GigaByte, 2024, 1–10

doi: xx.xxxx/xxxx
Manuscript in Preparation
Technical Release

(GIGA)byte

TECHNICAL RELEASE

SMARTER-database: a tool to integrate SNP array datasets for sheep and goat breeds

Paolo Cozzi[1,*], Arianna Manunza[1], Johanna Ramirez-Diaz[1], Valentina Tsartsianidou[2,3], Konstantinos Gkagkavouzis[2,3], Pablo Peraza[4], Anna Maria Johansson[5], Juan José Arranz[6], Fernando Freire[7], Szilvia Kusza[8], Filippo Biscarini[1], Lucy Peters[9], Gwenola Tosser-Klopp[9], Gabriel Ciappesoni[4], Alexandros Triantafyllidis[2,3], Rachel Rupp[9], Bertrand Servin[9] and Alessandra Stella[1]

[1]Institute of Agricultural Biology and Biotechnology, National Research Council, Milano, Italy and [2]Department of Genetics, Development & Molecular Biology, School of Biology, Aristotle University of Thessaloniki, Greece and [3]Genomics and Epigenomics Translational Research (GENeTres), Center for Interdisciplinary Research and Innovation (CIRI-AUTH), Greece and [4]Sistema Ganadero Extensivo, Instituto Nacional de Investigación Agropecuaria, INIA Las Brujas, Uruguay and [5]Department of Animal Breeding and Genetics, Swedish University of Agricultural Sciences, Uppsala, Sweden and [6]Departamento de Producción Animal, Facultad de Veterinaria, Universidad de León, León, Spain and [7]OVIGEN, Zamora, Spain and [8]Centre for Agricultural Genomics and Biotechnology, University of Debrecen, Debrecen, Hungary and [9]GenPhySE, Université de Toulouse, INRAE, ENVT, Castanet-Tolosan, France

[*]paolo.cozzi@ibba.cnr.it

Abstract

Underutilized sheep and goat breeds have the ability to adapt to challenging environments due to their genetic composition. Integrating publicly available genomic datasets with new data will facilitate genetic diversity analyses; however, this process is complicated by important data discrepancies, such as outdated assembly versions or different data formats. Here we present the SMARTER-database, a collection of tools and scripts to standardize genomic data and metadata mainly from SNP chips arrays on global small ruminant populations with a focus on reproducibility. SMARTER-database harmonizes genotypes for about 12,000 sheep and 6,000 goats to a uniform coding and assembly version. Users can access the genotype data via FTP and interact with the metadata through a web interface or programmatically using their custom scripts, enabling efficient filtering and selection of samples. These tools will empower researchers to focus on the crucial aspects of adaptation and contribute to livestock sustainability, leveraging the rich dataset provided by the SMARTER-database.

Availability & Implementation

The code is available as open source software under the MIT license at `https://github.com/cnr-ibba/SMARTER-database`.

Key words: sheep; goat; genotypes; adaptation; REST-API; standardization; reproducibility; database

Statement of Need

Background

The presence of small ruminant populations is crucial to the socio-economic prosperity of human settlements, particularly in European marginal regions. In these areas, sheep and goat breeds that are not fully utilized have the potential to significantly increase the profitability of small ruminant farming. The reason for their value is their distinctive and often unusual genetic composition (e.g. [1, 2]), which renders them an exceptionally valuable resource for adapting to challenging environments, withstanding harsh farming conditions, combating biotic and abiotic stressors, and producing high-quality animal-derived products. In this context, the SMARTER project[3] developed innovative strategies to improve resilience and efficiency related traits of sheep and goats in diverse environments. Here we present the SMARTER-database, a collection of tools and scripts to gather, standardize and provide the scientific community with a comprehensive dataset of genomic data and metadata information on worldwide small ruminant populations. Existing datasets were scouted from public repositories and complemented with newly produced data within the context of the SMARTER project[3]. Our system provides a single entry point and standardization tools to explore genetic diversity and demography of goat and sheep breeds, and to understand the genetic basis of resilience and adaptation, especially in under-utilized breeds.

Data composition

SMARTER-database is mainly composed of two types of data: i. genotype data, derived from low/high density genome chips and Whole Genome Sequencing (WGS); ii. phenotype data, including a wide range of information such as GPS data relative to sampled populations, morphological description of animals and other production measurements. The objective of our work was to integrate numerous publicly available and newly generated datasets into a single entry point. For genotype datasets, the process of data integration is complicated by the fact that several public datasets were generated years ago and refer to outdated genome assembly versions, using different variant names assigned by different SNP array manufacturers, or encoding the same information in different ways. Phenotype data is even more heterogeneous. Missing information is hard to retrieve since animals may no longer be available. Phenotypic data collection often aligns closely with specific experimental objectives, yet descriptions of such data seldom adhere to validated ontological frameworks (e.g. PATO - the Phenotype And Trait Ontology[4]). This lack of standardization complicates cross-study comparisons, hindering robust cross-referencing and comprehensive analytical endeavors. Since the focus of the SMARTER project was on adaptation, a minimal set of attributes was identified for newly collected data: country of origin, breed name, internal IDs used in the genotype file, GPS coordinates and the main purpose of the breed when available (dairy, meat, wool). We however encourage submission of any type of information relevant to resilience, efficiency and adaptation potential.

Since most of the data are derived from SNP chip arrays, the SMARTER-database is structured primarily around this type of information. WGS data, which represents a smaller fraction of the total dataset, is filtered and integrated by aligning the SNPs present in the SNP arrays. While WGS data can provide deeper insights into genetic variation across species, including the ability to perform complex genomic analyses such as pangenome analysis due to recent advances in sequencing technologies [5], bead-chip genotyping remains a highly valuable tool. This is largely because of its cost-effectiveness in analyzing large populations, especially in studies involving livestock or animal breeding, and the fact that SNP arrays yield highly accurate and reproducible genotypes. In contrast, WGS data at low coverage often results in numerous false homozygote calls due to insufficient sequencing depth, whereas SNP arrays are specifically designed to minimize such errors, particularly for large-scale genotyping studies[6, 7].

To integrate genotype datasets generated at different times and with different technologies, all files must first be converted to the same format. The majority of data submissions were formatted in PLINK[8], followed by Illumina and Affymetrix formats. Most software packages lack support for proprietary file formats, such as Illumina's row files or Affymetrix's cell data files. Consequently, prior to merging, these proprietary files must undergo conversion into universally accepted formats.

Furthermore, it is imperative to ensure consistency in the reference genome assembly across all datasets. Discrepancies in assembly versions can lead to variations in SNP positions. Even within the same assembly, SNPs may exhibit divergent locations if the mapping procedures between two genotyping array types are not identical[9, 10]. Reliance solely on SNP names was found to be inadequate, given that SNPs from different manufacturers frequently possess unique identifiers and may lack comprehensive information required for unambiguous SNP identification across diverse datasets.

The encoding of genotypes presents another compatibility challenge, as two different formats may be employed to convey identical SNP information. SNPs are identified by aligning a SNP probe (comprising a brief DNA sequence around the SNP) to the reference genome, which may be on either the forward or reverse strand across various genome assemblies. As a result, the same SNPs can be represented by different bases: for example, an [A/G] polymorphism becomes [T/C], if the probe matches on different strands. In such cases, the FAO Guidelines for Genomic characterization of animal genetic resources[11] recommend to reverse the SNP before merging the datasets, and to exclude the SNP when the first base is complementary to the second (ie [A/T] - [C/G]), since in these cases it is not possible to verify if the SNP is on the same strand on both assemblies without additional information. Besides, in the A/B format the letters A and B don't represent the real genotype but the first and second letters of the SNP recorded in the manifest file: in such cases it's mandatory to get access to the information in the same manifest file used in the genotyping process. To avoid problems caused by probe alignment and ensure consistent SNP representation regardless of genome version, the company Illumina has proposed a TOP/BOTTOM format[12], which relies on the SNP probe sequences themselves, rather than on probe alignments as in the forward/reverse convention. This solution is not widely adopted: for example, to submit data to EBI-EVA[13], one must provide SNPs in forward orientation with respect to the reference genome (the reference allele needs to match the reference genome in the same position[14]). In 2015, we responded to the need to standardize SNP data[15] by developing SNPchiMp[16] and a series of tools to work with different data sources and to convert data in the same format[17]. In SMARTER, we followed up on the same concept: development of tools to convert genomic data to a reference format (i.e. Illumina TOP), and then

Compiled on: September 17, 2024.
Draft manuscript prepared by the author.

Table 1. Genotype conversion of DU186191_327.1 (A/G) SNP for four different samples in the SMARTER database. The source version is the assembly version of the received samples, and source coding and genotype are the inferred coding and the received genotype respectively. The TOP genotype is the final genotype present in database according to Illumina TOP[12] convention

smarter_id	source version	source coding	source genotype	TOP genotype
UYOA-CRR-000003890	Oar_v4.0	forward	T C	A G
UYOA-CRL-000000382	Oar_v3.1	A/B	A B	A G
NAOA-ADP-000001020	Oar_v3.1	top	G A	G A
GROA-CHI-000004137	Oar_v3.1	forward	T T	A A

Table 2. Genotype conversion of OAR1_103790218.1 C/G SNP for four different samples in the SMARTER database. The source version is the assembly version of the received samples, and source coding and genotype are the inferred coding and the received genotype respectively. The TOP genotype is the final genotype present in database according to Illumina TOP[12] convention

smarter_id	source version	source coding	source genotype	TOP genotype
UYOA-CRR-000003890	Oar_v4.0	forward	G G	G G
UYOA-CRL-000000382	Oar_v3.1	A/B	B B	G G
NAOA-ADP-000001020	Oar_v3.1	top	G G	G G
GROA-CHI-000004137	Oar_v3.1	forward	C C	G G

giving the possibility to convert the data according to user needs, such as publishing data to public repositories or before merging data with other datasets encoded in a different way. An example of genotype conversion for *unambiguous* and *ambiguous* SNP is presented in Table 1 and Table 2 respectively. During the genotype data import, genotypes are converted to Illumina TOP, in order to refer to the same SNP coding across samples from different datasets. It is important to note that after the Illumina TOP conversion, the only available genotypes will be [A/T], [C/G], [A/G], or [A/C], as all possible allele combinations can be converted to one of these four.

Collecting phenotypes and other metadata

Besides genotype data, other types of data can also be useful when processing and analyzing genotypes in the context of adaptation and genomic selection. Genotype file formats are not suitable for the inclusion of data like geographical coordinates, or morphological description of animals. Even more advanced standards like Variant Calling Format (VCF) provide an INFO field supporting user defined data[18], accessing this kind of information is inefficient because this field cannot be indexed. Preferably, metadata information should be stored in a dedicated resource where it can easily be queried. In the context of the SMARTER project, a schemaless database was more desirable since metadata don't follow a standard format. A minimal set of requirements should be provided in order to make useful queries: for instance, breed, country and GPS locations are required in order to retrieve data relevant for adaptation analyses.

Methods

The SMARTER-database project

In order to standardize data and merge genotypes from different datasets, we collected information from SNP chip manufacturers both by accessing publicly available manifest files and by directly contacting the manufacturer in case of custom manifest files. Data stored in the SNPchiMp database[16] was also collected as a unique source of information for SNP positions and code conversion. In addition, information from public databases like dbSNP[19] and EVA[20] is integrated in order to provide external reference IDs to SNPs. To store metadata information, we decided to employ a MongoDB[21] database. This type of database, characterized by its schemaless nature and support for spatial queries, enables the flexible modeling of data with the capability to dynamically add or remove attributes as needed. In addition to sample information, the database is designed to also accommodate variant information, thereby facilitating genotype conversion through reliance on the database. The variant class model defined with the help of the MongoEngine[22] library, a Python-based Object-Document Mapper (ODM)[23] that facilitates working with MongoDB in an object-oriented way, is presented in Figure 1: a VariantSpecie abstract class defines all the attributes that can be referred to the same SNP, like the different names present in different chip manufacturers or external accession IDs. Multiple Locations are then embedded in the same document in order to track different positions across different assemblies with their specific allele coding. Finally, the VariantSpecie abstract class is extended by the VariantGoat and VariantSheep classes, which serve as the final Object Document Mappers[23] for each sheep and goat variant in the database. All information related to the same SNP is present in the same document allowing to track the same SNPs even when named differently by each technology. Moreover, it is possible to identify which SNPs are in common between different datasets and to query data relying on manufacturer's variant name, rsID and genomic locations. Finally, the MongoEngine[22] implementation defined some accessory methods which have no direct representation in MongoDB itself but can be exploited to determine the genotype coding or the genotype conversion of a provided SNP according to the assembly information present in the database. More information about the variant model is in the SMARTER-database documentation[24].

Although VCF[18] file format was proposed as a standard for the distribution of genotype information, we choose to first merge all genotypes we received in a unique PLINK binary file, one for goats and one for sheep. Proprietary file formats (e.g. illumina row and Affymetrix cell files) were converted to PLINK files and merged with other data using PLINK[8] software. The reason for adopting this format is related to its popularity in the research community: many software applications, libraries and pipelines focusing on adaptation or

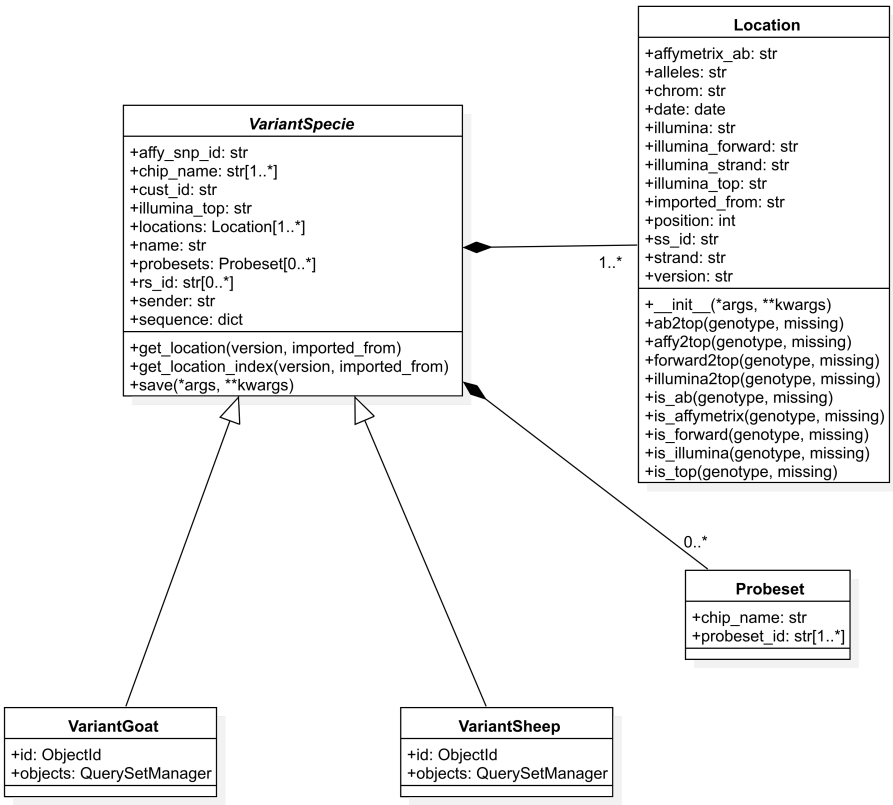

Figure 1. A UML diagram of the Variant classes model implemented in Python: the boxes in the diagram show class attributes and methods, with data types like `str` for string and `dict` for dictionary. The central `VariantSpecie` class serves as an *abstract* base class, containing common attributes and methods relevant to different species, such as *SNP IDs*, *chip names*, and *genotype data*. The `Location` class models assembly-specific information, including *chromosomal positions*, *alleles*, and *strand orientation*, and is embedded within the `VariantSpecie` class to handle multiple *locations* per variant. The `VariantSheep` and `VariantGoat` classes inherit from `VariantSpecie`, meaning they extend the base class by specializing it for sheep and goat data, respectively. This inheritance allows for efficient reuse of common functionality while tailoring specific attributes or behaviors for each species. The `Probeset` class, associated with `VariantSpecie`, represents specific data types that come from *Affymetrix* chips. It captures metadata linked to SNP probes, such as the *chip name* and *probeset ID*.

genetic diversity rely on this format. In addition, the much smaller size of the PLINK binary file makes it easy to manage (to subset, to index, and more in general to analyze) data using the PLINK software.

The Illumina TOP/BOTTOM coding convention was selected due to its reliance solely on the probe itself[12]. Its primary advantage lies in ensuring consistent encoding of SNPs across various assembly versions. Consequently, updating SNP positions to accommodate a new reference assembly merely entails a straightforward adjustment.

Different attributes linked to the same SNP, including varying names used by Affymetrix and Illumina or differences in coding types, are leveraged to produce a unified genotype file. This consolidated file guarantees SNP standardization across different datasets by assigning them uniform names and codes, thereby streamlining analyses across samples from various datasets. Furthermore, sample metadata stored in SMARTER-database helps to identify relevant samples, simplifying their extraction from the comprehensive genotype file relying on the same sample IDs.

The final intersection among all different chip types in sheep will be close to 30K SNPs, including various versions of Illumina 50K chip (which ranges between 50K and 60K SNPs depending on the release), HD chip and public and custom Affymetrix chips. This implies that at least 30K SNPs are theoretically shared across all 12K sheep samples, however all remaining SNPs are reported in the final genotype to support nearly all the 620K sheep SNPs managed by the database. Samples without information on SNPs outside the chip used to generate them will have missing data. The same approach was adopted for goats, however, the number of supported chips is lower resulting in a final intersection of nearly 93% of the goat supported SNPs. The data are provided as-is, with only minimal filtering applied based on Identical By State (IBS) to remove duplicate samples when there is overlap between background datasets. This is the only filtering applied to the final dataset; users are expected to apply their own filters after selecting the data they need, avoiding unnecessary filtering by missingness on samples that are irrelevant to their analyses. Standard Minor Allele Frequency (MAF) filters applied as a percentage cannot be applied to the entire dataset, as stated by the FAO guidelines[11], since we would lose all the variability associated with local adaptation, and MAF could change depending on different subsets of samples. Additionally, filtering based on assembly position or sex chromosomes could result in a loss of information: for example, when updating an assembly, unmapped SNPs might be mapped in the new assembly, while previously mapped SNPs could become unplaced. Ideally, a custom remapping of Illumina and Affymetrix probes against new genome assemblies could increase the intersection between different chip technologies and this can be added in a future release of the database. Sex chromosomes can also be informative, particularly for users interested in reproduction studies, therefore they aren't removed. Consequently, we believe that this dataset should be presented without any filtering, leaving it to the user to document all the steps needed to produce the final dataset required for their analyses.

Table 3. Amount of samples with GPS coordinates. Foreground samples are produced within the SMARTER project, while background samples are collected from already published datasets

	Total	With GPS
Foreground Sheep	5945	79,34%
Background Sheep	5892	82,30%
Foreground Goats	464	87,28%
Background Goats	5540	58,34%

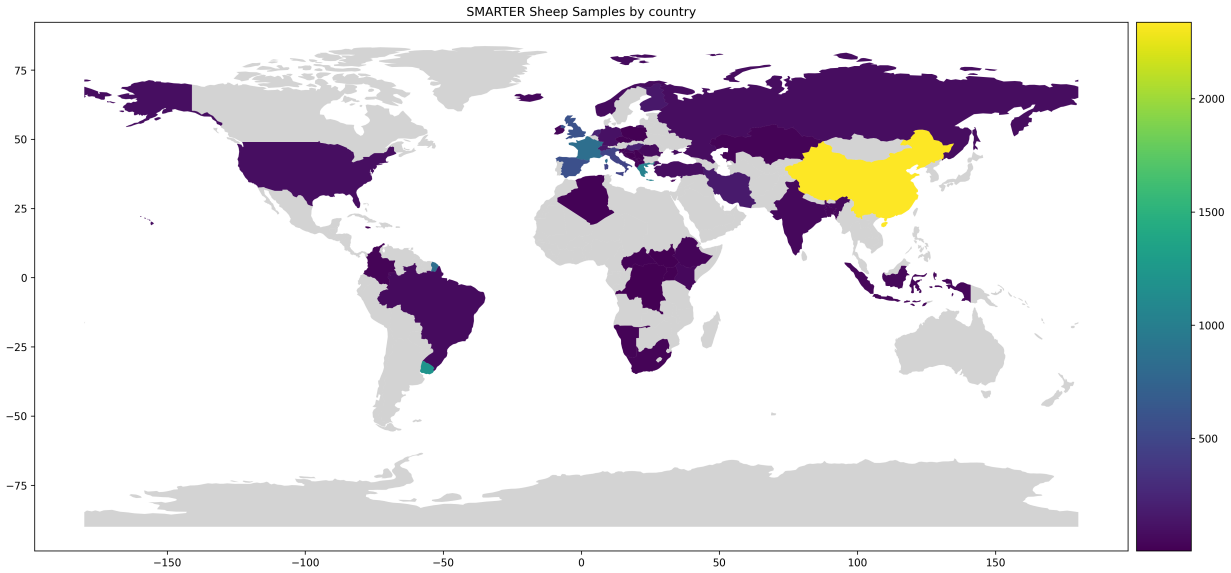

Figure 2. Geographical distribution of sheep samples in the SMARTER database. Countries are colored based on sample count using the viridis scale, where brighter colors represent higher sample counts.

Reproducibility

The code developed in the SMARTER-database project was enhanced to develop utilities used to maintain the database updated, such as adding new data sources, adding new breeds and samples and convert genotypes in the desired formats in order to produce the final genotype file, as described by our data import guide[25] in the project documentation. The idea behind this implementation was to adhere to the FAIR principles[26] by providing a reproducible and transparent workflow to create and manage the final dataset. Our project is based on the Cookiecutter Data Science Project[27], aiming to standardize data science projects for sharing purposes. It adheres to conventions outlined in the Cookiecutter framework[28], which includes organizing data into specific folders such as raw data, external data, processing, and final data. Additionally, it includes source folders for importing scripts and libraries, which can be installed as a Python package in a Conda[29] environment. All software dependencies are managed using Conda - a package and environment management system for Python - and tracked through requirements files. Furthermore, it introduces a database folder not found in the original Cookiecutter template. This folder manages database installation and initialization through Docker[30] and docker-compose[31], where Docker is a platform for containerizing applications to ensure they run consistently across different environments. Raw data undergoes initial exploration using IPython notebooks stored in the notebook folder, aiming to comprehend its structure and potential issues before importing data into the database: efforts are made to infer genotype coding, and to ensure all necessary information is available for genotype conversion and sample addition to the database. Subsequently, the dataset is integrated into the database, with updates to available breeds, if new are added, and the assignment of SMARTER unique and stable IDs to samples, which are used in the resultant genotype files. If metadata is provided, it is appended to the corresponding samples. Currently, the SMARTER database tracks information on approximately 12,000 sheep and 6,000 goats, with nearly 80% and 60% of samples possessing GPS coordinates for sheep and goats respectively, as indicated in Table 3. In Figure 2 and 3 a graphical representation of samples by country for sheep and goats respectively is reported. Figure 4 illustrates the distribution of 12,000 sheep samples based on the genotype technology employed. Despite variations in genotyping technologies, a subset of SNPs are in common: this enables comparison among samples from different datasets. All data processing steps, including environment setup, database initialization, and data processing, are handled using GNU Make[32] in order to provide simple commands to execute all the steps required to produce the final database. These steps, and the importing scripts executed by these steps, are idempotent, meaning that repeating the same command has no side effects, ensuring consistency in the final dataset. This also means that a new dataset can be added simply by updating the Makefile with the new required steps and then calling the proper make command again. The database follows Semantic Versioning[33] to track updates and changes effectively. This project is publicly accessible on GitHub[34], where GitHub workflows automate the process of running Continuous Integration (CI) tests which validate critical steps, such as genotype conversion, to ensure that everything functions correctly after each new code contributions.

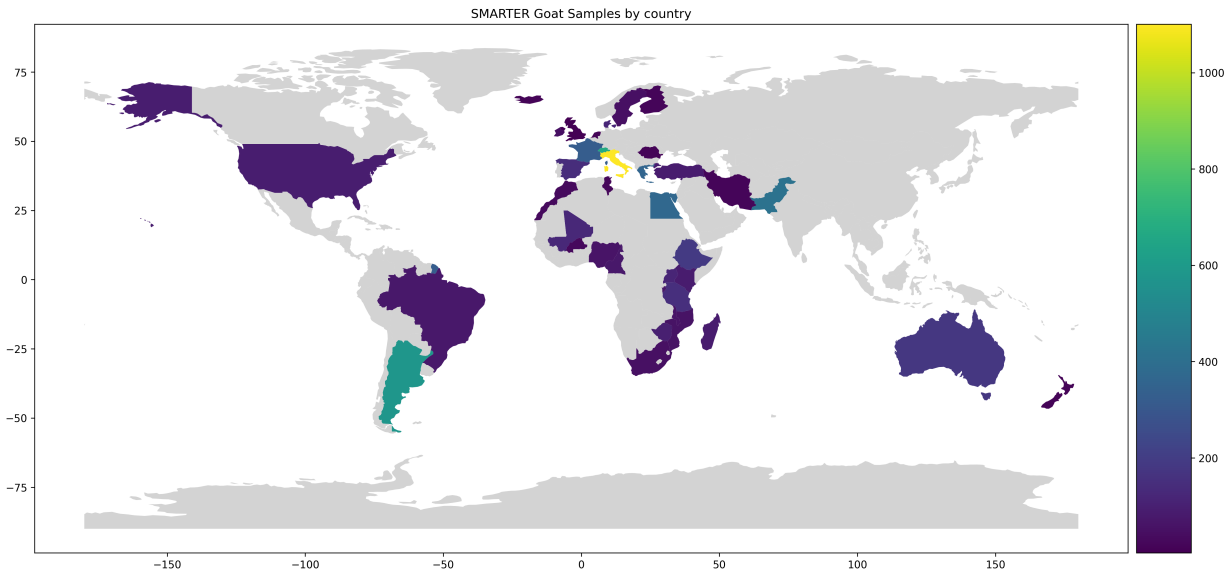

Figure 3. Geographical distribution of goat samples in the SMARTER database. Countries are colored based on sample count using the viridis scale, where brighter colors represent higher sample counts.

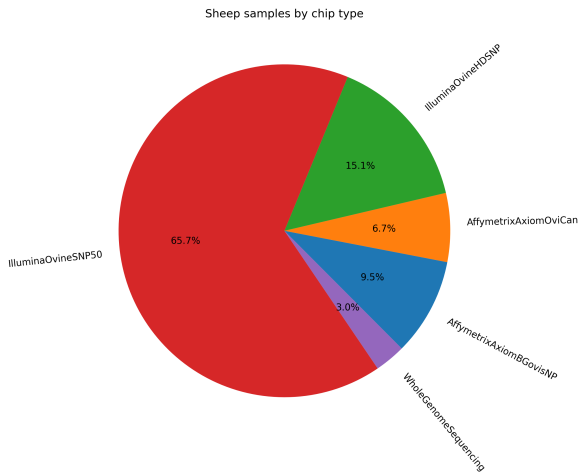

Figure 4. Distribution of sheep samples by genotyping technology

Figure 5. Structure of SMARTER-backend API URL. The endpoint part specifies the desired data type and the parameters let to filter out results

Table 4. SMARTER-backend available endpoints. The available data types are described in the SMARTER-database online documentation[24]

Endpoint suffix	Data type	Description
/auth/login	Users	User authentication (removed after public release)
/breeds	Breed	Available breeds as a list
/datasets	Dataset	Available datasets as a list
/info	SmarterInfo	Information about database status
/samples/sheep	SampleSheep	Sheep samples as a list
/samples/goat	SampleGoat	Goat samples as a list
/samples.geojson/sheep	GeoJSON	Sheep samples in GeoJSON format
/samples.geojson/goat	GeoJSON	Goat samples in GeoJSON format
/supported-chips	SupportedChip	Supported genotype platforms as a list
/variants/sheep/OAR3	VariantSheep	Supported sheep SNPs in Oar_v3.1 assembly
/variants/sheep/OAR4	VariantSheep	Supported sheep SNPs in Oar_v4.0 assembly
/variants/goat/ARS1	VariantGoat	Supported goat SNPs in ARS1.2 assembly
/variants/goat/CHI1	VariantGoat	Supported goat SNPs in CHIR_1.0 assembly

Data access

Data can be accessed through two distinct channels: the fully processed PLINK binary genotype files for both goat and sheep are accessible via FTPS, while sample information and metadata are retrievable through SMARTER-backend[35], a RESTful API interface[36]. Users are required to identify their desired samples using the API interface and then to extract their genotypes from the global file. This also means that the resource can be programmatically accessed, thereby enhancing reproducibility and facilitating data integration with other resources.

SMARTER-backend is a Python-Flask application built on top of the SMARTER-database MongoDB, offering a specific URL (i.e endpoint) to interact with each object stored in the SMARTER-database. Users can query for SNPs, samples, datasets, and breeds by making HTTP requests to the appropriate endpoint and providing the necessary parameters to obtain information in JSON format (Fig. 5). Endpoint parameters documentation is detailed using Swagger and can be accessed via the `/docs` location of the SMARTER-backend itself. Table 4 provides a list of the available endpoints. Figure 6 showcases an example of documentation for the `/sample/goat` endpoint, which enables users to gather information on goat samples. All available parameters are listed along with their descriptions and the supported data types. When the input data type is an array, users can provide the same parameter multiple times. For instance, it is possible to specify multiple breed codes in a single query to obtain all desired samples as illustrated above in Figure 5.

To facilitate data access for partners primarily focused on data analysis with R rather than retrieving data via REST interfaces, we developed the smarterapi R package[37]. This package aims to streamline data retrieval from SMARTER-backend by abstracting the complexities of handling HTTP requests and result pagination. It offers simple R functions that return data as R data frames, utilizing the same parameters described in the backend documentation. This package facilitates URL generation and data retrieval within R. To assist users, we created an online vignette outlining common operations such as data retrieval and filtering, as well as more advanced tasks like working with variants, conducting spatial queries, and extracting data from raster objects using WorldClim database[38]. These functionalities are valuable for performing landscape genomic analyses on SMARTER data.

The latest tool provided to the community is SMARTER-frontend[39], a web application developed in Angular[40] on the top of SMARTER-backend. This application allows users to browse SMARTER data and gain insights into database contents without the need for additional code or software installation. Regardless of the chosen method for collecting samples (API request, R package, or web application), users must identify their samples of interest and extract the required genotypes using PLINK.

Conclusions and future developments

The SMARTER database project provides valuable information on sheep and goat populations around the world. It is an essential tool for researchers, enabling them to generate new insights and offer the possibility to store new genotypes and drive progress in this field. Comprising a suite of scripts, it standardizes genotype data sourced from various methods and origins, resulting in a unified dataset primed for analysis across different assembly versions for both sheep and goats. Data access is granted to users through the SMARTER-backend, using R packages or the web interfaces, while genotypes are available over FTPS. The entire project was developed with the goal of facilitating reproducibility and programmatic data accessibility. The SMARTER database is open to additional dataset integration: given the schemaless nature of the database and the ability to collect data using the REST API, the most valuable contributions should include metadata, such as GPS coordinates and phenotypes. This type of data will contribute significantly to understanding adaptation and resilience in small ruminants. We plan to support additional assemblies with the option to collect data in VCF format. This will help the community reuse this data and provide the opportunity to upload genotypes to public archives like EBI-EVA. Moreover, by utilizing Illumina TOP coding, users can easily convert data between different assembly versions and project coordinates from older assemblies onto newer ones. This

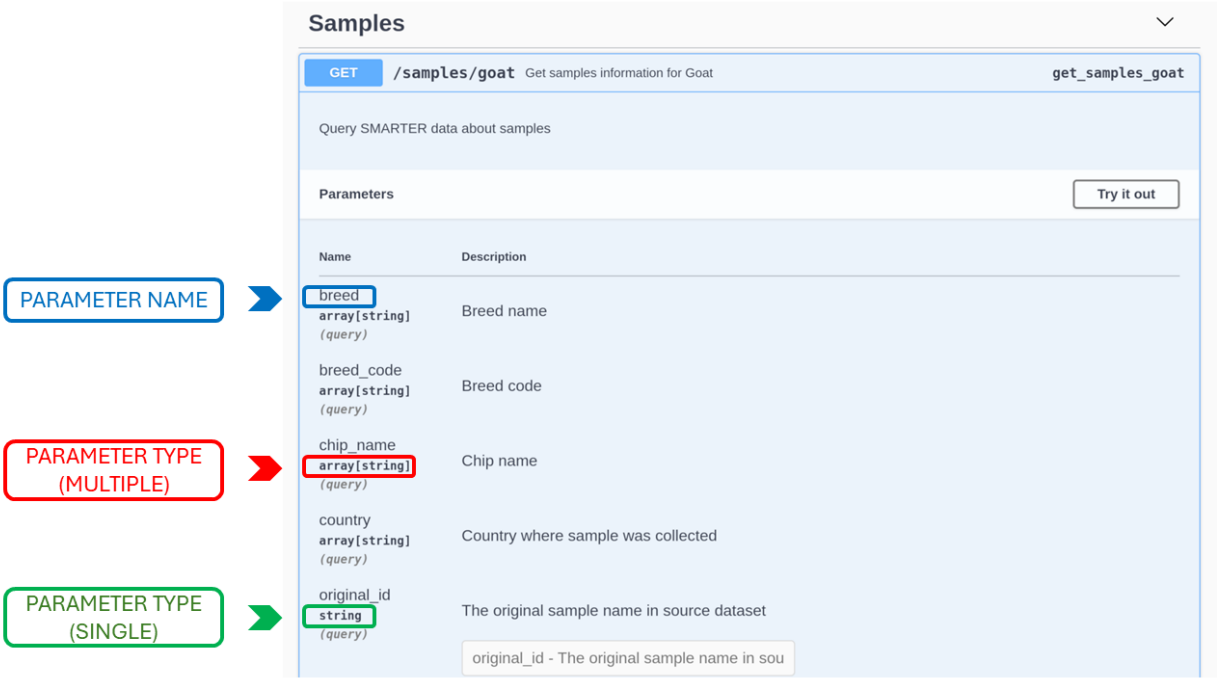

Figure 6. Screenshot of the Swagger interface for the SMARTER-backend API. The interface displays available endpoints and parameters, allowing users to interact with and test the API directly. In this example, the `GET /samples/goat` endpoint is shown, with parameters such as `breed`, `breed_code`, and `chip_name`. The `array[string]` type indicates that multiple string values can be entered, enabling filtering by multiple criteria. Users can click the *'Try it out'* button to input values, execute the query, and view the results with the URL to generate them, making it easy to explore the API's functionality.

approach ensures accurate SNP positioning across assemblies without concerns about probe orientation. All new improvements are tracked as GitHub issues[41], and when the changes are finalized, they are released in a new dataset version, along with any enhancements to data management. All release changes can be reviewed in the HISTORY.rst[42] file available on GitHub and in the SMARTER-database Read The Docs documentation[24].

Availability of source code and requirements

- Project name: SMARTER-database
- Project home page: `https://github.com/cnr-ibba/SMARTER-database`
- Documentation: `https://smarter-database.readthedocs.io/en/latest/`
- Operating system(s): Linux
- Programming language: Python 3.x
- Other requirements: docker, docker-compose, anaconda
- License: MIT

Availability of Test Data

The genotype files supporting the results of this article are available via anonymous FTP at `ftp://webserver.ibba.cnr.it` or through *smarterapi* R package (see `https://cnr-ibba.github.io/r-smarter-api/articles/smarterapi.html#collect-genotypes`). Metadata and sample information can be accessed via the SMARTER-backend REST API at `https://webserver.ibba.cnr.it/smarter-api/` using *smarterapi* R package (`https://cnr-ibba.github.io/r-smarter-api/`) and through the database portal at `https://webserver.ibba.cnr.it/smarter/`.

Declarations

List of abbreviations

- API: Application Programming Interface
- CI: Continuous Integration
- EBI: European Bioinformatics Institute
- EVA: European Variation Archive
- FAO: Food and Agriculture Organization
- FTPS: File Transfer Protocol Secure
- GNU: GNU's Not Unix
- GPS: Global Positioning System

- HTTP: HyperText Transfer Protocol
- IBS: Identical By State
- JSON: JavaScript Object Notation
- MAF: Minor Allele Frequency
- ODM: Object-Document Mapper
- PATO: Phenotype And Trait Ontology
- REST: Representational State Transfer
- RESTful: REST-compliant systems
- rsID: Reference SNP cluster ID
- SNP: Single Nucleotide Polymorphism
- UML: Unified Modeling Language
- URL: Uniform Resource Locator
- VCF: Variant Calling Format
- WGS: Whole Genome Sequencing

Ethical Approval

Not applicable

Competing Interests

The author(s) declare that they have no competing interests

Funding

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 772787 (SMARTER)

Author's Contributions

- Conceptualization: Alessandra Stella, Bertrand Servin, Gabriel Ciappesoni, Alexandros Triantafyllidis
- Resources: Valentina Tsartsianidou, Pablo Peraza, Anna Maria Johansson, Juan José Arranz, Fernando Freire, Szilvia Kusza, Gwenola Tosser-Klopp, Gabriel Ciappesoni, Alexandros Triantafyllidis, Bertrand Servin, Alessandra Stella
- Methodology: Paolo Cozzi, Bertrand Servin, Alessandra Stella, Alexandros Triantafyllidis, Gwenola Tosser-Klopp
- Software: Paolo Cozzi, Konstantinos Gkagkavouzis
- Data curation: Paolo Cozzi, Valentina Tsartsianidou, Konstantinos Gkagkavouzis, Pablo Peraza, Anna Maria Johansson, Juan José Arranz, Fernando Freire, Szilvia Kusza, Gwenola Tosser-Klopp
- Formal analysis: Arianna Manunza, Johanna Ramirez-Diaz, Valentina Tsartsianidou, Filippo Biscarini, Lucy Peters
- Writing - original draft preparation: Paolo Cozzi, Alessandra Stella
- Writing - review & editing: Arianna Manunza, Johanna Ramirez-Diaz, Bertrand Servin, Filippo Biscarini, Rachel Rupp
- Visualization: Paolo Cozzi
- Supervision: Alessandra Stella, Bertrand Servin
- Funding acquisition: Alessandra Stella, Bertrand Servin
- Project administration: Rachel Rupp

Acknowledgements

Not applicable

References

1. Stella A, Nicolazzi EL, Van Tassell CP, Rothschild MF, Colli L, Rosen BD, et al. AdaptMap: exploring goat diversity and adaptation. Genetics Selection Evolution 2018 dec;50(1):61. https://gsejournal.biomedcentral.com/articles/10.1186/s12711-018-0427-5.
2. Kijas JW, Lenstra JA, Hayes B, Boitard S, Neto LR, Cristobal MS, et al. Genome-wide analysis of the world's sheep breeds reveals high levels of historic mixture and strong recent selection. PLoS Biology 2012;10(2).
3. Smarter Project;. https://smarterproject.eu/, accessed: 2024-04-16.
4. PATO - the Phenotype And Trait Ontology;. https://www.ebi.ac.uk/ols4/ontologies/pato, accessed: 2024-04-16.
5. Gong Y, Li Y, Liu X, Ma Y, Jiang L. A review of the pangenome: how it affects our understanding of genomic variation, selection and breeding in domestic animals? Journal of Animal Science and Biotechnology 2023 May;14(1).
6. Fan J, Gunderson KL, Bibikova M, Yeakley JM, Chen J, Wickham Garcia E, et al. In: [3] Illumina Universal Bead Arrays Elsevier; 2006. p. 57–73.
7. Sun Y, Liu F, Fan C, Wang Y, Song L, Fang Z, et al. Characterizing sensitivity and coverage of clinical WGS as a diagnostic test for genetic disorders. BMC Medical Genomics 2021 Apr;14(1).

8. Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D, et al. PLINK: A tool set for whole-genome association and population-based linkage analyses. American Journal of Human Genetics 2007;81:559–575.
9. Fadista J, Bendixen C. Genomic position mapping discrepancies of commercial SNP chips. PLoS ONE 2012 2;7.
10. Gershoni M, Shirak A, Raz R, Seroussi E. Comparing BeadChip and WGS Genotyping: Non-Technical Failed Calling Is Attributable to Additional Variation within the Probe Target Sequence. Genes 2022 3;13.
11. Ajmone-Marsan P, Boettcher PJ, Colli L, Ginja C, Kantanen J, Lenstra JA. Genomic characterization of animal genetic resources. FAO; 2023. http://www.fao.org/documents/card/en/c/cc3079en.
12. "TOP/BOT" Strand and "A/B" Allele;. https://www.illumina.com/documents/products/technotes/technote_topbot.pdf, accessed: 2024-04-16.
13. European Variation Archive - Submit;. https://www.ebi.ac.uk/eva/?Submit-Data, accessed: 2024-04-16.
14. VCF Validator;. https://github.com/EBIvariation/vcf-validator?tab=readme-ov-file#assembly-checker, accessed: 2024-04-16.
15. Nicolazzi EL, Biffani S, Biscarini F, Wengel POT, Caprera A, Nazzicari N, et al. Software solutions for the livestock genomics SNP array revolution. Animal Genetics 2015;46:343–353.
16. Nicolazzi EL, Caprera A, Nazzicari N, Cozzi P, Strozzi F, Lawley C, et al. SNPchiMp v.3: Integrating and standardizing single nucleotide polymorphism data for livestock species. BMC Genomics 2015;16:1–6.
17. Nicolazzi E, Marras G, Stella A. SNPConvert: SNP Array Standardization and Integration in Livestock Species. Microarrays 2016;5:17.
18. Danecek P, Auton A, Abecasis G, Albers CA, Banks E, DePristo MA, et al. The variant call format and VCFtools. Bioinformatics 2011;27:2156–2158.
19. Sherry ST, Ward M, Sirotkin K. dbSNP—Database for Single Nucleotide Polymorphisms and Other Classes of Minor Genetic Variation. Genome Research 1999 8;9:677–679. http://genome.cshlp.org/lookup/doi/10.1101/gr.9.8.677.
20. Cezard T, Cunningham F, Hunt SE, Koylass B, Kumar N, Saunders G, et al. The European Variation Archive: A FAIR resource of genomic variation for all species. Nucleic Acids Research 2022 1;50:D1216–D1220.
21. MongoDB;. https://www.mongodb.com/, accessed: 2024-04-17.
22. MongoEngine;. https://mongoengine-odm.readthedocs.io/, accessed: 2024-04-17.
23. What is the difference between ODM and ORM?;. https://medium.com/@julianam.tyler/what-is-the-difference-between-odm-and-orm-267bbb7778b0 accessed: 2024-04-17.
24. The SMARTER Database documentation;. https://smarter-database.readthedocs.io/en/latest/index.html, accessed: 2024-08-08.
25. The Data Import Process;. https://smarter-database.readthedocs.io/en/latest/data-import.html, accessed: 2024-04-17.
26. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, et al. Comment: The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data 2016 3;3.
27. Cookiecutter Data Science;. https://drivendata.github.io/cookiecutter-data-science/, accessed: 2024-04-17.
28. Cookiecutter;. https://www.cookiecutter.io/, accessed: 2024-04-17.
29. Anaconda Software Distribution. Anaconda Inc.; 2020. https://docs.anaconda.com/.
30. Merkel D. Docker: lightweight linux containers for consistent development and deployment. Linux journal 2014;2014(239):2.
31. Docker Compose;. https://docs.docker.com/compose/, accessed: 2024-04-17.
32. Stallman RM, McGrath R, Smith PD. GNU Make: A Program for Directing Recompilation, for version 3.81. Free Software Foundation; 2004.
33. Preston-Werner T, Semantic Versioning; 2013. http://semver.org/. web.
34. The SMARTER Database;. https://github.com/cnr-ibba/SMARTER-database, accessed: 2024-05-20.
35. SMARTER-backend API;. https://webserver.ibba.cnr.it/smarter-api/docs/, accessed: 2024-04-17.
36. REST;. https://en.wikipedia.org/wiki/REST, accessed: 2024-04-17.
37. Cozzi P. smarterapi: Fetch SMARTER Data Through REST API; 2024, https://cnr-ibba.github.io/r-smarter-api/, r package version 0.2.0.
38. Fick SE, Hijmans RJ. WorldClim 2: new 1-km spatial resolution climate surfaces for global land areas. International Journal of Climatology 2017 10;37:4302–4315.
39. SMARTER-frontend;. https://webserver.ibba.cnr.it/smarter/, accessed: 2024-04-17.
40. Angular;. https://angular.io/, accessed: 2024-04-17.
41. SMARTER database issues;. https://github.com/cnr-ibba/SMARTER-database/issues, accessed: 2024-08-08.
42. SMARTER database history;. https://github.com/cnr-ibba/SMARTER-database/blob/master/HISTORY.rst, accessed: 2024-08-08.