BMC Bioinformatics

SOFTWARE Open Access

PyAGH: a python package to fast construct kinship matrices based on different levels of omic data

Wei Zhao[2], Qamar Raza Qadri[2], Zhenyang Zhang[1], Zhen Wang[1], Yuchun Pan[1,3], Qishan Wang[1*] and Zhe Zhang[1*]

*Correspondence: wangqishan@zju.edu.cn; zhe_zhang@zju.edu.cn

[1] Department of Animal Science, College of Animal Sciences, Zhejiang University, 866# Yuhangtang Road, Hangzhou 310058, China
[2] Department of Animal Science, School of Agriculture and Biology, Shanghai Jiao Tong University, 800# Dongchuan Road, Shanghai, China
[3] Hainan Research Institute, Zhejiang University, 11# Yonyou Industrial Park, Yazhou Bay Science and Technology City, Sanya 572025, China

Abstract

Background: Construction of kinship matrices among individuals is an important step for both association studies and prediction studies based on different levels of omic data. Methods for constructing kinship matrices are becoming diverse and different methods have their specific appropriate scenes. However, software that can comprehensively calculate kinship matrices for a variety of scenarios is still in an urgent demand.

Results: In this study, we developed an efficient and user-friendly python module, PyAGH, that can accomplish (1) conventional additive kinship matrces construction based on pedigree, genotypes, abundance data from transcriptome or microbiome; (2) genomic kinship matrices construction in combined population; (3) dominant and epistatic effects kinship matrices construction; (4) pedigree selection, tracing, detection and visualization; (5) visualization of cluster, heatmap and PCA analysis based on kinship matrices. The output from PyAGH can be easily integrated in other mainstream software based on users' purposes. Compared with other softwares, PyAGH integrates multiple methods for calculating the kinship matrix and has advantages in terms of speed and data size compared to other software. PyAGH is developed in python and C++ and can be easily installed by pip tool. Installation instructions and a manual document can be freely available from https://github.com/zhaow-01/PyAGH.

Conclusion: PyAGH is a fast and user-friendly Python package for calculating kinship matrices using pedigree, genotype, microbiome and transcriptome data as well as processing, analyzing and visualizing data and results. This package makes it easier to perform predictions and association studies processes based on different levels of omic data.

Keywords: Kinship matrices, Python package, Omic data

Background

Kinship matrix, a symmetrical matrix representing the pairwise relatedness between individuals, was initially proposed to account for variance–covariance structure of breeding values (additive genetic effects) implemented in the best linear unbiased prediction (BLUP) method using pedigree information. With the development of high throughput genotyping methods during the last two decades, the kinship matrices were calculated using genome-wide markers and can be used to account for cryptic relatedness between pairwise individuals in genome wide association studies (GWAS) and to fit polygenicity in genomic prediction (GP), such as genomic best linear unbiased prediction (GBLUP) method [1, 2]. Therefore, how to construct kinship matrix is an important factor to control false discoveries in GWAS and obtain high accuracy in GP for different traits or diseases.

Actually, methods for constructing kinship matrices are becoming diverse and different methods have their specific appropriate application scenarios. For instance, under the occasion that only part of individuals are genotyped in the population, the kinship matrix can be calculated in combination of pedigree and genotypes, which can be further implemented in GP [3] and GWAS [4], which are well known as single-step methods in animal and plant breeding area. Meanwhile, when GWAS or GP is applied in a large cohort composed of multiple populations, Wientjes et al. (2017) suggested to construct a kinship matrix considering the heterogenous minor alle frequencies (MAF) across different populations. And the results obtained by simulated data showed that when the across-population genomic relationships were scaled by the within-population allele frequency, the genetic correlation was estimated unbiasedly. In addition, nonadditive effects, including the interaction between the effects of alleles either at the same locus (dominance) or between the allels of multiple genetic loci (epistasis), contributes significantly to phenotypic variation associated with the expression of polygenic complex traits [5]. Nonadditive effects are considered as a possible explanation for the "missing heritability", that is, marginal genetic effects that cannot be accounted for in GWAS or GP. Some studies have shown that considering nonadditive effects can improve the accuracy of predictions [6, 7]. However, the mainstream GWAS and GP software such as GCTA [8] or DMU [9] either fail to calculate such extended kinship matrices or can't output these matrices, which limits its further application.

Furthermore, due to the development of multi-omics, kinship matrix can be calculated not only at the genomic level, but actually at multi-omic level. The concept of kinship should therefore be extended to improve its application in association and prediction studies. For instance, prediction of phenotypes based on transcriptome [10] or microbiome [11] improves the accuracy by utilizing more data. Microbiome-wide association studies [12] and transcriptome association studies [13] can further explore the mechanism of different omics on polygenic complex traits. However, there is no software available to meet such a need to calculate kinship matrices based on abundance data from transcriptome or microbiome.

Therefore, we developed the PyAGH package to calculate kinship matrices using a variety of methods based on different levels of omics data for different application scenarios. PyAGH can calculate additive, dominant and epistatic kinship matrices based on genomic data within one population and different additive kinship matrices across multiple populations efficiently. It also supports construction of kinship matrices using pedigree, microbiome and transcriptome data. In addition, the output of PyAGH can be easily provided to downstream mainstream software, such as DMU [9], GCTA [8], GEMMA [14] and BOLT-LMM [15]. Thus, these user-friendly features allow novice users to focus on the analysis rather than technical aspects of installation and execution.

Implementation

The PyAGH package is implemented in Python programming language. It contains multiple Python3 and C++scripts with all the functions required for the program to execute. Some functions are written in C++via pybind11 (https://github.com/pybind/pybind11) to accelerate the computation speed. To be able to handle high-density genomic data, PyAGH supports multi-threaded computation, as well as split-chromosome computation based on the chunk matrix theory. In addition, for large-scale pedigree data, sparse matrices are used to save memory as well as to increase speed through multithreading. PyAGH has been successfully tested on machines running Unix-based operation system (OS) (macOS/Linux) and Windows. A detailed description of all algorithms and functions of PyAGH is provided in the user manual available at https://github.com/zhaow-01/PyAGH. Basic information of PyAGH's functions has been summarized in Table 1.

Construction of kinship matrix based on pedigree data

Kinship matrix is the core of traditional breeding in best linear unbiased prediction (BLUP) which is a famous classical method and widely used in breeding since 1950s [16]. *makeA() and makeD()* are the functions in PyAGH used to construct kinship matrices based on pedigree information for additive and dominant effects, respectively. The additive effect refers to the cumulative effect between alleles and

Table 1 Summary of PyAGH's functions

Category	Function	Description
Pedigree	sortPed()	Sort the pedigree data according to the correct birth date of individuals and check for various errors in the pedigree like offspring born before its parents, same offspring have different parents, loop in pedigree and etc
	selectPed()	Select pedigree based on specific individuals and generations
	makeA()	Construct kinship matrix based on pedigree information for additive effect. Option to use sparse matrix for memory saving
	makeD()	Construct kinship matrix based on pedigree information for dominant effect. Option to use multithreading when there are multiple CPU
Genome	makeG()	Construct kinship matrix based on genotype data for additive effect. Option to use different methods
	makeG_inter()	Construct kinship matrix based on genotype data for dominance and epistatic effect. Option to use multithreading
	makeH()	Combine information of pedigree and genotypes to construct kinship matrix for both genotyped and ungenotyped individuals
Microbiome	makeM()	Construct kinship matrix based on microbiome data
Transcriptome	makeT()	Construct kinship matrix based on transcriptome data
Composition analysis and visualization	coefKinship()	Calculate the ancestry coefficients using kinship matrix
	coefInbreeding()	Calculate the inbreeding coefficients using kinship matrix
	cluster()	Cluster analysis of the kinship matrix and plot the result
	pca()	Principal component analysis of kinship matrix and plot the result
	heat()	Plot the heatmap of a kinship matrix
	gragh()	Plot family tree tracing back up to three generations of an individual

non-alleles, which is a fixed component of intergenerational inheritance and is also called breeding value in breeding. The dominant effect refers to the difference between the effect value of each gene and its additive effect value, which is derived from the effect of the interaction between alleles and is a non-additive effect, called dominant deviation. This effect can be inherited but not fixed and is the main part of the heterozygous advantage. The program for function *makeA()* improves the speed of computation by referring to the algorithm propose by Meuwissen and Luo [17]. And *makeD()* further calculates the dominant effects on the basis of *makeA().* To improve the speed of computation, both functions were written in C + + and support multi-threaded operation, while using sparse matrices to save memory.

Construction of kinship matrix based on genomic data

With the development of genotyping technology, more and more genomic data are available for GP, like GBLUP method [18]. The *makeG()* function provides four methods for calculating the additive effects kinship matrix. In a single population, the first method to calculate G matrix was developed by VanRaden [1]:

$$G = \frac{\sum (x_{ij}-2p_i)(x_{ik}-2p_i)}{\sum 2p_i(1-p_i)} \tag{1}$$

where x_{ij} and x_{ik} are the genotypes of the ith marker in individuals j and k (denoted as 0, 1 and 2). p_i is the minor allele frequency (MAF) of the ith marker. The second method of calculating G matrix was developed by Yang et al. [2]:

$$G = \frac{1}{m} \times \sum \frac{(x_{ij}-2p_i)(x_{ik}-2p_i)}{2p_i(1-p_i)} \tag{2}$$

where m is the number of markers, while other symbols represent the same meaning as formula (1). When computing G matrix in a combined popualtion, due to the differences in MAF between different populations, the direct use of the above method may bring bias. PyAGH provides two alternative methods to consider the heterogeneity of genetic structure of the combined population. One method for calculationg G matrix considering MAF differences between populations was developed by Chen et al. [19]:

$$G = \begin{bmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \\ \mathbf{G}_{21} & \mathbf{G}_{22} \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}}}{\sum 2\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})} & \frac{\mathbf{W}_1\mathbf{W}_2^{\mathrm{T}}}{\sum 2\sqrt{\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})}} \\ \frac{\mathbf{W}_2\mathbf{W}_1^{\mathrm{T}}}{\sum 2\sqrt{\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})}} & \frac{\mathbf{W}_2\mathbf{W}_2^{\mathrm{T}}}{\sum 2\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})} \end{bmatrix} \tag{3}$$

where $\mathbf{G}_{11}$ and $\mathbf{G}_{22}$ represent the genomic kinship matrix of individuals in two independent populations, respectively. $\mathbf{G}_{12}$ and $\mathbf{G}_{21}$ represent the genomic kinship matrix of individuals cross two populations. $\mathbf{W}_1$ and $\mathbf{W}_2$ are standardized genotypes of individuals in two populations, respectively. $\mathrm{p}_{1\mathrm{j}}$ and $\mathrm{p}_{2\mathrm{j}}$ are the minor allele frequencies of the jth marker calculated based on population 1 and population 2, respectively. Another method calculationg G matrix in combined populations was developed by Wientjes et al. [20]:

$$G = \begin{bmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \\ \mathbf{G}_{21} & \mathbf{G}_{22} \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}}}{\sum 2\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})} & \frac{\mathbf{W}_1\mathbf{W}_2^{\mathrm{T}}}{\sqrt{\sum 2\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})}\sqrt{\sum 2\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})}} \\ \frac{\mathbf{W}_2\mathbf{W}_1^{\mathrm{T}}}{\sqrt{\sum 2\mathrm{p}_{1\mathrm{j}}(1-\mathrm{p}_{1\mathrm{j}})}\sqrt{\sum 2\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})}} & \frac{\mathbf{W}_2\mathbf{W}_2^{\mathrm{T}}}{\sum 2\mathrm{p}_{2\mathrm{j}}(1-\mathrm{p}_{2\mathrm{j}})} \end{bmatrix} \# \tag{4}$$

where the symbols represent the same meaning as formula (3).

The *makeG_inter()* function calculates the dominant effect and epistatic effect kinship matrix based on genomic data according to algorithms proposed by Xu [21]. Epistatic effects refer to the effects of interactions between non-allelic genes at different loci, where one pair of genes suppresses or masks the other pair of genes. The formulas for dominant kinship matrix (d) and four epistatic kinship matrices (aa, dd, ad, da) are show in Table 2, where the **Z** and **W** represent the genotype matrices of different coding modes. For *k*th marker in individual *j*:

$$Z_{jk} = \begin{cases} +1 & for\ A \\ 0 & forH \\ -1 & for\ B \end{cases} \quad W_{jk} = \begin{cases} 0 & for\ A \\ 1 & for\ H \\ 0 & for\ B \end{cases} \tag{5}$$

Z_{jk} and W_{jk} represent the codes for additive and dominance effects, respectively, and A (the first homozygote), H (heterozygote), and B (the second homozygote) indicate the three genotypes. Z_{k} # W_{k} represents element-wise vector multiplication.

The original kinship matrix were normalized by dividing the mean of all diagonal elements of the original matrix so that the diagonal elements are approximately equal to 1. Using the normalized kinship matrix will result in the estimated genetic variance having the same scale as the residual variance.

The *makeH()* function combines information of pedigree and genotypes to construct kinship matrix **H** for both genotyped and ungenotyped individuals used in single-step genomic best linear unbiased prediction (ssGBLUP) method [3, 22]. The formula for **H** matrix is:

$$H = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{21} & \mathbf{H}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11} + \mathbf{A}_{12}\mathbf{A}_{22}^{-1}(\mathbf{G}_{\mathbf{w}} - \mathbf{A}_{22})\mathbf{A}_{22}^{-1}\mathbf{A}_{21} & \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{G}_{\mathbf{w}} \\ \mathbf{G}_{\mathbf{w}}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} & \mathbf{G}_{\mathbf{w}} \end{bmatrix} \tag{6}$$

where subscript 1 represent the individuals without genotypes and subscript 2 represent the individuals with genotypes. $\mathbf{A}_{11}$, $\mathbf{A}_{12}$, $\mathbf{A}_{21}$ and $\mathbf{A}_{22}$ are constructed by pedigree information.$\mathbf{G}_{\mathbf{w}}$ is calculated by $\mathbf{G}_{\mathbf{w}} = (1-w)\mathbf{G}^* + w\mathbf{A}_{22}$. The parameter w is used adjust the relative weights of the **G** matrix and the **A** matrix. $\mathbf{G}^* = a + b\mathbf{G}$, a and b are achieved by:

Table 2 Formulas used to calculate marker generated kinship matrices

Type of effect	Original kinship matrix	Kinship matrix
Dominance(d)	$K_d^* = \sum_{k=1}^{m} W_k W_k^T$	$K_d = (\frac{1}{mean[diag(K_d^*)]})K_d^*$
Additive × additive (aa)	$K_{aa}^* = \sum_{k=1}^{m-1}\sum_{k'=k+1}^{m}(Z_k\#Z_{k'})(Z_k\#Z_{k'})^T$	$K_{aa} = (\frac{1}{mean[diag(K_{aa}^*)]})K_{aa}^*$
Dominance × dominance (dd)	$K_{dd}^* = \sum_{k=1}^{m-1}\sum_{k'=k+1}^{m}(W_k\#W_{k'})(W_k\#W_{k'})^T$	$K_{dd} = (\frac{1}{mean[diag(K_{dd}^*)]})K_{dd}^*$
Additive × dominance (ad)	$K_{ad}^* = \sum_{k=1}^{m-1}\sum_{k'=k+1}^{m}(Z_k\#W_{k'})(Z_k\#W_{k'})^T$	$K_{ad} = (\frac{1}{mean[diag(K_{ad}^*)]})K_{ad}^*$
Dominance × additive (da)	$K_{dd}^* = \sum_{k=1}^{m-1}\sum_{k'=k+1}^{m}(W_k\#Z_{k'})(W_k\#Z_{k'})^T$	$K_{da} = (\frac{1}{mean[diag(K_{da}^*)]})K_{da}^*$

$$Avg\left(diag(\mathbf{G})\right) \times b + a = Avg\left(diag(\mathbf{A}_{22})\right)$$

$$Avg\left(offdiag(\mathbf{G})\right) \times b + a = Avg\left(offdiag(\mathbf{A}_{22})\right) \quad (7)$$

where $\mathbf{A}_{22}$ is a submatrix of $\mathbf{A}$ related to the genotyped individuals; $\mathbf{G}$ is a additive genomic relationship matrix of genotyped individuals. $Avg\left(diag(\mathbf{G})\right)$ is the average of the diagonal of the $\mathbf{G}$ matrix. And $Avg\left(offdiag(\mathbf{G})\right)$ is the average of the off-diagonal of the $\mathbf{G}$ matrix.

Construction of kinship matrix based on microbiome and transcriptome data

The host associated microbiome is known to influence many traits. A number of studies have reported that combining microbiome and genomic information could improve the prediction accuracy compared with only genomic data [23]. The *makeM()* function can easily normalize operational taxonomic units (OTU) as well as calculate the kinship matrix based on microbiome data. The formula for **M** matrix is as follows:

$$M = \frac{\mathbf{O}\mathbf{O}^{\mathbf{T}}}{n} \# \quad (8)$$

where n is the number of OUT in population. **O** is the original OTU matrix after natural logarithmic variation and normalization. And for each OUT j of individual i, the transformation formula is:

$$O_{ij} = \frac{log(X_{ij}) - \overline{log(X_{ij})}_j}{sd\left(log(X_{ij})\right)_j} \# \quad (9)$$

where $\mathrm{X_{ij}}$ is the abundance of the jth OTU of the ith individual. $\mathrm{sd}\left(\log\left(\mathrm{X_{ij}}\right)\right)_{\mathrm{j}}$ is the standard deviation of jth OUT in all individuals.

In addition, modeling transcriptome data as predictors in genomic prediction is expected to explain more nonlinear variation or complex biological regulatory processes and has the potential to improve the accuracy of prediction [24]. The *makeT()* function in PyAGH can simply calculate the kinship matrix based on transcriptome data. The formula for **T** matrix is as follows:

$$T = R\mathbf{R}^{\mathbf{T}} \# \quad (10)$$

R is the normalized gene expression matrix, and the normalization formula is:

$$r_{ij} = \frac{X_{ij} - \overline{X_j}}{sd_j} \# \quad (11)$$

where X_{ij} is the expression of gene j in individual i, $\overline{X_j}$ is the mean of the expression of gene j in all individuals, and sd_j is the standard deviation of the expression of gene j in all individuals.

Pedigree and composition analysis and visualization

Pedigree provides important information for estimating breeding values in the field of plant and animal breeding. To make it easier to use such information, PyAGH provides targeted tools for specific demands, such as detecting common pedigree errors (like offspring born before its parents, individual with two genders, same offspring have different parents, and etc.), selection target individuals (a subset of the whole pedigree), sorting pedigree by birthdate, pedigree visualization, calculating inbreeding coefficients and ancestry coefficients. In addition, principal component analysis (PCA), heatmap and cluster analysis functions were involved in PyAGH to reveal population structure conveniently.

Results

To support the robustness and speed of the package, we tested the performance of main functions with different cases data in a Linux machine with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30 GHz and 256 GB RAM. First, we compared the makeA function with the Nadiv (https://github.com/matthewwolak/nadiv) package using a dataset containing 100,000 pedigree records. Nadiv is a widely used R package for processing pedigree data. The results of the comparison between the two softwares were shown in Table 3. When the number of records is small, the computational speed of PyAGH and Nadiv is not much different, and even Nadiv is slightly faster than PyAGH. But PyAGH can support a larger number of pedigree data. For example, when the number of records reache 100,000, Nadiv was unable to perform the calculation, while PyAGH took only about 13 min to complete the calculation. This indicates that PyAGH can support a larger amount of pedigree data while maintaining speed compared to Nadiv package when calculating the pedigree additive kinship matrix. In addition, because the first step in calculating the dominance effect kinship matrix based on pedigree data is to calculate the additive effect kinship matrix, i.e., the makeA function is the basis of the makeD function, PyAGH can also support a larger amount of pedigree data for the calculation of the dominance effect kinship matrix.

Next, we tested functions that perform calculations based on genomic data. We compared the makeG function with GCTA software using a dataset containing 10,000 individuals and 1 million SNPs for one chromosome. The runing time for the two software to calculate the additive genomic kinship matrix for different number of individuals are shown in Table 4. Regardless of the number of individuals, PyAGH computed the G matrix faster than GCTA. In addition, PyAGH provides two additional methods for calculating additive kinship matrices in combined populations, whereas GCTA does not calculate. Therefore, using PyAGH makes it easier and faster to perform matrix calculations based on research needs.

The function makeG_inter in PyAGH, which calculates the dominance effect kinship matrix based on genomic data, was compared with PEPIS platform [25]. PEPIS

Table 3 Runtime and RAM of construction kinship matrix based on pedigree in PyAGH and Nadiv

Number of records	PyAGH	Nadiv
10,000	3 s (0.3 GB)	2 s (0.3 GB)
20,000	32 s (1.5 GB)	23 s (1.6 GB)
50,000	2min8s (11.9 GB)	1min47s (16.4 GB)
100,000	13min35s (100.6 GB)	–

The numbers of pedigree records verify from 10,000 to 100,000

– means that it cannot be calculated

Table 4 Running time of constructing kinship matrices based on genotypic data in PyAGH and GCTA

Number of individuals	PyAGH	GCTA
1000	10 s	2min32s
3000	51 s	2min36s
5000	1min39s	3min44s
10,000	4min50s	7min40s

Both PyAGH and GCTA used all 64 threads of the machine for computation

We fixing the number of snps at 1,000,000 while varying the numbers of individuals from 1,000 to 10,000

Table 5 Running time of construction kinship matrix for epistatic effect in PyAGH and PEPIS

Number of SNPs	PyAGH	PEPIS
4000	4 min	5 min
10,000	16 min	34 min
20,000	48 min	121 min
40,000	140 min	488 min

The test data is simulation data used in PEPIS (http://bioinfo.noble.org/PolyGenic_QTL/). We fixing the number of individuals at 1,000 while varying the numbers of SNPs from 4,000 to 40,000

is a pipeline for estimating epistatic effects in quantitative trait locus mapping and genome-wide association studies. Since PEPIS is a cloud-based platform, we used the test data provided by PEPIS including 1000 individuals and 40,000 SNP for PyAGH testing. Table 5 shows the running time for PyAGH and PEPIS to calculate the kinship matrices of the four dominance effects aa, dd, ad, da, from which it can be seen that the advantage of PyAGH over PEPIS increases as the number of loci increases. At 40,000 loci, the computational speed of PyAGH was about 3 ~ 4 times faster than that of PEPIS. Whether using pedigree or genomic information, PyAGH has speed advantages over other softwares and can support larger data sizes.

Because there is no software to calculate the kinship matrix based on microbiome data, we tested PyAGH in a dataset containing 16 s RNA sequencing data of 4500 pigs [26]. The results show that the package can quickly calculate the **M** matrix in the case of meeting the data size of a conventional study. When we fix the number of OTU at 100,000 and the number of individuals varies from 1,000 to 4,500, the time taken increases linearly (Fig. 1A). When we fix the number of individuals at 4,500 while varying the numbers of OTU from 10,000 to 100,000, the time taken increases as a quadratic function (Fig. 1B). For all 4,500 individuals and 100,000 OTU, PyAGH took about 20 s, and it can be seen that our software can quickly normalize the OUT matirx and calculate the kinship matrix.

Gene expression data can provide additional information in genomic prediction and can also be used to further explore the genetic mechanisms of traits in association studies. With the increase of transcriptome sequencing data, the application of transcriptome data in GP and GWAS will increase. PyAGH can quickly and easily calculate kinship matrix based on gene expression abundance data. And we used the gene expression data in muscle tissue of 1321 pigs from FarmGTEx (https://www.farmgtex.org/) as an example [27]. We performed PCA of the kinship matrices based

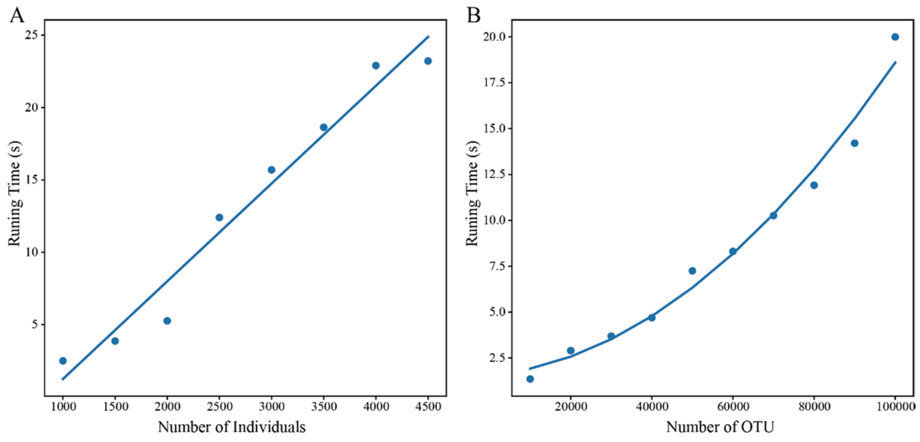

Fig. 1 Running time of PyAGH function makeM in different data. **A** fix the number of OTU at 100,000 while varying the numbers of individuals from 1,000 to 4,500. **B** fix the number of individuals at 4,500 while varying the numbers of OTU from 10,000 to 100,000

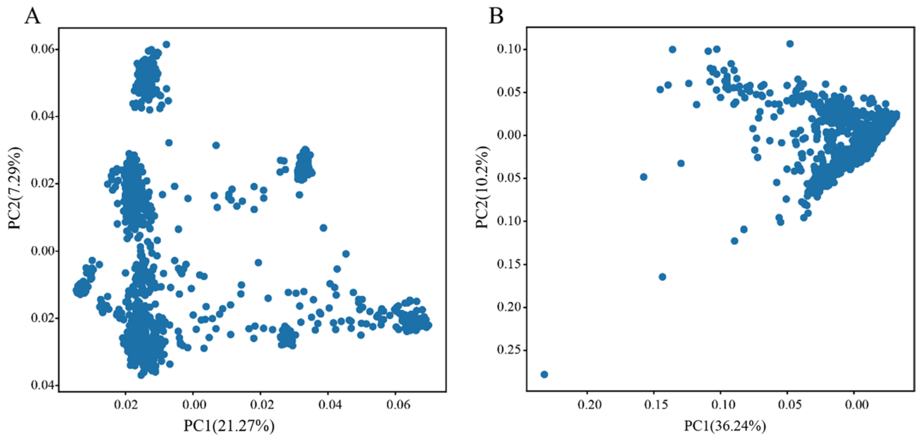

Fig. 2 Scatter plots of the first and the second principal components from kinship matrices based on genome (**A**) and transcriptome (**B**), respectively

on genomic data (Fig. 2A) and transcriptomic data (Fig. 2B), respectively. The results show that the kinship matrices calculated based on the two data were different, indicating that the transcriptome data provide additional information different from the genome.

In addition to calculating a variety of kinship matrices, PyAGH can also quickly check pedigree data, extract specific subsets of individuals on demand, and calculate ancestry coefficients and inbreeding coefficients. These features allow the user to easily organize the pedigree data to focus on the next analysis process. At the same time, PyAGH allows for a variety of visualizations including PCA, Heatmap, clustering and family trees. Figure 3A, B shows the heat map and clustering diagram drawn using the example data in the package. Figure 3C shows the results of PCA analysis of the genomic data of two populations using PyAGH. Data were obtained from previous study of two large white pig populations [28]. The left figure is PCA variance explained based on custom PCA. The right figure is PCA plot of top 2 PCs. Figure 3D

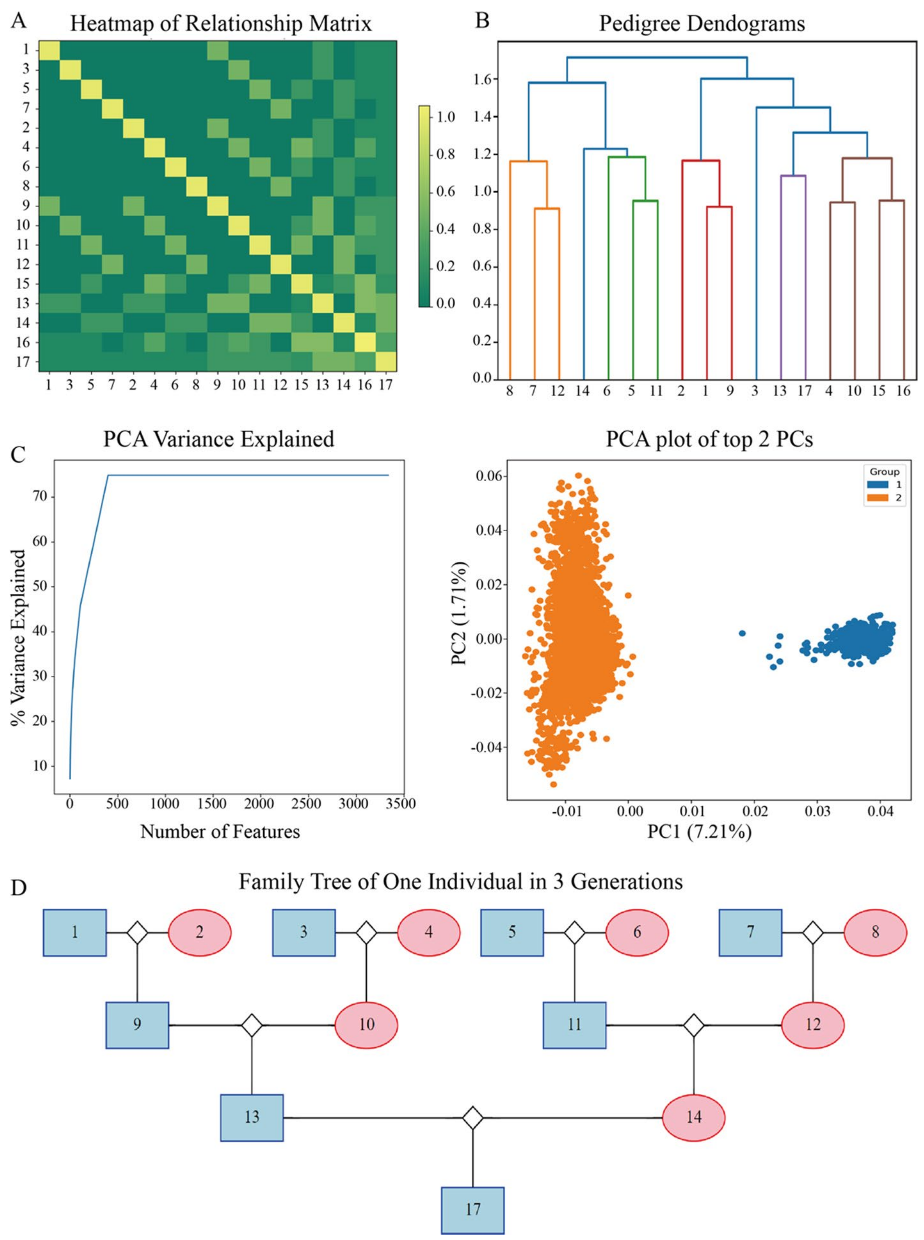

Fig. 3 **A** Heat-map of a kinship matrix. **B** The pedigree dendograms of the cluster results. **C** PCA analysis results. The left figure is PCA variance explained based on custom PCA. The right figure is PCA plot of top 2 PCs. **D** Family tree of one individual in three generations

was a family tree of one specific individual in three generations. This function is useful in production practice.

Conclusions

In this study, we have presented PyAGH, which is a robust and fast Python package for calculating kinship matrices using pedigree, genotype, microbiome and transcriptome data as well as processing, analyzing and visualizing data and results. This package provides various methods for kinship matrices construction based on additive, dominant and epistatic effects in a single population or combined populations. The PyAGH package has been intensively tested to guarantee the computation correctness and speed. Compared to existing tools, PyAGH exhibited the best performance for constructing a variety of matrices. And the calculation results can be easily used in other softwares, making the process of genome prediction and association studies more convenient. PyAGH is a python package that completes the process of using python for bioinformatics analysis. In the future work, we plan to apply more comprehensive kinship matrix calculation methods and multi-omics data processing to the coming version of PyAGH. In conclusion, PyAGH simplifies the procedure of calculating kinship matrices that are important for prediction or association studies.

Availability and requirements

Project name: PyAGH.

Project homepage: https://github.com/zhaow-01/PyAGH

Operating System(s): Mac Os, Linux, Windows.

Programming language: Python, C++.

Other requirements: All dependencies are handled during the installation.

License: MIT.

Any restrictions to use by non-academic: PyAGH has no restriction.

Abbreviations

BLUP	Best linear unbiased prediction
GWAS	Genome wide association studies
GP	Genomic prediction
GBLUP	Genomic best linear unbiased prediction
MAF	Minor alle frequencies
ssGBLUP	Single-step genomic best linear unbiased prediction
OTU	Operational taxonomic units
PCA	Principal component analysis

Acknowledgements
Not applicable.

Author contributions

W.Z. and Z.Y.Z. wrote the code; Z.Z., Q.R.Q. and Q.S.W. tested the code and prepared the diagrams and figures; W.Z. wrote the manuscript. W.Z., Y.C.P. and Z.Z. helped check and improve the manuscript. All authors contributed into design of the study. All authors read and approved the final manuscript.

Funding
This work was supported by National Natural Science Foundation of China [32102503], Zhejiang Provincial Key R&D Program of China [2021C02008, 2021C02068] and Shanghai Agricultural Science and technology innovation project No.1-5 (2020).

Availability of data and materials

The pedigree data underlying this article are available at https://github.com/zhaow-01/PyAGH/tree/main/PyAGH/data. The two pig populations genomic data are available in Alphaindex platform at http://alphaindex.zju.edu.cn/ALPHADB/download.html. The 16 s RNA sequencing data are available in GSA at https://ngdc.cncb.ac.cn/gsa/ database, and can be accessed with accession numbers: CRA006230, CRA006239, CRA006240, CRA006216. The transcriptome data are available in FarmGTEx at http://piggtex.farmgtex.org/. The source code of PyAGH is deposited in a Github repository https://github.com/zhaow-01/PyAGH/tree/main/PyAGH.

Declarations

Ethics approval and consent to participate

No ethics approval and consent required for this study.

Consent for publication

Not applicable.

Competing interests
The authors declare no competing fnancial interests.

Received: 3 December 2022 Accepted: 10 April 2023
Published online: 18 April 2023

References

1. VanRaden PM. Efficient methods to compute genomic predictions. J Dairy Sci. 2008;91:4414–23.
2. Yang J, Benyamin B, McEvoy BP, et al. Common SNPs explain a large proportion of the heritability for human height. Nat Genet. 2010;42:565–9.
3. Christensen OF, Lund MS. Genomic prediction when some animals are not genotyped. Genet Sel Evol. 2010;42:2.
4. Wang H, Misztal I, Aguilar I, et al. Genome-wide association mapping including phenotypes from relatives without genotypes. Genet Res. 2012;94:73–83.
5. Varona L, Legarra A, Toro MA, et al. Genomic prediction methods accounting for nonadditive genetic effects. genomic predict. Complex Traits Methods Protoc. 2022; 219–243
6. Momen M, Morota G. Quantifying genomic connectedness and prediction accuracy from additive and non-additive gene actions. Genet Sel Evol GSE. 2018;50:45.
7. Calleja-Rodriguez A, Chen Z, Suontama M, et al. Genomic predictions with nonadditive effects improved estimates of additive effects and predictions of total genetic values in *Pinus sylvestris*. Front Plant Sci. 2021;12: 666820.
8. Yang J, Lee SH, Goddard ME, et al. GCTA: a tool for genome-wide complex trait analysis. Am J Hum Genet. 2011;88:76–82.
9. Madsen P, Jensen J. A package for analysing multivariate mixed models. Version 6, release 5.2. 2013;
10. Azodi CB, Pardo J, VanBuren R, et al. Transcriptome-based prediction of complex traits in maize. Plant Cell. 2020;32:139–51.
11. Hughes RL, Marco ML, Hughes JP, et al. The role of the gut microbiome in predicting response to diet and the development of precision nutrition models—part I: overview of current methods. Adv Nutr. 2019;10:953–78.
12. Awany D, Allali I, Dalvie S, et al. Host and microbiome genome-wide association studies: current state and challenges. Front Genet. 2019;9:637.
13. Wainberg M, Sinnott-Armstrong N, Mancuso N, et al. Opportunities and challenges for transcriptome-wide association studies. Nat Genet. 2019;51:592–9.
14. Zhou X, Stephens M. Genome-wide efficient mixed-model analysis for association studies. Nat Genet. 2012;44:821–4.
15. Loh P-R, Kichaev G, Gazal S, et al. Mixed-model association for biobank-scale datasets. Nat Genet. 2018;50:906–8.
16. Henderson CR. Estimation of variance and covariance components. Biometrics. 1953;9:226–52.
17. Meuwissen T, Luo Z. Computing inbreeding coefficients in large populations. Genet Sel Evol. 1992;24:305.
18. Meuwissen TH, Hayes BJ, Goddard ME. Prediction of total genetic value using genome-wide dense marker maps. Genetics. 2001;157:1819–29.
19. Chen L, Schenkel F, Vinsky M, et al. Accuracy of predicting genomic breeding values for residual feed intake in Angus and Charolais beef cattle. J Anim Sci. 2013;91:4669–78.
20. Wientjes Y, Bijma P, Vandenplas J, et al. Multi-population genomic relationships for estimating current genetic variances within and genetic correlations between populations. Genetics 2017; genetics.300152.2017
21. Xu S. Mapping quantitative trait loci by controlling polygenic background effects. Genetics. 2013;195:1209–22.
22. Legarra A, Aguilar I, Misztal I. A relationship matrix including full pedigree and genomic information. J Dairy Sci. 2009;92:4656–63.
23. Ross EM, Hayes BJ. Metagenomic predictions: a review 10 years on. Front Genet. 2022;13: 865765.
24. Li Z, Gao N, Martini JWR, et al. Integrating gene expression data into genomic prediction. Front. Genet. 2019; 10:
25. Zhang W, Dai X, Wang Q, et al. PEPIS: a pipeline for estimating epistatic effects in quantitative trait locus mapping and genome-wide association studies. PLOS Comput Biol. 2016;12: e1004925.
26. Yang H, Wu J, Huang X, et al. ABO genotype alters the gut microbiota by regulating GalNAc levels in pigs. Nature. 2022;606:358–67.
27. Consortium TF-P, Teng J, Gao Y, et al. A compendium of genetic regulatory effects across pig tissues. 2022; 2022.11.11.516073
28. Zhao W, Zhang Z, Ma P, et al. The effect of high-density genotypic data and different methods on joint genomic prediction: a case study in large white pigs. Anim. Genet. n/a:

Publisher's Note