quincunx: an R package to query, download and wrangle PGS Catalog data

Ramiro Magno[1,2,¶] Isabel Duarte[1,2,¶] Ana -Teresa Maia[1,2,3,*]

[1] Centre for Biomedical Research (CBMR), Universidade do Algarve, Gambelas Campus, 8005-139 Faro, Portugal
[2] Algarve Biomedical Center (ABC), Universidade do Algarve, Gambelas Campus, 8005-139 Faro, Portugal
[3] Faculty of Medicine and Biomedical Sciences (FMCB), Universidade do Algarve, Gambelas Campus, 8005-139 Faro, Portugal

¶ These authors contributed equally to this work.
* To whom correspondence should be addressed. Email: atmaia@ualg.pt

Abstract

Motivation: The Polygenic Score (PGS) Catalog is a recently established open database of published polygenic scores that, to date, has collected, curated, and made available 721 polygenic scores from over 133 publications. The PGS Catalog REST API is the only method allowing programmatic access to this resource.
Results: Here, we describe *quincunx*, an R package that provides the first client interface to the PGS Catalog REST API. *quincunx* enables users to query and quickly retrieve, filter and integrate metadata associated with polygenic scores, as well as polygenic scoring files in tidy table format.
Availability: *quincunx* is freely available under an MIT License, and can be accessed from https://github.com/maialab/quincunx.

Keywords

Polygenic Scores, PGS, disease susceptibility, Genomics, PGS Catalog, R, SNP, trait, Software, human, REST client

Introduction

For two decades, GWAS identified individual variants associated with risk for complex diseases, raising the hopes of a polygenic approach for disease prevention. However, until recently, integration of these results was challenging delaying its prompt application to the clinical setting. In 2020 alone, over 1,400 publications on polygenic risk scores (PGS) appeared in PubMed, raising the need for a standardised distribution of studies' key data, assuring their wide evaluation and accurate use.

The Polygenic Score (PGS) Catalog, created in 2019, is a publicly available, manually curated, open database of PGS and relevant metadata, that responds to this need [1]. Its current release [date 2021-02-03] includes manually curated data from 133 publications and 721 PGS associated with 194 traits. Currently, there are three alternative ways to access the data: (i) the web graphical user interface (GUI); (ii) by downloading database dumps; and (iii) the recently implemented PGS Catalog representational state transfer (REST) application programming interface (API), released in [date 2020-06-03], which provides direct programmatic access to the database, being this the preferred method for batch analyses.

We developed *quincunx*, the first R package [2] to programmatically access the PGS Catalog REST API. This package provides a simple user-friendly interface for querying the most updated Catalog data, retrieve and map it to in-memory relational databases of tidy data tables, allowing its prompt integration with tidyverse packages for subsequent data transformation, visualisation and modelling [3, 4].

Results

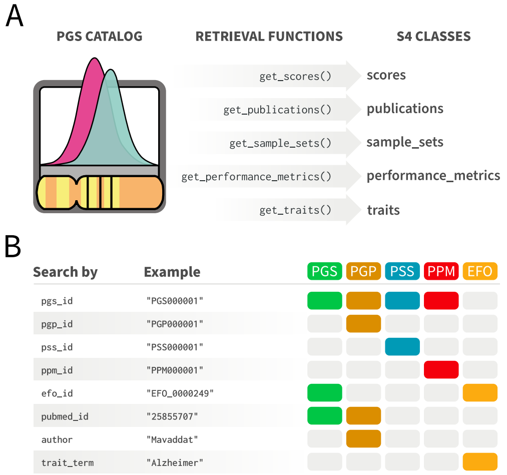
Retrieving data from the PGS Catalog REST API

The PGS Catalog REST API is an EBI service hosted at `https://www.pgscatalog.org/rest/`. The REST API uses hypermedia with resource responses following the OpenAPI Specification (`https://swagger.io/docs/specification/about/`). Response data is provided as hierarchical data in JSON format and can be paginated, i.e., split into multiple responses (`https://www.pgscatalog.org/rest/`).

To ease the conversion from the hierarchical to the relational tabular format — the preferred format for data analysis in R [4] — we developed a set of retrieval functions (Fig. 1A). Since the REST API data is organised around five core data entities — *Polygenic Scores*, *PGS Publications*, *PGS Sample Sets*, *PGS Performance Metrics* and *EFO traits*— we implemented five corresponding retrieval functions that encapsulate the technical aspects of resource querying and format conversion: `get_scores()`, `get_publications()`, `get_sample_sets()`, `get_performance_metrics` and `get_traits()` (Fig. 1A). These functions simplify the querying of PGS Catalog entities, by providing a complete and consistent interface to the Catalog. For example, to query for *scores*, the user needs only to know the function `get_scores()`, whereas the REST API itself exposes three separate resource URL endpoints for *scores* with different querying parameters. Moreover, the user can choose directly the arguments of the retrieval functions from any number of available search criteria exposed by the REST API (Fig. 1B). All arguments are vectorised, meaning that multiple queries are promptly available from a single function call. Results obtained from multiple queries can be combined with the logical operators `OR` or `AND` using the `set_operation` parameter. If `set_operation` is set to `OR` (default behaviour), results are collated while removing duplicates, if any. If `set_operation` is set to `AND`, only entities that concomitantly match all search criteria are returned. If finer control is needed on combining results, the following functions can be used: `bind()`, `union()`, `intersect()`, `setdiff()`, and `setequal()`. These are S4 methods that work with the S4 classes created in *quincunx*. An example of a case study (in tutorial style) can be found in Additional file 1.

Representation of PGS Catalog entities

All S4 classes share the same design principles that make them relational databases: (i) each slot corresponds to a table (dataframe in R); (ii) the first slot corresponds to the main table that lists observations of the respective PGS Catalog entity, e.g., *scores*; and (iii) all tables have a primary key, the identifier of the respective PGS Catalog entity: `pgs_id`, `pgp_id`, `pss_id`, `ppm_id` or `efo_id`. For easy consultation of the variables present in the retrieved data tables, we provide a cheatsheet (Additional file 2); and for detailed descriptions, the user can issue the following help commands to open the respective help



Search by	Example	PGS	PGP	PSS	PPM	EFO
pgs_id	"PGS000001"	🟩	🟧	🟦	🟥	
pgp_id	"PGP000001"		🟧			
pss_id	"PSS000001"			🟦		
ppm_id	"PPM000001"				🟥	
efo_id	"EFO_0000249"	🟩				🟨
pubmed_id	"25855707"	🟩	🟧			
author	"Mavaddat"		🟧			
trait_term	"Alzheimer"					🟨

Figure 1. *quincunx* retrieval functions. **(A)** Functions for retrieving data from the PGS Catalog: `get_scores()`, `get_publications()`, `get_sample_sets()`, `get_performance_metrics()` and `get_traits()`. **(B)** *quincunx* search criteria (function parameters) to be used with retrieval functions. Coloured boxes indicate which entities can be retrieved by each search criteria.

pages for each class: `class?scores`, `class?publications`, `class?sample_sets`, `class?performance_metrics` or `class?traits`.

Improvements & Limitations

Compared to the exposed REST API, we have improved data accessibility in *quincunx* in several ways. Firstly, we harmonised the nomenclature of the variables in tidy tables with the nomenclature used by the GWAS Catalog [5], namely for the variables that are also used by the R package *gwasrapidd* [6]: an analogous R package that provides access to the GWAS Catalog REST API. This permits a frictionless wrangling of variables between the two R packages, allowing crosstalk between the data from the two Catalogs. Secondly, by recognising that in some cases the values of a variable are provided in its name and not in its value (a case of untidy data), we decided to perform the required refactoring to make those variables explicit columns in the relational tables, thus making the data more analysis friendly. For example, the *stage* of a sample comes implicitly coded in the JSON keys `samples_variants` and `samples_training` and are mapped in *quincunx* to the variable `stage`, with values `‘‘discovery’’` and `‘‘training’’`, respectively. Additionally, the PGS Catalog REST API does not offer specific endpoints allowing direct mapping between the PGS entities, as this information is deeply nested in the hierarchical structure of the JSON responses. *quincunx* facilitates the retrieval of relationships between entities, by providing a set of mapping functions based on the entities' identifiers, e.g., `pgs_to_pgp`, `pgp_to_ppm()`, `ppm_to_pss()`, including mapping (when applicable) from PGS scores to GWAS studies: `pgs_to_study()` and `study_to_pgs()` (see online documentation for the complete list). Finally, *quincunx* provides a set of helper functions to easily browse linked web resources, such as PubMed (`open_in_pubmed()`), dbSNP (`open_in_dbsnp()`), and the PGS Catalog Web interface itself (`open_in_pgs_catalog()`).

Despite the availability of some R software packages, e.g. *bigsnpr* [7], *RápidoPGS* [8], or *SummaryLasso* [9], that allow the application of polygenic scores to particular datasets, i.e. for analyses downstream of *quincunx* data retrieval, the most popular software tools for these calculations, e.g. PRSice [10,11], *LDpred* [12], *PRS-CS* [13], *JAMPred* [14], *lassosum* [15], *PLINK* [16,17], do not run in R. This could present an obstacle to pursuing a full PGS analysis within the same R framework (using the PGS scoring files), therefore delaying the process of polygenic score application (for an overview of the currently available methods, please see [18]).

Conclusion

We have developed the first R client for the PGS Catalog REST API, thus greatly facilitating the programmatic access to the database. The main advantages of *quincunx* are: (i) providing a simple user-friendly interface to the REST API, allowing the programmatic querying of the most updated data from within R; (ii) the retrieval of the data in an analysis friendly format, with tidy data representations of the PGS entities, i.e., of *scores*, *publications*, *sample sets*, *performance metrics* and *traits* in the form of in-memory relational databases; (iii) allowing the automatic retrieval of polygenic scoring files from the PGS Catalog FTP server, making the data immediately available for analysis in R (as an extra feature not available via the REST API); and (iv) dedicated functions to export the retrieved objects to Excel (.xlsx) format for data inspection and sharing outside of R. *quincunx* is a package that will greatly improve the research community's ability for data mining within R, therefore accelerating the evaluation and subsequent application of published and manually curated polygenic scores.

Availability and requirements

- **Project name**: quincunx.
- **Project home page**: `https://github.com/maialab/quincunx`.
- **Operating system(s)**: Platform independent.
- **Programming language**: R.
- **Other requirements**: None.
- **License**: MIT.
- **Any restrictions to use by non-academics**: None.

Competing interests

The authors declare that they have no competing interests.

Funding

This work was supported by national Portuguese funding through FCT—Fundação para a Ciência e a Tecnologia and CRESC ALGARVE 2020: POCI-01-0145-FEDER-022184 ”GenomePT” and ALG-01-0145-FEDER-31477 ”DEvoCancer”.

Author's contributions

RM & ID devised and wrote the package, and wrote the manuscript. ATM supervised the project and wrote the manuscript. All authors read and approved the final manuscript.

Acknowledgements

We would like to thank the PGS Catalog team, particularly Samuel Lambert and Laurent Gil, for feedback and for supporting the development of *quincunx*. We also would like to thank the members of the Cancer Functional Genomics lab for user experience feedback, and UAIC-UAlg for administrative support.

References

1. Lambert SA, Gil L, Jupp S, Ritchie SC, Xu Y, Buniello A, et al. The Polygenic Score Catalog: an open database for reproducibility and systematic evaluation. medrxiv. 2020 may. Available from: `https://www.medrxiv.org/content/10.1101/2020.05.20.20108217v1`.

2. R Core Team. R: A Language and Environment for Statistical Computing. Vienna, Austria; 2017. Available from: `https://www.R-project.org/`.

3. Wickham H, et al. Tidy data. Journal of Statistical Software. 2014;59(10):1–23.

4. Wickham H, Grolemund G. R for Data Science: Import, Tidy, Transform, Visualize, and Model Data. 1st ed. O'Reilly Media; 2017. Available from: `http://r4ds.had.co.nz/`.

5. Welter D, MacArthur J, Morales J, Burdett T, Hall P, Junkins H, et al. The NHGRI GWAS Catalog, a curated resource of SNP-trait associations. Nucleic Acids Research. 2013;42(D1):D1001–D1006.

6. Magno R, Maia AT. gwasrapidd: an R package to query, download and wrangle GWAS catalog data. Bioinformatics. 2019 aug.

7. Privé F, Aschard H, Ziyatdinov A, Blum MGB. Efficient analysis of large-scale genome-wide data with two R packages: bigstatsr and bigsnpr. Bioinformatics. 2018 mar;34(16):2781–2787.

8. Reales G, Vigorito E, Kelemen M, Wallace C. RápidoPGS: A rapid polygenic score calculator for summary GWAS data without validation dataset. biorxiv. 2020 jul.

9. Chen TH. SummaryLasso: Building Polygenic Risk Score Using GWAS Summary Statistics; 2019. R package version 1.2.1. Available from: `https://CRAN.R-project.org/package=SummaryLasso`.

10. Euesden J, Lewis CM, O'Reilly PF. PRSice: Polygenic Risk Score software. Bioinformatics. 2014 dec;31(9):1466–1468.

11. Choi SW, O'Reilly PF. PRSice-2: Polygenic Risk Score software for biobank-scale data. GigaScience. 2019 jul;8(7).

12. Vilhjálmsson BJ, Yang J, Finucane HK, Gusev A, Lindström S, Ripke S, et al. Modeling Linkage Disequilibrium Increases Accuracy of Polygenic Risk Scores. The American Journal of Human Genetics. 2015 oct;97(4):576–592.

13. Ge T, Chen CY, Ni Y, Feng YCA, Smoller JW. Polygenic prediction via Bayesian regression and continuous shrinkage priors. Nature Communications. 2019 apr;10(1).

14. Newcombe PJ, Nelson CP, Samani NJ, Dudbridge F. A flexible and parallelizable approach to genome-wide polygenic risk scores. Genetic Epidemiology. 2019 jul;43(7):730–741.

15. Mak TSH, Porsch RM, Choi SW, Zhou X, Sham PC. Polygenic scores via penalized regression on summary statistics. Genetic Epidemiology. 2017 may;41(6):469–480.

16. 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. The American Journal of Human Genetics. 2007 sep;81(3):559–575.

17. Chang CC, Chow CC, Tellier LCAM, Vattikuti S, Purcell SM, Lee JJ. Second-generation PLINK: rising to the challenge of larger and richer datasets. GigaScience. 2015 feb;4(1).

18. Choi SW, Mak TSH, O'Reilly PF. Tutorial: a guide to performing polygenic risk score analyses. Nature Protocols. 2020 jul;15(9):2759–2772.

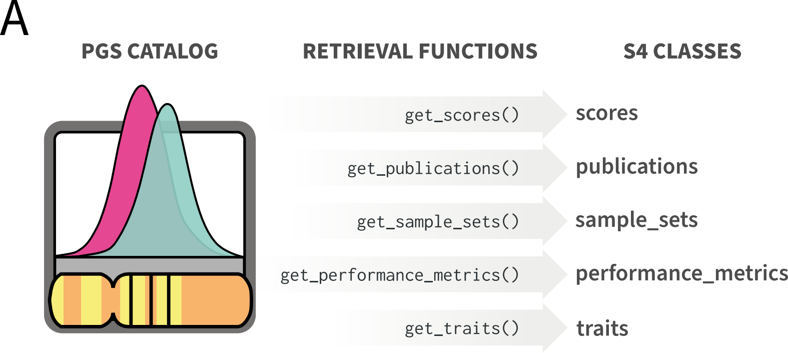
Additional Files

Additional file 1 — Example of a case study.

- **File name**: `additional_file_1.pdf`.
- **File format**: Portable Document Format (PDF).
- **Title**: Example Study Case.
- **Description**: Example of a study case exploring the PGS scores by Mavaddat et al. (2018).

Additional file 2 — quincunx cheatsheet.

- **File name**: `additional_file_2.pdf`
- **File format**: Portable Document Format (PDF)
- **Title**: quincunx cheatsheet
- **Description**: Additional file 2 contains an infographics: quincunx cheatsheet.



B

Search by	Example	PGS	PGP	PSS	PPM	EFO
pgs_id	"PGS000001"	■	■	■	■	
pgp_id	"PGP000001"		■			
pss_id	"PSS000001"			■		
ppm_id	"PPM000001"				■	
efo_id	"EFO_0000249"	■				■
pubmed_id	"25855707"	■	■			
author	"Mavaddat"		■			
trait_term	"Alzheimer"					■