|  |
| --- |
| DeepGS: Predicting phenotypes from genotypes using Deep Learning  Wenlong Ma1, †, Zhixu Qiu1, †, Jie Song1, Qian Cheng1,2, Chuang Ma1,2,\*  1State Key Laboratory of Crop Stress Biology for Arid Areas, Center of Bioinformatics, College of Life Sciences, Northwest A&F University, Yangling, Shaanxi 712100, China, 2Key Laboratory of Biology and Genetics Improvement of Maize in Arid Area of Northwest Region, Ministry of Agriculture, Northwest A&F University, Shaanxi, Yangling 712100, China.  \*To whom correspondence should be addressed.  † These authors contributed equally to this work. |

# Optimization of weights in the integrated GS model based on particle swarm optimization (PSO) algorithm

An integrated GS model (*I*) was constructed using the ensemble learning approach by linearly combining the predictions of DeepGS (*D*) and RR-BLUP (*R*), using the formula:

.

where, *predict* represents the predicted values, *W* represents the weights in the *I* GS model.

For each fold of 10-cross fold procedure, parameters (and *)* were optimized on the corresponding validation dataset using the particle swarm optimization (PSO) algorithm, which was developed by inspiring from the social behavior of bird flocking or fish schooling ([Kennedy and Eberhart, 1995](#_ENREF_1)). This optimization process repeated ten times. The averaged weights were used to build the integrated GS model, the prediction performance of which was evaluated on the testing dataset.

To perform the PSO-based weight optimization, we defined the particles as the weights (*W*) and corresponding updated velocities (*V*) of the two GS models (RR-BLUP and DeepGS), the swarm as the gather of the particles, and *L* as the length of the swarm. Additionally, we also set the inertia weight (*IW*) of 1, the accelerated factor 1 (*AF*1) and 2 (*AF*2) of 2, the *W* interval of [0, 1], the *V* interval of [-0.01, 0.01]. The fitness function was the mean normalized discounted cumulative gain value (MNV) of the integrated GS model (*I*) values. The optimal objective was to maximize the MNV of the integrated GS model at the top-ranked level of α = 1% (). The steps and corresponding R scripts in DeepGS of implementing PSO-based weight optimization are shown below.

Firstly, *W* and *V* were randomly initialized by the uniform distribution (function “runif” in R programming language):

,

1. #Ensemble Learning Based on Particle Swarm Optimization (ELBPSO) initialization function
2. ELBPSO\_initialized<-function(weight\_dimension,weight\_min,weight\_max,rate\_min,rate\_max,paticle\_number,para\_deliver,IW,AF1,AF2)
3. #Prepare the predicted matrix including the real phenotypic values, predicted RR-BLUP phenotypic values and the predicted DeepGS phenotypic values
4. pred\_matrix <- cbind(real\_pheno,pre\_rr\_pheno,pre\_deepgs\_pheno)
5. #Prepare the parameters’ deliver list
6. para\_deliver <- list()
7. para\_deliver$predict\_models <- pred\_matrix[,2:ncol(pred\_matrix)]
8. para\_deliver$indepedent\_pheno <- pred\_matrix[,1]
9. para\_deliver$train\_num <- c(1:length(pred\_matrix[,1]))
10. best\_weight\_rep <- matrix(,ncol(pred\_matrix)-1,rep\_times)
11. #ELBPSO initialized
12. Parameter\_matrix<-ELBPSO\_initialized(weight\_dimension,weight\_min,weight\_max,rate\_min,rate\_max,paticle\_number,para\_deliver,IW,AF1,AF2)

Secondly, *W* and *V* were iteratively updated by the following formula:

,

where, , , *t* denotes the iteration time, *r*1, *r*2, *r*3 represent three stochastic parameters randomly generated from [0, 1] by the uniform distribution (function “runif”).

1. #ELBPSO optimized function
2. ELBPSO\_optimized<-function(Parameter\_matrix,weight\_dimension,weight\_min,weight\_max,rate\_min,rate\_max,paticle\_number,para\_deliver,IW,AF1,AF2)
3. #ELBPSO fitness function
4. ELBPSO\_fitness<-function(weight, para\_deliver)
5. #ELBPSO global update function
6. ELBPSO\_global\_update<-function(Parameter\_matrix,weight\_dimension,paticle\_number
7. #ELBPSO optimized
8. Parameter\_matrix<-ELBPSO\_optimized(Parameter\_matrix,weight\_dimension,weight\_min,weight\_max,rate\_min,rate\_max,paticle\_number,para\_deliver,IW,AF1,AF2)

Then, the first and second steps repeated, until the pre-defined iteration time was reached or a pre-defined stop criterion was reached.

Finally, the RR-BLUP and DeepGS were integrated in a linear fashion:

1. #Global best parameters
2. Global<-Parameter\_matrix[(paticle\_number+1),1:weight\_dimension]
3. #Best weights for each repeated times
4. best\_weight\_rep[,rep] <- Global
5. #Average of best weights
6. weight<- apply(best\_weight\_rep,1, mean)
7. #Integrated GS predicted values
8. new\_pre<- apply(t(t(pred\_matrix[,2:ncol(pred\_matrix)])\*weight),1,sum)/sum(weight)

# Supplementary figures



**Supplementary Fig. S1.** MNV curves of DeepGS, FNN, and random selection for predicting phenotypic values of the eight tested traits using 33,709 DArT markers.



**Fig. S2.** Outlier individuals (OIs) and their effects on prediction performance. (A) Boxplots of observed phenotypic values of the eight tested traits. The numbers OIs with extremely high (or low) phenotypic values are shown in pink (or in light blue). (B) MNV improvement of RR-BLUP (blue) and DeepGS (red) on GS dataset without OIs compared to that with OIs. (C) MNV improvement of DeepGS (red) and the integrated GS model (green) over RR-BLUP for five traits, testing using the GS dataset without OIs.



**Fig. S3.** MNV improvement for DeepGS (red) and the integrated GS model (green) over RR-BLUP, when different subsets of marker were used. (A) 20,000 markers, (B) 10,000 markers, (C) 5,000 markers.

# Supplementary references

Kennedy, J. and Eberhart, R. (1995) Particle swarm optimization. *IEEE Intl Conf Neural Networks* 1995;4:1942-1948.