Dear editor and reviewers,

First, we would like to thank the reviewers because their comments and suggestions have led to a significant improvement of the new version of the paper. We addressed all reviewers’ concerns to the best of our ability.
Secondly, we added some new comparisons between software. For GWAS computations, we now also compare bigstatsr and bigsnpr to R packages SNPRelate and GWASTools and also added a GWAS on a continuous outcome (before, we had compared only GWAS of binary outcomes). Then, for the comparison of PRS computations, we now make a PRS analysis based on external summary statistics and compare bigstatsr and bigsnpr to PRSice-2. Finally, we added PLINK 2.0 to the comparison of fast PCA software (computation time and precision).
Thirdly, the major changes of the main text have been highlighted in red in order to assist reviewers in tracking changes. In particular, the introduction and method section “Data management, preprocessing and imputation” includes major changes in order to avoid exaggerated claims of providing a single comprehensive framework.
Finally, please note that we added an author to this paper who contributed to improve the manuscript.

##########################   Reviewer 1   #########################

Comments to the Author
1) Overall, these are two very useful packages that combine widely-used methods for analysis of SNP data, and bring them into the R environment.

2) Specific comments:
a) Major comments:
[1] I disagree that SNP clumping is necessary or useful before PCA (for most common uses of PCA). PCA is usually done in an 'unsupervised' manner, i.e., ignoring the phenotype association status of the SNPs. Batch effects / population structure, by their nature, are not limited to the top associated SNPs, but have a roughly uniform effect genome-wide. But we expect the proportion of highly-associated SNPs to be low for most phenotypes (i.e., quite localised signal). So pruning/thinning is useful for detecting 'global' signals by throwing out 'duplicated' signals that can distort the PCA (which assumes SNPs are independent so that we can detect if the individuals are independent or not...), regardless of association status. But clumping will tend to favour SNPs associated with your phenotype (local signal), which is not what you want most of the time.

The thinning strategy we proposed never used the phenotype information (i.e. is fully independent of the genotype-phenotype association test). We understand that confusion arises due to the use of the term “clumping”, which is commonly used for post-processing of GWAS results in order to keep only the most significant SNP per region of the genome. Here, we propose to use a similar strategy but applying clumping based on the minor allele frequency (MAF) of SNPs (as the statistic of importance to order the SNPs). For comparison between pruning and clumping based on maf, see our answer to question [14].

We modified the text accordingly (page 3, section 3.3, paragraphe 2).

[2] How was imputation error measured (in the comparison against Beagle), and does the imputation error matter in practice? i.e., it could be that for GWAS (detecting SNPs associated with phenotype) the imputation error matters more since it dilutes the signal, but for predictive modelling the impact could be minimal, depending on the effect size and if there are enough other SNPs in LD with the causal variants. It would be good to know if your imputation models are "safe" to use in either situation.
In response to this comment and comments from the other reviewers, we would like first to clarify that we do not impute SNPs that are not present in the data. Such task requires the use of reference panels and, in agreement with the literature, we recommend using existing tools such as imputation servers for human data (McCarthy et al., 2016). Instead, our goal is to impute missing values for available SNPs. It addresses cases where the proportion of missing values is minimal (e.g. < 5%). Indeed, most modern SNP chips provide genotype data with large call-rates. For example, the Celiac data we use in this paper presents only a proportion of 0.04% of missing values for the genotyped SNPs after quality control.

In the comparison against Beagle, the imputation error is measured by introducing some missing data in a data set with no missing data. Then, the imputation error is the proportion of imputed genotypes that are different from the true genotypes.

In practice, imputation of missing values is performed using only neighboring SNPs (independently of the phenotype). As we now rephrase in the main text: “Our algorithm is the following: for each SNP, we divide the individuals in the ones which have a missing genotype (test set) and the ones which have a non-missing genotype for this particular SNP. Those latter individuals are further separated in a training set and a validation set (e.g. 80% training and 20% validation). The training set is used to build the XGBoost model for predicting missing data. The prediction model is then evaluated on the validation set for which we know the true genotype values, providing an estimator of the number of genotypes that have been wrongly imputed for that particular SNP. The prediction model is then projected on the test set (missing values) in order to impute them.”.
Using a subset of the non-missing values as a test set (i.e. 20%) allow us to keep track of SNP that have been poorly imputed. We show that the estimation of the number of imputation errors provided by function snp_fastImpute is accurate (**Figure S8**), and can be used for imputation post-processing by removing SNPs with too many errors (**Figure S9**). This ensure that our imputation procedure is “safe” to use. We also note that snp_fastImpute makes less 0/2 switching errors than Beagle (supplementary notebook “imputation”).

We updated the main text to make all these points clearer (page 3, section 3.2, last paragraph & page 6, section 4.5).

[3] It's not clear, how are dosages supported in the pipelines? If I have impute2 dosage data (.bgen), can I convert it to your format without going through PLINK hard calls first?

First, the bgen format offers various accuracy (0 to 4 decimal) depending on the number of bits used to store each dosage (from 1 to 16, see the post from its author here: https://goo.gl/mFdLWK). Our format stores each genotype using 8 bits. It follows that we can store dosages rounded to two decimal places. Note that this strategy is for example used in modern large-scale datasets such as the UK Biobank. However, if the standard was to change, it would be relatively easy to modify our code to allow for expanded accuracy (by using a “FBM.code65536”, coding each dosage on 16 bits, allowing for 4-decimal accuracy). We now provide a supplementary notebook (“dosage”) that presents two methods to convert from the bgen format to our format “FBM.code256”, as proofs of concept.

b) Minor comments:
[4] The grammar can use a bit of cleaning up and making the language a bit more formal, e.g. 'this might be a brake on data exploration'.

We updated some phrases accordingly.

[5] The number of SNPs in the celiac disease dataset is quite small (280,000), is that due to QC or the chip being used?

This is due to both. The chip used was an Illumina Hap300 v1-1 (Dubois 2010). There were 295,453 genotyped SNPs (method section “Data analyzed”) and 281,122 SNPs remaining after quality control (result section “Application”).

##########################   Reviewer 2   #########################

Comments to the Author
I think the aim of the authors of producing a unified pipeline for genetic analyses is excellent, albeit ambitious, and the FBM matrix structure to allow memory mapping will be useful for R users wishing to perform genetic analyses on the very large data that is becoming common now. The authors have also incorporated some nice speed-ups of existing functions/methods, which could be useful to the field. However, there are a number of significant issues that would need to be well addressed before I could consider recommending this manuscript for publication in Bioinformatics. The points are outlined below in order of appearance in the manuscript rather than significance (although minor points are prefaced as such).

[6] (minor) I think a potential selling point of these packages (and specifically the matrix structure) is the potential to re-establish tools that have become defunct due to the size of contemporary data - this is a point made at the end of the 1st paragraph of the Introduction (although 'limited access' doesn't make the point clearly I think). However, it would be good if the authors could provide some examples of tools that are not feasible for use on the latest UK Biobank data but that are now feasible via these packages (from this manuscript it's not clear that there are any).

First, in regards of this comment and further comments, we modified the introduction and other sections of the manuscript.

Regarding this specific point we now highlight that the growing size of human genetic datasets requires a continuous optimization of method and software to allow for reasonable computation time. We believe that for the R language, the implementation of scalable tools for GWAS and other genomic analyses is still lacking. Consider for example the popular packages snpStats and GenABEL: package snpStats still uses in-memory data and package GenABEL doesn't handle binary PLINK files and has not been updated since 2013. Note that we now compare bigsnpr to R packages SNPRelate and GWASTools (Table 1).

[7] (minor) Is there really the claimed 'need' for combining software into a single data analysis pipeline? I think most Statistical Geneticists are comfortable learning a few programs, given the gain in accuracy or speed in certain settings relative to being restricted to a single pipeline. Also, the start of 2nd paragraph of the Introduction (file format conversions) seems somewhat disingenuous because it appears to be being used as part of an argument that there is too much pre-processing work required to make data/programs consistent, hence the need for these packages, yet bigsnpr simply makes system calls to PLINK to perform file conversions and QC. Thus, the wording here should be reframed to avoid such a line of argument.

We understand the reviewer's concern. We removed the sentences referring to file format transformation in the introduction. We now simply refer to possible data format transformation through call system in the method section (section 3.2, paragraph 2).

[8] (minor) pg.2 middle of 1st paragraph - You should also state that PLINK-2 (or PLINK 1.9), which I believe is now used far more than PLINK v.1, does handle imputed data.

This is a typo from us. By version 1, we actually meant version 1.9, which indeed does not allow for the analysis of dosage data. Note that we don't talk about dosage data in the introduction anymore.

[9] Also, the sentence: 'As a result, one has to make extensive bash/perl/R/python scripts..' is misleading, because programs such as PRSice (mentioned the sentence before) already combine PLINK/C++/R operations offering full (PRS in this case) analyses in a single command line (with several input file formats) - and so the sentence should be reworded to reflect this.

PRSice is certainly an excellent tool for standard PRS analysis and it is true that the user of PRSice doesn't need to combine different software and languages because PRSice does it for them. The point we wished to make was that our packages allow to perform a range of analysis block by block all within R. Combining all these blocks in R should be appealing for researchers using R. We removed the cited sentence about scripting and now focus on only two aspects in the introduction: the need for optimized software and the possibility to do all the analysis in R.

[10] Does this really provide 'a single comprehensive framework'? - seems far less comprehensive than alternatives such as PLINK, and not sure it can be described as such when it does not include imputation of non-genotyped SNPs. It's not obvious to me why users would switch from using PLINK, and a few other programs (such as Imputation servers), other than gaining the impression from this paper that this is in fact 'a single comprehensive framework'. There are users who may prefer to use R, but the R package 'GWASTools' offers a good all-in-one package for genetic analyses - wouldn't extending this package have created a more comprehensive framework? To avoid exaggerated claims, I think the wording should be changed to more specifically reflect the main benefits of these packages: (1) They are in R (for those who prefer R), (2) They exploit memory mapping, which allows standard analyses on very large data in R, (3) Efficiency gains of *some* standard analysis functions have been incorporated.

We thank very much the reviewer for these helpful suggestions – the summary of main benefits made by the reviewer reflects very clearly our packages. We now re-phrased relevant sections in order to avoid exaggerated claims. In particular, we now make clear that our packages primarily allows for the analysis of large-scale genome-wide data in R, whereas the preprocessing is only integrated via system calls to existing software (to help with simplicity and reproducibility). Also, as discussed in response to previous comment [2], we now make clear that our packages do not allow for imputation of unavailable SNP, and that we recommend using imputation servers to address this specific point.

We update the introduction and section 3.2 / paragraph 2 accordingly.

[11] (The following point could be addressed by my previous point, if well executed): While the authors are upfront about their use of existing packages, I do wonder whether such heavy reliance on combining existing software warrants a publication in Bioinformatics. While the authors have made some adjustments to existing methods,

these seem minimal, and in the examples provided I would recommend using the existing software instead of these packages (imputation - Beagle/Imputation-server; pruning/clumping - PLINK-2; polygenic risk scoring - PRSice) perhaps apart from PCA calculation given the reported gains in speed from bigsnpr (although I'd question the value in switching from the more comprehensive PLINK for this reason). The effect of this is that these packages may add to statistical geneticists pipeline, rather than unify it.

It is true that we make use of existing software. For example, for our fast implementation of partial PCA, we build on top of the *RSpectra* package, and for providing some fast yet accurate in-sample imputation, we use the widely-used prediction algorithm *XGBoost*, for learning a joint prediction on million of SNPs, we build on top of previous work about efficient screening rules (package *biglasso*). However, we believe it is a good illustration of the strength of our packages. As we mention in the discussion, our packages will provide access to "the vast and diverse R libraries".

Moreover, we also provide some completely reimplemented and optimized algorithms such as pruning and clumping, as well as GWAS for binary and continuous traits. We now show in the results section that the GWAS performed with bigstatsr for a continuous outcome is much faster than the GWAS performed in PLINK 1.9. As noted by the reviewer, the implementation of PCA in bigstatsr is both faster and more precise than EIGENSOFT ("fastmode" option) and more precise than PLINK 2.0, while being very competitive in terms of computation time.

[12] (minor) The authors should make it absolutely clear that their imputation function only performs imputation of genotyped SNPs, not non-genotyped SNPs for which imputation is mostly known in genetics (the risk is user time being wasted in misunderstanding this). Also, the imputation performed here only improves on standard alternatives in terms of computational time, which I think is a lower priority than accuracy in this case and so I see this function as little/no benefit to the field.

We now clarify this point as best as we can. See response to previous comment [2].

[13] Does the linear-time SVD produce vectors highly correlated with quadratically calculated PCs or is there some loss of information?

As illustrated in **Figure 5** (figure 7 in the previous version), we carefully addressed this point in the original version and we did not observed any loss of information, as opposed to the estimates of FastPCA (EIGENSOFT) and PLINK 2.0 (that we added in this new version).

[14] The authors' recommendation of using clumping rather than pruning as a pre-processing step prior to PCA is unjustified. Firstly, it certainly shouldn't be performed indexing according to P-value as then the resultant PCs would likely reduce the power of the GWAS dramatically, and the explanation given by the 1st author on his webpage for indexing by MAF is incorrect I believe. The reason that he retained only 1 SNP in the pruning case and 5 SNPs in the MAF-indexed case is not due to the indexing but because of the way that the pruning is implemented, with the index SNP removed if correlated with other SNPs in the former case but with the index

SNP retained at each step in the latter case. If the index SNP had been instead retained in the former (pruning) case, then all the even SNPs would have been retained (if the index SNP had been removed in the MAF-indexed scenario at each step then only the 1st SNP would remain). If PLINK does not retain the index SNP in pruning then this appears to be a clear flaw, but perhaps more likely that the 1st author has made an error in his implementation of PLINK's pruning algorithm? In any case, if PLINK's pruning algorithm can be efficiently improved upon (and I can see a potential case for indexing by MAF being sensible) then that would be welcome, but the benefits would need to be properly and accurately demonstrated. I think it was rather careless to have suggested that the field switch from performing pruning to MAF-indexed clumping on the basis of such flimsy (and I believe flawed) evidence, which was not even included in the manuscript itself.

Also see previous response to [1].

To confirm there was no error in our implementation of the PLINK pruning algorithm, we reported a similarity of 99.95% between the two sets of SNPs returned by PLINK and bigsnpr (supplementary notebook "GWAS-comparison").

In order to provide the reader with an illustration of the possible limitation of PLINK's pruning algorithm, we now added the document of package bigsnpr's website as a supplementary document ("pruning-vs-clumping") while using a system call to PLINK instead of our reimplemented function.

Note that using the index SNP would also potentially lead to a worst-case scenario. Imagine if all index SNPs have a very low MAF, one would not capture much variation with such a set of SNPs. So, the optimal algorithm would be to keep the index SNPs AND that these index SNPs surely capture some variation (more likely with large MAFs). This is exactly what we suggest to do by using the clumping algorithm instead, using the MAF as the statistic of importance. n practice, PC scores obtained after using either pruning or clumping (with the maf to rank SNPs) are almost identical (and we now include it in the main text), so that we only suggest to prefer clumping (as a safety measure for some worst-case scenario).

We updated the main text accordingly (page 3, section 3.3, paragraph 2).

[15] Data analysed: The replication of individuals by 5x and 10x surely induces much higher levels of population structure and LD (your data are now enriched with 'monozygotic twins'). I think you'll need to either simulate a large data set or else obtain one, in order for your large-data results to be reliable.

Using the terms "population structure" and "LD" was misleading. Actually, we meant that the replicated data have the same eigen decomposition (up to a constant) and pairwise SNP correlations (see the proof at the end of this document). This means that the pruning step returns the same set of SNPs as for the initial dataset (because SNPs still have the same pairwise SNP correlations). And we can also show that replicated datasets will end up with the same eigen decomposition (up to a constant). This enables us to check the precision of the compared approximated algorithms for computing partial PCA on very large datasets by simply checking the squared correlation of PC scores with those of the initial dataset (using an exact algorithm). Indeed, it would be impossible to apply an exact eigen decomposition on a genotype dataset with more than 100,000 individuals.

[16] Reproducibility: The availability of code and execution times is very welcome, but I think in the light of the pruning error above and several issues that I have with the reported execution times discussed below, I think the authors need to also provide data so that users (and reviewers!) can repeat the software comparisons themselves and identify where the speed gains are coming from. If these packages truly offer significant gains over alternatives then this will be an excellent way of convincing the field of that.

While we totally understand the point, we cannot share the Celiac and POPRES data. We now provide some open-access data of domestic dogs so that users can test our code and functions on a moderate size dataset with 4342 samples and 145,596 SNPs (Hayward et al., 2016).

Results / Execution times

[17] From the provided R code it appears that the execution times provided do not include the time required to read in the data to FBM.code256 - this could be particularly important for performing analyses across multiple traits (eg. many GWAS) or in performing simulation/permutation studies (where, e.g., PLINK can read each data set directly but bigsnpr needs to first store the data as FBM.code256 and then read in/out to/from R). Full execution times should instead be included.

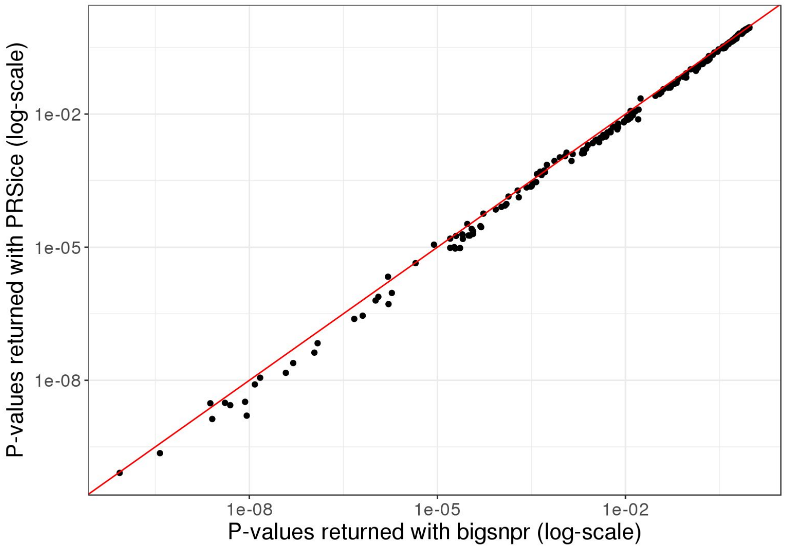
Reading binary PLINK files in our format “FBM.code256” is reported in **table 1** as well as in the analysis of 500K individuals. Note that reading the data in our format is only done once (each time, we only need to “attach” the data as an R object, which takes only a few seconds at most, even for very large datasets), this is why we don’t report the execution time in the PCA comparisons. Our algorithm for PCA can be directly used on a subset of SNPs (e.g. remaining after pruning) without having to write/read again a subset of the data in order to perform PCA (which some algorithms have to do, e.g. FastPCA and FlashPCA2).

[18] The speed-up in clumping time seems surprising given the much faster speed of pruning using PLINK (the algorithms are very similar) - given the issue highlighted above regarding the pruning algorithm I think it will be important to better demonstrate that their clumping algorithm is working properly. One check would be whether the results of their PRS analyses using the different software are extremely similar or not (I don’t think their removal of long-range correlated SNPs - after the usual clumping has been applied - should make a substantial difference to results).

We agree with the reviewer that the algorithms of pruning and clumping are very similar. Nevertheless, in PLINK, pruning is usually performed by measuring LD with the Pearson correlation coefficient (using “--indep-pairwise”) whereas in PLINK clumping algorithm, the only option is to use some maximum likelihood haplotype frequency estimation, which takes longer to compute. On the contrary, in package bigsnpr, we use the Pearson correlation coefficient in both algorithms, likely explaining why our clumping algorithm is faster than the one in PLINK. Also note that in practice, the way the LD is assessed doesn’t dramatically change the results because “the correlation between genotypic values is approximately equal to that between genic values” (Rogers, 2009).

Finally, we report in the supplementary notebooks (“GWAS-comparison” and “PRS-comparison”) the similarities between results from the software compared. For example, similarity between the SNP sets returned pruning (comparing PLINK and bigsnpr) have a similarity of 99.95%.

We also show here the plot of the p-values obtained when testing for correlation between Polygenic Risk Scores on height and the celiac disease status. To compare PRSice and bigstatsr/bigsnpr, we consider several thresholds of p-values to compute Polygenic Risk Scores of height. Each point in the plot below corresponds to a given threshold of p-value to compute PRS. We show that the results obtained by the 2 software are similar (correlation of more than 99.9%), while the differences are explained by the difference in the way LD is assessed in PRSice/PLINK and bigstatsr/bigsnpr.



[19] Is a full PRS analysis - including performing the usual regression analyses involved (on the test data) - faster using these packages than the popular PRS software PRSice? It appears that bigsnpr only computes PRS, yet in practice whenever researchers compute PRS the next step is almost always to perform a regression on a phenotype of interest in some test data set of individual-level data. Moreover, since the standard PRS analyses involve formatting and reading in GWAS summary statistics, as well as the individual-level test data, then won’t users of bigsnpr have to do all this pre-processing themselves? (already handled by PRSice). I can see that bigsnpr could be useful for users performing prediction entirely on individual-level data, but if performing the standard PRS analyses then the present comparison is misleading since it does not reflect full PRS analyses as researchers will consider them - to include these a comparison should be done with PRSice-2 with GWAS summary results and test data as input and regression results from testing in the test data at a range of PRS P-value thresholds as output (execution times need not include performing the GWAS since PRS are typically performed using pre-computed GWAS statistics available online). It would be surprising if these packages are faster than PRSice-2 given that it is written in C++.

Indeed, the reviewer again well capture the purpose of our packages. We made a comparison on computing a PRS from/to individual-level data because our research will primarily focus on genomic prediction from individual-level data. Yet, as the reviewer stated, this is not the most standard use of PRS. So, we changed the second comparison (Table 2) in order to compare a standard PRS analysis with our packages and the most recent version of PRSice-2. We report similar computation times.
We hope this will convince the reviewer that our packages are both fast and flexible. Indeed, this type of analysis was not initially intended to be part of our packages, but this analysis shows that we can use existing blocks provided by our packages in order to make this specific analysis in a satisfactory computation time.
Finally, note that bigstatsr and bigsnpr also make largely use of C++ code via the package Rcpp.

Wording/minor:

[20] English needs checking in places, and abbreviations (eg. "don't") should be expanded (eg. "do not").

We updated some phrases accordingly.

[21] The 'Pruning + Thresholding (P+T)' model is incorrectly named. I know that this has been stated several times in the literature but nobody performs 'Pruning + Thresholding' in Polygenic Score analyses, they perform 'Clumping + Thresholding (C+T), as the authors here go on to describe after introducing 'P+T'. This would be a good opportunity to rectify the error in the literature.

We agree with the reviewer. We now use the rectified term and will use it from now on in our forthcoming research work.

##########################   Reviewer 3   #########################

Comments to the Author
The authors present two R packages, one for working with large matrices (bigstatsr) and one for working with GWAS datasets (bigsnpr). The packages integrate various existing methods, and implement two new ones (for imputation, and for finding long-range LD regions).

[22] The paper would benefit from a clearer statement of the supported analyses, before delving into implementation details. It would also help to have better comparisons with existing work. E.g. the advantages of bigstatsr over existing R packages like bigmemory should be more clearly stated.

A comprehensive list of features available in bigstatsr and bigsnpr are provided on the packages' websites. In the supplementary materials, we now include a vignette ("bigstatsr-and-bigmemory") providing a comprehensive comparison between packages bigstatsr and bigmemory. We also add table S5, which presents a comparison of features available for different software.

[23] The new imputation method should be described in more detail, and comparisons done with leading software such as IMPUTE2.

In response to this comment and comments from the other reviewers, we would like to clarify that we do not impute SNPs that are not present in the data. Such task requires the use of reference panels and, in agreement with the literature, we recommend using existing tools such as imputation servers for human data (McCarthy et al., 2016). Instead, our goal is to impute missing values for available SNPs. It addresses cases where the proportion of missing values is minimal (e.g. < 5%). Indeed, most modern SNP chips provide genotype data with large call-rates. For example, the Celiac data we use in this paper presents only a proportion of 0.04% of missing values for the genotyped SNPs after quality control.

As mentioned in that response, the aim the new imputation method is to perform a quick in-sample imputation for a small number of missing values amongst available genotyped SNPs without using any reference panel.

[24] Also, authors state that FlashPCA/FlashPCA2 does not support parallelism, but these tools' descriptions say otherwise.

FlashPCA does support parallelism but FlashPCA2 does not. We now add PLINK 2.0, which supports parallelism, to the comparison of fast partial PCA algorithms.

[25] The supplement could use some explanatory text to tie together the figures and tables.

It is true that we didn’t design the supplementary to be a readable document on its own. It is more a collection of supplementary tables and figures that are referred from the main text.

Same eigen decomposition: proof

Proof for the same pearson correlation coefficient

Let us note x and y two SNPs and $\overline{x}$ the mean of x.

Then the pearson correlation coefficient between x and y is:

$$r = \frac{\sum\limits_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum\limits_{i=1}^{n} (x_i - \overline{x})^2} \sqrt{\sum\limits_{i=1}^{n} (y_i - \overline{y})^2}} \ .$$

If we note x_K and y_K the SNPs x and y replicated K times, then, $\overline{x_K} = \frac{K \sum\limits_{i=1}^{n} x_i}{K\ n} = \frac{1}{n} \sum\limits_{i=1}^{n} x_i = \overline{x}$ and r_K the pearson correlation coefficient between x_K and y_K is

$$r_K = \frac{K \sum\limits_{i=1}^{n} (x_i - \overline{x_K})(y_i - \overline{y_K})}{\sqrt{K \sum\limits_{i=1}^{n} (x_i - \overline{x_K})^2} \sqrt{K \sum\limits_{i=1}^{n} (y_i - \overline{y_K})^2}} = \frac{\sum\limits_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum\limits_{i=1}^{n} (x_i - \overline{x})^2} \sqrt{\sum\limits_{i=1}^{n} (y_i - \overline{y})^2}} = r \ .$$

Proof for the same eigen analysis

First, as $\overline{x_K} = \overline{x}$, then the allele frequency of x_K is the same as for x so that the scaling $\frac{x-2p}{\sqrt{2p(1-p)}}$ is also the same (because $p = \overline{x}/2$).

For the exact singular value decomposition of $G = U \Delta V^T$ (where G is the scaled genotype matrix), we can first compute $\Sigma = G^T G = V \Delta^2 V^T$, then remark that $\Sigma V = V \Delta^2$ so that V is the matrix of the eigen vectors of Σ and Δ^2 is the matrix of the eigen values of Σ. Finally to get U, we can compute $G V \Delta^{-1} = U \Delta V^T V \Delta^{-1} = U \Delta \Delta^{-1} = U$.

For replicated individuals, we want the decomposition of $G_K = U_K \Delta_K V_K^T$. Then, $\Sigma_K = G_K^T G_K = K G^T G = K \Sigma$ so that $V_K = V$ (same PC loadings) and $\Delta_K^2 = K \Delta^2$ resulting in $\Delta_K = \sqrt{K} \Delta$ (same eigen values, up to a constant). Finally $U_K = G_K V_K \Delta_K^{-1} = G_K V \left(\sqrt{K} \Delta\right)^{-1} = \frac{1}{\sqrt{K}} G_K V \Delta^{-1}$ (PCs scores are the same, up to a constant).