely, in noncanonical counting mode ("-b" option), the k-mer and its reverse complement are counted separately. Next, the "kmers_add_strand_information" function from the kmersGWAS library is used to combine the output of two KMC runs. This generates a single list of k-mers per sample, along with their strand information. Subsequently, *k*-mer lists from each sample are merged into a single binary file (kmer_to_use) and then filtered using the "list_kmers_found_in_multiple_samples" feature of the kmersGWAS library. As described in Voichek et al. (2020), the two-step filtering criteria are applied as follows: First, *k*-mers are filtered if they are not present in at least N individuals. Second, *k*-mers are filtered based on their canonical and noncanonical counts. A *k*-mer is dropped if it is not found in both canonical and noncanonical forms in at least X percent of the individuals/samples in which it appeared (Voichek and Weigel 2020). Users can easily modify filtering criteria within the configuration file. Upon completing the *k*-mer counting and filtering phases, kGWASflow generates summary statistics for *k*-mer counts and visually displays the results through a variety of plots (Fig. 1).

Next, kGWASflow generates the *k*-mer presence/absence genotype matrix, using the filtered *k*-mers obtained from all individuals/samples and utilizing the "build_kmers_table" function from the kmersGWAS library. This binary *k*-mers presence/absence table features *k*-mers as rows and individuals as columns. kGWASflow also allows users to convert this binary *k*-mers table into PLINK (Chang *et al.* 2015) format. The option to export the *k*-mers table in PLINK format enables users to utilize various other GWAS tools of their choice outside of the kGWASflow environment. This flexibility allows for the adoption of diverse GWAS models or settings, such as incorporating covariates into their GWAS model, an action not supported by the current kmersGWAS method. After creating the *k*-mer table, kGWASflow constructs a kinship-relatedness matrix based on either the *k*-mers table or a user-provided SNP file in PLINK format. Users can choose between these two options via the configuration file. If users opt for a k-mer-based kinship matrix, kGWASflow executes the "emma_kinship_kmers" function from the kmersGWAS library, generating an EMMA-based (Kang *et al.* 2008) relatedness matrix. Alternatively, if an SNP-based kinship matrix is preferred, the pipeline employs the "emma_kinship" function from the same library. Users can specify the minor allele frequency and minor allele count parameters for this step by modifying the config file.

Upon generating the *k*-mers table and the kinship matrix, kGWASflow proceeds to perform k-mers-based GWAS utilizing the methodology developed by Voichek et al. (2020). In this phase, the *k*-mers table, kinship matrix, and phenotype file serve as inputs. The kmersGWAS method is applied independently to each provided phenotype using the "kmers_gwas.py" script (Voichek and Weigel 2020). In short, this method initially permutes the given phenotype N times (with N specified by the user in kGWASflow) and employs a linear mixed model (LMM) to associate *k*-mer presence/absence patterns with the phenotype and its permutations. By utilizing this approximated model, the top-ranking *k*-mers (determined by the user in kGWASflow, with a default value of 10,000) are identified and subsequently passed to the next stage. Afterward, these top *k*-mers are utilized as input in the following step, where the actual model implemented in GEMMA (Zhou and Stephens 2012) is employed to generate exact *P*-values for the phenotype and its permutations. The kinship matrix is used to account for the relatedness between individuals. A permutation-based 5% family-wise error-rate threshold is identified, and the *k*-mers surpassing this threshold are deemed statistically significant. After the completion of the kmersGWAS run, kGWASflow produces a results summary table (Fig. 1), which incorporates the significant *k*-mers derived from the kmersGWAS stage.

Post-GWAS analysis

Post-GWAS analyses are all optional and highly dependent on the user's preference, the organism of interest, and the research question. Users can easily activate or deactivate various Post-GWAS options within the configuration file. This phase concentrates on identifying the genomic location and context of significantly associated *k*-mers. This step can offer deeper insights into the context of previously recognized associations, while also aiding in the discovery of new associations that may contain a broader range of genetic variants. kGWASflow includes 3 post-GWAS analysis options, which differ based on whether the user wants to map the kmers themselves, the reads they originated from, or contigs assembled from those reads.

The first post-GWAS analysis option is to directly map the trait-associated *k*-mers to a reference genome FASTA file (as defined in the configuration file). To map significant *k*-mers to a reference genome, users have the option to select from two read-mapping algorithms: bowtie or bowtie2 (Langmead and Salzberg 2012). kGWASflow maps the associated *k*-mers to a given genome FASTA using the preferred alignment algorithm and then converts the results into a sorted and indexed BAM file using samtools (Li *et al.* 2009). Finally, kGWASflow generates a Manhattan plot by incorporating the *P*-values from the kmersGWAS step and the genomic locations of the aligned *k*-mers obtained during this mapping stage.

The second option is to identify which sequencing reads the significant *k*-mers came from and map those reads to the genome FASTA file. When this option is enabled, kGWASflow initially retrieves the source reads for each associated *k*-mer from the FASTQ files of samples containing those *k*-mers. kGWASflow executes the"fetch_source_reads.py" script, which incorporates the "fetch_reads_with_kmers" tool (https://github.com/voichek/fetch_reads_with_kmers) at its core to identify the source reads of each trait-associated *k*-mer. After finding the source reads of trait-associated *k*-mers, the pipeline first merges and then sorts the reads using seqkit (Shen *et al.* 2016). kGWASflow maps the sorted reads to a reference genome FASTA file using bowtie2 (Langmead and Salzberg 2012) with "--very-sensitive-local" parameters. Alignments are filtered based on a mapping quality score defined by the user. kGWASflow also converts the alignment outputs into BAM and BED files for downstream analysis. Optionally, the workflow generates IGV reports of the alignment results in HTML format using the igv-reports tool (https://github.com/igvteam/igv-reports).

The third option also utilizes the source reads of the associated *k*-mers. If not previously obtained, the source reads of *k*-mers are retrieved as outlined above. In this step, instead of mapping the raw reads to a reference genome, kGWASflow first performs a de novo assembly of the source reads using SPADES (Prjibelski *et al.* 2020) with the "--careful" parameter. After the assembly step, kGWASflow runs minimap2 (Li 2018) to map assembled contigs onto a reference genome FASTA file. Users also have the option to perform a BLAST (Altschul *et al.* 1990) query using blastn on the resulting contigs. The output of this step is the sorted, indexed, and quality-filtered (user-defined) BAM files of mapped contigs. If the BLAST option is enabled, the workflow outputs a BLAST results file. As in the previous step, kGWASflow optionally generates IGV reports of the contig mapping results using the igv-reports software.

Testing the workflow

In order to test and evaluate the kGWASflow, we generated a test mock dataset consisting of 100 individuals with a corresponding mock genome FASTA file and phenotype (Fig. 2). A test run of the workflow using this mock dataset can be performed by executing a single command:

```
kgwasflow test --threads 16 \
  --work-dir working_dir
```

With this command, kGWASflow initiates a test run, first conducting QC analysis and producing a MultiQC report. After QC, it progresses to the *k*-mer counting phase, followed by *k*-mer filtration, table construction, and kinship matrix generation based on *k*-mers. The workflow will then perform kmersGWAS using the k-mers table and the kinship matrix. Finally, kGWASflow provides a summary of the kmersGWAS results and carries out options 1 and 2 of the post-GWAS phase, as illustrated in Fig. 1, thereby successfully completing the test run.

Results and discussion

To illustrate our workflow, we selected two distinct datasets that had been previously analyzed using different k-mer-based GWAS methods. The first dataset is a public *Escherichia coli* (*E. coli*) ampicillin resistance dataset that contains 241 strains of *E. coli* (Earle *et al.* 2016), which was used by Rahman *et al.* (2018) to test their k-mers-based association mapping tool HAWK (Rahman *et al.* 2018). Among 241 strains, 189 had ampicillin resistance and 52 were susceptible. By applying our workflow to this dataset, kGWASflow yielded results comparable to the findings from Rahman *et al.* (2018) (Supplementary Data S1: Tables 1 and 2). Figure 3 shows examples of kGWASflow output from this test run, including *k*-mer count summary statistics (Fig. 3a–c) and k-mers-based GWAS results (Fig. 3d,e). The configuration files and the HTML summary report from this kGWASflow run can be found in the Supplementary Data (Data S6–S10). This *E. coli* ampicillin resistance dataset is also included with the pipeline as an alternative test dataset and its results can be reproduced by executing a single command:

```
kgwasflow test --dataset ecoli --threads 16 \
  --work-dir working_dir
```

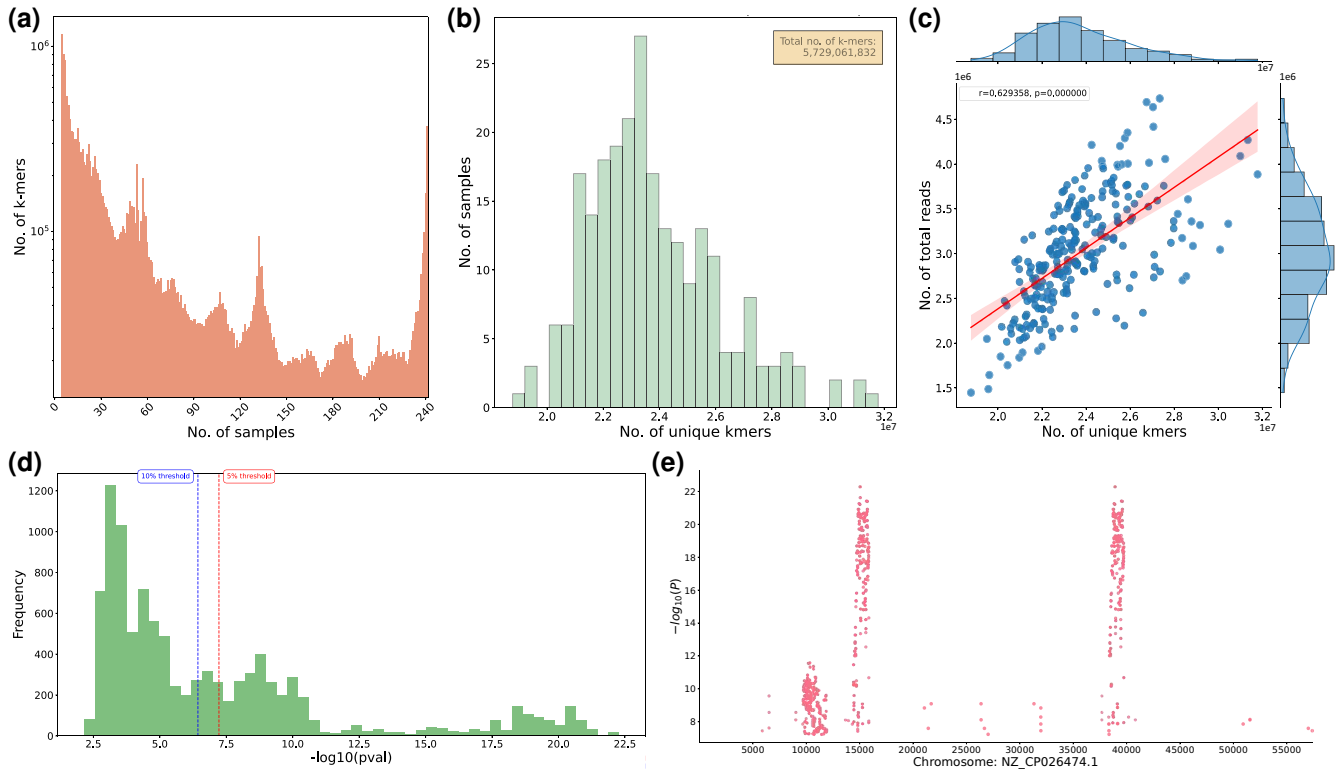

Fig. 3. Example outputs obtained from kGWASflow by processing the *E. coli* ampicillin resistance dataset (Rahman *et al.* 2018). a) Bar plot showing the number of *k*-mers that appeared in exactly "N" number of samples ("N" goes between 1 to the total number of samples). Only the *k*-mers that passed the initial filtering step were used. b) Histogram plot showing the distribution of noncanonical *k*-mer counts. The *x*-axis shows the unique *k*-mer counts and the *y*-axis shows the number of samples. The legend at the top right shows the total number of unique *k*-mers (noncanonical). The histogram plot for canonical counts can be found in the Supplementary Data (Supplementary Figure S1a). c) Joint plot showing the relationship between the noncanonical unique *k*-mer counts and the number of reads. The *x*-axis represents the number of unique *k*-mers (noncanonical), and the *y*-axis represents the number of total reads. The red line represents the linear regression line. The *r*-value is the Pearson correlation coefficient, and the *P*-value is the two-tailed *P*-value. The marginal distributions of the *x* and *y* axis are also shown on the top and right sides of the plot, respectively. The joint plot for canonical counts can be found in the Supplementary Data (Supplementary Figure S1b). d) Histogram of the -log10 *P*-values of each *k*-mer that passed the first kmersGWAS step. The red dashed line indicates the 5% family-wise error-rate threshold, while the blue dashed line indicates the 10% family-wise error-rate threshold. Only the *P*-values of the best *k*-mers from the first kmersGWAS step are used. *P*-values are obtained from GEMMA during the second step of kmersGWAS (a detailed explanation can be found in the k-mers-based GWAS section). e) Manhattan plot showing −log10 *P*-values of *k*-mers that are significantly associated with ampicillin resistance, mapped to their genomic locations. *k*-mers were mapped to *E. coli* plasmid pKBN10P04869A reference genome (PRJNA430286) using bowtie2.

Executing this command will trigger an automated download of the sequencing reads of all 241 strains of *E. coli* from NCBI. Following this, kGWASflow will execute all preprocessing, k-mers-based GWAS, and post-GWAS analysis stages and complete the *E. coli* test run. Ultimately, to produce an HTML report summarizing the workflow and the results from the *E. coli* test run, users only need to execute a single command after the test workflow run is completed:

```
kgwasflow test --dataset ecoli --threads 16 \
  --work-dir working_dir \
  --generate-report
```

We employed a second dataset consisting of whole genome sequencing data of 261 maize lines from the Goodman-Buckler Maize Association Panel (Flint-Garcia *et al.* 2005) and three different maize phenotypes, including kernel color, upper leaf angle, and cob color. He *et al.* (2021) previously used this dataset to test their k-mer-based GWAS tool, which relies on *k*-mer occurrence count (KOC) rather than presence/absence (He *et al.* 2021). Of the 281 maize lines in this association panel, only 261 were used due to the phenotype information available, as described in He *et al.* (2021). As in the previous test run, using kGWASflow on this dataset generated results that closely mirrored the findings of He *et al.* (2021). We identified trait-associated *k*-mers that passed the *P*-value threshold for each of the three phenotypes (Supplementary Data S1: Tables S3–S8). Furthermore, using the post-GWAS analysis module of kGWASflow, we have determined the probable genomic locations of these trait-associated *k*-mers and the results were comparable to He *et al.* (2021) (Supplementary Data S2: Figures S2–S5). The configuration files and the HTML report from this kGWASflow run can be found in the Supplementary Data (Data S11–S15).

Conclusion

kGWASflow is an easy-to-install, reproducible, scalable, and user-friendly workflow written in Snakemake. It employs the kmersGWAS method (Voichek and Weigel 2020) to conduct k-mer-based GWAS while offering enhanced pre- and post-GWAS analysis capabilities. kGWASflow offers extensive customization, either via the command line or a configuration file, enabling users to modify the workflow to their specific requirements. It takes advantage of Snakemake's parallelization and scalability capabilities, making the workflow deployable in local computers, high-performance computing, or cloud computing

environments. By utilizing Conda, kGWASflow effectively circumvents software dependency issues and prevents library/version conflicts. The easy expansibility of kGWASflow enables seamless integration of future enhancements and the incorporation of new or improved k-mers-based GWAS methods. Taken together, by creating a modular, customizable, and user-friendly workflow, we aim to enhance the accessibility and streamline k-mer-based GWAS, empowering a broader research community to leverage this approach.

Data availability

kGWASflow is freely available (under MIT license) from GitHub (https://github.com/akcorut/kGWASflow) and also on Bioconda (https://anaconda.org/bioconda/kgwasflow), including necessary inputs to perform a test run. The GitHub repository contains a wiki (https://github.com/akcorut/kGWASflow/wiki) explaining in detail how to install, configure, and use the workflow. Supplemental material available at FigShare: https://figshare.com/s/6f54d35d6f8dfc79e2f9.

Acknowledgments

The majority of the computation for developing and testing the pipeline was conducted on the high-performance computing (HPC) resource SAPELO2 at the Georgia Advanced Computing Resource Center (GACRC).

Funding

Funding for this project was provided by the University of Georgia and the National Science Foundation (grant #1764127).

Conflicts of interest

The authors declare no conflicts of interest.

Literature cited

Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. 1990. Basic local alignment search tool. J Mol Biol. 215:403–410. doi:10.1016/S0022-2836(05)80360-2

Andrews S. 2010. FastQC: A Quality Control Tool for High Throughput Sequence Data. Cambridge (UK): Babraham Bioinformatics, Babraham Institute.

Boyle EA, Li YI, Pritchard JK. 2017. An expanded view of complex traits: from polygenic to omnigenic. Cell. 169:1177–1186. doi:10.1016/j.cell.2017.05.038

Cano-Gamez E, Trynka G. 2020. From GWAS to function: using functional genomics to identify the mechanisms underlying complex diseases. Front Genet. 11:424. doi:10.3389/fgene.2020.00424

Chang CC, Chow CC, Tellier LC, Vattikuti S, Purcell SM, Lee JJ. 2015. Second-generation PLINK: rising to the challenge of larger and richer datasets. Gigascience. 4:7. doi:10.1186/s13742-015-0047-8

Coletta RD, Della Coletta R, Qiu Y, Ou S, Hufford MB, Hirsch CN. 2021. How the pan-genome is changing crop genomics and improvement.

Colque-Little C, Abondano MC, Lund OS, Amby DB, Piepho HP, Andreasen C, Schmöckel S, Schmid K. 2021. Genetic variation for tolerance to the downy mildew pathogen peronospora variabilis in genetic resources of quinoa (*Chenopodium quinoa*). BMC Plant Biol. 21:41. doi:10.1186/s12870-020-02804-7

Deorowicz S, Kokot M, Grabowski S, Debudaj-Grabysz A. 2015. KMC 2: fast and resource-frugal k-mer counting. Bioinformatics. 31:1569–1576. doi:10.1093/bioinformatics/btv022

Earle SG, Wu CH, Charlesworth J, Stoesser N, Gordon NC, Walker TM, Spencer CCA, Iqbal Z, Clifton DA, Hopkins KL, *et al.* 2016. Identifying lineage effects when controlling for population structure improves power in bacterial association studies. Nat Microbiol. 1:16041. doi:10.1038/nmicrobiol.2016.41

Ewels P, Magnusson M, Lundin S, Käller M. 2016. MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics. 32:3047–3048. doi:10.1093/bioinformatics/btw354

Faye LL, Machiela MJ, Kraft P, Bull SB, Sun L. 2013. Re-ranking sequencing variants in the post-GWAS era for accurate causal variant identification. PLoS Genet. 9:e1003609. doi:10.1371/journal.pgen.1003609

Flint-Garcia SA, Thuillet AC, Yu J, Pressoir G, Romero SM, Mitchell SE, Doebley J, Kresovich S, Goodman MM, Buckler ES. 2005. Maize association population: a high-resolution platform for quantitative trait locus dissection. Plant J. 44:1054–1064. doi:10.1111/j.1365-313X.2005.02591.x

Génin E. 2020. Missing heritability of complex diseases: case solved? Hum Genet. 139:103–113. doi:10.1007/s00439-019-02034-4

Göktay M, Fulgione A, Hancock AM. 2021. A new catalog of structural variants in 1,301 a. thaliana lines from Africa, Eurasia, and North America reveals a signature of balancing selection at defense response genes. Mol Biol Evol. 38:1498–1511. doi:10.1093/molbev/msaa309

Gupta PK. 2021. GWAS for genetics of complex quantitative traits: genome to pangenome and SNPs to SVs and k-mers. Bioessays. 43:e2100109. doi:10.1002/bies.202100109

Gupta PK, Kulwal PL, Jaiswal V. 2019. Association mapping in plants in the post-GWAS genomics era. Adv Genet. 104:75–154. doi:10.1016/bs.adgen.2018.12.001

He C, Washburn JD, Hao Y, Zhang Z, Yang J, Liu S. 2021. Trait association and prediction through integrative k-mer analysis.

Kale SM, Schulthess AW, Padmarasu S, Boeven PHG, Schacht J, Himmelbach A, Steuernagel B, Wulff BBH, Reif JC, Stein N, *et al.* 2022. A catalogue of resistance gene homologs and a chromosome-scale reference sequence support resistance gene mapping in winter wheat. Plant Biotechnol J. 20:1730–1742. doi:10.1111/pbi.v20.9

Kang HM, Zaitlen NA, Wade CM, Kirby A, Heckerman D, Daly MJ, Eskin E. 2008. Efficient control of population structure in model organism association mapping. Genetics. 178:1709–1723. doi:10.1534/genetics.107.080101

Kokot M, Dlugosz M, Deorowicz S. 2017. KMC 3: counting and manipulating k-mer statistics. Bioinformatics. 33:2759–2761. doi:10.1093/bioinformatics/btx304

Langmead B, Salzberg SL. 2012. Fast gapped-read alignment with bowtie 2. Nat Methods. 9:357–359. doi:10.1038/nmeth.1923

LaPierre N, Taraszka K, Huang H, He R, Hormozdiari F, Eskin E. 2021. Identifying causal variants by fine mapping across multiple studies. PLoS Genet. 17:e1009733. doi:10.1371/journal.pgen.1009733

Lemane T, Chikhi R, Peterlongo P. 2022. k mdiff, large-scale and user-friendly differential k-mer analyses. Bioinformatics. 38:5443–5445. doi:10.1093/bioinformatics/btac689

Li H. 2018. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics. 34:3094–3100. doi:10.1093/bioinformatics/bty191

Li YR, Glessner JT, Coe BP, Li J, Mohebnasab M, Chang X, Connolly J, Kao C, Wei Z, Bradfield J, *et al.* 2020. Rare copy number variants in

over 100,000 European ancestry subjects reveal multiple disease associations. Nat Commun. 11:255. doi:10.1038/s41467-019-13624-1

Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R. 2009. The sequence alignment/map format and SAMtools. Bioinformatics. 25:2078–2079. doi:10.1093/bioinformatics/btp352

Li S, Kong L, Xiao X, Li P, Liu A, Li J, Gong J, Gong W, Ge Q, Shang H. 2022. Genome-wide artificial introgressions of *Gossypium barbadense* into *G. hirsutum* reveal superior loci for simultaneous improvement of cotton fiber quality and yield traits. J Advert Res. doi:10.1016/j.jare.2022.11.009.

MacArthur J, Bowler E, Cerezo M, Gil L, Hall P, Hastings E, Junkins H, McMahon A, Milano A, Morales J, *et al.* 2017. The new NHGRI-EBI catalog of published genome-wide association studies (GWAS catalog). Nucleic Acids Res. 45:D896–D901. doi:10.1093/nar/gkw1133

Manolio TA, Collins FS, Cox NJ, Goldstein DB, Hindorff LA, Hunter DJ, McCarthy MI, Ramos EM, Cardon LR, Chakravarti A, *et al.* 2009. Finding the missing heritability of complex diseases. Nature. 461:747–753. doi:10.1038/nature08494

Manuweera B, Mudge J, Kahanda I, Mumey B, Ramaraj T, Cleary A. 2019. Pangenome-wide association studies with frequented regions. In: BCB '19. New York, NY, USA: Association for Computing Machinery. 627–632.

Martin M. 2011. Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet J. 17:10–12. doi:10.14806/ej.17.1

Mehrab Z, Mobin J, Tahmid IA, Rahman A. 2021. Efficient association mapping from k-mers-an application in finding sex-specific sequences. PLoS One. 16:e0245058. doi:10.1371/journal.pone.0245058

Mölder F, Jablonski KP, Letcher B, Hall MB, Tomkins-Tinch CH, Sochat V, Forster J, Lee S, Twardziok SO, Kanitz A, *et al.* 2021. Sustainable data analysis with snakemake. F1000Res. 10:33. doi:10.12688/f1000research.29032.1

Nolte IM, van der Most PJ, Alizadeh BZ, de Bakker PI, Boezen HM, Bruinenberg M, Franke L, van der Harst P, Navis G, Postma DS, *et al.* 2017. Missing heritability: is the gap closing? an analysis of 32 complex traits in the lifelines cohort study. Eur J Hum Genet. 25:877–885. doi:10.1038/ejhg.2017.50

Onetto CA, Sosnowski MR, Van Den Heuvel S, Borneman AR. 2022. Population genomics of the grapevine pathogen *Eutypa lata* reveals evidence for population expansion and intraspecific differences in secondary metabolite gene clusters. PLoS Genet. 18:e1010153. doi:10.1371/journal.pgen.1010153

Prinsen RTMM, Rossoni A, Gredler B, Bieber A, Bagnato A, Strillacci MG. 2017. A genome wide association study between CNVs and quantitative traits in Brown Swiss cattle. Livest Sci. 202:7–12. doi:10.1016/j.livsci.2017.05.011

Prjibelski A, Antipov D, Meleshko D, Lapidus A, Korobeynikov A. 2020. Using SPAdes de novo assembler. Curr Protoc Bioinformatics. 70:e102. doi:10.1002/cpbi.102

Qin P, Lu H, Du H, Wang H, Chen W, Chen Z, He Q, Ou S, Zhang H, Li X, *et al.* 2021. Pan-genome analysis of 33 genetically diverse rice accessions reveals hidden genomic variations. Cell. 184:3542–3558.e16. doi:10.1016/j.cell.2021.04.046

Rahman A, Hallgrímsdóttir I, Eisen M, Pachter L. 2018. Association mapping from sequencing reads using k-mers. Elife. 7:e32920. doi:10.7554/eLife.32920

Schulthess AW, Kale SM, Liu F, Zhao Y, Philipp N, Rembe M, Jiang Y, Beukert U, Serfling A, Himmelbach A, *et al.* 2022. Genomics-informed prebreeding unlocks the diversity in genebanks for wheat improvement. Nat Genet. 54:1544–1552. doi:10.1038/s41588-022-01189-7

Shen W, Le S, Li Y, Hu F. 2016. SeqKit: a cross-platform and ultrafast toolkit for FASTA/Q file manipulation. PLoS One. 11:e0163962. doi:10.1371/journal.pone.0163962

Song JM, Guan Z, Hu J, Guo C, Yang Z, Wang S, Liu D, Wang B, Lu S, Zhou R, *et al.* 2020. Eight high-quality genomes reveal pan-genome architecture and ecotype differentiation of *Brassica napus*. Nat Plants. 6:34–45. doi:10.1038/s41477-019-0577-7

Sul JH, Martin LS, Eskin E. 2018. Population structure in genetic studies: confounding factors and mixed models. PLoS Genet. 14:e1007309. doi:10.1371/journal.pgen.1007309

Sun S, Dong B, Zou Q. 2021. Revisiting genome-wide association studies from statistical modelling to machine learning. Brief Bioinform. 22:bbaa263. doi:10.1093/bib/bbaa263

Tam V, Patel N, Turcotte M, Bossé Y, Paré G, Meyre D. 2019. Benefits and limitations of genome-wide association studies. Nat Rev Genet. 20:467–484. doi:10.1038/s41576-019-0127-1

Theunissen F, Flynn LL, Anderton RS, Mastaglia F, Pytte J, Jiang L, Hodgetts S, Burns DK, Saunders A, Fletcher S, *et al.* 2020. Structural variants may be a source of missing heritability in sALS. Front Neurosci. 14:47. doi:10.3389/fnins.2020.00047

Tian D, Wang P, Tang B, Teng X, Li C, Liu X, Zou D, Song S, Zhang Z. 2020. GWAS atlas: a curated resource of genome-wide variant-trait associations in plants and animals. Nucleic Acids Res. 48:D927–D932. doi:10.1093/nar/gkz828

Tibbs Cortes L, Zhang Z, Yu J. 2021. Status and prospects of genome-wide association studies in plants. Plant Genome. 14:e20077. doi:/10.1002/tpg2.20077

Tripodi P, Rabanus-Wallace MT, Barchi L, Kale S, Esposito S, Acquadro A, Schafleitner R, van Zonneveld M, Prohens J, Diez MJ, *et al.* 2021. Global range expansion history of pepper (*Capsicum* spp.) revealed by over 10,000 genebank accessions. Proc Natl Acad Sci USA. 118.

Uffelmann E, Huang QQ, Munung NS, de Vries J, Okada Y, Martin AR, Martin HC, Lappalainen T. 2021. Genome-wide association studies. Nat Rev Methods Primers. 1:1–21. doi:10.1038/s43586-021-00056-9

Visscher PM, Wray NR, Zhang Q, Sklar P, McCarthy MI, Brown MA, Yang J. 2017. 10 years of GWAS discovery: biology, function, and translation. Am J Hum Genet. 101:5–22. doi:10.1016/j.ajhg.2017.06.005

Voichek Y, Weigel D. 2020. Identifying genetic variants underlying phenotypic variation in plants without complete genomes. Nat Genet. 52:534–540. doi:10.1038/s41588-020-0612-7

Wei X, Qiu J, Yong K, Fan J, Zhang Q, Hua H, Liu J, Wang Q, Olsen KM, Han B, *et al.* 2021. A quantitative genomics map of rice provides genetic insights and guides breeding. Nat Genet. 53:243–253. doi:10.1038/s41588-020-00769-9

Wray NR, Purcell SM, Visscher PM. 2011. Synthetic associations created by rare variants do not explain most GWAS results. PLoS Biol. 9:e1000579. doi:10.1371/journal.pbio.1000579

Yang N, Liu J, Gao Q, Gui S, Chen L, Yang L, Huang J, Deng T, Luo J, He L, *et al.* 2019. Genome assembly of a tropical maize inbred line provides insights into structural variation and crop improvement. Nat Genet. 51:1052–1059. doi:10.1038/s41588-019-0427-6

Young AI. 2019. Solving the missing heritability problem. PLoS Genet. 15:e1008222. doi:10.1371/journal.pgen.1008222

Zhou Y, Connor EE, Wiggans GR, Lu Y, Tempelman RJ, Schroeder SG, Chen H, Liu GE. 2018. Genome-wide copy number variant analysis reveals variants associated with 10 diverse production traits

in holstein cattle. BMC Genom. 19:314. doi:10.1186/s12864-018-4699-5

Zhou X, Stephens M. 2012. Genome-wide efficient mixed-model analysis for association studies. Nat Genet. 44:821–824. doi:10.1038/ng.2310

Zhou Y, Zhang Z, Bao Z, Li H, Lyu Y, Zan Y, Wu Y, Cheng L, Fang Y, Wu K, *et al.* 2022. Graph pangenome captures missing heritability and empowers tomato breeding. Nature. 606:527–534. doi:10.1038/s41586-022-04808-9

Editor: A. Lipka