A fast algorithm to factorize high-dimensional Tensor Product matrices used in Genetic Models

Marco Lopez-Cruz[1*], Paulino Pérez-Rodríguez[2] & Gustavo de los Campos[1,3,4]

[1] Department of Epidemiology and Biostatistics, Michigan State University, East Lansing, MI 48824, USA

[2] Socioeconomía, Estadística e Informática, Colegio de Postgraduados, Edo. de México 56230, Montecillos, México.

[3] Department of Statistics and Probability, Michigan State University, East Lansing, MI 48824, USA

[4] Institute for Quantitative Health Science and Engineering, Michigan State University, East Lansing, MI 48824, USA

* Corresponding author: MLC (lopezcru@msu.edu)

Running head: A fast algorithm to factorize large Tensor Product matrices used in Genetic Models

Keywords: R package, genetic model, covariance matrix, eigenvalue decomposition

Abstract

Many genetic models (including models for epistatic effects as well as genetic-by-environment) involve covariance structures that are Hadamard products of lower rank matrices. Implementing these models require factorizing large Hadamard product matrices. The available algorithms for factorization do not scale well for big data, making the use of some of these models not feasible with large sample sizes. Here, based on properties of Hadamard products and (related) Kronecker products we propose an algorithm that produces an approximate decomposition that is orders of magnitude faster than the standard eigenvalue decomposition. In this article, we describe the algorithm, show how it can be used to factorize large Hadamard product matrices, present benchmarks, and illustrate the use of the method by presenting an analysis of data from the northern testing locations of the G×E project from the Genomes-to-Fields Initiative (n~60,000). We implemented the proposed algorithm in the open-source 'tensorEVD' R-package.

Introduction

Hadamard products of positive definite matrices appear in many genetic models including gene-by-gene (e.g., additive-by-additive or additive-by-dominance, Henderson 1985) and gene-by-environment interactions (Crossa *et al.* 2006) as well as in hybrid prediction models (Bernardo 1998). In this article, we focus on high-dimensional Hadamard products derived from two positive semi-definite matrices, each with a dimension considerably smaller than the resulting Hadamard product.

To motivate this problem, consider a reaction norm infinitesimal model (Falconer and Mackay 1996) for n_G genotypes tested over n_E locations (environments). Following Jarquín *et al.* (2014), interactions between genetic and environmental factors can be modeled using a Gaussian random effect with a covariance matrix $\mathbf{K}$ which is the product of a genetic ($\mathbf{K}_G$, derived from DNA or pedigree data) and an environmental ($\mathbf{K}_E$, typically derived from environmental covariates) relationship matrix. If all genotypes are tested in all environments, $\mathbf{K}$ is a Kronecker product $\mathbf{K} = \mathbf{K}_G \otimes \mathbf{K}_E$ of dimension $n = n_G \times n_E$. However, usually, not all genotypes are tested in all environments and genotypes may be replicated. In these cases, the $\mathbf{K}$ matrix takes the form $\mathbf{K} = (\mathbf{Z}_1\mathbf{K}_G\mathbf{Z}_1') \circ (\mathbf{Z}_2\mathbf{K}_E\mathbf{Z}_2')$ where $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are incidence matrices connecting phenotypes with the rows (and columns) of $\mathbf{K}_G$ and $\mathbf{K}_E$, respectively, and '$\circ$' denotes the Hadamard product. A very similar problem arises when modelling hybrids' effects where

$\mathbf{K}_G$ and $\mathbf{K}_E$ are replaced by additive relationship matrices between the female and male parental lines (Bernardo 1998).

Fitting Gaussian models with dense covariance structures such as the one presented above requires factorizing $\mathbf{K}$ using, for example, the eigenvalue decomposition (EVD) of $\mathbf{K}$. The EVD has an $O(n^3)$ computational complexity; therefore, a standard decomposition of $\mathbf{K}$ does not scale well to large sample sizes. To tackle this problem, we use results about the EVD of Kronecker products, and the fact that Hadamard products are sub-matrices of Kronecker products, to propose an algorithm that derives a basis for $\mathbf{K}$ which only requires factorizing $\mathbf{K}_G$ and $\mathbf{K}_E$ matrices which usually are much smaller than $\mathbf{K}$. We show that the proposed approach provides a very good approximation to the target matrix ($\mathbf{K}$) and that, in large-*n* problems, the proposed approach can be orders of magnitude faster than performing EVD on $\mathbf{K}$ directly. Finally, we provide real data analyses showing that the proposed approach yields very close variance components estimates and almost an identical prediction accuracy in cross-validation that an exact EVD. The methods described in this article are implemented in the open-source 'tensorEVD' R-package which is available through CRAN and the GitHub repository.

Methods

Recall the eigenvalue decomposition (EVD) of an $N \times N$ positive semi-definite matrix $\mathbf{K}$ which has the form

$$\mathbf{K} = \mathbf{V}\mathbf{D}\mathbf{V}'$$

where $\mathbf{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N]$ is an orthonormal matrix (i.e., $\mathbf{V}'\mathbf{V} = \mathbf{I}$) whose columns $\boldsymbol{v}_k$ ($k = 1, \ldots, N$) are the eigenvectors and $\mathbf{D} = diag(d_1, \ldots, d_N)$ is a diagonal matrix with the eigenvalues $d_1 \geq \cdots \geq d_N \geq 0$.

Consider the Kronecker product ('$\otimes$') of two symmetric positive semi-definite matrices, $\mathbf{K}_1$ and $\mathbf{K}_2$,

$$\mathbf{K} = \mathbf{K}_1 \otimes \mathbf{K}_2. \quad (1)$$

Let the EVD of the two matrices in the right-hand side be $\mathbf{K}_1 = \mathbf{V}_1\mathbf{D}_1\mathbf{V}_1'$ and $\mathbf{K}_2 = \mathbf{V}_2\mathbf{D}_2\mathbf{V}_2'$, respectively. Replacing these matrices with their EVD we get:

$$\mathbf{K} = (\mathbf{V}_1\mathbf{D}_1\mathbf{V}_1') \otimes (\mathbf{V}_2\mathbf{D}_2\mathbf{V}_2').$$

Using properties of Kronecker products (e.g., Searle 1982, p. 265), it can be shown that the eigenvectors of $\mathbf{K}$ are Kronecker products of the eigenvectors of $\mathbf{K}_1$ and $\mathbf{K}_2$. Likewise, the eigenvalues

of $\mathbf{K}$ are Kronecker products of the eigenvalues of $\mathbf{K}_1$ and $\mathbf{K}_2$ (see Supplementary Note 1 for a proof). Specifically, we have that (a numerical example of the above results is presented in Supplementary Note 2):

$$\mathbf{K} = \mathbf{VDV}' = (\mathbf{V}_1 \otimes \mathbf{V}_2)(\mathbf{D}_1 \otimes \mathbf{D}_2)(\mathbf{V}_1 \otimes \mathbf{V}_2)'.$$

A Hadamard product ('$\circ$') of two matrices is a sub-matrix of the corresponding Kronecker product. For example, an $n \times n$ matrix:

$$\mathbf{K}_0 = (\mathbf{Z}_1\mathbf{K}_1\mathbf{Z}_1') \circ (\mathbf{Z}_2\mathbf{K}_2\mathbf{Z}_2'), \quad (2)$$

is a sub-matrix of $\mathbf{K}_1 \otimes \mathbf{K}_2$ in Equation (1). Therefore, the linear space spanned by $(\mathbf{Z}_1\mathbf{K}_1\mathbf{Z}_1') \circ (\mathbf{Z}_2\mathbf{K}_2\mathbf{Z}_2')$ in Equation (2) is a sub-space of the linear space spanned by $\mathbf{K}_1 \otimes \mathbf{K}_2$. This suggests that we can find a basis for a Hadamard product from the EVD of the corresponding Kronecker product. The Tensor EVD algorithm is inspired by this idea.

Tensor EVD algorithm

We assume that the input data consist of the following:

- ***Covariance structures***: $\mathbf{K}_1$ and $\mathbf{K}_2$ of dimensions $n_1 \times n_1$ and $n_2 \times n_2$, respectively. For example, $\mathbf{K}_1$ may be a genomic relationship matrix and $\mathbf{K}_2$ may be an environmental relationship matrix describing environmental similarity between testing environments.
- ***IDs***: $\mathbf{ID}_1$ and $\mathbf{ID}_2$ are n-vectors (n here is the sample size) mapping from observations to the rows and columns of $\mathbf{K}_1$ and $\mathbf{K}_2$, respectively. (The row- and column-names of $\mathbf{K}_1$ and $\mathbf{K}_2$ are the unique entries of $\mathbf{ID}_1$ and $\mathbf{ID}_2$, respectively.) These IDs are used to form the incidence matrices $\mathbf{Z}_1$ and $\mathbf{Z}_2$ in Equation (2). For instance, the matrix $\mathbf{Z}_1\mathbf{K}_1\mathbf{Z}_1'$ can be obtained by indexing rows and columns of $\mathbf{K}_1$ by $\mathbf{ID}_1$, in R's (R Core Team 2021) notation this is $\mathbf{K}_1[\mathbf{ID}_1, \mathbf{ID}_1]$.

Using the above-described inputs, our algorithm (which we named *tensorEVD*) proceeds as follows:

1. Perform the EVD of $\mathbf{K}_1 = \mathbf{V}_1\mathbf{D}_1\mathbf{V}_1'$ and $\mathbf{K}_2 = \mathbf{V}_2\mathbf{D}_2\mathbf{V}_2'$.
2. Derive the $N = n_1 \times n_2$ eigenvalues of the Kronecker product as $\widetilde{\mathbf{D}} = diag(\tilde{d}_1, \ldots, \tilde{d}_N) = \mathbf{D}_1 \otimes \mathbf{D}_2$.
3. Derive the N eigenvectors $\widetilde{\mathbf{V}} = [\widetilde{\boldsymbol{v}}_1, \ldots, \widetilde{\boldsymbol{v}}_N]$ of the Kronecker product. Each column $\widetilde{\boldsymbol{v}}_k$ ($k = 1, \ldots, N$) is the Hadamard product of the i_k^{th} and j_k^{th} eigenvectors of $\mathbf{V}_1$ and $\mathbf{V}_2$, respectively, that is

$\tilde{\boldsymbol{v}}_k = \left(\mathbf{Z}_1 \boldsymbol{v}_{1i_k}\right) \circ \left(\mathbf{Z}_2 \boldsymbol{v}_{2j_k}\right)$. As before, the terms $\mathbf{Z}_1 \boldsymbol{v}_{1i_k}$ and $\mathbf{Z}_2 \boldsymbol{v}_{2i_k}$ are obtained using indexing, i.e., $\boldsymbol{v}_{1i_k}[\mathbf{ID}_1]$ and $\boldsymbol{v}_{2j_k}[\mathbf{ID}_2]$.

4. For unbalanced or replicated data, the eigenvectors in $\tilde{\mathbf{V}}$ may not have a norm equal to one; thus, the sum of the eigenvalues $\tilde{d}_k$ will no longer be equal to $trace(\mathbf{K})$. Therefore, we normalize each eigenvector $\tilde{\boldsymbol{v}}_k$ to have unit norm.
5. Order the eigenvalues $\tilde{d}_k$ and eigenvectors $\tilde{\boldsymbol{v}}_k$ according to $\tilde{d}_k$.

The *tensorEVD* algorithm described above renders orthonormal vectors only for the balanced case (i.e., for the Kronecker product of $\mathbf{K}_1$ and $\mathbf{K}_2$). For unbalanced cases the eigenvectors are not guaranteed to be mutually orthogonal; however, they provide a basis for the Kronecker product. Therefore, the eigenvectors are also a basis for Hadamard products which spans a sub-space of the corresponding Kronecker product.

Note that the *tensorEVD* algorithm produces the complete basis containing $N = n_1 \times n_2$ eigenvectors for the Kronecker matrix product $\mathbf{K}_1 \otimes \mathbf{K}_2$. As consequence, this basis can include more vectors than the ones needed to span $(\mathbf{Z}_1 \mathbf{K}_1 \mathbf{Z}_1') \circ (\mathbf{Z}_2 \mathbf{K}_2 \mathbf{Z}_2')$. This can be particularly relevant if the size of the Hadamard product is considerably smaller than the corresponding Kronecker product. Furthermore, most of those vectors will have a very small eigenvalue (resulting from the product of a small eigenvalue of $\mathbf{K}_1$ and a small eigenvalue from $\mathbf{K}_2$). Therefore, instead of forming all possible eigenvectors, we allow for the user to specify a proportion of variance explained ($0 < \alpha \leq 1$, e.g., $\alpha = 0.95$) and build only the eigenvectors needed to achieve such proportion of variance.

The 'tensorEVD' R-package can be installed from CRAN using the following instruction:

```
install.packages('tensorEVD')
```

Alternatively, it can be installed from the GitHub platform via, for instance, the 'remotes' R-package (Csárdi *et al*. 2023) using the instructions below:

```
install.packages('remotes')
library(remotes)
install_github('MarcooLopez/tensorEVD')  # Install tensorEVD
```

The following script shows how to perform EVD using the *tensorEVD* function (see Supplementary Note 3 for an actual numerical example).

```
EVD = tensorEVD(K1, K2, ID1, ID2, alpha = 0.95)
ncol(EVD$vectors)          # Number of eigenvectors
sum(EVD$values)/EVD$totalVar  # Variance explained
```

Results and Discussion

We benchmarked the *tensorEVD* routine against the *eigen* function of the 'base' R-package (R Core Team 2021) in terms of the computational time used to derive eigenvectors, the accuracy of the approximation provided by *tensorEVD*, and the dimension of the resulting basis. All the analyses were performed in R v4.2.0 (R Core Team 2021) run on the High-Performance Computing Center (HPCC) from Michigan State University (https://icer.msu.edu/hpcc/hardware) using nodes equipped with Intel Xeon Gold 6148 CPUs at 2.40 GHz with 84 GB of RAM memory in a single computing thread.

The data used in these benchmarks was generated by the Genomes-To-Fields (G2F) Initiative (Lima *et al.* 2023) which was curated and expanded by adding environmental covariates by Lopez-Cruz *et al.* (2023). This data set was used to derive a genetic (GRM) and an environmental relationship matrix (ERM, from the environmental covariates, see Supplementary Note 4) for 4,344 maize hybrids and 97 environments (year-locations), respectively, corresponding to the northern testing locations. We formed Hadamard products ($\mathbf{K}$ in Equation (2)) between the GRM (as $\mathbf{K}_1$) and the ERM (as $\mathbf{K}_2$) matrix of various sizes by sampling hybrids ($n_G = 100, 500,$ and 1000), environments ($n_E = 10, 30,$ and 50), and the level of replication needed to complete a total sample size ranging from n =10,000 to 30,000. Then, we factorized the resulting Hadamard product matrix using the R-base function *eigen* (R Core Team 2021) as well as using *tensorEVD*, deriving as many eigenvectors as needed to explain 90%, 95%, and 98% of the total variance.

The *tensorEVD* method was consistently orders of magnitude faster than *eigen* (see Supplementary Figures 1-3). The difference in computation time is particularly clear (e.g., *tensorEVD* ~10,000 faster than *eigen*) when the product of the dimensions of each of the relationship matrices ($n_G \times n_E$) was smaller than sample size (n)–compare the left, middle, and right columns of Figure 1.

The Cholesky decomposition is an alternative factorization that can be used to implement the models discussed in this study. This factorization has a computational complexity of $O\left(\frac{1}{3}n^3\right)$ which is smaller than the complexity of the EVD ($O(n^3)$). However, the Cholesky decomposition can be numerically

unstable for matrices that are (near) singular, a situation that is not uncommon in G×E models. Another approach would be to use partial EVD methods that compute only a fraction of the eigenvectors. To evaluate these approaches we benchmarked *tensorEVD* against Cholesky decomposition as per the *chol* function from the 'base' R-package (R Core Team 2021), and against partial eigenvalue decompositions computed using the *trlan.eigen* and *eigs_sym* functions from the 'svd' (Korobeynikov *et al.* 2023) and 'RSpectra' (Qiu and Mei 2022) R-packages, respectively. As expected, partial SVD methods were faster than *eigen* only when the product $n_G \times n_E$ was smaller than n (see Supplementary Figures 1-3); however, *tensorEVD* was much faster than the partial EVD methods (Supplementary Figure 4). Likewise, *tensorEVD* was faster than *chol* only in cases where $n_G \times n_E < n$.

Approximation Accuracy

We measured the accuracy of the approximation of the basis derived by the *eigen* and *tensorEVD* routines for each of the α-values by evaluating the Frobenius norm (i.e., a matrix-generalization of the Euclidean norm, Golub and Van Loan 1996) of the difference between the Hadamard product matrix ($\mathbf{K}$) and the approximation ($\widehat{\mathbf{K}}_\alpha = \widetilde{\mathbf{V}}_\alpha \widetilde{\mathbf{D}}_\alpha \widetilde{\mathbf{V}}'_\alpha$, where $\widetilde{\mathbf{V}}_\alpha$ and $\widetilde{\mathbf{D}}_\alpha$ are the eigenvectors and eigenvalues derived by each method and α-value, see Supplementary Note 5 for more details). In general, both methods provided a very good and similar approximations (Figure 2). As expected, the values of the norm decrease when α increased (smaller norm indicates better approximation). The values of the norm for different sample sizes cannot be compared because the Frobenius norm is a cumulative sum of $n \times n$ elements. Therefore, we also computed the Correlation Matrix Distance (CMD, Herdin *et al.* 2005) between the Hadamard product matrix ($\mathbf{K}$) and the approximation provided by each method and α-value ($\widehat{\mathbf{K}}_\alpha$, see Supplementary Note 5, Supplementary Figure 5). These CMD values are always between 0 and 1. In all the cases, the CMD was very small (<0.006), which indicates that both approximations were very good. As with the Frobenius norm metric, the CMD shows that both methods provide similar approximations (Supplementary Figure 5); however, there is evidence that whenever $n_G \times n_E$ becomes larger than sample size n, *tensorEVD* provides a slightly better approximation than the *eigen* method (e.g., bottom-right panel in Figure 2 and Supplementary Figure 5, see Supplementary Figure 6).

Dimension reduction

We also compared *eigen* and *tensorEVD* in terms of the number of eigenvectors provided by each method for a given α-value, relative to the rank of the $\mathbf{K}$ (i.e., the number of eigenvectors of $\mathbf{K}$ with positive eigenvalue). By construction, *eigen* is very efficient at maximizing the proportion of variance

explained in the derivation of eigenvectors. The *tensorEVD* function is as effective as the *eigen* method at dimension reduction only for cases where $n_G \times n_E < n$, for example the case $n_G = 100$ and $n_E = 10$ (top-left panel in Figure 3). However, *tensorEVD* becomes less effective at dimension reduction when $n_G \times n_E$ exceeds sample size n (e.g., bottom-right panel in Figure 3, see Supplementary Figure 7).

Application in Genomic Prediction

Finally, we evaluated the performance of the approximation of **K** provided by the *tensorEVD* method in Gaussian linear models in terms of variance components estimates and cross-validation prediction accuracies. For this evaluation, we used all the G2F data from the northern testing locations included in the data set presented by Lopez-Cruz *et al.* (2023). For the northern testing locations, this data set includes $n = 59{,}069$ records for 4 traits (grain yield, anthesis, silking, and anthesis-silking interval) from $n_G = 4{,}344$ hybrids and $n_E = 97$ environments.

We analyzed this data with a Gaussian reaction norm model (Jarquín *et al.* 2014) in which the response (y_{ijk}) is modeled as the sum of the main effect of hybrid (G_i), main effect of environment (E_j), and the interaction hybrid×environment (GE_{ij}) term, this is

$$y_{ijk} = \mu + G_i + E_j + GE_{ij} + \varepsilon_{ijk}. \quad (3)$$

Above, μ is an intercept and i, j, and k are indices for the hybrids, environment, and replicate, respectively. The term ε_{ijk} is an error term assumed to be Gaussian distributed as $\varepsilon_{ijk} \overset{iid}{\sim} N(0, \sigma_\varepsilon^2)$, with σ_ε^2 variance parameter associated to the error. Hybrid, environment, and interaction effects were assumed to be multivariate normally distributed with zero mean and effect-specific covariance matrices, specifically $\mathbf{G} \sim MVN(\mathbf{0}, \sigma_G^2 \mathbf{K}_G)$, $\mathbf{E} \sim MVN(\mathbf{0}, \sigma_E^2 \mathbf{K}_E)$, and $\mathbf{GE} \sim MVN(\mathbf{0}, \sigma_{GE}^2 \mathbf{K})$, where $\mathbf{K}$ takes the Hadamard form in Equation (2), $\mathbf{K} = (\mathbf{Z}_1 \mathbf{K}_G \mathbf{Z}_1') \circ (\mathbf{Z}_2 \mathbf{K}_E \mathbf{Z}_2')$ and σ_G^2, σ_E^2, and σ_{GE}^2 are variance parameters associated to $\mathbf{G}$, $\mathbf{E}$ and $\mathbf{GE}$, respectively. We fitted the model in Equation (3) to each trait in a Bayesian fashion using the 'BGLR' R-package (Pérez-Rodríguez and de los Campos 2022) with the decomposition of the GE kernel ($\mathbf{K}$) computed using *eigen* and *tensorEVD* methods for different percentages of variance of $\mathbf{K}$ explained ($\alpha = 0.90, 0.95,$ and 0.98). For these analyses we used computing nodes equipped with Intel Xeon E5-2680 v4 CPUs at 2.40 GHz with 96 GB of RAM memory using 3 computing threads.

As one would expect, reducing α from 1 to 0.98, 0.95, and 0.90, led to a slight reduction in the proportion of variance explained by the GE term and a small increase in the error variance (Figure 4). In

general, for the same α-value, the reduction in proportion of variance explained and the increase in the error variance was smaller with the *tensorEVD* compared to *eigen*. We obtained similar patterns for anthesis, silking, and anthesis-silking interval (Supplementary Figures 8-10).

To evaluate **prediction performance**, we conducted a 10-fold cross-validation with hybrids assigned to folds (this mimics the CV1 scheme of Burgueño *et al.* (2012)). For any given α-value the models fitted using the factorization derived with *tensorEVD* and *eigen* produced almost identical predictions (Figure 5). Furthermore, there was a negligible reduction in prediction accuracy associated to lower values of α. For instance, for grain yield, the prediction correlations with the *tensorEVD* method were 0.387, 0.386, and 0.384 for α-values of 0.98, 0.95, and 0.90, respectively (Figure 5).

In the analysis presented above, the Hadamard product matrix had a dimension of $n = 59{,}059$ and a rank (number of eigenvalues greater than zero) of 38,187. Table 1 gives the number of eigenvectors returned by *eigen* and *tensorEVD* by α-value. As previously noted, *tensorEVD* is less efficient than *eigen* at dimension reduction when $n_G \times n_E \gg n$, a condition met in the analysis just described ($4{,}344 \times 97 \gg 59{,}059$). However, using an $\alpha = 0.95$ *tensorEVD* already provides substantial dimension reduction which translates into a shorter total computation time (decomposition + model fitting, Table 1 and Supplementary Figure 11).

Concluding Remarks

The *tensorEVD* method can be used to factorize large Hadamard product matrices that are sub-matrices of Kronecker produces of smaller positive-semi-definite matrices. Examples where such matrices are key components of genomic models include hybrid prediction and G×E models involving genetic and environmental relationship matrices. The proposed algorithm can be several orders of magnitude faster than a standard eigenvalue decomposition, with relatively negligible effect on variance components estimates and prediction performance. The proposed method can be very advantageous in terms of speed and dimensionality reduction in cases where the dimensions of the low-rank matrices are very small relative to the sample size.

Web resources

The 'tensorEVD' R-package is freely available on CRAN (https://CRAN.R-project.org/package=tensorEVD) and on the GitHub repository

(https://github.com/MarcooLopez/tensorEVD). All the scripts used for analyses can be found in the 'tensorEVD' R-package documentation.

Data availability

The data set used in this study for the simulation and genomic prediction applications is fully described in Lopez-Cruz *et al.* (2023) and it is publicly available in the Figshare repository (https://doi.org/10.6084/m9.figshare.22776806). All Supplementary Notes 1-5 and Supplementary Figures 1-11 are included in the Supplementary Material file which is provided along with this manuscript.

Acknowledgments

The authors thank the Genomes-To-Fields (G2F) Initiative for generating and making publicly available the data used in this study.

Conflict of interest

The authors declare no conflict of interests.

Funding

Financial support was provided by the Plant Genome Research Program of the National Science Foundation (NSF PGRP-Tech grant #2035472) and by the National Institute for Food and Agriculture of the United States Department of Agriculture (USDA-NIFA award #2021-6701533413).

References

Bernardo R., 1998 A model for marker-assisted selection among single crosses with multiple genetic markers. Theor. Appl. Genet. 97: 473–478.

Burgueño J., G. de los Campos, K. Weigel, and J. Crossa, 2012 Genomic prediction of breeding values when modeling genotype × environment interaction using pedigree and dense molecular markers. Crop Sci. 52: 707–719.

Crossa J., J. Burgueño, P. L. Cornelius, G. McLaren, R. Trethowan, *et al.*, 2006 Modeling genotype × environment interaction using additive genetic covariances of relatives for predicting breeding values of wheat genotypes. Crop Sci. 46: 1722–1733

Csárdi G., J. Hester, H. Wickham, W. Chang, M. Morgan, et al., 2023 remotes: R package installation from remote repositories, including 'GitHub'. R-package version 2.4.2.1. https://CRAN.R-project.org/package=remotes

Falconer D. S., and T. F. C. Mackay, 1996 *Introduction to quantitative genetics*. Prentice Hall, Essex, UK.

Golub G. H., and C. F. Van Loan, 1996 *Matrix computations*. Johns Hopkings University, Baltimore, MD.

Henderson C. R., 1985 Best linear unbiased prediction of nonadditive genetic merits in noninbred populations. J. Anim. Sci. 60: 111–117.

Herdin M., N. Czink, H. Özcelik, and E. Bonek, 2005 Correlation Matrix Distance, a Meaningful Measure for Evaluation of Non-Stationary MIMO Channels, pp. 136–140 in *IEEE 61st Vehicular Technology Conference*, Stockholm, Sweden.

Jarquín D., J. Crossa, X. Lacaze, P. Du Cheyron, J. Daucourt, *et al.*, 2014 A reaction norm model for genomic selection using high-dimensional genomic and environmental data. Theor. Appl. Genet. 127: 595–607.

Korobeynikov A., R. M. Larsen, and L. B. National Laboratory, 2023 svd: Interfaces to Various State-of-Art SVD and Eigensolvers. R-package version 0.5.4.1. https://CRAN.R-project.org/package=svd

Lima D. C., J. D. Washburn, J. I. Varela, Q. Chen, J. L. Gage, *et al.*, 2023 Genomes to Fields 2022 Maize genotype by Environment Prediction Competition. BMC Res. Notes 16: 148.

Lopez-Cruz M., F. Aguate, J. Washburn, S. K. Dayane, C. Lima, et al., 2023 Leveraging Data from the Genomes to Fields Initiative to Investigate genotype-by-environment interactions in Maize in North America. Nat. Commun. 14: 6904.

Perez-Rodriguez P., and G. de los Campos, 2022 Additions to the BGLR R-package: a new function for biobank size data and Bayesian multivariate models, pp. 1486–1489 in *Proceedings of 12th World Congress on Genetics Applied to Livestock Production (WCGALP)*, Rotterdam.

Qiu Y., and J. Mei, 2022 RSpectra: Solvers for Large-Scale Eigenvalue and SVD Problems. R-package version 0.16-1. https://CRAN.R-project.org/package=RSpectra

R Core Team, 2021 *R: A Language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria.

Searle S. R., 1982 *Matrix Algebra Useful for Statistics*. John Wiley & Sons, Inc, New Jersey

Tables and Figure legends

Table 1. Number of eigenvectors and the times required to perform eigenvalue decomposition and to generate 50,000 posterior samples by method and target proportion of variance explained.

$\alpha \times 100\%$ of variance	Method	Number of eigenvectors	Time to compute the EVD (min)[a]	Time in the Gibbs sampling[b]		Total time (min)[e]
				Per sample	Per 50,000	

				(sec)[c]	samples (min)[d]	
100%	eigen	38187	267.56 (2.82)	0.670 (0.058)	576.61 (38.99)	844.17
98%	eigen	12294	266.39 (3.10)	0.111 (0.003)	98.64 (2.36)	365.02
	tensorEVD	37442	3.13 (0.17)	0.631 (0.023)	545.51 (18.38)	548.65
95%	eigen	7260	266.01 (3.55)	0.062 (0.002)	56.60 (1.08)	322.61
	tensorEVD	19843	2.78 (0.17)	0.217 (0.007)	190.53 (5.87)	193.31
90%	eigen	3839	267.66 (2.76)	0.035 (0.001)	33.18 (0.68)	300.85
	tensorEVD	9512	2.39 (0.11)	0.080 (0.002)	72.31 (1.12)	74.70

[a]Average (SD) across 10 replicates of the decomposition. [b]Gibbs sampler was implemented using BGLR for model in Equation (3) fitted to each trait (grain yield, anthesis, silking, and anthesis-silking interval). Each model was run with 50,000 MCMC iterations (discarding 5,000 as burning and using a thinning of 10 samples) and replicated 5 times. [c]Average (SD), across 4 traits and 5 replicates, time per iteration (median value across iterations). [d]Average (SD), across 4 traits and 5 replicates, time to perform the Gibbs sampler which includes the initial over-heading time (matrices preparation and hyperparameters setting) plus time to complete all iterations. [e]Estimated total computing time (EVD computation + Gibbs sampling). All the computations were carried out on the MSU's High-Performance Computing Center in nodes with Intel processors with 96 GB of RAM memory using 3 computing threads.

Figure 1. Computation time ratio ($\log_{10}$ scale, average across 20 replicates) of the EVD of the matrix **K** using the *eigen* method relative to *tensorEVD* method, by sample size ($n = 10000,\ 20000,$ and 30000 in the x-axis) and proportion α of variance of **K** explained ($\alpha = 0.90, 0.95,$ and 0.98). Each panel represents a combination of number of hybrids (n_G) and number of environments (n_E).

Figure 2. Frobenius norm (average $\pm$ SD across 20 replicates) of the difference between the Hadamard matrix **K** and the approximation ($\widehat{\mathbf{K}}_\alpha$) provided by the *eigen* and *tensorEVD* procedures, by sample size ($n = 10000, 20000,$ and 30000) and proportion α of variance of **K** explained ($\alpha = 0.90, 0.95,$ and 0.98). Each panel represents a combination of number of hybrids (n_G) and number of environments (n_E). Smaller norm indicates better approximation.

Figure 3. Number of eigenvectors (average $\pm$ SD across 20 replicates) produced by the *eigen* and *tensorEVD* methods, relative to the rank of matrix **K**, by α-value (i.e., proportion of variance explained, $\alpha = 0.90, 0.95,$ and 0.98) and sample size ($n = 10000, 20000,$ and 30000). Each panel represents a combination of number of hybrids (n_G) and number of environments (n_E).

Figure 4. Proportion of the phenotypic variance (average $\pm$ SD across 5 replicates) of grain yield explained by each model term (G, E, GE, Error) in Equation (3). The EVD of the Hadamard matrix **K** (covariance matrix of GE) was performed using *eigen* and *tensorEVD* methods for different α-values

($\alpha = 1.00, 0.98, 0.95, 0.90$). Numbers on the top represent the percentage of change (%) relative to the variance explained by each term in the model that uses full information in **K** (i.e., $\alpha = 1.00$) obtained with the *eigen* method (horizontal dotted line).

Figure 5. Within environment prediction correlation (r) for the model in Equation (3) in cross-validation with EVD of the Hadamard matrix **K** performed using the *tensorEVD* (x-axis) and the *eigen* (y-axis) method. Each point gives the prediction correlation obtained within an environment (r_i, $i = 1, \ldots, 97$) with each of the methods, by trait (rows) and α-value (in columns, proportion of variance of **K** explained captured by the selected eigenvectors). Numbers in gray (below and above the diagonal) represent the weighted mean across the 97 environments for each of the methods. The numbers in parenthesis (in red) are 95% confidence intervals for the coefficient b in the regression $r_{eigen} = a + b\, r_{tensorEVD} + \varepsilon$.

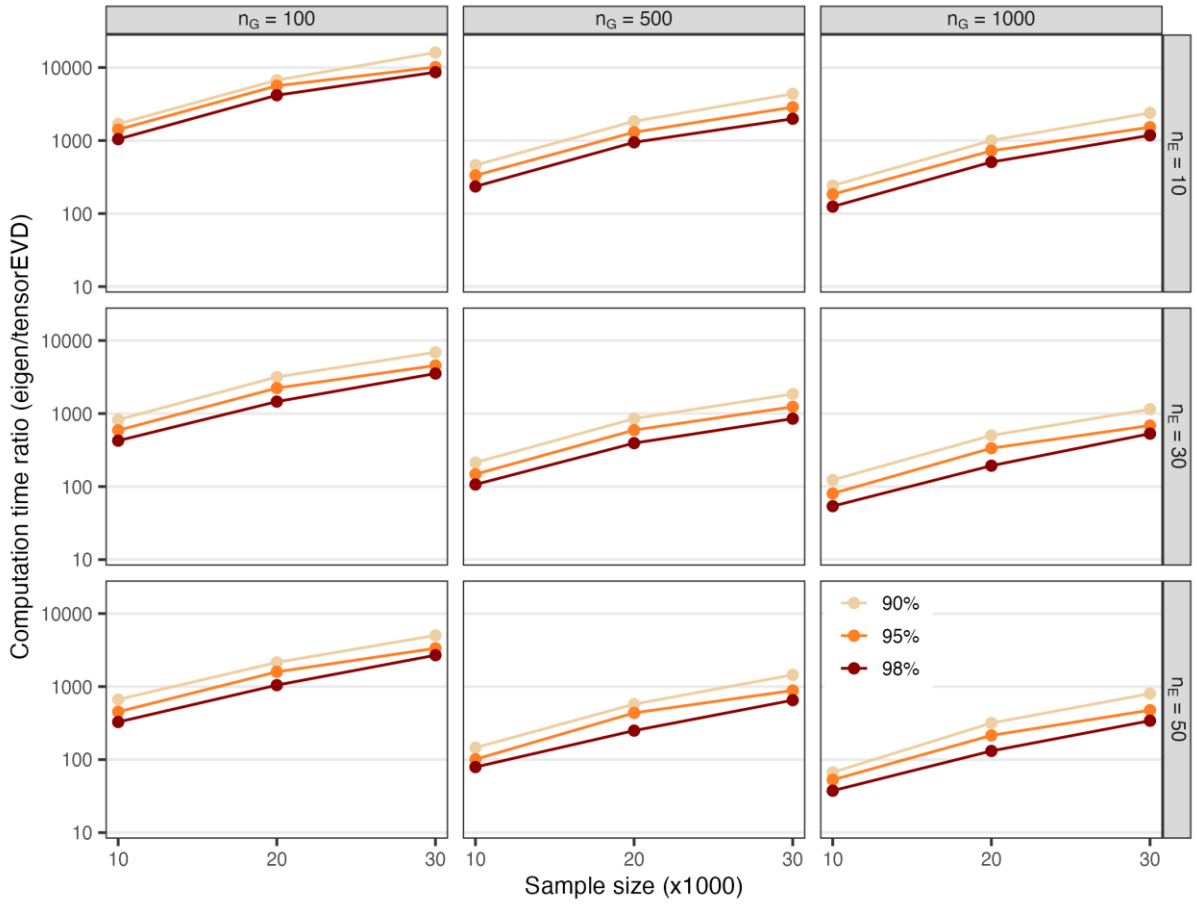


Figure 1
160x122 mm (x DPI)

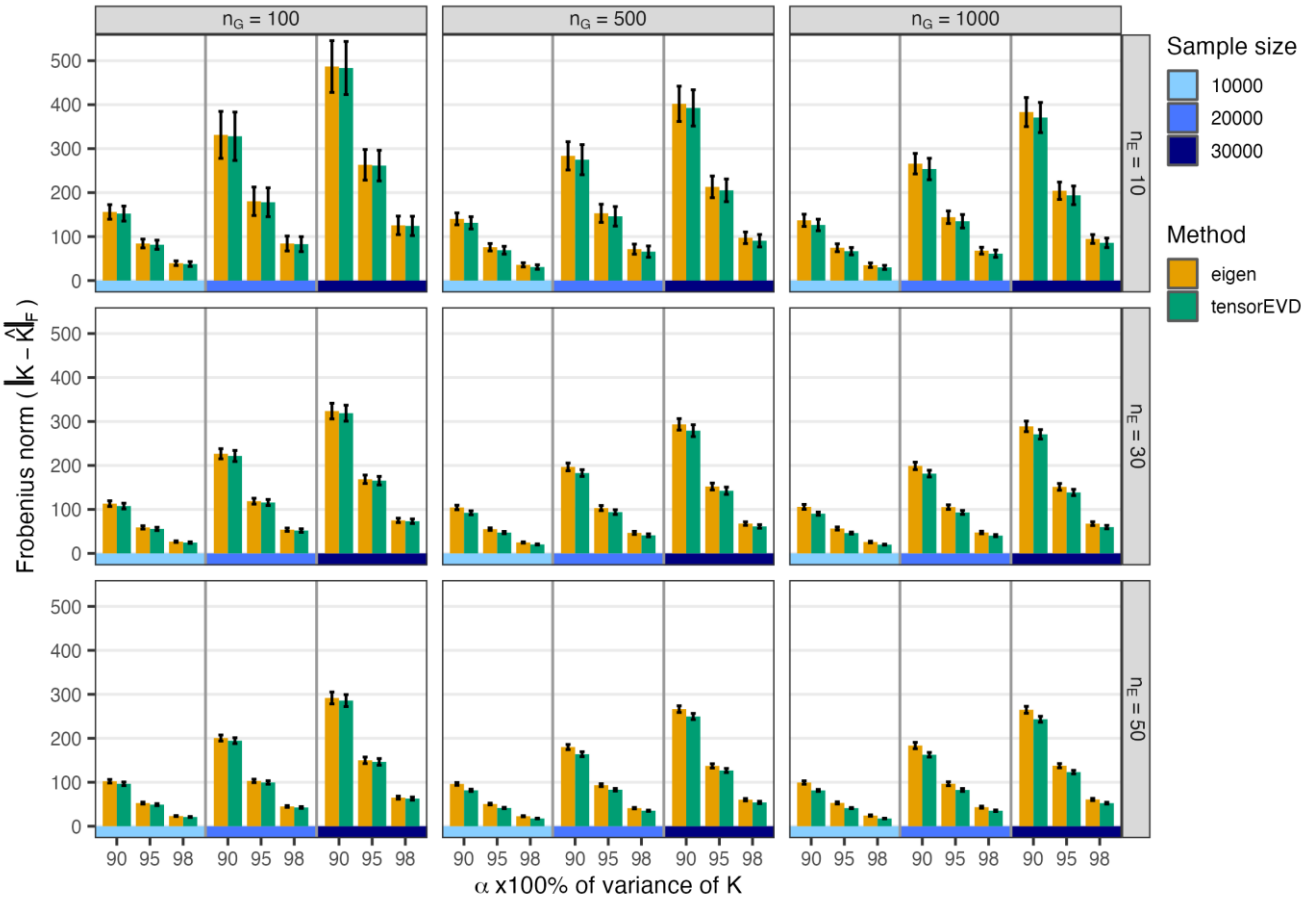


Figure 2
175x122 mm (x DPI)

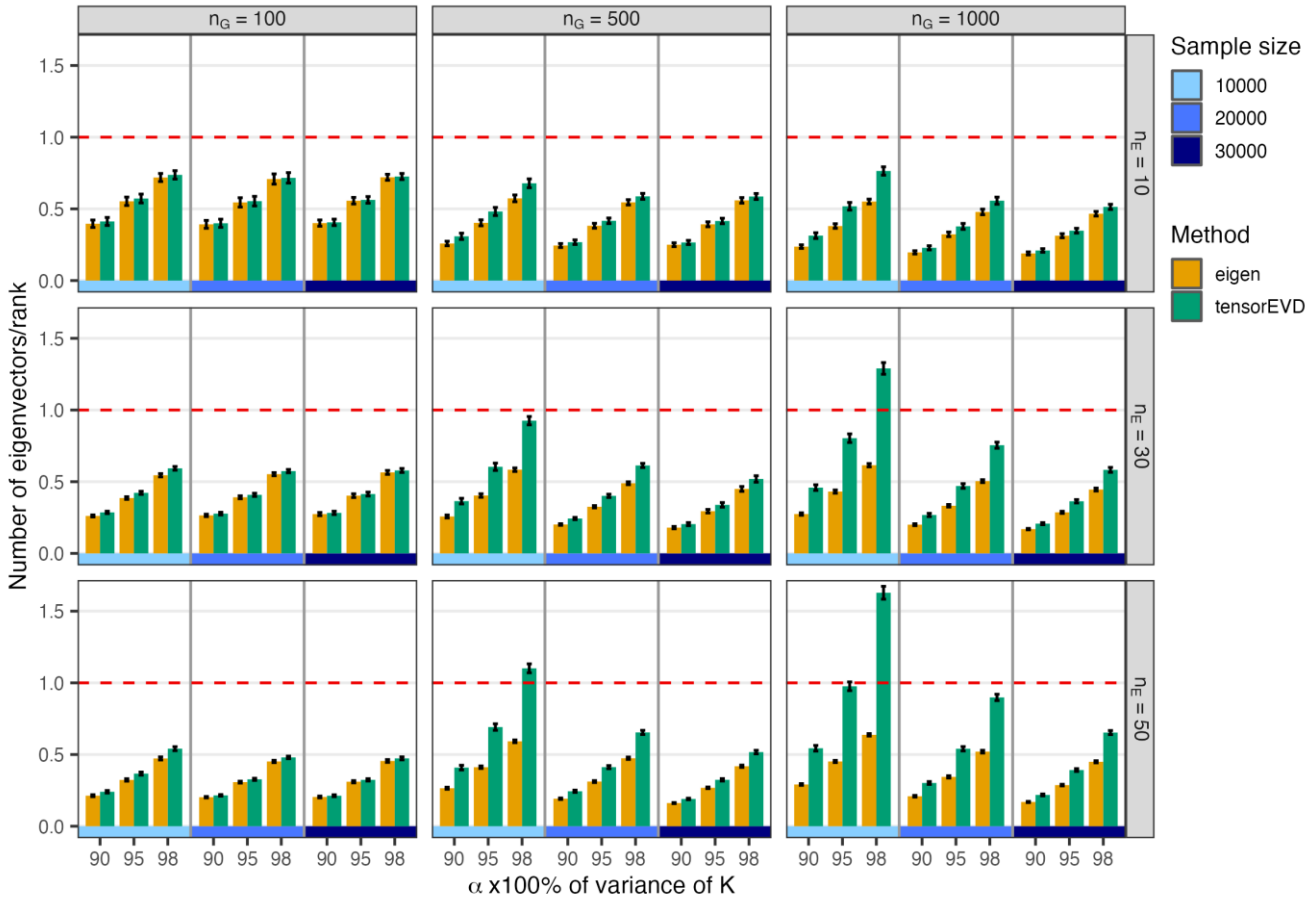


Figure 3
175x122 mm (x DPI)

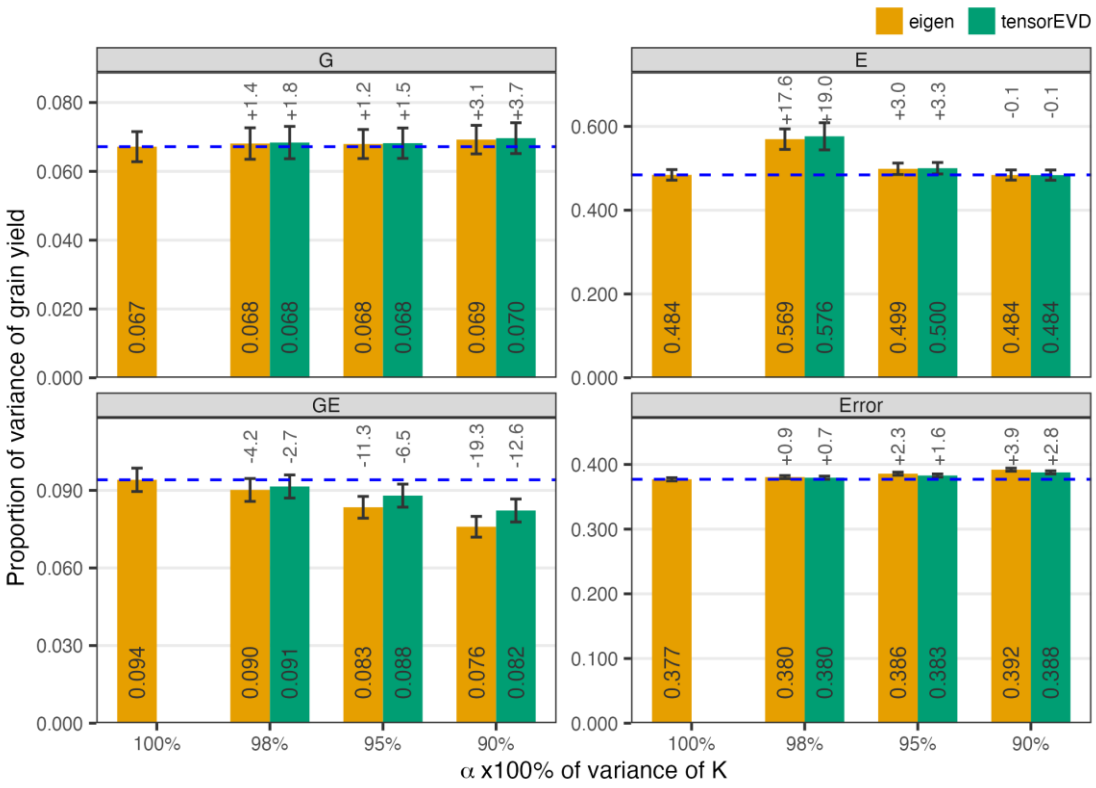
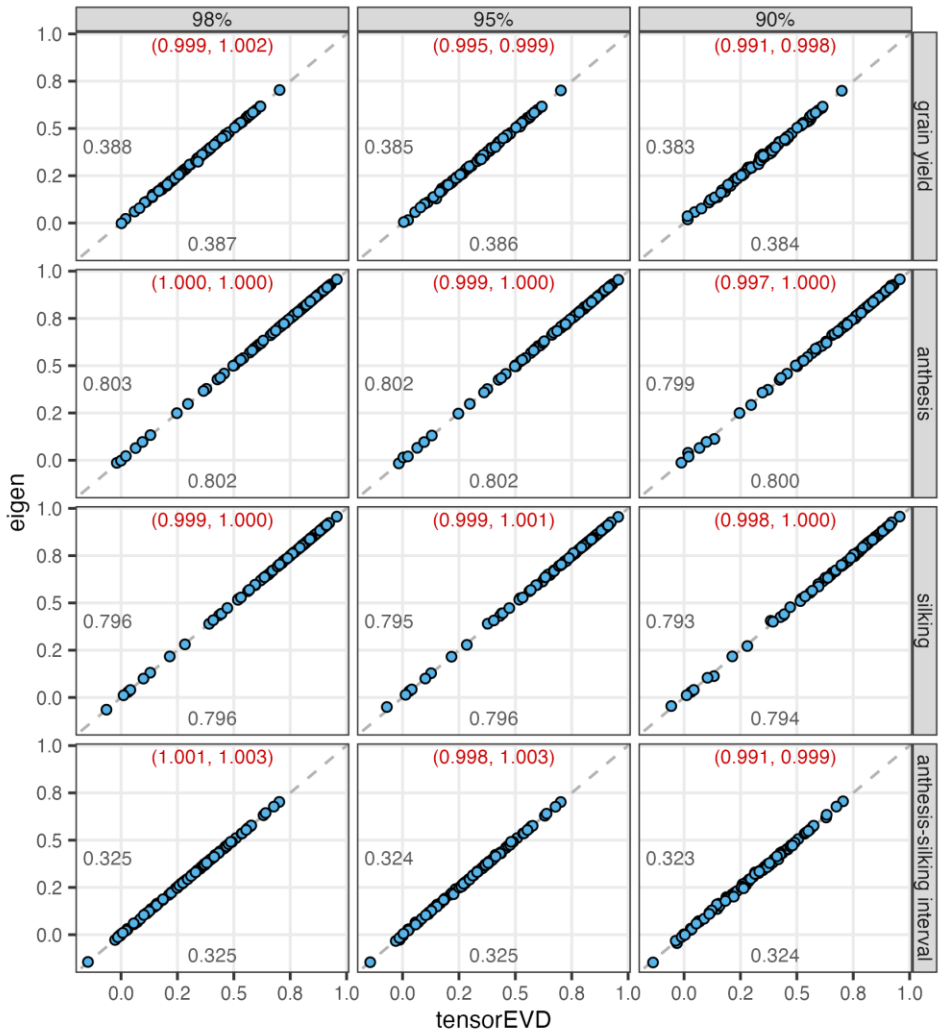


Figure 4
147x107 mm (x DPI)



1

2 *Figure 5*
3 *127x140 mm (x DPI)*