GigaScience, 2023, **12**, 1–10
DOI: 10.1093/gigascience/giad025
Technical Note

DivBrowse—interactive visualization and exploratory data analysis of variant call matrices

Patrick König[1,*], Sebastian Beier[1,2], Martin Mascher[3,4], Nils Stein[3,5], Matthias Lange[1] and Uwe Scholz[1]

[1]Department of Breeding Research, Leibniz Institute of Plant Genetics and Crop Plant Research (IPK) Gatersleben, 06466 Seeland, Germany
[2]Institute of Bio- and Geosciences, IBG-4, Forschungszentrum Jülich GmbH, 52425 Jülich, Germany
[3]Department of Genebank, Leibniz Institute of Plant Genetics and Crop Plant Research (IPK) Gatersleben, 06466 Seeland, Germany
[4]German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig, 04103 Leipzig, Germany
[5]Center for Integrated Breeding Research, Georg-August University, 37075 Göttingen, Germany
*Correspondence address. Patrick König, Department of Breeding Research, Leibniz Institute of Plant Genetics and Crop Plant Research (IPK) Gatersleben, 06466 Seeland, Germany; E-mail: koenig@ipk-gatersleben.de

Abstract

Background: The sequencing of whole genomes is becoming increasingly affordable. In this context, large-scale sequencing projects are generating ever larger datasets of species-specific genomic diversity. As a consequence, more and more genomic data need to be made easily accessible and analyzable to the scientific community.

Findings: We present DivBrowse, a web application for interactive visualization and exploratory analysis of genomic diversity data stored in Variant Call Format (VCF) files of any size. By seamlessly combining BLAST as an entry point together with interactive data analysis features such as principal component analysis in one graphical user interface, DivBrowse provides a novel and unique set of exploratory data analysis capabilities for genomic biodiversity datasets. The capability to integrate DivBrowse into existing web applications supports interoperability between different web applications. Built-in interactive computation of principal component analysis allows users to perform ad hoc analysis of the population structure based on specific genetic elements such as genes and exons. Data interoperability is supported by the ability to export genomic diversity data in VCF and General Feature Format 3 files.

Conclusion: DivBrowse offers a novel approach for interactive visualization and analysis of genomic diversity data and optionally also gene annotation data by including features like interactive calculation of variant frequencies and principal component analysis. The use of established standard file formats for data input supports interoperability and seamless deployment of application instances based on the data output of established bioinformatics pipelines.

Keywords: genomics, data visualization, variation data, Variant Call Format

Introduction

In times of ever-increasing quantity and quality of sequencing data that is driven by the ongoing reduction of sequencing costs and simultaneously increasing computing power, an ever-increasing amount of data of genomic diversity is being generated, which is almost impossible to keep track of in its entirety [1, 2]. Rather, there is a need to reduce complexity through models that aggregate the data in a meaningful way but preserve the information capacity.

One goal of diversity analyses is to find the underlying changes in genomic traits by observing the type of change, frequency, and correlations to the phenotype. A prerequisite for this is dense genomic diversity datasets to perform genome-wide association studies, which can directly identify related nucleotide polymorphisms, genes, and other genetic features like promoters or enhancers [3].

Visualization of such huge genomic datasets as used in diversity analysis remains a challenge for software tools and their developers, as the amount of generated data grows exponentially fast and the development of appropriate visualization and analysis tools is a time-consuming effort [4, 5].

In recent years, numerous tools for visualizing genomes and genomic diversity data have been developed and published. Tools like JBrowse2 [6] and igv.js [7] focus on the general visualization of assemblies, genome annotations, and read alignments, whereas tools like SnpHub [8], SNPversity [9], and Flapjack [10] put their main focus on the visualization of variants and variant calls. In reviewing the existing software tools that have been used in genomics research for decades, it became apparent that they all had several shortcomings and that none of them could meet the main requirements of our community for visualizing genomic data. By comparing existing software tools with user feedback from genomics projects [11], the following major requirements were identified:

- A web-based software that can be run standalone or can be integrated as a plugin into existing data web portals.
- A performant interactive visualization of variant call matrices with hundreds of millions of variants and thousands of samples.
- Usage of standardized and established bioinformatics file formats for data import and data export.
- Availability of a Javascript-API to control the tool from a hosting web portal (e.g., to control the list of genotypes to be displayed).

In order to combine the features of existing tools and these advanced features, we developed DivBrowse as a tool specifically for the interactive visualization of very large variant call matrices. It

Received: September 23, 2022. **Revised:** January 23, 2023. **Accepted:** March 23, 2023

allows users to interactively navigate through variant matrices on the order of hundreds of millions of variants and several thousand to tens of thousands of genotypes. It allows the user to keep a visual overview of very large but also small datasets of genomic diversity, supplemented by interactive analysis features like principal component analysis for ad hoc investigation of the population structure on the level of genetic features like genes, promotors, enhancers, or silencers. DivBrowse can serve as a daily used entry point for exploratory data analysis of datasets of genomic diversity aligned to a reference genome for all species. It uses standardized and established bioinformatics file formats like the Variant Call Format (VCF) [12] for single-nucleotide polymorphisms and the Generic Feature Format Version 3 (GFF3) for genome annotation data [13].

DivBrowse was initially developed as part of a web-based information system for visual analytics in the frame of a barley genebank genomics project to serve and visualize VCF-based genotypic data with 22,621 genotypes and 171,263 variants from a genotyping-by-sequencing (GBS) approach [11, 14, 15]. A new barley reference genome expanded the number of variants of this genotype panel to 775,283 [16]. A VCF file derived from whole-genome sequencing data with 223,387,147 variants and 300 barley genotypes was used for one of the demo instances of DivBrowse available on the project website, along with demo instances for human and mouse diversity data [16, 17].

Findings

DivBrowse combines the approach of genome browsers with the capability to visualize and interactively analyze thousands to millions of genomic variants for thousands of genotypes in the style of an exploratory data analysis [18]. The graphical user interface (GUI) of DivBrowse is accessed via a web browser. It shows genomic features such as nucleotide sequence, associated gene models, and genomic variants. Their physical positions in the currently selected chromosome or contig are shown in the upper section of the GUI, and the genotypes are listed row-wise in the lower section (see Fig. 1). Besides the visualization of the variant calls per variant and genotype, DivBrowse also calculates and displays variant statistics such as minor allele frequencies, proportion of heterozygous calls, or missing variant calls for each visualized genomic window. Furthermore, variant effect predictions according to SnpEff [19] can be displayed, provided they are present in the underlying VCF file.

General usability concept

The main concept for the visualization of the variant matrices is a tabular-like gapless display with the variants as columns and genotypes as rows alongside a genome track that visualizes the physical positions of the variants together with physical positions of genes, exons, and other genetic features in the current visible genomic range. Whereas the horizontal axis of the genome track at the top is scaled with respect to the physical positions and distances, the variant calls at the bottom are visualized gapless. Since variants generally do not cover all physical positions, this creates a positional divergence between the variant calls and the corresponding physical positions of the variants in the genome track. As a solution for this divergence, Bézier curves have been implemented that connect the physical positions of the variants with the corresponding columns in the variant calls matrix below. With this approach, more variant calls can be visualized at the same time on the available screen width while maintaining the physical coordinates of each variant along the reference genome. Each variant column is connected to the physical coordinate of the variant through a Bézier curve.

The user is able to switch between 2 zoom levels for the horizontal and vertical axis independently from each other by dedicated buttons in the menu at the top (see Fig. 1A). Thus, the user can decide whether to display more variants, more genotypes, or both at the same time on the viewport. We envision that the common use of DivBrowse proceeds in 3 successive stages, as illustrated and explained in Fig. 2.

Efficient handling of large variation data matrices

DivBrowse is able to efficiently handle and visualize very large variant matrices in the VCF file format containing variant calls for thousands to tens of thousands of genotypes and thousands to hundreds of millions of variants. Typically, these VCF files can exceed file sizes of 100 gigabytes, even when compressed with GZip or BGZip. Uncompressed file sizes can exceed 1 terabyte easily. It is therefore important to efficiently handle those large amounts of data to enable interactive access to slices of the variation data matrix with request-response-cycle times below 1 second for a pleasant user experience. DivBrowse is able to do so by using the Zarr Python package for storing the variant calls as *n*-dimensional compressed chunked arrays [20]. The Zarr format is a cloud-native storage format and is increasingly used in areas of life sciences that need to handle very large array- or matrix-like datasets (e.g., bioimaging data) [21]. The VCF file to be visualized is converted to the Zarr format with the vcf_to_zarr() method of the Python package *scikit-allel* [22] beforehand. The usage of the Zarr format has the advantage that the ASCII-formatted content of a VCF file is converted into an optimized and compressed binary format that can be natively consumed with libraries of the Scientific Python Stack like Numpy, Pandas, and scikit-allel. This eliminates the need to repetitively read and parse the contents of the VCF file while executing API requests by the DivBrowse server. Zarr also provides built-in index-based access to the rows and columns of matrices, which eliminates the need of system calls to Tabix [23] for random access to the rows in a VCF file.

Command-line interface

The DivBrowse command-line interface (CLI) provides a way to start an ad hoc session of an instance of DivBrowse that uses VCF and GFF3 data files present in the current working directory of the user's shell. The CLI automatically detects multiple VCF and GFF3 files and provides an interactive dialogue where the user can select the files to be used to start an instance of DivBrowse. The CLI automatically infers basic configuration settings from the VCF and GFF3 files (e.g., the number of chromosomes and its labels and a mapping from chromosome labels in the VCF file to those in the GFF3 file). It is also possible to provide a manually created DivBrowse configuration file (divbrowse.config.yml) in the YAML format [24]. An automatically inferred configuration can also be saved as a configuration YAML-file in the current working directory. Such an inferred configuration file can be then used as a skeleton for a more advanced configuration file to use all features of DivBrowse. Since not all functions of DivBrowse can be derived from the underlying files, some functions have to be activated or configured specifically via the YAML configuration file. Please refer to the official documentation available on the project website for more information [17].

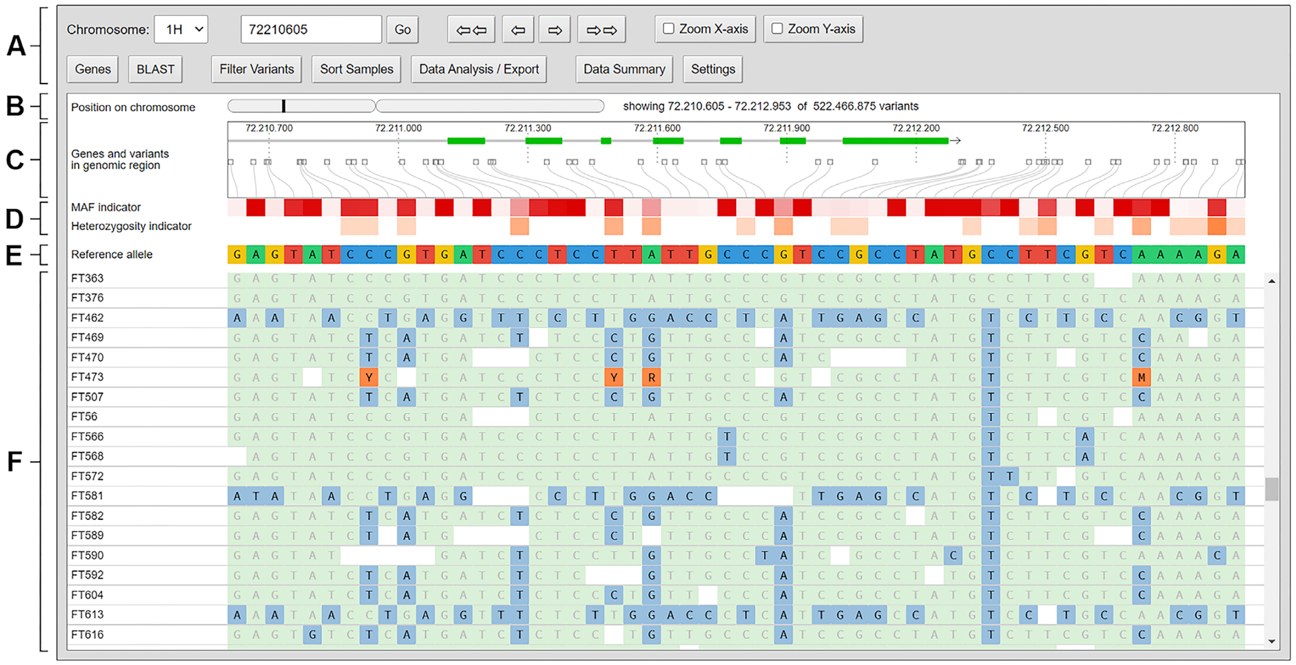

Figure 1: Screenshot of the initial view of the graphical user interface of DivBrowse. It is organized from top to bottom into the following main sections: (A) The navigation and tool menu where users can enter a physical position where to jump to, buttons for navigating left and right through the variant matrix, and buttons to open dialogues for data analysis and export features. (B) An overview map of the chromosome with the current visible genomic region highlighted. (C) The track truly scaled in terms of physical positions and distances for genetic traits and variant positions. (D) The track with the visual indicator for the minor allele frequency of each variant where boxes with darker red indicate smaller minor allele frequencies and the track for the heterozygosity frequency of each variant, with white indicating low frequencies and red shades indicating high frequencies. (E) The track with the reference allele for this variant present in the reference genome. (F) The scrollable track with all variant calls for each combination of variant and genotype.

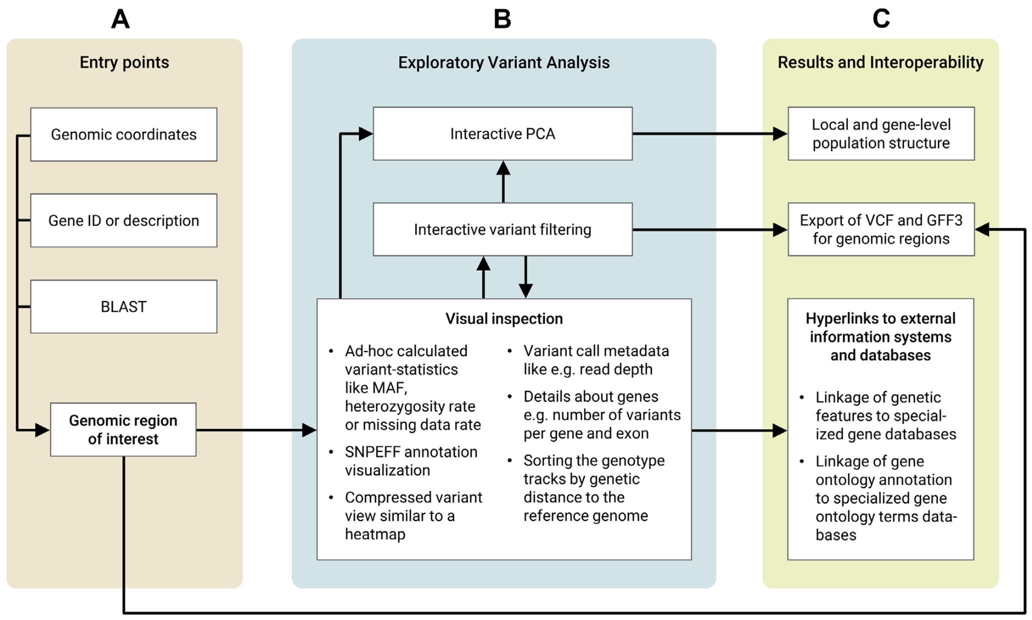
A
B
C
Entry points
Genomic coordinates
Gene ID or description
BLAST
Genomic region of interest
Exploratory Variant Analysis
Interactive PCA
Interactive variant filtering
Visual inspection
Ad-hoc calculated variant-statistics like MAF, heterozygosity rate or missing data rate
SNPEFF annotation visualization
Compressed variant view similar to a heatmap
Variant call metadata like e.g. read depth
Details about genes e.g. number of variants per gene and exon
Sorting the genotype tracks by genetic distance to the reference genome
Results and Interoperability
Local and gene-level population structure
Export of VCF and GFF3 for genomic regions
Hyperlinks to external information systems and databases
Linkage of genetic features to specialized gene databases
Linkage of gene ontology annotation to specialized gene ontology terms databases

Figure 2: The workflow concept of DivBrowse can be divided into 3 successive workflow stages. (A) As an initial step, users can access the variant data by 3 independent entry points: (i) by providing a genomic coordinate consisting of the chromosome label and the physical position, (ii) by searching for a gene by its annotation ID or description and then jumping to the genomic coordinate of this gene, and (iii) by performing a BLAST search with a user-defined nucleotide or protein sequence and then jumping to the genomic coordinates of the BLAST results. (B) After the user has found a genomic region of interest, they can now visually inspect and exploratively analyze the variants. Variant statistics like minor allele frequency or heterozygosity frequency are visualized by a heatmap-style track. The user is able to filter the variants based on variables of the variant statistics. It is also possible to perform a principal component analysis for a genomic window or a gene with or without application of the variant filter settings. We call this workflow "exploratory variant analysis." (C) As a result of the iterative and interactive exploratory variant analysis workflow, the user is able to get a better understanding of local (e.g., promoter or enhancer/silencer regions) or gene-level population structure. The user can also export the variation data together with genome annotation data for a genomic window or gene in VCF and GFF3 files to use those in further downstream analysis steps. It is also possible to integrate data and knowledge from other specialized databases like gene or gene ontology databases that are linked via hyperlinks.

Gene search

If a gene annotation of the genome exists as a GFF3 file and has been loaded into a DivBrowse instance, it is possible to use the "Gene search" feature. The "Gene search" dialogue allows searching all genes by their ID or annotated label. It is also possible to list all genes within a user-defined genomic range on a specific chromosome. Both search capabilities can be used in combination to search for genes by ID or annotation on a specific chromosome or within a user-defined genomic range on a specific chromosome. In addition to the number of variants on the entire gene, the number of variants located on exons of a gene is also displayed in the search results table when using the configuration setting "count_exon_variants: true."

BLAST as an entry point

As an additional entry point besides the entry points "genomic position" and "search genes by ID/description," an optional BLAST entry point has also been implemented in DivBrowse [25]. It allows performing a BLAST search for a given DNA (blastn) or protein sequence (tblastn) of the user's choice. The BLAST search is implemented by using the Galaxy API of an existing Galaxy instance via the Python package *BioBlend* [26]. The Galaxy instance to be used must have the NCBI BLAST+ tools [27] and the corresponding BLAST databases for the reference genome used in the original VCF installed. In order to use a Galaxy instance for a BLAST search by DivBrowse, the user must have an API key or user account credentials for the Galaxy instance to be used. After the BLAST search has been conducted by the Galaxy server, results are then displayed in a table that consists of a column that reports about the number of variants within the aligned target sequence for each BLAST hit. A DivBrowse user can directly see whether their BLAST hits contain variants or not. The BLAST result table also includes a column with links that can be used to jump directly to the genomic region of each BLAST hit. The BLAST results are stored within the current web browser session so that a DivBrowse user can review all BLAST hits without repeating the whole BLAST search again. In addition, multiple BLAST search results are stored in a session-based cache and can be loaded within microseconds without repeating the BLAST search.

Variant filtering

The "Filter variants" dialogue allows one to filter the variants by different variant-level attributes. It is possible to filter the variants by ad hoc calculated variant statistics like minor allele frequency/fraction, heterozygosity fraction, and missing data fraction. These ad hoc calculated variant statistics are dependent on the currently selected genotype panel. It is also possible to filter by variant-level attributes that have been derived from the VCF input file (e.g., by the QUAL- or MQ-attribute of a variant). As those metadata attributes are derived from the variant calling process, it is not possible to calculate those metrics in an ad hoc manner. The user-defined filter settings can also be used in the interactive principal component analysis (PCA) to exclude certain variants from the PCA calculation. This allows users of DivBrowse to interactively study PCA results based on different variant filter criteria. Furthermore, it is also possible to use the user-defined variant filtering in the VCF export to export only those variants that match the specific variant filter criteria to a newly created VCF file. This enables DivBrowse users to export custom-filtered variants to a VCF file for their personal downstream analysis scenarios.

Sample sorting

The "Sort samples" feature allows sorting the list of genotypes according to different criteria. On the one hand, the genotype tracks can be sorted alphabetically in ascending or descending order. This could be helpful if the IDs or labels of the genotypes can be sorted alphabetically in a meaningful way (e.g., if they are actual names of plants or animals and the names can infer ancestry or different traits). Another option is to sort the genotype tracks based on the respective genetic distance from the reference genome. For the calculation of genetic distances, the pairwise Euclidean distances between all genotype vectors is used as the simplest estimator for the genetic distance [28]. The genotype vectors used for the pairwise distance calculation contain the number of alternative alleles for each variant. Here, the user can select which genomic region should be used to calculate the genetic distance for sorting the tracks. It is possible to calculate the genetic distance (i) within the currently visible genomic range in the viewport, (ii) within the boundaries of a currently visible gene or subfeature of a gene (e.g., specific exon), or (iii) within a user-defined genomic window, which could be useful to sort by the genetic distances of an intragenic region, gene flanking region, or regulatory region.

Interactive principal component analysis

The built-in PCA functionality allows for interactive principal component analysis of (i) the variants that are currently visible in the viewport, (ii) the variants on the complete gene or subfeatures of the gene (e.g., a specific exon) that is currently visible in the viewport, or (iii) the variants within a user-defined genomic range. Interactive in this context refers to the PCA being computed on-demand with a runtime in the time frame of seconds. Due to possible hardware limitations on the server-side compute power, the PCA calculation can be limited to a maximum number of included variants by the configuration setting "pca_max_variants" in the configuration YAML file of DivBrowse. This allows the user to adjust the amount of variants to be included in the PCA calculation depending on the computational capacity of the web server. For smaller slices up to 10,000 genotypes and 100 variants of the whole variant matrix, the PCA calculation can be done under 1 second on average contemporary server hardware.

Export of VCF and GFF3 files

DivBrowse features data export in standard formats, namely, VCF and GFF3. Such exported datasets can be analyzed further with downstream analysis tools that are able to consume VCF and GFF3 files directly or indirectly by using appropriate file format converters. The VCF export feature allows the export of variation data of a given genomic range to standard-compliant VCF files [12]. The user can export (i) the genomic region that is displayed in the current viewport of DivBrowse, (ii) the genomic region encompassed by a currently visible gene or subfeature of that gene (e.g., exon), or (iii) a genomic window defined by a start and end position within a single chromosome by the user. The GFF3 export feature works similarly to the VCF export feature but instead exports a GFF3 file containing the gene annotation.

Linkage of genetic features to other web-based information systems

DivBrowse allows setting up URLs to external websites for each gene of a given GFF3 file by a configuration setting in the divbrowse.config.yml file. Furthermore, external links for each available gene ontology term can be set up. This allows users of DivBrowse to get specific and more detailed information about a gene and its ontology terms on specialized external websites and databases, for example, by linking to species-specific gene expression databases or by linking to the QuickGO service of the European Bioinformatics Institute for the ontology terms provided in the GFF3 file for each gene [29].

Standalone or plugin usage

DivBrowse can be used either standalone or as a Javascript plugin to complement existing functionality in an already existing web application. Using it as a plugin makes sense if there is a web application with existing omics data (e.g., phenotypic data), which should be supplemented with a visualization of genotypic data that have a relational connection to the already existing omics data. If only genotypic data are available, the standalone use is sufficient and advised. The usage of DivBrowse as a plugin opens additional possibilities concerning the interaction between the hosting web application and the DivBrowse instance that are described more in detail in the use case "Plugin in existing web-based information systems" in the Discussion section of this article. In the case of plugin usage, the communication between the hosting web application and the DivBrowse instance can be realized by using the DivBrowse Javascript-API, which is described in the "Implementation" section of this work.

Methods

General application architecture

DivBrowse (RRID:SCR_022780) is implemented in a client–server architecture that uses a Python-based stack for the server-side part and a stack based on JavaScript in combination with the Svelte framework [30], HTML, and CSS for the client-side component (GUI) (see Fig. 3). The server-side part uses multiple third-party packages available on the Python Package Index [31] that are listed in Supplementary Table S1. The DivBrowse server-side component is configurable by a YAML format-based configuration file named "divbrowse.config.yml" per default. It is also possible to use alternative file names for the configuration file.

Architecture of the server-side component

The DivBrowse server-side component is a Python application that utilizes Flask as a micro web framework to provide and expose the DivBrowse REST-API used for communication between itself and the graphical user interface that runs client-side in the user's web browser. The server-side component uses third-party packages available on the Python Package Index [31] like bioblend, numpy, scikit-learn, zarr, scikit-allel, flask, pandas, pyyaml, and click (see Supplementary Fig. S1). The server-side component can also serve all the static files necessary for the GUI like HTML, Javascript, and CSS files. In a productive setup, we recommend using specialized web server software such as Nginx or Apache HTTP server to serve the static files of the GUI and to act as a proxy server for the WSGI-based Python processes. For instructions on how to set up such a productive web server environment with Nginx, see the official documentation. An overview of all endpoints of the DivBrowse REST-API can be found in Supplementary Table S1.

Architecture of the graphical user interface

The GUI of DivBrowse is implemented in Javascript, HTML, and CSS utilizing the Svelte framework for a modular and component-based application architecture (see Supplementary Fig. S2). In order to maximize compatibility in the case of a plugin usage of DivBrowse inside of an existing web portal, we decided not to use a CSS framework like Bootstrap to lower the risk of CSS conflicts due to the cascading nature of CSS. All CSS rules were created individually by hand. In addition, CSS rules were completely avoided to be applied globally at the HTML tag level. Instead, rules were applied only based on ID or class attributes with a common prefix "divbrowse" to simulate a CSS namespace. The DivBrowse Javascript component provides a JavaScript API that allows interaction and communication between a DivBrowse instance and a hosting web application. In such a scenario, DivBrowse acts as a plugin or sub-component inside of another web application. The Javascript API provided methods to control certain aspects of DivBrowse (e.g., to subset the list of genotypes or to jump to a specific genomic position). Furthermore, callback functions are provided that are called after specific events or user interactions in the DivBrowse GUI. For example, when the user performs an interactive PCA and then makes a lasso selection of some genotype samples in the PCA plot, a callback function "selectedSamples()" is called with an array of sample IDs that have been selected as the first function argument. In this way, selection of genotype samples can be transferred from DivBrowse to the host web application where the sample selection can be processed further.

JavaScript-API

The JavaScript-API allows controlling the GUI of DivBrowse from a web application that itself hosts a DivBrowse instance. This allows an interactive coupling of existing web applications for omics data with one or many DivBrowse instances. For example, it is then possible to force a DivBrowse instance to jump to a specific genomic range in its viewport. Communication and data transfer from a DivBrowse instance to the hosting portal are also possible. For example, if a user selects some genotypes of interest in a PCA result, the sample IDs of those genotypes can be transferred back to the hosting portal via a Javascript callback function. A detailed list of all available JavaScript-API commands is given in Supplementary Table S2.

Discussion

Since the visualization is performed by DivBrowse in the web browser, a convenient and low-barrier access to genotypic diversity data is possible even without special bioinformatics knowledge and without time-consuming download of VCF files and their computationally intensive processing on their own computer hardware. This supports easy access to the extensive biodiversity data that already exist and will be obtained in the future, making it easily available to a wide audience in the spirit of open data. To realize low latencies for the interactive visualization of variant matrices of the size of several hundred gigabytes, an efficient server-side backend is implemented in Python. For this purpose, the backend uses *n*-dimensional chunked arrays and is performant enough for a good user experience in contemporary hardware environments like notebooks, standard desktop computers, and bare metal server or virtual machine servers. This aspect lowers the financial and technical hurdles for the interactive visualization of genotypic diversity data using DivBrowse for institutions that generate, store, and make genotypic data available for public access. Moreover, for species-agnostic bioinformatics file formats, DivBrowse can be used with any species that has a haploid or diploid genome.

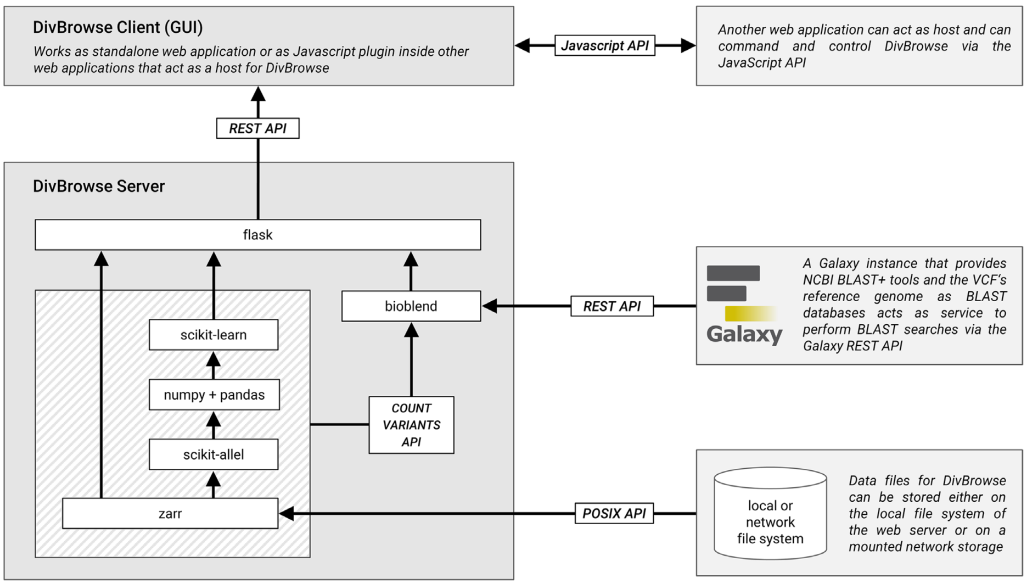
DivBrowse Client (GUI)
Works as standalone web application or as Javascript plugin inside other web applications that act as a host for DivBrowse
Javascript API
Another web application can act as host and can command and control DivBrowse via the JavaScript API
REST API
DivBrowse Server
flask
bioblend
REST API
Galaxy
A Galaxy instance that provides NCBI BLAST+ tools and the VCF's reference genome as BLAST databases acts as service to perform BLAST searches via the Galaxy REST API
scikit-learn
numpy + pandas
scikit-allel
zarr
COUNT VARIANTS API
POSIX API
local or network file system
Data files for DivBrowse can be stored either on the local file system of the web server or on a mounted network storage

Figure 3: The general architecture of DivBrowse showing the flow of data between the different components and between the server component and the storage layer.

Due to the progressing genetic inventory of plant genetic resources stored in genebanks in the frame of genebank genomics projects, we expect a significantly increasing amount and size of genotypic datasets in the future, whose easy accessibility and visualization are of great importance [32]. We have shown that DivBrowse is capable of delivering diversity matrices with thousands of genotypes and hundreds of millions of variants on standard server hardware or even virtual machines. It thus provides a reusable tool to efficiently deliver the increasing amount of genotypic data to the public in the form of an interactive visualization with easy to use basic analysis functions and multiple entry points.

The VCF, which acts as the input file format for genotypic data for DivBrowse, has suffered from a number of indeterminacies in the past with regard to the indable, Accessible, Interoperable, Reusable (FAIR) criteria [33, 34]. These indeterminacies have made fully automated processing difficult or impossible without manual intervention in terms of interoperability at the level of bioinformatics toolchains. As DivBrowse continues to evolve, we look forward to incorporating recent advances and improvements related to metadata in VCF files to better take advantage of the FAIR data paradigm, such as improved interoperability and reusability [34]. For example, the usage of standardized sample IDs for the genotypes based on BioSamples-IDs in the VCF metadata section would allow DivBrowse to easily read those BioSamples-IDs and use them to automatically interconnect the genotypes in the GUI of DivBrowse with other omics-related web-based information systems [35]. As another example, the appropriate standardized usage of the VCF metadata field "*Contig*," as recommended in Beier et al. [34], would allow one to automatically derive human-readable chromosome labels, the chromosome length, and the species from the VCF file. This would make corresponding manual entries for the chromosome labels and length in the DivBrowse configuration file unnecessary in the future. Monomorphic variants are often filtered out of VCF files by variant calling pipelines, even though they may contain information relevant to clinical applications [36]. DivBrowse applies no prior filtering to the content of the provided VCF file. Therefore, monomorphic variants are displayed together with polymorphic variants if they are contained in the VCF file.

Today, APIs are an integral part in modern data-driven life science [37]. They allow programmatic and automatable access to data of any kind. In terms of smooth interoperability between different institutions and their data-holding systems, standardized and community-accepted APIs are of particularly central importance. One such standardized API in the field of plant breeding is the Breeding API specification project (BrAPI) [38]. BrAPI, as an attempt to establish a widely accepted application programming interface for plant breeding applications and related use cases, is under ongoing development by a global community of scientists of institutes and companies related to plant sciences and plant breeding [38]. Therefore, it will also be a great opportunity to integrate API calls of BrAPI related to genotypic data to support and improve the interoperability between different existing omics-related information systems and data warehouses for germplasm, genebank material, and breeding material. Another interesting feature would be to enhance DivBrowse to be capable of reading data from BrAPI endpoints that serve genotypic data. Thus, DivBrowse could deliver genotypic data through its server-side component via BrAPI and consume and visualize genotypic data via an external BrAPI endpoint through the client-side GUI.

One major limitation is that only single-nucleotide polymorphisms can be visualized. It would also be advantageous to visualize structural variations such as insertions, deletions, copy number variations, and translocations. However, at the moment of writing this article, the VCF specifications regarding structural variations were still under revision by the specification maintainers [39]. A new, stable version of the VCF specification regarding structural variations is planned for version 4.4, which has not yet been released at the time of writing this article.

Another limitation of DivBrowse is that it can currently only visualize haploid and diploid genomes. In the case of allopolyploidy, it is possible in most cases to dissect the genome into diploid subgenomes that can be then visualized as independent chromosomes. For example, the hexaploid wheat genome is such a case, because its genome consists of 3 diploid subgenomes with 7 chro-

Table 1: Feature comparison between different genome and variant call visualization tools. "+" means that the feature is available; "(+)" means that the feature is partially implemented or was not working at the time it was checked in the online demo; "–" means that the feature is not provided by the software. [1]SNPversity only accepts HDF5 files as input for variant matrices; for genome annotations, it accepts GFF3 files as input. [2]SNPversity does not provide a dedicated track for genome annotations or features, but it can display IDs of genes as text labels if a variant is located within the bounds of a gene. [3]SNPversity offers export as VCF files, but at the time of the test, this function did not work and a "Proxy Error" message from the webserver occurred. [4]DivBrowse supports exporting of scatterplots as a result of a principal component analysis as PNG image files.

Criterion or feature (category)	JBrowse2	igv.js	SnpHup	Flapjack	SNPversity	DivBrowse
Usage of standardized bioinformatics file formats (A)	+	+	+	–	(+)[1]	+
Genome track to show position on the assembly (B)	+	+	–	+	–	+
Display of genome annotation/features (B)	+	+	–	–	(+)[2]	+
Display of quantitative trait loci (QTL) (B)	+	+	–	+	–	–
Display of variant calls (B)	+	+	+	+	+	+
Display of variant calls metadata (e.g., read depth) (B)	+	+	+	–	–	+
Filtering of variants by minor allele frequency (C)	–	–	+	–	–	+
Filtering of variants by missing rate (C)	–	–	+	+	–	+
Filtering of variants by heterozygosity frequency (C)	–	–	–	+	–	+
Calculation and display of minor allele frequency (D)	–	–	–	–	–	+
Calculation and display of heterozygosity frequency (D)	–	–	–	–	–	+
Calculation and display of Principal coordinate analysis (PCA/PCoA) (D)	–	–	–	+	–	+
Calculation and display of distance matrices (D)	–	–	–	+	–	–
Calculation and display of phylogenetic tree (D)	–	–	+	+	–	–
Export of variant calls in VCF (E)	–	–	–	–	(+)[3]	+
Export of genome annotations as GFF files (E)	+	–	–	–	–	+
Export if images or diagrams (SVG, PNG) (E)	+	+	+	+	–	(+)[4]
Installation as desktop software (F)	+	–	–	+	–	–
Setup/installation as web-based standalone tool (F)	+	+	+	–	+	+
Integration as plugin into existing web-based tools (F)	+	+	+	–	+	+

mosomes each [40]. The wheat genome can thus be visualized with DivBrowse by dividing it into 21 diploid chromosomes. But there are, of course, plant cultivars or animal species that have true higher ploidy levels. As a consequence, there is still potential for future research about how genomes with higher levels of ploidy can be visualized in a meaningful and effective way.

A possible future extension regarding integration with existing software could be, for example, to make DivBrowse available as a plugin for Galaxy [41], so that genomic diversity data can be visualized and analyzed directly within a Galaxy workflow.

Use cases and workflow scenarios

Use case "A tool to visualize data of genebank genomics projects"

Genebank genomics projects create a vast amount of genotypic and phenotypic data [11, 32]. DivBrowse can be one particular tool to serve and visualize the genotypic diversity stored in genebanks and make this data foundation easily and interactively accessible from every personal computer with an Internet connection and an installed web browser. This reduces the barrier to accessing genotypic diversity data in general and makes it easier and faster for non-bioinformaticians to access critical information. As a reusable and configurable application that uses standardized and established bioinformatics file formats like VCF and GFF3, it makes the installation and setup of instances for many different plant species easy and affordable. The deployment of multiple DivBrowse instances for different plant species can be automated by customized Shell scripts or by pipelines for continuous integration and deployment based on GIT repositories [42, 43].

Use case "plugin in existing web-based information systems"

While DivBrowse can be used as a standalone tool, it also plays very well if being integrated in existing web applications that are focused on visualization and analysis of omics data. One example for such integration is the BRIDGE web portal, where collections of germplasm derived from a search by passport attributes can be directly and seamlessly transferred to the integrated DivBrowse instance, to only visualize the variant calls of those genotypes derived from the previously defined collection of germplasm [14]. Conversely, it is possible to create new germplasm collections from within the DivBrowse plugin by transferring corresponding sample IDs to the BRIDGE web portal via an API call from DivBrowse. As an example, users are able to create new germplasm collections in the BRIDGE web portal by selecting genotypes in the result of an interactive PCA via a lasso selection in DivBrowse. We can imagine that likewise, other existing web applications in the field of plant genomics, like Gigwa, Germinate, Ensembl, or GrainGenes, could be functionally enriched in a similar way using DivBrowse as a plugin for variant visualization and analysis [44–47].

Use case "fast insights into the genomic diversity of genes and genomic regions"

Geneticists and other scientists interested in available genetic variants of a specific gene can use DivBrowse to get insights about how many variants for a gene or a genomic range are available and which genotypes are carrying specific alleles. This could be useful to support the understanding of the genetic diversity of specific traits or to get an overview about the genotypic diversity of a specific genomic range. Furthermore, if the genotypic data are of high enough resolution, DivBrowse allows the calculation of a principal component analysis to gain initial insights into the population structure of a gene or genetic feature. The integrated BLAST entry point is useful to find variants based on a given nucleotide or protein sequence [25]. The variants within the genomic region determined by a BLAST can then be quickly and easily exported as a VCF file and are thus available for further processing steps within a genome editing workflow. This feature can support the design of single-guide RNA in a CRISPR/Cas9 genome editing experiment [48, 49].

Comparison with existing tools

We compared DivBrowse in terms of features with existing tools that offer similar functionality and were published before this work. We included only those software tools in the comparison for which either a working online demo or a downloadable and installable version is available. Because there are a variety of bioinformatics tools for visualizing genomic diversity data, only a limited selection of software tools can be compared here. We ended up comparing the following tools with DivBrowse: JBrowse2 [6], igv.js [7], SnpHub [8], Flapjack [10], and SNPversity [9]. Based on our experiences, we selected a list of comparison criteria, focusing on features useful for the visualization and analysis of genomic biodiversity, which can be grouped into the following categories: data import (A); visualization and display of genome, variants, and variant calls (B); filtering of variants (C); calculation and visualization of summary statistics (D); data export (E); and installation possibilities (F). The result of our comparison, including the assignment to the 6 mentioned categories, is summarized in Table 1.

This comparison is intended to assist users in selecting a visualization tool suitable for their use case. Since software is usually constantly being developed further, it should be noted that this comparison is only valid for the time of its creation.

Availability of Source Code and Requirements

- Project name: DivBrowse
- Project homepage: [51]
- Code repository: [52]
- Operating system: Linux, macOS, Windows
- Programming language: Python 3.9, JavaScript
- Other requirements: conda, docker or podman
- License: MIT
- RRID: SCR_022,780
- biotools ID: divbrowse

Additional Files

Supplementary Fig. S1. The architecture of the DivBrowse server. The divbrowse package imports and uses numerous packages from the Python Standard Library as well as third-party packages that are published on the Python Package Index [31].
Supplementary Fig. S2. The architecture of the DivBrowse client. The main entry point is the Svelte component "App," which holds the basic layout of the GUI, which itself is divided into multiple modules and components. There are 2 main top-level modules: Components and Utilities. Components inside of the Components module are all Svelte components that consist of logic directly responsible for the presentation layer of the GUI. Components inside of the Utilities module are plain Javascript components that consist of cross-component business logic (like the ApiController) and helper functions.
Supplementary Table S1. Information about the resources of the REST-API.
Supplementary Table S2. Information about the methods of the Javascript-API.

Abbreviations

API: Application Programming Interface; ASCII: American Standard Code for Information Interchange; BLAST: Basic Local Alignment Search Tool; BrAPI: Breeding API; CLI: command-line interface; GFF3: General Feature Format Version 3; GUI: Graphical user interface; NCBI: The National Center for Biotechnology Information; PCA: principal component analysis; VCF: Variant Call Format.

Authors' Contributions

N.S., M.M., and U.S. designed the study. N.S. supervised the experiments and M.M. the data analysis for the barley datasets. P.K. designed and developed application software. P.K. wrote the manuscript with contributions from all coauthors. All authors read and approved the final manuscript.

Competing Interests

The authors declare that they have no competing interests.

Data Availability

All supporting data and materials are available in the *GigaScience* GigaDB database [50].

Funding

This work was supported by the Leibniz Association (Pakt für Forschung und Innovation: SAW-2015-IPK-1 "BRIDGE" to U.S., M.M., N.S.) and the German Ministry of Education and Research (BMBF) (grant 031A536A "de.NBI" to U.S., grant 031B0884A "SHAPE-II" to N.S., U.S., and M.M.). Costs for open access publishing were partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation, grant 491250510).

Acknowledgments

We thank J. Bauernfeind, T. Münch, and H. Miehe for the administration of the IT infrastructure that is used to host the demo instances.

References

1. Christensen, K, Dukhovny, D, Siebert, U, *et al.* Assessing the costs and cost-effectiveness of genomic sequencing. *J Pers Med* 2015;**5**:470–86.
2. Bayle, A, Droin, N, Besse, B, *et al.* Whole exome sequencing in molecular diagnostics of cancer decreases over time: evidence from a cost analysis in the French setting. *Eur J Health Econ* 2021;**22**:855–64.
3. Korte, A, Farlow, A. The advantages and limitations of trait analysis with GWAS: a review. *Plant Methods* 2013;**9**:29.
4. Stephens, ZD, Lee, SY, Faghri, F, *et al.* Big data: astronomical or genomical? *PLOS Biol* 2015;**13**:e1002195.
5. Grüning, BA, Lampa, S, Vaudel, M, *et al.* Software engineering for scientific big data analysis. *Gigascience* 2019;**8**:giz054.
6. Diesh, C, Stevens, GJ, Xie, P, *et al.* JBrowse 2: a modular genome browser with views of synteny and structural variation. *Biorxiv*. 2022. https://doi.org/10.1101/2022.07.28.501447.
7. Robinson, JT, Thorvaldsdottir, H, Turner, D, *et al.* igv.js: an embeddable JavaScript implementation of the Integrative Genomics Viewer (IGV).*Bioinformatics* 2023;**39**:(1):1–2.
8. Wang, W, Wang, Z, Li, X, *et al.* SnpHub: an easy-to-set-up web server framework for exploring large-scale genomic variation data in the post-genomic era with applications in wheat. *Gigascience* 2020;**9**:giaa060.
9. Schott, DA, Vinnakota, AG, Portwood, JL, *et al.* SNPversity: a web-based tool for visualizing diversity. *Database* 2018;**2018**:1–9.
10. Milne, I, Shaw, P, Stephen, G, *et al.* Flapjack–graphical genotype visualization. *Bioinformatics* 2010;**26**:3133–4.
11. Milner, SG, Jost, M, Taketa, S, *et al.* Genebank genomics highlights the diversity of a global barley collection. *Nat Genet* 2019;**51**:319–26.
12. Danecek, P, Auton, A, Abecasis, G, *et al.* The variant call format and vcftools. *Bioinformatics* 2011;**27**:2156–8.
13. Stein, L. Generic feature format version 3 (GFF3). *GitHub*. 2020. https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md. Accessed 2022 May 18.
14. König, P, Beier, S, Basterrechea, M, *et al.* BRIDGE—a visual analytics web tool for Barley Genebank Genomics. *Front Plant Sci* 2020;**11**: 1–15.
15. Mascher, M. Variant matrices for a global barley diversity panel. e!DAL—Plant Genomics and Phenomics Research data Repository (PGP). 2018. http://dx.doi.org/10.5447/IPK/2018/9. Accessed 2022 April 15.
16. Monat, C, Padmarasu, S, Lux, T, *et al.* TRITEX: chromosome-scale sequence assembly of Triticeae genomes with open-source tools. *Genome Biol* 2019;**20**:(284):1–18.
17. König, P. DivBrowse project website. 2023. https://divbrowse.ipk-gatersleben.de. Accessed 2023 Feb 2.
18. de Mast, J, Kemper, BPH. Principles of exploratory data analysis in problem solving: what can we learn from a well-known case? *Qual Eng* 2009;**21**:366–75.
19. Cingolani, P, Platts, A, Wang, LL, *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 (Austin)* 2012;**6**:80–92.
20. Miles, A, Jakirkham, Bussonnier M, Moore, J *et al.*, Zarr-Python. *Zenodo*. 2022. https://zenodo.org/record/6419641. Accessed 2022 May 20.
21. Moore, J, Allan, C, Besson, S, *et al.* OME-NGFF: a next-generation file format for expanding bioimaging data-access strategies. *Nat Methods* 2021;**18**:1496–8.
22. Miles, A, Pyup.Io, Bot, Murillo, R, *et al. cggh/scikit-allel: v1.3.3*. *Zenodo*. 2021. https://zenodo.org/record/4759368. Accessed 2022 May 20.
23. Li, H. Tabix: fast retrieval of sequence features from generic TAB-delimited files. *Bioinformatics* 2011;**27**:718–9.
24. Ben-Kiki, O, Evans, C. döt Net I: YAML^{TM} Specification index. 2009. https://yaml.org/spec/. Accessed 2022 Jul 1.
25. Altschul, SF, Gish, W, Miller, W, *et al.* Basic local alignment search tool. *J Mol Biol* 1990;**215**:403–10.
26. Sloggett, C, Goonasekera, N, Afgan, E. BioBlend: automating pipeline analyses within Galaxy and CloudMan. *Bioinformatics* 2013;**29**:1685–6.
27. Cock, PJA, Chilton, JM, Grüning, B, *et al.* NCBI BLAST+ integrated into Galaxy. *Gigascience* 2015;**4**:39.
28. Nei, M. *Molecular Evolutionary Genetics*. New York: Columbia University Press, 1987.
29. Binns, D, Dimmer, E, Huntley, R, *et al.* QuickGO: a web-based tool for gene ontology searching. *Bioinformatics* 2009;**25**:3045–6.
30. Harris, R. Svelte. *GitHub*. 2022. https://github.com/sveltejs/svelte. Accessed 2022 May 18.
31. Python Software Foundation: python Package Index—PyPI. 2023. https://pypi.org/. Accessed 2023 Mar 15.
32. Mascher, M, Schreiber, M, Scholz, U, *et al.* Genebank genomics bridges the gap between the conservation of crop diversity and plant breeding. *Nat Genet* 2019;**51**:1076–81.
33. Wilkinson, MD, Dumontier, M, Aalbersberg, IJ, *et al.* The FAIR Guiding Principles for scientific data management and stewardship. *Sci Data* 2016;**3**:(160018):1–9.
34. Beier, S, Fiebig, A, Pommier, C, *et al.* Recommendations for the formatting of Variant Call Format (VCF) files to make plant genotyping data FAIR. *F1000Res* 2022;**11**:231.
35. Courtot, M, Gupta, D, Liyanage, I, *et al.* BioSamples database: fAIRer samples metadata to accelerate research data management. *Nucleic Acids Res* 2022;**50**:D1500–7.
36. Ferrarini, A, Xumerle, L, Griggio, F, *et al.* The Use of Non-Variant Sites to Improve the Clinical Assessment of Whole-Genome Sequence Data. *PLoS One* 2015;**10**:1–15.
37. Woody, SK, Burdick, D, Lapp, H, *et al.* Application programming interfaces for knowledge transfer and generation in the life sciences and healthcare. *npj Digit Med* 2020;**3**:(24):1–5.
38. Selby, P, Abbeloos, R, Backlund, JE, *et al.* BrAPI—an application programming interface for plant breeding applications.*Bioinformatics* 2019;**35**:(20):4147–4155.
39. Cameron, D. Improved structural variant support by d-cameron · pull request #465 · samtools/hts-specs. 2019. https://github.com/samtools/hts-specs/pull/465. Accessed 2022 Jul 8.
40. The International Wheat Genome Sequencing Consortium (IWGSC), Appels, R, Eversole, K, *et al.* Shifting the limits in

wheat research and breeding using a fully annotated reference genome. *Science* 2018;**361**:661:1–13.

41. Goecks, J, Nekrutenko, A, Taylor, J, *et al.* Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol* 2010;**11**:R86.
42. Perez-Riverol, Y, Gatto, L, Wang, R, *et al.* Ten Simple Rules for Taking Advantage of Git and GitHub. *PLoS Comput Biol* 2016;**12**:1–12.
43. Gruening, B, Sallou, O, Moreno, P, *et al.* Recommendations for the packaging and containerizing of bioinformatics software. *F1000Res* 2019;**7**:742.
44. Sempéré, G, Pétel, A, Rouard, M, *et al.* Gigwa v2—Extended and improved genotype investigator. *Gigascience* 2019;**8**:giz051.
45. Raubach, S, Kilian, B, Dreher, K, *et al.* From bits to bites: advancement of the Germinate platform to support prebreeding informatics for crop wild relatives. *Crop Sci* 2021;**61**:1538–66.
46. Cunningham, F, Allen, JE, Allen, J, *et al.* Ensembl 2022. *Nucleic Acids Res* 2022;**50**:D988–95.
47. Yao, E, Blake, VC, Cooper, L, *et al.* GrainGenes: a data-rich repository for small grains genetics and genomics. *Database* 2022;**2022**:1–11.
48. Wang, H, La Russa, M, Qi, LS. CRISPR/Cas9 in genome editing and beyond. *Annu Rev Biochem* 2016;**85**:227–64.
49. Jiang, F, Doudna, JA. CRISPR–Cas9 structures and mechanisms. *Annu Rev Biophys* 2017;**46**:505–29.
50. König, P, Beier, S, Mascher, M, *et al.* Supporting data for “DivBrowse—Interactive Visualization and Exploratory Data Analysis of Variant Call Matrices.” *GigaScience Database*. 2023. http://dx.doi.org/10.5524/102358.
51. https://academic.oup.com/gigascience/article/doi/10.1093/gigascience/giad020/7113330?searchresult=1#401289755.
52. DivBrowse. Github repository. https://github.com/IPK-BIT/divbrowse. accessed 14/04/2023.