





RESEARCH PAPER

Evaluation of Predictive Ability of Bayesian Regularized Neural Network Using Cholesky Factorization of Genetic Relationship Matrices for Additive and Non-additive Genetic Effects

Hayrettin Okut^{1,2,*} , Daniel Gianola^{1,3} , Kent A. Weigel¹ , Guilherme J. M. Rosa^{1,3} 

¹Department of Animal and Dairy Sciences, University of Wisconsin, Madison, 53706, USA

²University of Kansas, School of Medicine-Wichita, 67214, USA

³Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, 53706, USA

*Corresponding Author

Article History

Received: 22 June 2022

Accepted: 06 July 2022

First Online: 09 August 2022

Corresponding Author*

Tel.:

E-mail: hokut@kumc.edu

Keywords

Artificial neural networks

Bayesian regularization

Additive and non-additive genetic effects

Abstract

This study aimed to explore the effects of additive and non-additive genetic effects on the prediction of complex traits using Bayesian regularized artificial neural network (BRANN). The data sets were simulated for two hypothetical pedigrees with five different fractions of total genetic variance accounted by additive, additive x additive, and additive x additive x additive genetic effects. A feed forward artificial neural network (ANN) with Bayesian regularization (BR) was used to assess the performance of different nonlinear ANNs and compare their predictive ability with those from linear models under different genetic architectures of phenotypic traits. Effective number of parameters and sum of squares error (SSE) in test data sets were used to evaluate the performance of ANNs. Distribution of weights and correlation between observed and predicted values in the test data set were used to evaluate the predictive ability. There were clear and significant improvements in terms of the predictive ability of linear (equivalent Bayesian ridge regression) and nonlinear models when the proportion of additive genetic variance in total genetic variance (σ_a^2 / σ_G^2) increased. On the other hand, nonlinear models outperformed the linear models across different genetic architectures. The weights for the linear models were larger and more variable than for the nonlinear network, and presented leptokurtic distributions, indicating strong shrinkage towards 0. In conclusion, our results showed that: a) inclusion of non-additive effects did not improve the prediction ability compared to purely additive models, b) The predictive ability of BRANN architectures with nonlinear activation function were substantially larger than the linear models for the scenarios considered.

Introduction

Additive genetic variance generally accounts for most or all of total genetic variance in complex traits (Van Tassel 2000; Nagy *et al.* 2013). Nonetheless, there is a long-standing controversy and paradox about the importance of non-additive genetic effects in breeding programs. For example, some authors argue that the prediction of genetic response may be biased upwards if non-additive genetic variances are not included in genetic models (Cheverud, & Routman 1995; Carlborg & Haley 2004; Hallander & Waldmann 2007). In addition, many authors have reported that accounting for non-additive effects in the genetic effects might improve the estimation of additive effects, resulting in less biased

prediction (e.g., Wittenburg *et al.* 2011; Colleen *et al.* 2013; Powell *et al.* 2013).

Interactions between genes are known to be common and are a key concept for understanding adaptation and evolution of species as well as long-term response to selection in breeding programs (Alvarez-Castro & Carlborg 2007; Ingileif & Yuster 2008). An alternative to better understand the genetic architecture of complex traits is to include intralocus (dominance) and inter-locus (epistasis) interaction of alleles when fitting a model to a trait (Jamrozik *et al.* 2005; Oakey *et al.* 2006; Valentina *et al.* 2007; Wittenburg *et al.* 2011). However, modelling additive and non-additive effects simultaneously poses challenges in terms of computational demand, data

analysis and interpretation of results. Several statistical and computational methods have been devised for studying the association between complex traits and high dimensional data sets. Classical single-marker regressions and, more recently, Bayesian linear regression models of various types (Gianola *et al.* 2011; Meuwissen *et al.* 2009; de los Campos *et al.* 2009; de los Campos *et al.* 2010) are focused on additive inheritance and ignore interactions and non-linearity. In fact, a study by Wittenburg *et al.* (2011) confirmed that parametric methods have difficulties in identifying and estimating some non-genetic effects. Alternatively, artificial neural networks (ANNs) provide a powerful technique for learning about complex traits by predicting the future state of an outcome variable based on training data. ANNs can capture non-linear relationships between predictors (e.g. SNP markers) and responses (e.g. phenotypes) and learn about functional forms in an adaptive manner (Okut *et al.* 2011; Gianola *et al.* 2011). The ability of computing complex nonlinear relationships between response and input variables, including any kind of interactions between input variables, with many training algorithms available makes ANN extremely interesting for analysis of complex traits (Lampinen & Vehtari 2001; Okut 2016; Okut 2021).

One of the most serious problems that can occur when training an ANN is overfitting. Since the ultimate goal of ANN is attaining generalization such overfitting problems should be detected and addressed carefully. Bayesian Regularized Artificial Neural Network (BRANN) is more resistant to overcome the problem of overfitting problem than conventional ANN, resulting in improved generalization performance (Gianola *et al.* 2011; Okut *et al.* 2013). The idea of Bayesian regularization (BR) is to make the network response smoother, through modification of the objective function by adding a penalty term consisting of the mean square of all network coefficients (Gencay & Qi 2001). BR minimizes a combination of squared errors and weights, and then determines an appropriate combination so as to produce a network that generalizes well (Marwala 2007; Ripley 2007; MacKay 2008). As such, BR can be viewed as a nonlinear analog of ridge regression. The Bayesian approach to neural network modeling consists of arriving at the posterior probability distribution of weights by updating a prior probability distribution by means of a training data set (Okut *et al.* 2013).

The objectives of this paper were to use the ANNs to explore the impact of non-additive genetic effects (additive x additive, and additive x additive x additive) on the prediction of complex traits when additive and non-additive effects were jointly considered when fitting a model to a trait. For such, BRANN architectures differing in terms of number of neurons and activation functions were used and compared in terms of their prediction ability. Details on data simulation and modeling are presented in the next

section. The scenarios used to generate data by including only additive or a certain portion of non-additive genetic effects is explained in this section. On the second section, we summarize the results obtained from the different BRANNs. A final section presents a discussion and concluding remarks.

Materials and Methods

Data sets

The data considered in this paper were simulated for 5 non-overlapping generations. Two hypothetical pedigrees, referred to as population 1 (Pop1) and population 2 (Pop2), were generated. The number of animals in the base populations were 100 (50 nuclear families) and 300 (150 nuclear families), respectively. Animals used in the base populations were assumed to be unrelated and not inbred. Four additional generations were created with 100 and 300 animals in each generation for each population. Each family in the base population and subsequent generations had only 2 offspring. In both populations throughout the five generations (base and four additional generations), animals were randomly mated within their generations. At the end of the simulation the total number of animals in Pop1 was 500 (274 female and 226 male) and in Pop2 was 1500 (767 female and 733 male). A relationship matrix ($\mathbf{A}=\mathbf{C}\mathbf{C}'$, here \mathbf{C} is lower triangular Cholesky factor decomposition) from the animals of the 5 generations was then constructed for both populations and the elements of the lower triangular Cholesky factor decomposition of these relationship matrices (a_{ki}) were considered as input variables ($\mathbf{p}_i = a_{ki}$) in BRANN architectures.

The phenotypic records (t_i) were generated for all animals in the pedigree structure and were a function of genetic and random environmental effects from a normal distribution. For each generation, 50 and 150 nuclear families consisting of the two sibs values were obtained by adding a normally distributed environmental effect with 0 mean and unit variance. Target variables (t_i) which represent animals' phenotypes were generated according to the following equation:

$$\begin{aligned} \mathbf{t} &= \mathbf{a}_1 + \mathbf{a}_2 + \mathbf{a}_3 + \mathbf{e} \\ &= [\mathbf{A}^{1/2} \times \mathbf{u}_1 \times \sigma_a] + [(\mathbf{A}\#\mathbf{A})^{1/2} \times \mathbf{u}_2 \times \sigma_{aa}] \\ &\quad + [(\mathbf{A}\#\mathbf{A}\#\mathbf{A})^{1/2} \times \mathbf{u}_3 \times \sigma_{aaa}] + [\mathbf{u}_4 \\ &\quad \times \sigma_e] \end{aligned}$$

where \mathbf{a}_1 , \mathbf{a}_2 and \mathbf{a}_3 are vectors of additive, additive x additive, and additive x additive x additive genetic effects, and \mathbf{e} is a vector of random residual effects. In the equation above, the matrix \mathbf{A} represents the numerator relationship matrix, is the Hadamard product, and \mathbf{u}_1 , \mathbf{u}_2 , \mathbf{u}_3 and \mathbf{u}_4 are random vectors from multivariate standard normal distribution, i.e. $\mathbf{u}_j \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$, $j=1\dots 4$, where $\mathbf{0}$ is a column vector of zeros

and I is an identity matrix. With these settings we have that:

$$\begin{aligned} \mathbf{a}_1 &\sim N(\mathbf{0}, \mathbf{A}\sigma_a^2), \\ \mathbf{a}_2 &\sim N(\mathbf{0}, \mathbf{A}\#\mathbf{A}\sigma_{aa}^2), \\ \mathbf{a}_3 &\sim N(\mathbf{0}, \mathbf{A}\#\mathbf{A}\#\mathbf{A}\sigma_{aaa}^2) \text{ and} \\ \mathbf{e} &\sim N(\mathbf{0}, \mathbf{I}\sigma_e^2) \end{aligned}$$

and that the total phenotypic variance is given by:
 $\text{Var}(\mathbf{t}) = \mathbf{A}\sigma_a^2 + \mathbf{A}\#\mathbf{A}\sigma_{aa}^2 + \mathbf{A}\#\mathbf{A}\#\mathbf{A}\sigma_{aaa}^2 + \mathbf{I}\sigma_e^2$.

In the simulations, the total variance was assumed to be $V(t) = \sigma_T^2 = 2$, and the variance of

genetic and non-genetic effects was assumed to be $\sigma_G^2 = 1$ and $\sigma_e^2 = 1$, respectively. Here, σ_G^2 is the total genetic variance and σ_e^2 is the residual (random environmental) variance. Five different scenarios were considered as $\sigma_a^2 / \sigma_G^2 = 0, 0.1, 0.5, 0.9, \text{ and } 1$ (namely, five different fractions of variance accounted by additive genetic effect) for simulating phenotypic values. Heritability (h^2) was kept to be 0.5 for the all scenarios. The fractions of both non-additive effects (additive x additive, $\sigma_{aa}^2 / \sigma_G^2$, and additive x additive x additive, $\sigma_{aaa}^2 / \sigma_G^2$) were assumed to be equal in each model. For example, when the fraction of variance accounted by additive genetic effects was $\sigma_a^2 / \sigma_G^2 = 0.5$ then the

fractions of non-additive genetic effects were assumed $\sigma_{aa}^2 / \sigma_G^2 = 0.25$ and $\sigma_{aaa}^2 / \sigma_G^2 = 0.25$ (Table 1).

Feed-forward neural networks

A fully connected, two-layer feed-forward BRANN with backpropagation was used in this study and is illustrated in Figure 1. In Figure 1, \mathbf{p}_i is an input vector of elements of the relationship matrix (a_{kl}) at the left-most layer. The elements of relationship matrix (Cholesky factorization) are connected to the neurons in a single hidden (middle) layer via weights (w_{jk}) with a bias (intercept) specific to each neuron. For example, if there are S neurons in the architecture (Figure 1 depicts four neurons), the biases in the hidden layer are $b_1^{(1)}, b_2^{(1)}, \dots, b_S^{(1)}$. A hyperbolic tangent and a linear activation function were applied to the hidden and output layers. The input into neuron j , prior to

activation, is $b_j + \sum_{k=1}^R w_{jk} p_k$. This weighted input is transformed (“activated”) using hyperbolic tangent

activation function $f(.)$ (Figure 1) as $f_j \left(b_j + \sum_{k=1}^R w_{jk} p_k \right)$. This activated emission is then sent to the output layer

and collected as $\sum_{j=1}^S w_j f_j \left(b_j + \sum_{k=1}^R w_{jk} p_k \right) + b$, where w_j ($j = 1, 2, \dots, S$) are weights specific to each neuron and b is another bias parameter. Finally, this is activated again

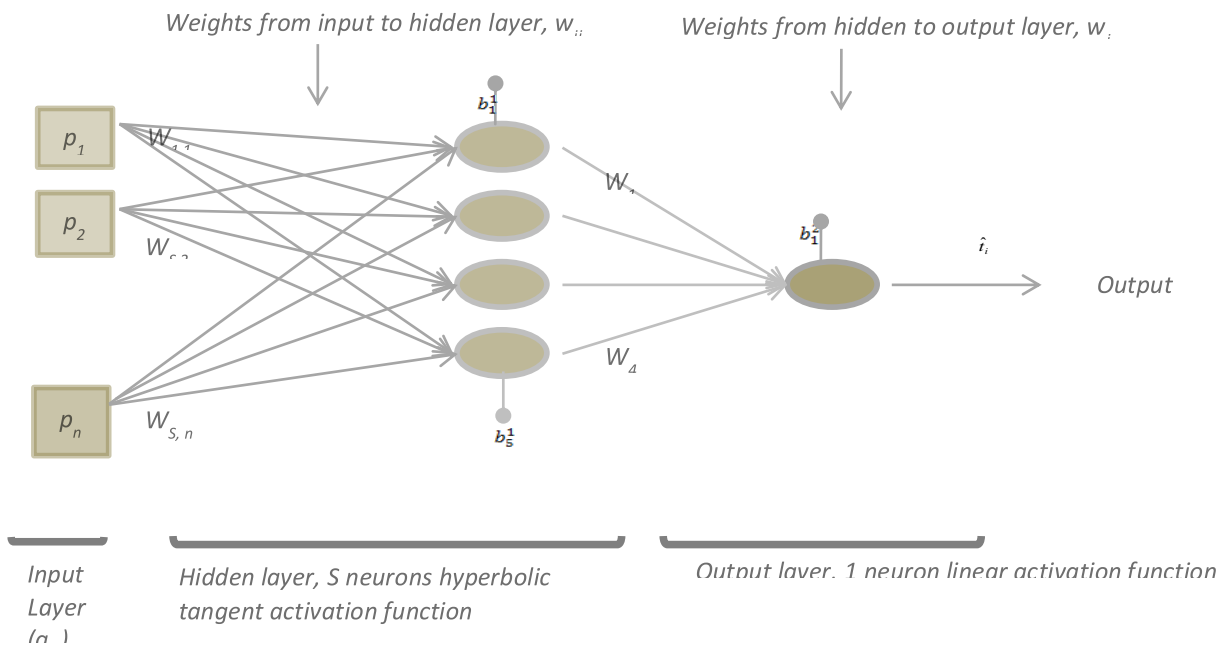


Figure 1. Artificial neural network design used in this study. The elements (a_{kl}) of relationship matrix (\mathbf{A}) were used as inputs (p_i). Each input, p_i is connected to up to S neurons via coefficients w_{ij} (j denotes neuron, i denotes input). Each hidden and output neuron has a bias $b_j^{(k)}$ (j denotes neuron, k denotes layer).

Table 1. The fraction of additive and non-additive genetic variances for the 5 different models considered.

Models	σ_a^2 / σ_G^2	$\sigma_{aa}^2 / \sigma_G^2$	$\sigma_{aaa}^2 / \sigma_G^2$	σ_e^2
1	0	0.5	0.5	1
2	0.1	0.45	0.45	1
3	0.5	0.25	0.25	1
4	0.9	0.05	0.05	1
5	1	0	0	1

σ_G^2 = Total genetic variance, σ_a^2 = additive genetic variance, σ_{aa}^2 = additive*additive genetic variance, σ_{aaa}^2 = additive*additive*additive genetic variance. σ_e^2 = residual variance

with linear function $g(.)$ as $g\left[\sum_{j=1}^s w_j' f_j(.) + b\right]$. Thus, the output is assumed to be a linear combination of output from hidden neurons and a Gaussian error term $N(0, \sigma^2)$. This becomes the predicted value of t_i in the training set. Both input and target variables were normalized prior to network training and BRANN training was implemented according to the Levenberg-Marquardt optimization (Foresee & Hagan 1997). After training, the output (i.e., the predicted value of phenotypes) is calculated as:

$$\hat{t}_i = g\left\{\sum_{j=1}^s w_j' f_j\left(\sum_{k=1}^R w_{jk} p_{ik} + b_j\right) + b\right\}; \quad i = 1, 2, \dots, n \quad (1)$$

Bayesian regularization

ANN conventional training aims to reduce the sum of squared error, $F = E_D$. Pre-Bayesian “training” of neural networks involved finding the network parameters, w , to minimize the error term equivalent in the probabilistic interpretation to maximum likelihood. As with other highly parameterized or ill-posed problems, this led to overfitting (Titterton 2004). In Bayesian ANNs (e.g., BRANNs), the objective function F has an additional quadratic penalty term that penalizes large weights to achieve a smoother mapping. Gradient-based optimization is then used to minimize the following function, which is equal to a penalized natural log-likelihood:

$$F = \beta E_D(D|w, M) + \alpha E_w(w|M), \quad (2)$$

$$E_D(D|w, M) = \sum_{i=1}^n (\hat{t}_i - t_i)^2$$

In equation (2), the and

$E_w(w|M) = \sum_{i=1}^m w_i^2$ are the sum of squares of error and network weights, m is the number of weights, and α and β are positive regularization parameters which need to be estimated. M denotes a specific network architecture which consists of a specification of the number of layers, the number of neurons in each layer, and the type of activation functions used. The second part of equation (2) is called weight decay, which penalizes large weights to achieve a smoother mapping. Therefore, the Bayesian approach involves a probability distribution of network weights, so the predictions from the network can also be casted in a probabilistic framework (Sorich *et al.*

2003). Conditional on the data, given α , β , and M , the posterior distribution of w is:

$$P(w|D, \alpha, \beta, M) = \frac{P(D|w, \beta, M)P(w|\alpha, M)}{P(D|\alpha, \beta, M)}, \quad (3)$$

where D is the training data set. In (3) the

$$P(w|\alpha, M) = \left(\frac{\alpha}{2\pi}\right)^{m/2} \exp\left\{-\frac{\alpha}{2} w'w\right\}$$

is the prior distribution of weights and $P(D|w, \beta, M)$ is the likelihood function, which is the probability of the data given w , while $P(D|\alpha, \beta, M) = \int P(D|w, \beta, M)P(w|\alpha, M)dw$ is a normalization factor, which does not depend on w (Kumar *et al.* 2004). In this Bayesian framework, the optimal weights should maximize the posterior probability $P(w|D, \alpha, \beta, M)$. Maximizing the posterior probability of w is equivalent to minimizing the regularized objective function $F = \beta E_D + \alpha E_w$ (Foresee and Hagan 1997). While minimization of F is identical to finding the (locally) maximum a posteriori estimates w^{MP} , minimization of E_D by back-propagation is identical to finding the maximum likelihood estimates w^{ML} if $n > m$, where m is the number of the parameters (MacKay 2008). Consider the joint posterior density

$$P(\alpha, \beta|D, M) = \frac{P(D|\alpha, \beta, M)P(\alpha, \beta|M)}{P(D|M)} \quad (4)$$

If the prior density $P(\alpha, \beta|M)$ is uniform, maximization of $P(\alpha, \beta|D, M)$ with respect to α is equivalent to the maximization of the likelihood $P(D|\alpha, \beta, M)$ in equation (4). This likelihood is evidence for α and β , which is the normalization factor for equation (3). According to MacKay (1992) we have;

$$P(D|\alpha, \beta, M) = \frac{P(D|w, \beta, M)P(w|\alpha, M)}{P(w|D, \alpha, \beta, M)} = \frac{Z_F(\alpha, \beta)}{(\pi/\beta)^{n/2}(\pi/\alpha)^{m/2}}, \quad (5)$$

where n and m are the number of observation and parameters, respectively. The objective function, $F = \beta E_D(D|w, M) + \alpha E_w(w|M)$ has the shape of a quadratic in a small area surrounding the minimum point of the posterior density w^{MAP} , where the gradient is zero. A Laplacian approximation to $Z_F(\alpha, \beta)$ in equation (5) yields;

$$Z_F(\alpha, \beta) \propto |\mathbf{H}^{MAP}|^{-\frac{1}{2}} \exp(-F(w^{MAP})), \quad (6)$$

where \mathbf{H}^{MAP} is the Hessian matrix of the objective function evaluated at w^{MAP} .

Bayesian optimization of the regularization parameters requires computation of the Hessian matrix of the objective function F evaluated at the optimum point w^{MAP} (Xu *et al.* 2006). However, directly computing the Hessian matrix is not always necessary. As proposed by MacKay (1992), the Gauss-Newton approximation to the Hessian matrix can be used if the Levenberg-Marquardt (LM) optimization algorithm is employed to locate the minimum of F (Shaneh & Butler 2006). The LM algorithm is a robust method for approximation function and LM modification to Gauss-Newton is

$$(J'J + \mu I) \delta = J'e, \quad (7)$$

and the Hessian matrix can be approximated as:

$$\mathbf{H} = \mathbf{J}'\mathbf{J}, \quad (8)$$

where J is the Jacobian matrix that contains first derivatives of the network errors with respect to network parameters (the weights and biases), δ is the parameter update vector and μ is the Levenberg's damping factor. The gradient is computed as $g=J'e$. The μ is adjustable at each iteration and guides to the optimization process. If reductions of the cost function F are rapid, then the parameter μ is divided by a constant (c) to bring the algorithm closer to the Gauss-Newton. On the other hand, if an iteration gives insufficient reduction in F , then μ is multiplied by the same constant giving a step closer to the gradient descent direction. Therefore, the Marquardt-Levenberg

algorithm can be considered a trust-region modification to Gauss-Newton designed to serve as an intermediate optimization algorithm between the Gauss-Newton method and the Gradient-Descent algorithm (Battiti 1992).

If the expression $\gamma = m - 2\alpha^{MAP}tr(H^{MAP})^{-1}$ refers to the effective number of parameters in the neural network, where m is the total number of parameters ($0 \leq \gamma \leq m$), then it can be shown (MacKay 1992; Xu *et al.* 2006) that:

$$\alpha^{MAP} = \frac{\gamma}{2E_w(w^{MAP})} \quad \text{and} \quad \beta^{MAP} = \frac{n - \gamma}{2E_D(w^{MAP})} \quad (9)$$

Analyses

MATLAB (2009) was used for fitting the BRANN. The neural networks considered had two layers (hidden and output) and were fully connected feed-forward networks as shown in Figure 1. To avoid overtraining, improve predictive ability, and eliminate spurious effects caused by the starting values, the BRANNs were trained independently 12 times. Results were recorded as the average of the 12 independent runs. The number of epochs used was 1000. Training was stopped if: 1) the maximum number of epochs was reached; 2) performance had met a suitable level; 3) the gradient was below a suitable target; and 4) the Levenberg-Marquardt μ parameter exceeded a suitable maximum (training stopped when it became larger than 10^{10}). Each of these targets and goals were set at the default values set by the MATLAB implementation.

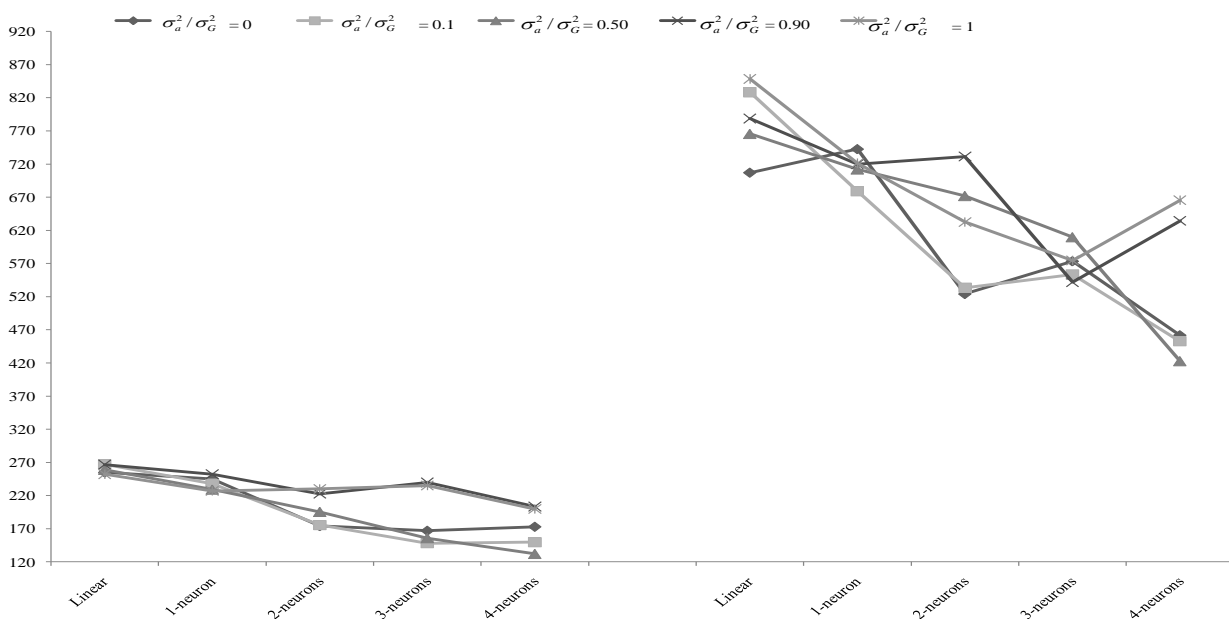


Figure 2. Effective number of parameters (γ) in populations I (at the left side) and population II (at the right side).

Table 2. Effective number parameters and their standard deviations after 12 running for different network architectures for the both populations (results are average of 12 independent running).

	Population I					Population II				
	Effective number of parameters γ					Effective number of parameters γ				
	Linear	1- neuron	2- neuron	3- neuron	4- neuron	Linear	1- neuron	2- neuron	3- neurons	4- neuron
$\sigma_a^2 / \sigma_G^2 = 0$	255.0±4 0	245.0±3 0	174.0±5 9	167.0±5 9	173.0±4 4	707±83	742.5±6 3	524.0±1 63	573.4±165	461.8±1 79
$\sigma_a^2 / \sigma_G^2 = .1$	267.0±3 0	237.8±3 6	175.6±3 6	148.0±4 4	149.7±4 0	828.3±5 0	679.1±6 0	533.5±6 2	553.4±168	452.7±1 56
$\sigma_a^2 / \sigma_G^2 = .5$	259.4±3 1	229.3±3 0	195.4±4 3	155.8±4 1	132.0±4 2	765.5±1 68	711.9±6 6	671.8±1 72	609.7±118	422.7±9 6
$\sigma_a^2 / \sigma_G^2 = .9$	266.6±3 4	252.2±5 0	222.6±3 3	239.7±3 5	203.4±4 4	788.6±1 08	719.8±8 0	731.4±6 4	541.8±140	634.4±1 24
$\sigma_a^2 / \sigma_G^2 = 1$	252.2±2 7	226.9±2 7	230.0±4 9	235.0±3 8	199.7±5 1	848.4±6 6	721.2±7 7	632.6±1 40	574.5±122 .9	665.5±9 5

σ_a^2 =Additive genetic variance, σ_G^2 =Total genetic variance ($\sigma_G^2 = \sigma_a^2 + \sigma_{aa}^2 + \sigma_{aaa}^2$)

Results

Degree of complexity and performance of BRANN

The performance of BRANNs was evaluated in terms of effective number of parameters (γ) and residuals sum of squares in the test data set (SSE_{test}). The estimated effective number of parameters associated with each network evaluated is summarized in Table 2. For both populations, a total of 600 BRANN architectures were examined (50 ANN architectures and 12 replications for each). Our focus was on comparing performances from different linear and nonlinear ANN architectures under practical conditions while varying the proportion of total genetic variance contributed by additive genetic effects.

Except for $\sigma_a^2 / \sigma_G^2 = 0$ in Pop2, the highest effective number of parameters (γ) was obtained from linear ANN architectures (linear activation function in hidden and output layers), indicating explicitly penalizing the complex models by nonlinear BRANN. For example, γ was equal to 255.0±30 and 173.0±44 in BRANNs for linear and nonlinear, respectively, with 4 neurons when the genetic model was assumed to be purely non-additive ($\sigma_a^2 / \sigma_G^2 = 0$). Here, γ reduced drastically (32%) even though the nominal number of parameters (m) increased from 501 to 2509 for the same BRANN architectures (Table 2 and Figure 2). In most cases (except for $\sigma_a^2 / \sigma_G^2 = 0$ and $\sigma_a^2 / \sigma_G^2 = 0.1$ in Pop2) the smallest effective number of parameters attained was obtained either with 3 or 4 neurons in nonlinear (hyperbolic tangent activation function in hidden layer) BRANN architectures, suggesting the penalization ability of ANNs via Bayesian regularization in complex nonlinear models to attain better performance.

The residuals sum of squares (SSE) in test data sets showed the same pattern in both populations across

BRANNs: SSE substantially increased as the proportion of genetic variance accounted for by additive genetic effect (σ_a^2 / σ_G^2) increased. The SSE of linear models for Pop1 and Pop2 ranged from 164.3 to 668.6 and from 525.1 to 1941.6, respectively. The minimum and maximum values of SSE in both populations were obtained for $\sigma_a^2 / \sigma_G^2 \geq 0.1$ and $\sigma_a^2 / \sigma_G^2 = 1$. Except for $\sigma_a^2 / \sigma_G^2 = 0.5$ in Pop2 with two neurons, the SSE of nonlinear models were smaller than those of linear models when the ratio of additive genetic variance increased ($\sigma_a^2 / \sigma_G^2 > 0$). For example, the smallest SSE values in Pop1 and Pop2 for $\sigma_a^2 / \sigma_G^2 = 0.9$ were observed with 4 neuron nonlinear models which were 14 and 16% less than the linear models in the both populations. In general, the smallest SSE as well as γ values in both populations for $\sigma_a^2 / \sigma_G^2 \geq 0.5$ was attained with 3 or 4 neuron architectures, indicating the capability of BRANN to improve performance of complex models via shrinking and reducing the SSE (Table 3).

Predictive ability

The predictive ability (generalization) of the networks was assessed by means of the correlation between predicted and phenotypic observed values in the testing set and the distributions of weights. Such predictive correlations are given in Table 4 and Figure 3.

There were clear and significant improvements in terms of the predictive ability for both linear (equivalent Bayesian ridge regression) and nonlinear models when the proportion of additive genetic variance increased. In

Table 3. Residual sum of squares for BRANNs in testing data set for five models and two populations.

	$\sigma_a^2/\sigma_G^2=0$	$\sigma_a^2/\sigma_G^2=0.1$	$\sigma_a^2/\sigma_G^2=0.5$	$\sigma_a^2/\sigma_G^2=0.9$	$\sigma_a^2/\sigma_G^2=1$
Population I					
Linear	164.3	165.2	271.1	651.8	668.6
Nonlin-1neuron	174.1	150.9	257.7	543.6	620.5
Nonlin-2neurons	165.7	149.4	247.2	515.0	611.0
Nonlin-3neurons	162.6	151.4	246.6	571.5	593.1
Nonlin-4neurons	152.6	157.1	243.1	492.9	576.0
Population II					
Linear	525.1	530.8	782.8	1716.2	1941.6
Nonlin-1neuron	542.8	486.8	758.5	1608.6	1771
Nonlin-2neurons	495.6	502.9	791.9	1546.4	1740.8
Nonlin-3neurons	492.2	486.3	733.4	1461.7	1771
Nonlin-4neurons	471.9	495.4	730.1	1446.7	1770

general, increasing in predictive ability was quite distinct for the nonlinear BRANN, but more markedly so for genetic models with $\sigma_a^2/\sigma_G^2 > 0.5$.

The predictive abilities were smallest when $\sigma_a^2/\sigma_G^2 < 0.5$, intermediately for $\sigma_a^2/\sigma_G^2 = 0.5$ and highest when $\sigma_a^2/\sigma_G^2 > 0.5$ for all BRANNs, suggesting that the high ratio of additive genetic variance to total variance might be a good approximation for analyzing the quantitative traits. For example, improvements $\sigma_a^2/\sigma_G^2 = 1$ vs $\sigma_a^2/\sigma_G^2 = 0$ in Pop2, were 57, 67, 54, 53 and 68 % for linear and nonlinear BRANN architectures of 1, 2, 3 and 4 neurons,

respectively. There was no consistent upward trend on predictive correlations from linear to nonlinear BRANN when $\sigma_a^2/\sigma_G^2 \leq 0.1$. Further, the same findings were observed for $\sigma_a^2/\sigma_G^2 < 0.1$ when the number of neurons in the hidden layer gradually increased in the both populations (Table 4). For example, the predictive correlations for the linear model for the Pop1 and Pop2 when $\sigma_a^2/\sigma_G^2 = 0$ were 0.406 and 0.358, respectively. The highest correlations for both population for $\sigma_a^2/\sigma_G^2 = 0$ with nonlinear BRANN were 0.441 (in 3-neuron BRANN architecture) and 0.407 (in 2-neuron BRANN architecture). Similar patterns were found for $\sigma_a^2/\sigma_G^2 = 0.1$ for both populations. In contrast, the

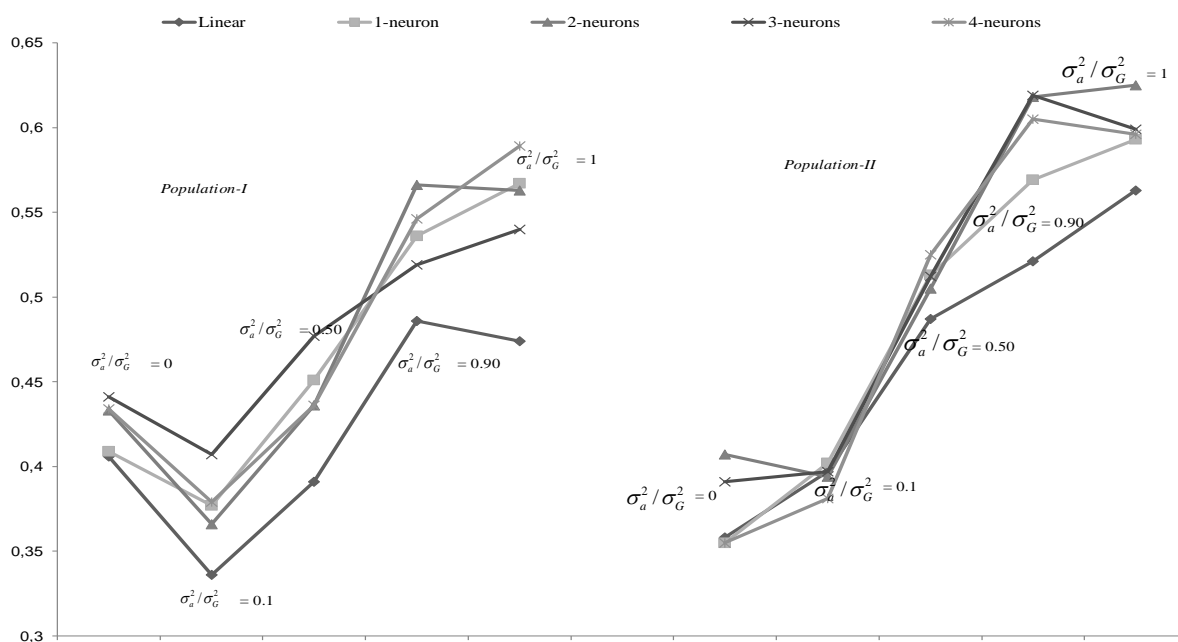


Figure 3. Predictive correlations in testing data set (r_{test}) for different ANN architectures for both populations' data sets for network architectures used in both populations.

Table 4. Predictive correlations of BRANN in testing data set for five models and two populations.

	$\sigma_a^2 / \sigma_G^2 = 0$	$\sigma_a^2 / \sigma_G^2 = 0.1$	$\sigma_a^2 / \sigma_G^2 = 0.5$	$\sigma_a^2 / \sigma_G^2 = 0.9$	$\sigma_a^2 / \sigma_G^2 = 1$
Population 1					
Linear	0.406	0.336	0.391	0.486	0.474
Nonlin-1neuron	0.409	0.377	0.451	0.536	0.567
Nonlin-2neurons	0.433	0.366	0.436	0.566	0.563
Nonlin-3neurons	0.441	0.407	0.477	0.519	0.54
Nonlin-4neurons	0.434	0.379	0.436	0.546	0.589
Population 2					
Linear	0.358	0.397	0.487	0.521	0.563
Nlin-1neuron	0.355	0.402	0.513	0.569	0.593
Nlin-2neurons	0.407	0.394	0.505	0.618	0.625
Nlin-3neurons	0.391	0.397	0.512	0.619	0.599
Nlin-4neurons	0.355	0.381	0.525	0.605	0.596

predictive ability of nonlinear BRANN architectures increased as the proportion of additive genetic variance increased beyond 0.5 ($\sigma_a^2 / \sigma_G^2 \geq 0.5$).

The distribution of weights after training a network also provides some indication of predictive ability; smaller values suggest better generalization while larger weights indicate a more local representation. The weight decay penalty term in the objective function in BRANN causes the weights to converge to smaller absolute values. Figure 4 depicts the distribution of weights in Pop2 for the linear and nonlinear networks with 4-neuron architectures for the $\sigma_a^2 / \sigma_G^2 = 0$, $\sigma_a^2 / \sigma_G^2 = 0.5$ and $\sigma_a^2 / \sigma_G^2 = 1$ scenarios. The average sum of squares of weights when $\sigma_a^2 / \sigma_G^2 = 0$ were 7.05 and 2.8 for linear and nonlinear specifications, respectively;

however, the 7.05 for the linear model was the sum of squares of about 1500 weights whereas the 2.8 for the nonlinear model with 4 neurons was the sum of squares of about 6000 weights (4 x 1500). Similar results were obtained for Pop1 (not depicted here), indicating how strong the shrinkage is (towards 0) when utilizing BRANN. The weights in ANN are the measure the importance of the inputs. The contribution value of an individual input is simply the product of the absolute value of the weights going from a specific input unit to a specific output, summed over the *S* hidden units. Figure 4 shows that the contribution of an input variable to the target variable becomes larger as the proportion of additive genetic variance is increased. Therefore, the majority of weights lie between ± 0.05 and ± 0.1 for $\sigma_a^2 / \sigma_G^2 = 0$ and $\sigma_a^2 / \sigma_G^2 = 1$ with four neurons, indicating the relative importance of input on the output increases as proportion of additive genetic variance increase.

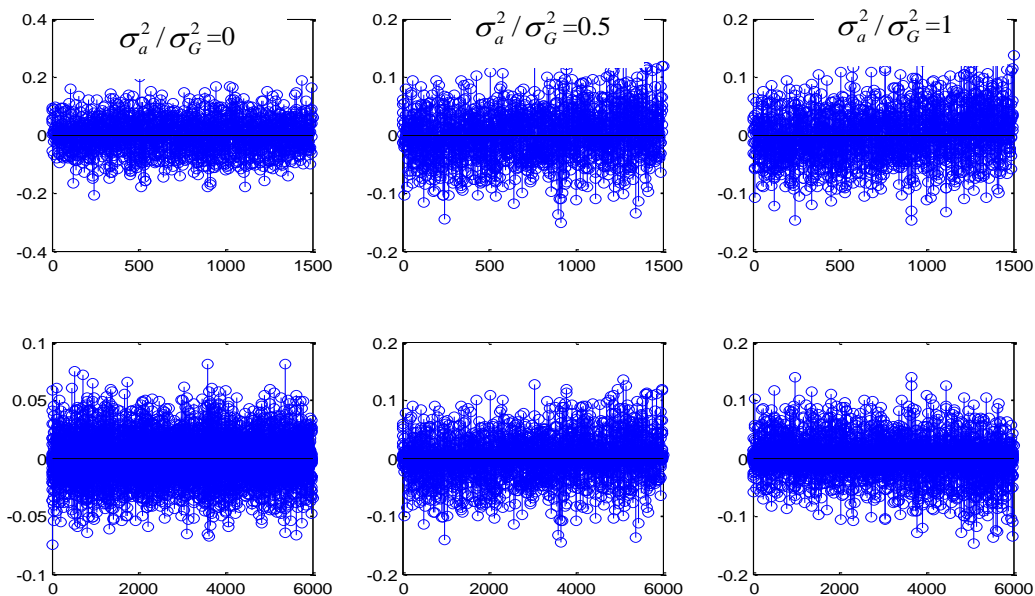


Figure 4. Distributions of weights in Pop2 for $\sigma_a^2 / \sigma_G^2 = 0$, $\sigma_a^2 / \sigma_G^2 = 0.5$ and $\sigma_a^2 / \sigma_G^2 = 1$ for linear (upper row) and nonlinear BRANN with 4 neurons in hidden layer (bottom row).

Discussion

In contrast to the early generation of the quantitative genetics, the non-additive components have a clear mechanistic knowledge in the framework of scientific advancement of the network model and the theories of systems quantitative genetics (Zhu *et al.* 2009). Many research articles (Bagheri & Wagner 2004; Bradshaw *et al.* 2005; Weinreich *et al.* 2005; Le Rouzic *et al.* 2008) have documented that there exists a widening consensus about the evolutionary operability of non-additive genetic components. With this study, the BRANN architectures differing in terms of number of neurons and activation functions were used to make predictions when additive and non-additive effects were jointly involved in fitting a model to a trait. It was shown that the inclusion of non-additive effects did not improve the predictive ability compared to purely

additive models (Table 4). Only the $\sigma_a^2 / \sigma_G^2 = 0.9$ scenario generated similar prediction abilities with all BRANN architectures compared to purely additive models. In contrast, the predictive ability of non-linear models decreased drastically when the proportion of non-additive effects ($\sigma_{aa}^2 / \sigma_G^2$ and $\sigma_{aa}^2 a / \sigma_G^2$) were assumed to be 0.5 or more. These results are in agreement with Allison *et al.* (2009; Calus, 2010) as they reported that partitioning into a complex model for non-additive effects is unnecessary, as these often represent a relatively small proportion of the total genetic variance. Results from our findings and other studies (Allison *et al.* 2009; Hill *et al.* 2008; Wittenburg *et al.* 2011) suggest that even when non-additive genetic effects are present, the total genetic variation may be explicable to a large degree by additive genetic variance (Pirchner 1983).

Results documented here are based on the additive genetic relationship information. Even though the data set used here is from a simulation study, it does provide a practical illustration of the methods presented. Given advances in molecular technology it is now easy to have genome-wide scans with more than one million SNPs for developing additive and non-additive relationship matrices with information from molecular markers. Genomic information can provide a more accurate representation of the relationships between individuals rather than using relationships based on pedigree information only. Gianola *et al.* (2011) reported that the use of genomic relationships led to a more reliable prediction of phenotypes than the use of pedigree information. The relative increase in strength of association, as measured by the correlation, is much larger in those predicted from genomic information than those from pedigrees. The same conclusion has been portrayed by Habier *et al.* (2007).

Traditionally, statistical theory and algorithms have been well developed for linear relationship models. However, quite often the relationship between

variables requires nonlinear methods to allow for successful prediction of properties of interest (Hoffman *et al.* 2008). However, linear and nonlinear parametric statistical approaches have limited flexibility for modeling the high-order, non-linear interactions that may be important in complex traits (Gianola *et al.* 2006; Moore 2010). On the other hand, neural networks have the potential to capture non-linear relationships and may be useful in the study of quantitative traits under complex gene action, given suitable inputs. A nonlinear transformation (the hyperbolic tangent sigmoidal used herein) in $\hat{t}_i = g \left\{ \sum_{j=1}^s w_j' f_j \left(\sum_{k=1}^R w_{jk} p_{ik} + b_j \right) + b_i \right\}$, modifies the connection strength between additive relationship (Cholesky factor decomposition) and phenotype in an adaptive manner underlying the potential for an improvement in predictive ability (Okut *et al.* 2013). Our results revealed that the sigmoidal-type activation function used in the hidden layer in BRANN models outperformed the linear models. The predictive ability of BRANN architectures with nonlinear activation function were substantially larger than the linear models for the scenarios considered, except for $\sigma_a^2 / \sigma_G^2 = 0.1$ in the Pop1. In a recent study, it was shown that non-linear neural networks outperformed a benchmark linear model when predicting phenotypes, especially in inbred wheat lines where cryptic gene actions and interactions are expected (Gianola *et al.* 2011).

The Levenberg-Marquardt algorithm was adopted to optimize weights and biases because previous evaluations with networks used a smaller number of weights to indicate that it was a suitable method (Demuth *et al.* 2009). However, the Levenberg-Marquardt is sensitive to initial values of weights as well as outliers in the data. These may lead to a overfitting problem in ANN. In the training process, overfitting often occurred, leading to loss of generalization of the predictive model. Bayesian regularization proposed by MacKay (1992) was used to avoid over-fitting and improve generalization. Adding Bayesian regularization to the Levenberg-Marquardt enables it to overcome the problem invoked in interpolating noisy data and overfitting. Since evidence provide an objective Bayesian criterion for stopping training, they are difficult to overtrain (Winkler & Burden 2004). The problem of overfitting and overtraining are also dealt with by this method so that the production of a definitive and reproducible model is attained.

Because highly parameterized models are penalized in the Bayesian approach, we were able to explore complex BRANN architectures. The complexity of a network is related to both the number of weights and the size of the weights. A model selection criterion related to complex BRANN is concerned with the number of weights. The more weights there are, relative to the number of training cases, the more overfitting amplifies noise in the target variables. For the networks trained with Bayesian regularization, we examined how

the effective number of parameters γ varied with architecture. As shown in Table 2, even though the fold number of parameters varied from about 501 to 2009 in Pop1 and from 1501 to 6009 in Pop2, the effective number of parameters varied only from 199.7 to 252

and from 665 to 848, for $\sigma_a^2 / \sigma_G^2 = 1$ on average, indicating the effect of regularization was effective. The weight decay penalty term in objective function, $F = \beta E_D + \alpha E_W$ in BRANN causes the weights to converge to smaller absolute values than what they otherwise would. Thus, the effective number of parameters used in the models is less than the number of weights, as some weights do not contribute to the models. In the linear model this form of weight is equivalent to the ridge regression.

Conclusions

The Bayesian learning with a nonlinear model outperformed the linear model when the entire or considerable part of the total genetic variance was due to additive genetic effects. This was evident in our study when 50% or more of the total genetic variance was explained by additive genetic effects. BRANN architecture with a nonlinear activation function and linear models had similar predictive ability when the entire ($\sigma_a^2 / \sigma_G^2 = 1$) or a considerable part ($\sigma_a^2 / \sigma_G^2 = 0.9$) of the total genetic variance was due to additive genetic effects. ANNs are a promising tool to handle complex data situations. Bayesian regularization ANN allowed estimation of all connection strengths even when $n \ll p$, and the effective number of parameters was much smaller than the corresponding nominal number. The optimal values of posterior distribution of the connection strengths in BRANN can be automatically obtained through Bayesian regularized methods. This facilitates the selection of regularization parameters, possesses good robustness and excellent fitting. These advantages might even be more pronounced if further research can be done concerning the interpretation of their parameters.

In summary a feed forward ANN with BR was used to assess the performance and predictive ability of different nonlinear ANNs and linear models for complex traits with genetic architectures. It was shown that the inclusion of non-additive effects did not improve the predictive ability compared to purely additive models.

Author Contributions

HO conceived and performed computations, coordinate the study and drafted the manuscript; DG conceived, coordinate the study and provided critical insights and revised the manuscript; GJMR conceived, carried out the study, advised for computations, provided critical insights and revised the manuscript, KAW helped conceive and revised the manuscript.

References

- Allison, M. K., Cullus, C. B., Gilmour, A. R., Eccleston, J. A. and Thompson, R. (2009) Estimation in a multiplicative mixed model involving a genetic relationship matrix. *Genetics Selection Evolution* 41:33.
- Bagheri, H. C., Wagner, G. P. (2004) Evolution of dominance in metabolic pathways. *Genetics*, 168, 1713-35.
- Battiti, R. (1992) First- and second order methods for learning: Between steepest descent and Newton's method. *Neural Computation*, 4, 2, pp. 141-166.
- Bradshaw, W. E., Haggerty, B. P., Holzapfel, C. M. (2005) Epistasis underlying a fitness trait within a natural population of the pitcher-plant mosquito, *Genetics*, 169, 485-8.
- Calus, M. P. L. (2010) Genomic breeding value prediction: methods and procedures. *Animal*, 4, 157-164.
- Carlborg, O., Haley, C. S. (2004) Epistasis: too often neglected in complex trait studies? *Nat Rev Genet* 5, 618-625.
- Cheverud, J., Routman, J. (1995) Epistasis and its contribution to genetic variance components. *Genetics*, 139, 1455-61.
- Demuth, H., Beale, M., Hagan, M. (2009) *Neural Network Toolbox™ 6 User's Guide*. The MathWorks, Inc., Natick, MA, USA.
- Foresee, F. D., Hagan, M. T. (1997) Gauss-Newton approximation to Bayesian learning, In *Proc. IEEE Int. Conf. Neural Networks*. Edited by Martin T. Hagan, 1930-1935.
- Gencay, R., Qi, M (2001) Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging. *IEEE Trans. Neural Networks*, 12, 726-734.
- Gianola, D., de los Campos, G. (2008) Inferring genetic values for quantitative traits non-parametrically. *Genetics Research*, 90(6), 525-540.
- Gianola, D., Okut, H., Weigel, K. A., Rosa, G. J. M. (2011). Predicting complex quantitative traits with Bayesian neural networks: a case study with Jersey cows and wheat. *BMC genetics*, 12-87 <http://www.biomedcentral.com/1471-2156/12/87>
- Habier, D., Fernando, R. L., Dekkers, J. C. M. (2007) The impact of genetic relationship information on genome-assisted breeding values. *Genetics*, 177(4), 2389-2397.
- Hallander, J., Waldmann, P. (2007) The effect of non-additive genetic interactions on selection in multi-locus genetic models. *Heredity*, 98, 349-359.
- Hill W.G., Goddard M.E., Visscher P.M. (2008) Data and theory point to mainly additive genetic variance for complex traits. *PloS Genet.*, 4, e1000008.
- Hofmann, T., Scholkopf, B., Smola, J. A. (2008) Kernel Methods In Machine Learning. *The Annals of Statistics*, 36:3, 1171-1220.
- Ingileif, B. H., Yuster, S. D. (2008) A complete classification of epistatic two-locus models. *BMC Genetics*, 9:17.
- Jamrozik, J., Fatehi, J., Kistemaker, G. J., Schaeffer, L.R. (2005) Estimates of genetic parameters for Canadian Holstein female reproduction traits. *J. Dairy Sci.* 88, 2199-2208.
- Alvarez-Castro, J. M., Orjan, C. A. (2007) Unified Model for Functional and Statistical Epistasis and Its Application

- in Quantitative Trait Loci Analysis. *Genetics*, 176, 1151–1167.
- Kumar, P., Merchant, S. N., Desai, U. B. (2004) Improving performance in pulse radar detection using Bayesian regularization for neural network training. *Digital Signal Processing*, 14,438–448.
- Lampinen, J., Vehtari, A. (2001) Bayesian approach for neural networks review and case studies, *Neural Networks*, 14, 257-274.
- Le Rouzic, A., Alvarez-Castro, J. M., Carlborg, O. (2008) Dissection of the genetic architecture of body weight in chicken reveals the impact of epistasis on domestication traits. *Genetics*, 179, 1591-9.
- MacKay, D. J. C. (1992) Bayesian interpolation. *Neural Computation*, 4, 415–447.
- MacKay, J. C. D. (2008) *Information theory, inference and learning algorithms*. Cambridge University Press, Cambridge, UK.
- Marwala, T. (2007) Bayesian training of neural networks using genetic programming. *Pattern Recognition Letters*, 28,1452–1458.
- Nagy, I., Gorjanc, G., Curik, I., Farkas, J., Kiszlinger, H., Szendro, Z. (2013) The contribution of dominance and inbreeding depression in estimating variance components for litter size in Pannon White rabbits. *J. Anim. Breed. Genet.*, 130, 303–311.
- Okut, H., Gianola, D., Rosa, G. J. M., Weigel, K. A. (2011) Prediction of body mass index in mice using dense molecular markers and a regularized neural network. *Genet Res.* 93(3):189–201. Available from: <http://dx.doi.org/10.1017/S0016672310000662>
- Okut H. (2016) Artificial Neural Networks Model and Application. Joao Juis G. Rosa (Eds), Bayesian Regularized Neural Networks for Small n Big p Data (pp 27-48). London, UK. IntechOpen.
- Okut H. (2021) Deep Learning and Application, Pier Luigi Mazzeo and Paolo Spagnolo, (Eds), Deep Learning for Subtyping and Prediction of Diseases: Long-Short Term Memory (pp 27-48). London, UK. IntechOpen. DOI: 10.5772/intechopen.96180.
- Oakey, H., Verbyla, A. P., Pitchford, W., Cullis, B. R., Kuchel, H. (2006) Joint modeling of additive and non-additive genetic line effects in single field trials. *Theor Appl Genet*, 113, 809-819.
- Pirchner, F. (1983) *Population genetics in animal breeding*. Second Edition. Plenum Press, New York, USA.
- Ripley, B. D. (2007) *Pattern recognition and neural networks*. Cambridge University Press, Cambridge, UK.
- Sorich, M. J., Miners, J. O., Ross, A. M., Winker, D. A., Burden, F. R., Smith, P. A. (2003) Comparison of Linear and Nonlinear Classification Algorithms for the Prediction of Drug and Chemical Metabolism by Human UDP-Glucuronosyltransferase Isoforms. *J. Chem. Inf. Comput. Sci.*, 43, 2019-2024.
- Titterton, D. M. (2004) Bayesian methods for neural networks and related models. *Statistical Science*, 19, 128–139.
- Valentina, P., Lawrence, R. S., Filippo M, Vern, O. (2007) Non-additive genetic effects for fertility traits in Canadian Holstein cattle. *Genet. Sel. Evol.*, 39, 181–193.
- Wade, M. (2002) A gene's eye view of epistasis, selection and speciation. *J Evol Biol*, 15, 337-346.
- Weinreich, D. M., Watson, R. A., Chao, L. (2005) Perspective: sign epistasis and genetic constraint on evolutionary trajectories. *Evolution*, 59, 1165-74.
- Wittenburg, D., Melzer, N., Reinsch, N. (2011) Including non-additive genetic effects in Bayesian methods for the prediction of genetic values based on genome-wide markers. *BMC Genetic*, 12:74.
- Xu, M., Zengi, G., Xu, X., Huang, G., Jiang, R., Sun, W. (2006) Application of Bayesian regularized BP neural network model for trend analysis, acidity and chemical composition of precipitation in North. *Water, Air, and Soil Pollution*, 172, 167–184.
- Zhu, M., Yu, M., Zhao, S. (2009) Understanding Quantitative Genetics in the Systems Biology Era. *Int. J. Biol. Sci.*, 5(2), 161-170.