Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Frederick Nilsen

Modeling non-Gaussian traits using Gaussian quantitative genetic models

Master's thesis in Industrial Mathematics (MTFYMA-IM)
Supervisor: Stefanie Muff
June 2023

NTNU
Norwegian University of
Science and Technology

Frederick Nilsen

Modeling non-Gaussian traits using Gaussian quantitative genetic models

Master’s thesis in Industrial Mathematics (MTFYMA-IM)
Supervisor: Stefanie Muff
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Acknowledgements

First and foremost, I wish to start by expressing my gratitude to my supervisor, Associate Professor Stefanie Muff, for the exceptional guidance and support I received throughout the entire master's thesis process. From the very start, she has helped me get started with both the mathematical background and especially the biological background, the latter of which I had almost no prior knowledge. The feedback and discussions that we had during the semester, the frequent meetings and the fast response times have truly been invaluable in shaping my thesis.

Additionally, I would like to thank those who provided me with important methodology and data making this master's thesis possible. Thanks to Professor Jane Reid for allowing me to work with her dataset for my research, as it has been instrumental both in producing the results and in helping me understand the real-world applications. I would also like to thank Pierre de Villemereuil for publishing important methodologies that laid the foundations of this thesis.

Lastly, as this master's thesis concludes my five-year journey at NTNU and my M.Sc. in Industrial Mathematics, I would like to thank some people who have helped me over the past years. Thank you to wonderful friends and family for continued support and motivation. I especially thank my fiancé Lars Dahlgren for standing by my side through thick and thin and for much-needed encouragement over the last five years. Thank you to all the student organizations I have had the opportunity to be a part of, especially the Nabla student organization and the International Student Festival in Trondheim (ISFiT). The skills and competence I have gained from these organizations and the students I have met there have been formative in shaping who I am and taught me how to tackle prior and upcoming challenges of all calibers.

Abstract

In the field of quantitative genetics, the *animal model* has been widely used to model phenotypic traits, incorporating fixed and random effects. When modeling binary traits, a general linear mixed model (GLMM) with a nonlinear link function is often employed, transforming variance components onto a latent scale that differs from the observed scale.

In this thesis, we first examine how heritability, denoted as the proportion of variance in a trait explained by genetic factors, can be translated from an underlying scale to the observed scale using the threshold model. We also aim to determine whether fitting binary features to a Gaussian model can achieve results similar to a binomial model with more complex back-transformation techniques. The aim is to establish whether Gaussian models can be sufficient when assessing observation-scale heritability.

Using a Bayesian statistical framework, we fit animal models as Gaussian and binomial with a probit link function for a dataset of song sparrows and simulated data, calculating estimated heritabilities for both datasets. We compare the posterior density between the Gaussian and binomial models to determine if a simpler linear model may be sufficient.

The results demonstrate that one can fit a Gaussian model on a binary trait and re-scale it back to the underlying liability scale. The rescaling handles models with fixed effects, but it is prone to overestimation in the presence of highly unbalanced traits. We also demonstrate that Gaussian models obtain posterior distributions of heritability close to a binomial model's heritability back-transformed to the observation scales. However, the distributions deviate more by introducing fixed effects whose variance contributes significantly to the total variance.

Overall, the findings show that using a linear model rather than a binomial model to assess additive genetic variance in an animal model for binary data can be viable depending on the accuracy required. Under certain circumstances, such as a dominating fixed effect, the Gaussian model performs worse than a binomial model and requires including variance from fixed effects in the heritability computations. However, these constraints would not affect performance in most practical cases, indicating that simpler and more interpretable approaches can give valid estimates of heritability.

Sammendrag

Innenfor kvantitativ genetikk har *dyremodellen* blitt mye brukt til å modellere fenotypiske egenskaper, med både faste og tilfeldige effekter. Ved modellering av binære egenskaper brukes ofte en generell lineær blandet modell (GLMM) med en ikke-lineær lenkefunksjon, som transformerer varianskomponenter til en latent skala som er svært ulik den observerte skalaen.

Denne oppgaven undersøker først hvordan arvbarheten, betegnet som andelen av total varians som er forklart av genetiske faktorer, kan transformeres fra en underliggende skala til den observerte skalaen ved hjelp av terskelmodellen. Det er også ønskelig å finne ut om gaussisk modellering av binære fenotyper kan oppnå resultater sammenlignbart med en binomisk statistisk modell med tilbaketransformasjonsteknikker. Målet er å finne ut om gaussiske modeller kan være tilstrekkelige for å vurdere arvbarhet på en observasjonsskala.

Ved å bruke et bayesiansk statistisk rammeverk opprettes dyremodeller som gaussiske og binomiske med en probit lenkefunksjon, for et datasett med sangspurver og simulerte data, og beregner estimert arvbarhet i begge tilfellene. Den posteriore fordelingen av arvbarhet for den gaussiske modellen sammenlignes med binomialmodellen for å finne ut om en lineær modell kan være tilstrekkelig.

Resultatene viser for det første at man kan tilpasse en gaussisk modell på en binær fenotype og skalere den tilbake til den underliggende kontinuerlige skalaen. Reskaleringen håndterer modeller med faste effekter, men overestimerer arvbarhet i modeller med et svært ubalansert fenotypisk gjennomsnitt. I tillegg demonstreres det at gaussiske modeller oppnår posteriore fordelinger av arvelighet nær binomiske modeller tilbaketransformert til observasjonsskala. Fordelingene avviker imidlertid mer ved å introduksjon av overdispersjon eller faste effekter i kombinasjon med stor additiv genetisk varians.

Samlet sett viser funnene at det kan være aktuelt å bruke gaussisk modell i stedet for en binomisk modell, i sammenheng med den additive genetiske variansen i en dyremodell, dog avhengig av nøyaktigheten som kreves. Under visse omstendigheter, for eksempel én enkelt dominerende fast effekt, oppnår den gaussiske modellen dårligere resultater enn en binomisk modell, og krever i tillegg å inkludere varianskomponenten fra de faste effektene i beregningen av arvbarhet. Disse begrensningene ville derimot ikke påvirket arvbarheten i de fleste praktiske tilfeller, noe som indikerer at enklere, lineære og mer tolkbare modeller kan gi lovende estimater av arvbarhet innen kvantitativ genetikk.

Contents

Figures

Tables

Chapter 1

Introduction

Quantitative genetics is a subfield of genetics and biological research that relates statistical methods and models to genetic and biological observations. In particular, quantitative genetics is useful for inference to understand how different genetic components and environmental factors contribute to individual traits such as height, probability of disease, or overall fitness. The field has applications in scientific areas such as biodiversity research, agriculture, and medicine (Linney et al. 2003; Topp et al. 2016; Reid et al. 2021). Rather than studying alleles at any specific locus in the genome, quantitative genetics uses overall summaries of the individually minor effects of alleles at many different loci (location/position of a particular gene), making it viable for statistical analysis (Aase 2021).

In the context of quantitative genetics, the response variable in a statistical model is often the phenotypic trait, which is an observable characteristic of an individual within the studied population, either directly or indirectly. Examples of phenotypic traits can be categorical, such as hair color in humans, continuous, such as height, or binary, such as juvenile survival (i.e., an individual surviving to a defined adulthood threshold in a natural habitat). We can differentiate between the phenotype and the genotype, where the latter is the complete genetic composition of an individual (Baker et al. 2008). Traditionally, the distinction between phenotype and genotype is that the appearance (physical structure) of an individual is its phenotype (Watson 1970). However, it is important to reiterate that a phenotypic trait is not necessarily directly observable, but can be an aggregation of observations or otherwise implicitly inferred from observations of an individual.

A phenotypic value of an individual is determined by the individual's genotype and the environment (defined as nongenetic factors), expressed as $P = G+E$ (Conner, Hartl, et al. 2004). In other words, we consider an additive partitioning of the phenotypic trait into genetic components (G) and environmental components (E). By partitioning the genetic and environmental components of a phenotypic trait, we can alongside genetic data from, e.g., pedigrees, estimate the additive genetic variance. For a given allele, we define additive genetic variance as the deviation between the phenotypic mean from inheritance, and the allele's relative effect on a phenotype (Byers 2008). In cases with a large additive genetic variance, the rate of evolutionary selection becomes large, allowing the population to adapt faster to new factors such as environmental change or the emergence of new natural predators.

The heritability of a trait is another important parameter in quantitative genetics, defined as the proportion between additive genetic variance and total phenotypic variance (Falconer 1996). Therefore, heritability is a standardization of additive genetic variance that allows analysis and comparison between studies and populations. Understanding heritability is important to predict the response of populations to selection. However, there are apparent disadvantages to using heritability. Within wild population studies, environmental factors are not controlled, making it difficult to distinguish between environmental and genetic factors, and thus introducing bias to the heritability estimate. Furthermore, in human behavioral genetics, heritability can be misleading due to the multicausal nature of human traits (Moore and Shenk 2017). Also note that additive genetic variance is relative to total phenotypic variance and is only concerned with the extent to which individuals differ in terms of their genetic makeup within the studied group, and not the individuals themselves (Gazzaniga et al. 2010). An alternative metric for a trait's ability to evolve is evolvability, where the additive genetic variance is divided by the square of the phenotypic mean of the trait (Hansen et al. 2011). Although such alternative metrics exist, we will limit ourselves to estimating the heritability of a trait for the purposes of this thesis.

In general, modeling biological traits in nature can become a complex task due to the multitude of genetic and environmental causes that can influence a trait. Historically, researchers have used continuous Gaussian traits to model complex processes (Nelson et al. 2013). The *animal model* is a mixed effects model using measures of relatedness in the population using, e.g., pedigree data (Kruuk 2004). While Gaussian models have been useful in many cases, for instance, with continuous response types, this paradigm may not be able to capture the complex biological processes contributing to a phenotype, for example in the case of binary traits.

Another approach is to use binomial regression within the framework of the animal model. In a linear model, the variance components estimated are on the same scale as the response, which we call the observation scale. In binomial models, this is no longer the case due to the nonlinear relationship between the response and the fitted values, and consequently the variance components attain a *latent scale*, not directly related to the biological observation-level response. Recently, methods have been used to scale the latent variance back on the observation scale (de Villemereuil, Schielzeth, et al. 2016). With some closed-form exceptions, the method requires a series of numerical integrations, considerably increasing the time complexity. Therefore, our research question is whether a linear model could instead estimate the heritability of a binary trait. Although a linear model fitted onto a binary response will likely violate the model assumptions, our aim is to investigate if the violations lead to significant bias in the heritability estimate.

The goal of this thesis consists of two parts. The first part will investigate how one can use a Gaussian model to obtain estimates without nonlinear transformations. The second part will compare the new method to other modern techniques for back-transformation. Overall, the goal is to explore the degree to which we can approximate heritability by fitting a linear mixed model (LMM) instead of a generalized linear mixed model (GLMM) for binary traits. By comparing the performance of these types of models, we aim to shed light on whether a simpler approach may be sufficient for modeling heritability in certain cases. Ultimately, this research could contribute to a better understanding of the complex interplay between genetic and environmental factors in the shaping of biological traits and provide insights into the appropriate modeling techniques to study heritability. If using a linear model yields viable results, we would not need to fit generalized linear mixed models requiring back-transformations, effectively simplifying the statistical work for biological research in the context of heritability.

To achieve these goals, we performed statistical modeling on two datasets measuring a binary trait. The first dataset is from simulated data and the other dataset measures the juvenile survival of song sparrows (*Melospiza melodia*) living on Mandarte Island (Smith et al. 2006). The model specifications in the song sparrow data are inspired by previous work (Reid et al. 2021; Rekkebo 2021) on the same dataset, where logistic or probit regression models were used to analyze binary traits. Furthermore, we use a Bayesian statistical framework based on integrated nested Laplace approximations (INLA) to fit the different models and evaluate the deviation between the linear and binomial models. Since the variance and hence heritability obtain a latent scale, the thesis also reports on the different methods to transform the latent heritability back into an interpretable scale to be comparable to the Gaussian model.

Chapter 2

Theory

The theory chapter provides an overview of the preliminary theoretical groundwork, relevant for understanding the methodology employed. The first section of the mathematical foundation covers the underlying model statements and assumptions in linear models, extending onto GLMMs. Subsequently, we present the statistical models used in quantitative genetics. Lastly, we introduce the statistical framework used for model fitting, namely Bayesian statistics and INLA.

2.1 Mathematical foundation

2.1.1 Linear models

A linear model, hereafter denoted a Gaussian model, assumes that the continuous response variable y is a linear combination of a given number of covariates, an error term and an intercept. Linear regression can be formulated as

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \varepsilon \tag{2.1}$$

where y is the response variable, x_i are observations for covariate i out of the p number of covariates, β_i is the estimator for each covariate and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is the error term. That is, ε follows a standard normal distribution with mean 0 and variance σ^2. Equivalently, linear regression in matrix notation generalizes to n different responses using the design matrix $\boldsymbol{X}$, so that the expression becomes

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \ , \ \boldsymbol{y} = [y_1, \dots, y_n]^\top \ , \ \boldsymbol{\beta} = [\beta_1, \dots, \beta_n]^\top \ \boldsymbol{\varepsilon} = [\varepsilon_1, \dots, \varepsilon_n]^\top, \tag{2.2}$$
$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix} .$$

Linear regression relies on four major assumptions. The first assumption, exogeneity, assumes that there is no error in the data from the design matrix, i.e. $\mathrm{E}[\boldsymbol{\varepsilon}|\boldsymbol{X}] = 0$. The second assumption is that the relationship between the response and the covariates is linear, indicated by equations (2.1) and (2.2). The third assumption is that the estimated errors $\boldsymbol{e} = \widehat{\boldsymbol{y}} - \boldsymbol{y}$, also called residuals, are independent and identically distributed (iid) as a normal distribution with mean zero and constant variance. Finally, we assume that we do not have perfect multicollinearity in the predictors, which means that the covariates are linearly dependent and rank$(\boldsymbol{X})$ is not full. Violating these assumptions can compromise the performance of the model and lead to biased estimators.

There are several advantages to using linear regression for inference. Firstly, the maximum likelihood estimator has the closed form

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y} \ . \tag{2.3}$$

As such, both implementation and interpretability of the model fit are relatively easy. There are several methods for evaluating model fit, and easy-to-use libraries in most modern programming languages already exist. Furthermore, due to the linear relationship between the covariates and the response variable, we can see that an increase of one unit in $\boldsymbol{x}_i$ will increase the expected response by β_i, making it a powerful and flexible tool for statistical inference.

2.1.2 Mixed models

An assumption that limits the use cases for a linear regression model is that the variance of the residuals must be iid and constant, σ^2. This assumption makes it difficult to model intra-class variance, i.e., variance among the different observations within a given class. We can extend (2.1) by including a random intercept effect for cluster i, $\gamma_{0,i} \sim \mathcal{N}(0, \sigma_0^2)$. We consider clusters $i = 1, \ldots, m$, each of which is accompanied by observations x_{ij} for $j = 1, \ldots, n_i$. Then, the model becomes, for each y_{ij},

$$y_{ij} = \beta_0 + \beta_1 x_{ij} + \gamma_{0,i} + \varepsilon_{ij} \ . \tag{2.4}$$

This expression is a random intercept mixed linear model. Generalizing the random intercept model yields a random slope model by introducing $\gamma_{1,i} \sim \mathcal{N}(0, \sigma_1^2)$, such that, for instance,

$$y_{ij} = \beta_0 + \beta_1 x_{ij} + \gamma_{1,i} x_{ij} + \gamma_{0,i} + \varepsilon_{ij} \ ,$$

which is equivalent to the matrix expression

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{U\gamma} + \boldsymbol{\varepsilon} \tag{2.5}$$

$$\begin{pmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\varepsilon} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix}, \begin{pmatrix} \boldsymbol{G} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R} \end{pmatrix} \right) .$$

Note that $\boldsymbol{R}$ and $\boldsymbol{G}$ are block diagonal covariance matrices with diagonal elements $(\sigma^2 \boldsymbol{\Sigma}_{n_1}, \ldots, \sigma^2 \boldsymbol{\Sigma}_{n_m})$ and $(\boldsymbol{Q}, \ldots, \boldsymbol{Q})$, with $\boldsymbol{\Sigma}_{n_i}$ is the covariance matrices for fixed effects and $\boldsymbol{Q}$ for random effects, respectively. The design matrix for random effects, $\boldsymbol{U}$, is also a block diagonal matrix with elements $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_m$ (Fahrmeir et al. 2022).

2.1.3 Generalized linear models

When attempting the model more complex traits in nature, assuming a linear relationship between the covariates and the response variable may not be sufficient. A motivating example of the limits to a linear model is the modeling of a binary response,

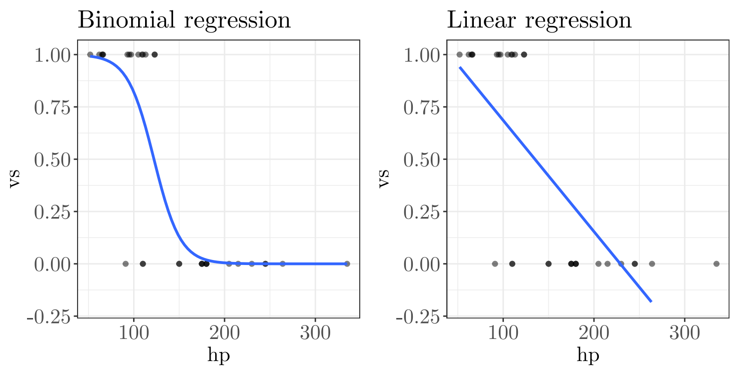


Figure 2.1: Illustrative example of fitting a linear model (to the right) versus a generalized (binomial) linear model (to the left) when the response is binary. The data used is from the standard `mtcars` set provided by R (R Core Team 2021). The response is the engine type (`vs`), either V-shaped or straight, and the covariate is gross horsepower (`hp`).

where the only values y_i attain are zero or one. Figure 2.1 illustrates how a linear estimator can struggle to estimate binary outcomes.

Binomial regression is a generalized linear model (GLM). In GLMs, we assume that the response follows a distribution of the exponential family, such as the binomial, Poisson, exponential, or beta distribution. The expected value of the response links to a linear predictor $\boldsymbol{\eta}$ through a link function. More precisely, a GLM can be written as

$$g(\mathrm{E}[\boldsymbol{Y}|\boldsymbol{X}]) = g(\boldsymbol{\mu}) = \boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta} \ , \tag{2.6}$$

where g is the link function, $\boldsymbol{\eta}$ is the linear predictor, $\boldsymbol{\mu}$ is the expected value of the distribution of $\boldsymbol{Y}|\boldsymbol{X}$, and $\boldsymbol{X}$ and $\boldsymbol{\beta}$ are as previously defined. For the example of binary binomial regression, we get $\boldsymbol{Y}|\boldsymbol{X} \sim \mathrm{Bern}(\boldsymbol{\mu})$, that is, $\boldsymbol{Y}|\boldsymbol{X}$ is assumed to have a Bernoulli distribution with mean $\boldsymbol{\mu}$. The common link functions for binomial regression are the logit, probit, and cloglog (complementary log-log) links, and are provided in Table 2.1.

Extending on the idea of an LMM, we have generalized linear mixed models (GLMMs) whose linear predictor $\boldsymbol{\eta}$ can include random effects, in addition to fixed effects. In general, a GLMM will have the form

$$\begin{aligned} \boldsymbol{\eta} &= \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{U}\gamma \ , \\ \boldsymbol{\mu} &= g^{-1}(\boldsymbol{\eta}) \ , \\ \boldsymbol{Y}|\boldsymbol{X} &\sim \mathcal{D}(\boldsymbol{\mu}, \boldsymbol{\theta}) \ , \end{aligned} \tag{2.7}$$

for a distribution $\mathcal{D}$ of the exponential family with expected value $\boldsymbol{\mu}$ and (potentially additional) parameters $\boldsymbol{\theta}$.

2.2 Models in quantitative genetics

We will use the animal model, which is a specific GLMM, to estimate the additive genetic variance. For context, we first introduce the infinitesimal and threshold model.

Link function	Expression
Logit	$\log(y_i/(1-\log(y_i)))$
Probit	$\Phi^{-1}(y_i)$
Cloglog	$\log(-\log(1-y_i))$

Table 2.1: Most common link functions for binomial regression. Here, Φ is the cumulative density function of a standard normal distribution, $\mathcal{N}(0,1)$.

2.2.1 The threshold model

The infinitesimal model in quantitative genetics assumes that any phenotypic trait is influenced by an infinite number of loci, all of which have some infinitesimal effect. In simpler terms, the trait of an offspring is Gaussian around the mean of its two parents with a variance independent of the values of the parents (Barton et al. 2017). The motivation behind this modeling choice dates back to Galton (1886), suggesting a law of ancestral heredity, where the phenotype is decided as the sum of geometrically declining contributions from parents and further up the pedigree. Each ancestor had a probability p of passing the trait to the next generation. This idea is equivalent to the framework we use today, which is an additive genetic trait with the heritability concept replacing, and generalizing, the probability p (Bulmer 1998).

The infinitesimal model motivates us to use a normal distribution with a random effect. However, Gaussian models struggle to provide accurate predictions of binary phenotypes. To remedy this, the infinitesimal model can be extended to allow for more complex trait modeling, namely through the threshold model. Rather than trying to model a specific trait, we consider a latent trait, also called the liability, with a normal distribution. In the example of modeling a binary trait, for a given threshold M, the values attained below M are cast to the one binary category, and all values above are set to the latter. Figure 2.2 shows an illustration of the threshold model. The heritability computed from the liability trait can be transformed into an

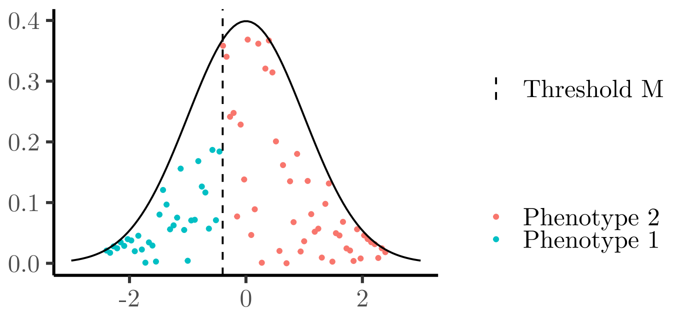


Figure 2.2: Illustration of the threshold model for a binary phenotypic trait. The blue points are realizations of the normal distribution less than the threshold value M and thus belong to phenotype 1. Consequentially, the realizations above M are the red points and belong to phenotype 2.

observation-scale heritability using the relation

$$h^2_{\text{liab}} = \frac{p(1-p)}{t^2}\, h^2_{\text{obs}} \ , \tag{2.8}$$

where p is the proportion of one of the binary phenotypes in the population, and t is density in $\mathcal{N}(0,1)$ at the pth quantile (Dempster and Lerner 1950; de Villemereuil 2018). The idea of the threshold model has been extended further into a multiple threshold model; see, for instance, Reich et al. (1972).

An important result is that (2.8) can be used for Gaussian models fitted to a binary trait when the binary trait is relatively well balanced (van Vleck 1972; Elston et al. 1977). This result suggests that a linear (mixed) model can be sufficient to model binary traits if one scales the outcome by a scalar value dependent on the data. Note that Elston et al. (1977) mention that the variation in liability among groups (in terms of the response) should be small for the approximation to be valid. There is also little to no literature exploring the stability of the approximation when including fixed effects in the model.

2.2.2 The animal model

The animal model is a mixed model that incorporates genetic information as a variance-covariance structure in a random effect. The model can use information from a complex pedigree encoded in a matrix, to estimate the causal components of the phenotypic variance (Kruuk 2004). Whereas alternatives often must assume that there is no assortative mating, inbreeding, or selection, the animal model can account for all three. In a Gaussian case, the animal model can be expressed as the linear mixed model

$$\begin{aligned} &\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}_a\boldsymbol{a} + \boldsymbol{Z}\boldsymbol{u} + \boldsymbol{\varepsilon} \ , \\ &\boldsymbol{a} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\sigma_A^2) \ , \\ &\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}\sigma_E^2) \ , \end{aligned} \tag{2.9}$$

where $\boldsymbol{Y}$ is the phenotypic trait of interest, $\boldsymbol{a}$ are called the breeding values, or the (total) additive genetic merit, and $\boldsymbol{\varepsilon}$ are the errors. Furthermore, $\boldsymbol{X}\boldsymbol{\beta}$ are the intercept and fixed effects, and each $\boldsymbol{Z}_j u_{ji}$ are the random effects other than the breeding values.

A key component of the animal model is the additive genetic relationship matrix $\boldsymbol{A}$. Before defining $\boldsymbol{A}$, we define the coefficient of ancestry, Θ_{ij}, to be the probability that an allele drawn from individual i is the same as an allele from individual j. Then we have that Θ_{ii}, that is, self-relatedness, is $\frac{1}{2}$ and if i and j have a parent-offspring relationship, $\Theta_{ij} = \frac{1}{4}$. We can then define $\boldsymbol{A}$ so that each $A_{ij} = 2\Theta_{ij}$. In terms of notation, the total phenotypic variance on the observed scale will be written as σ_P^2, which is the sum of additive genetic variance, environmental and / or residual variance. Recall that heritability is the proportion of phenotypic variance explained by the additive genetic variance, which now can be expressed mathematically as

$$h^2_{\text{obs}} = \frac{\sigma_A^2}{\sigma_P^2} \ . \tag{2.10}$$

2.2.3 The animal model for non-Gaussian traits

The threshold model has been used for binary and, in general, categorical traits. One can show that there is an equivalence between a binomial GLMM whose link function is the probit link, and the threshold model (de Villemereuil, Schielzeth, et al. 2016). With the extensions to GLMMs, we also have a broader framework that can accurately model non-Gaussian traits.

Specifically, we use a GLMM such as a binomial model with a probit link for the case of a binary trait. Although a probit-based animal model has a better predictive power than a Gaussian animal model, the nonlinearity of the link functions confounds the actual biological values of interest. To illustrate the scales more clearly, we consider a simple binomial probit animal model by

$$\boldsymbol{\eta} = \beta_0 + \boldsymbol{a} \ , \tag{2.11}$$

$$\boldsymbol{a} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\sigma_A^2) \ , \tag{2.12}$$

$$\boldsymbol{\mu} = \Phi(\boldsymbol{\eta}) \ , \tag{2.13}$$

$$\boldsymbol{Y}|\boldsymbol{X} \sim \text{Bern}(\boldsymbol{\mu}) \ . \tag{2.14}$$

The estimated variance components $\hat{\sigma}_A^2$ in the model are on the same scale as (2.11), namely the latent scale, meaning that the variance is additive on the linear predictor's scale. However, it is no longer additive in relation to observed values, which follow a Bernoulli distribution whose mean is linked through the nonlinear function Φ, that is, the cumulative density function for $\mathcal{N}(0, 1)$. We also illustrate this concept in Figure 2.3, where the posterior latent fitted values are shown with the true observations. While the fitted values on the latent scale are far from the true observations, passing the inverse of the link function will obtain the *expected* scale, which reflects the phenotypic mean. However, as apparent in the same figure, none of these scales is the same as the observed data, which are binary realizations of a Bernoulli distribution. In summary, since the link function g is nonlinear, we get a non-additive genetic variance.

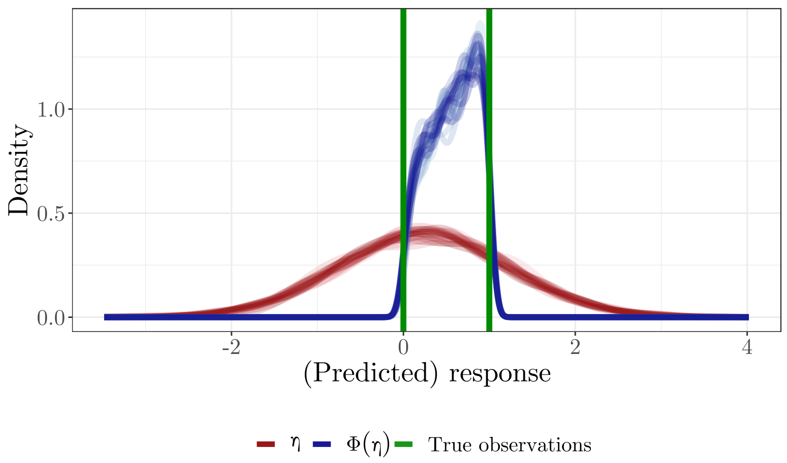


Figure 2.3: Illustrative figure showing the fitted values obtained in a binomial probit model. The red lines, $\eta \in (-\infty, \infty)$, are the marginal fitted values, i.e., the fitted values on the latent scale. By using the inverse of the link function, we obtain the dark blue estimate $\Phi(\eta) \in (0, 1)$, which can be interpreted as the expected scale. Lastly, the green lines show the true observations $y \in \{0, 1\}$. The estimates (η and $\Phi(\eta)$) show 20 posterior samples of these estimates, hence their blurry shape compared to the true observations.

We have motivated the interest in applying animal models to get heritability estimates. However, only reporting the heritability on the latent scale is not viable, as inferring parameters on the observation scale should be the standard approach (de Villemereuil 2018). This raises the issue of how one can get usable estimates on the original observation scale when using nonlinear models. There are several potential remedies to obtain an estimate comparable to the observation scale, one of them by introducing *link variance*. By sending the linear predictor through the nonlinear inverse link function $g^{-1}(\eta_i)$, one introduces variance. Traditionally, this is captured by adding a term to the denominator in the heritability estimate, for instance in the probit case

$$h^2_\Phi = \frac{\sigma^2_A}{\sum_i \sigma^2_i + 1} \ , \tag{2.15}$$

where $\sum_i \sigma^2_i$ is the sum of the variance components from the random effects. Including the term 1 in the denominator captures the link variance for a probit model and attains the same scale as would be achieved using the threshold model, that is, the *liability scale*. Similarly, the link variance in a binomial model with a logit link is $\pi^2/3$ and the cloglog link is $\pi^2/6$ (Nakagawa, Johnson, et al. 2017). Although this method is easy to use, it does not address the issue of fixed effects influencing heritability and does not yield heritability estimates on an observation scale.

In wild populations, we must interpret heritability as conditioned on any fixed effects chosen in the model (Wilson 2008). To remedy this, de Villemereuil, Schielzeth, et al. (2016) introduces a method that attempts to obtain heritability estimates on the observation scale by averaging over fixed effects. The algorithm is based on the same three hierarchical scales introduced by Figure 2.3, namely, the latent scale, the expected scale and the observation scale. The primary goal will be to convert from the latent scale to the observation scale, as presented below.

A general back-transformation algorithm

de Villemereuil, Schielzeth, et al. (2016) also suggests general formulae, both tackling the back-transformation of variables onto the observation scale and providing a method to average over the fixed effects. The purpose of averaging over fixed effects is to attempt to remove the heritability's dependence on model choices of fixed effects.

The algorithm is detailed in Algorithm 2.1. In the algorithm, g^{-1} is the inverse link function for a given model, σ^2_{RE} is the sum of variance from all random effects, $f_{\mathcal{N}}(x;\mu,\sigma^2)$ is the density function for the distribution $\mathcal{N}(\mu,\sigma^2)$, and $\mathcal{V}$ is a distribution-specific variance function.

In the case where we have a binomial probit model, the sequence obtains a closed-form solution, given by Algorithm 2.2, and will be significantly faster than, for instance, a binomial logit model.

2.3 Bayesian inference and INLA

2.3.1 The Bayesian statistical modeling framework

We can consider two main frameworks for statistical modeling: a frequentist approach and a Bayesian approach. When modeling with a frequentist approach, we assume that the parameters in a model have one true value, whereas, in a Bayesian approach, we interpret them as stochastic with their own distributions. The prior distribution in the Bayesian framework represents the knowledge that we have in the model before we observe the data itself. Using an appropriate prior can improve the accuracy of the model. Another key component in Bayesian statistics is that the methods used provide a probabilistic interpretation of the fitted values, which is helpful for further inference in the model of interest. In terms of notation, we define the operator MAP$[\boldsymbol{X}]$ as the maximum a posteriori of a random variable $\boldsymbol{X}$ with a posterior distribution and

Algorithm 2.1 General method for observation-scale heritability h^2_Ψ.

```
Input: Latent additive genetic variance $\sigma^2_{A,l}$ and total phenotypic variance $\sigma^2_{RE,l}$, either intercept $\mu$ or predict (latent marginal estimates). Optional: $w$, width for integration.
if predict given but not $\mu$ then
    $\tilde{\boldsymbol{\mu}} \leftarrow$ predict
else
    $\tilde{\boldsymbol{\mu}} \leftarrow [\mu, \dots, \mu]^\top$                ▷ Such that $\tilde{\boldsymbol{\mu}} = [\tilde{\mu}^{(1)}, \dots, \tilde{\mu}^{(N)}]^\top$
end if
$\bar{z} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \int_{\tilde{\mu}^{(i)} - w\sqrt{\sigma^2_{RE,l}}}^{\tilde{\mu}^{(i)} + w\sqrt{\sigma^2_{RE,l}}} g^{-1}(x) f_{\mathcal{N}}(x; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l}) dx$
$\sigma^2_{RE,exp} \leftarrow \left( \frac{1}{N}\sum_{i=1}^{N} \int_{\tilde{\mu}^{(i)} - w\sqrt{\sigma^2_{RE,l}}}^{\tilde{\mu}^{(i)} + w\sqrt{\sigma^2_{RE,l}}} \left(g^{-1}(x)\right)^2 f_{\mathcal{N}}(x; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l}) dx \right) - \bar{z}^2$
$\sigma^2_{RE,dist} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \int_{\tilde{\mu}^{(i)} - w\sqrt{\sigma^2_{RE,l}}}^{\tilde{\mu}^{(i)} + w\sqrt{\sigma^2_{RE,l}}} \mathcal{V}(x) f_{\mathcal{N}}(x; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l}) dx$
$\sigma^2_{RE,obs} \leftarrow \sigma^2_{RE,exp} + \sigma^2_{RE,dist}$
$\Psi \leftarrow \frac{1}{N}\sum_{i=1}^{N} \int_{\tilde{\mu}^{(i)} - w\sqrt{\sigma^2_{RE,l}}}^{\tilde{\mu}^{(i)} + w\sqrt{\sigma^2_{RE,l}}} \frac{\partial g^{-1}(x)}{\partial x} f_{\mathcal{N}}(x; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l}) dx$
$h^2_\Psi \leftarrow \frac{\Psi^2 \sigma^2_{A,l}}{\sigma^2_{RE,obs}}$
Returns: $h^2_\Psi$
```

Algorithm 2.2 Observation-scale heritability for a binomial probit model.

```
Input: Latent additive genetic variance $\sigma^2_{A,l}$ and total phenotypic variance $\sigma^2_{RE,l}$, either intercept $\mu$ or predict (latent marginal estimates).
if predict given but not $\mu$ then
    $\tilde{\boldsymbol{\mu}} \leftarrow$ predict
else
    $\tilde{\boldsymbol{\mu}} \leftarrow [\mu, \dots, \mu]^\top$                ▷ Such that $\tilde{\boldsymbol{\mu}} = [\tilde{\mu}^{(1)}, \dots, \tilde{\mu}^{(N)}]^\top$
end if
$p \leftarrow \frac{1}{N}\sum_{i=1}^{N} 1 - \Phi(0; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l} + 1)$
$\sigma^2_{RE,obs} \leftarrow p(1-p)$
$\Psi \leftarrow \frac{1}{N}\sum_{i=1}^{N} f_{\mathcal{N}}(0; \mu = \tilde{\mu}^{(i)}, \sigma^2 = \sigma^2_{RE,l} + 1)$
$h^2_\Psi \leftarrow \frac{\Psi^2 \sigma^2_{A,l}}{\sigma^2_{RE,obs}}$
Returns: $h^2_\Psi$
```

represents the mode of the posterior distribution computed for $\boldsymbol{X}$. Simulation studies have also shown that in the case of the estimation of heritability of binary traits, the use of Bayesian models is the most effective (de Villemereuil, Gimenez, et al. 2013), further motivating the use of Bayesian frameworks.

Penalized complexity priors

A suitable choice of priors can significantly improve the performance of a model, while choosing unsuitable priors can increase the bias in the model. Thus, choosing prior distributions in a Bayesian model is not a trivial task. When there is little prior knowledge, noninformative priors, and improper priors (such as the uniform distribution) can be used. An improper prior is any prior distribution that integrates to a non-finite value. However, improper priors can also lead to other issues, such as improper posterior distributions.

For this thesis, we will be using penalized complexity (PC) priors (Simpson et al. 2017). By introducing penalized complexity priors, simpler and more interpretable models will be prioritized, while more complex models that tend to overfit the data will be penalized. PC priors require only two parameters, namely U and α where $0 < \alpha < 1$. Informally, the prior $\text{PC}(U, \alpha)$ on some parameter $\boldsymbol{\theta}$ is expressed so that the probability $\Pr(\boldsymbol{\theta} \leq U) = 1 - \alpha$.

For the case of a random effect $[a_1, \dots, a_n]^\top = \boldsymbol{a} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\sigma_A^2) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\tau_A^{-1})$, the PC prior becomes a type-2 Gumbel distribution with respect to λ,

$$\pi(\tau_A) = \frac{\lambda}{2}\tau^{-3/2}\exp(-\lambda\tau_A^{-1/2}), \quad \tau_A > 0,\ \lambda > 0\ , \tag{2.16}$$

and with $\Pr(\tau_A^{-1/2} \leq U) = 1 - \alpha$, we obtain $\lambda = -\ln(\alpha)/U$ (Simpson et al. 2017). PC priors have a parameterization so that U resembles the standard deviation, for example, a PC prior $\text{PC}(1, 0.05)$ for the breeding values $\boldsymbol{a}$ would resemble $\Pr(\sigma_A \leq 1) = 0.95$.

2.3.2 The INLA computing scheme

INLA is an abbreviation for the term integrated nested Laplace approximation and is an alternative to fitting Bayesian models using Markov chain Monte Carlo (MCMC). INLA is a faster alternative to obtain posterior distributions for latent Gaussian models. In this brief review of INLA, we use the notation and definitions provided in Martino and Paige (2022).

A latent Gaussian model is a relatively broad class of models consisting of observations $\boldsymbol{y}$, a latent field $\boldsymbol{x}$, and hyperparameters $\boldsymbol{\theta}$ with a partition $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)^\top$. We have that

$$\boldsymbol{y}|\boldsymbol{x} \sim \prod_i \pi(y_i|x_i, \dots, \boldsymbol{\theta}_1) \tag{2.17}$$

$$\boldsymbol{x}|\boldsymbol{\theta}_1 \sim \mathcal{N}(0, \boldsymbol{Q}(\boldsymbol{\theta}_2)^{-1}),\ , \tag{2.18}$$

where (2.17) implies that $\boldsymbol{y}$ is conditionally independent given $\boldsymbol{x}$ and $\boldsymbol{\theta}_1$ and $\pi(\cdot)$ indicates the distribution of a random variable, and $\boldsymbol{Q}$ in (2.18) is the precision matrix for the latent field $\boldsymbol{x}$. Furthermore, we assume that $\boldsymbol{x}$ is a Gaussian Markov Random Field (GMRF). To obtain effective computations within the INLA framework, the precision matrix $\boldsymbol{Q}$ should be sparse. Within the context of the animal model,

$\boldsymbol{Q} = \boldsymbol{A}^{-1}$, which has been shown to be sparse (Steinsland and Jensen 2010). The linear predictor in a latent Gaussian model is

$$\eta_i = \alpha + \sum_l f_i(u_{l,i}) + \sum_k \beta_k z_{k,i} + \varepsilon_i \ , \tag{2.19}$$

which we can use to express both linear models, GLMs, GLMMs, and more complex models like geostatistical models.

The INLA computing scheme

The first step of the INLA computing scheme is to obtain estimates for the marginal hyperparameter distribution. That is, $\pi(\boldsymbol{\theta}_j|\boldsymbol{y}) = \int \pi(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta}_{-j}$, which we compute numerically. The notation $\boldsymbol{\theta}_{-j}$ means all elements of $\boldsymbol{\theta}$ except j. In particular, we can approximate the marginal hyperparameter distribution using the expanded form (Morrison 2017)

$$\pi(\boldsymbol{\theta}|\boldsymbol{y}) \approx \left. \frac{\pi(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}) \times \pi(\boldsymbol{x}|\boldsymbol{\theta}) \times \pi(\boldsymbol{\theta})}{\tilde{\pi}_G(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})} \right|_{\boldsymbol{x}=\text{MAP}[\pi(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})]} \ , \tag{2.20}$$

where $\tilde{\pi}_G$ is the GMRF approximation. With these, INLA selects suitable support points (for non-scalar $\boldsymbol{\theta}$) with Newton-like numerical methods such that each $\boldsymbol{\theta}_k$ has weight Δ_k (Morrison 2017).

With the marginal hyperpriors, we can get the distribution of the latent space by $\pi(x_i|\boldsymbol{y}) = \int \pi(x_i|\boldsymbol{\theta},\boldsymbol{y}) \times \pi(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta}$ with additional Laplace approximations, given that they are not Gaussian. The Laplace approximation can be done using either a full Laplace approximation or a simplified version based on a skew-normal distribution to a series expansion of $\tilde{\pi}(x_i|\boldsymbol{\theta},\boldsymbol{y})$ (Martino and Paige 2022). Lastly, we can obtain the posterior estimate $\tilde{\pi}(x_i|\boldsymbol{y})$ as

$$\tilde{\pi}(x_i|\boldsymbol{y}) = \sum_k \tilde{\pi}(x_i|\boldsymbol{\theta}_k,\boldsymbol{y}) \times \tilde{\pi}(\boldsymbol{\theta}_k|\boldsymbol{y}) \times \Delta_k \ . \tag{2.21}$$

The package R-INLA (Rue et al. 2009) implements the computing scheme and will be used to fit all models in this thesis.

Chapter 3

Methods

3.1 Overview

In this chapter, we will present the methods and model specifications used to answer the research question. We will also introduce the simulation and application data used in the study. Supplementary to the chapter is the complete R code provided in Appendix D.

The initial segment of the thesis will investigate different scales of heritability, with particular attention given to the threshold model. In the threshold model, we assume that the underlying scale is a Gaussian distribution. This distribution is not available in application contexts. However, by using simulations, we can construct this Gaussian linear predictor and produce a corresponding binary response variable using a dichotomization of the linear predictor. Then, by computing the heritability from a Gaussian model fitted with the dichotomized response, we can scale it onto the liability scale with the coefficient $p(1-p)/t^2$ from the threshold model (2.8), and compare the model's performance to the true underlying distribution. If the results from the Gaussian model coincide well with the true heritability, we would strengthen the claim that Gaussian modeling may be sufficient to accurately estimate heritability.

The second segment of the thesis aims to compare heritability from state-of-the-art back-transformation strategies applied to a probit model, with heritability from our Gaussian mixed model. The main focus will be to compare the posterior heritability density in the Gaussian and probit models, where the probit model attains the same scale as the Gaussian using back-transformations, namely the observation scale. If the Gaussian observation-scale heritability is similar to that of the more complex back-transformation algorithms, one may conclude that linear mixed models can be sufficient when primarily interested in the additive genetic variance (or heritability) on the observation scale. These comparisons will be the most important results when discussing a Gaussian model's performance on binary traits.

3.2 Statistical models

We define the general statistical models and heritability scales used throughout the thesis in this section. Methods specific to the simulation or application data are provided in the following sections.

In general, the Gaussian model is on the form

$$\boldsymbol{y} = \beta_0 + \sum_{i=1}^{n} \beta_i \boldsymbol{x}_i + \sum_{j=1}^{m} \gamma_{0,j} \boldsymbol{z}_j + \boldsymbol{a} + \boldsymbol{\varepsilon} \ , \tag{3.1}$$

where $\boldsymbol{y}$ is the response, β_0 is the intercept, $\beta_i \boldsymbol{x}_i$ are the n fixed effects, $\gamma_{0,j}\boldsymbol{z}_j$ are the random effects, $\boldsymbol{a} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\sigma_A^2)$ are the breeding values for a defined relatedness matrix $\boldsymbol{A}$, and $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}\sigma_E^2)$ is the error term. The corresponding binomial probit model is defined as

$$\boldsymbol{\eta} = \beta_0 + \sum_{i=1}^{n} \beta_i \boldsymbol{x}_i + \sum_{j=1}^{m} \gamma_{0,j} \boldsymbol{z}_j + \boldsymbol{a} \ , \tag{3.2}$$

$$\mathrm{E}\left[\boldsymbol{y}|\boldsymbol{X}\right] = \Phi(\boldsymbol{\eta}) \ , \tag{3.3}$$

where Φ is the cumulative density function of $\mathcal{N}(0,1)$. Moreover, all random effects are given a penalized complexity prior (PC prior) of the form $\mathrm{PC}(U, \alpha = 0.05)$ (Simpson et al. 2017). Note that PC priors are parameterized so that U denotes the standard deviation and not the variance (Rue et al. 2009). The value of U in the PC priors, as well as the specific fixed and random effects, are denoted in the subsequent sections regarding the application and simulation data, respectively. Since we are using a Bayesian statistical framework, we can get a posterior distribution of the random effects estimates and hence a posterior of the estimated heritability. We use the INLA framework (Rue et al. 2009) for Bayesian inference and model fitting.

3.2.1 Comparing with state-of-the-art techniques

Initially, we fit both a Gaussian and a binomial probit model to the datasets. With the fit of the models, we can compute the heritability. For the Gaussian models, this will simply be

$$h^2_{\text{obs}} = \frac{\hat{\sigma}^2_A}{\hat{\sigma}^2_P} = \frac{\hat{\sigma}^2_A}{\hat{\sigma}^2_A + \hat{\sigma}^2_E + \sum_j \widehat{\mathrm{Var}[\gamma_{0,j}]}} \ , \tag{3.4}$$

where $\sum_j \widehat{\mathrm{Var}\left[\gamma_{0,j}\right]}$ is the sum of variances from all j random effects, excluding the breeding values whose variance is $\hat{\sigma}^2_A$. The observation-scale heritability is thus directly computable from the estimated variance components without transformation. However, computing heritability for a probit model (3.2) would yield a latent scale estimate of heritability, denoted as h^2_{lat}. To obtain a comparable estimate to h^2_{obs}, it is necessary to use some of the techniques described in the theory chapter. Once the estimates are computed for the different scales, we can compare their posterior distributions to infer conclusions about how well the Gaussian model performs. The resulting heritability scales are described in Table 3.1. In particular, we want to compare h^2_{obs} with h^2_{Ψ} and h^2_{liab} with h^2_{Φ}.

Observation scale heritability from probit model

When estimating the heritability from a probit model on the observation scale, i.e., h^2_{Ψ}, we can choose to work with the mode $\mathrm{MAP}[\boldsymbol{\mu}]$ of the intercept estimate $\boldsymbol{\mu}$, or using the latent marginal estimates. By simply using the intercept, we do not average over fixed effects, which could lead to significantly different results (de Villemereuil, Schielzeth, et al. 2016), and is denoted in the results as *No averaging*. In particular, when not averaging over fixed effects, the algorithm simplifies to

$$p = 1 - \Phi(0; \mathrm{MAP}[\boldsymbol{\mu}]; \hat{\sigma}^2_{RE} + 1) \ , \tag{3.5}$$

$$h^2_{\Psi} = \frac{f_{\mathcal{N}}(0; \mathrm{MAP}[\boldsymbol{\mu}], \hat{\sigma}^2_{RE} + 1)\ \hat{\sigma}^2_A}{p(1-p)} \ , \tag{3.6}$$

where p is the observation-scale estimated phenotypic mean. Using latent marginal estimates, which is $\boldsymbol{X}\hat{\boldsymbol{\beta}}$ marginal to random effects, we also have two options, denoted by *Frequentist* or *Bayesian*. The frequentist approach is to use $\mathrm{MAP}[\boldsymbol{X}\hat{\boldsymbol{\beta}}]$ for each predictor so that each sample of h^2_{Ψ} achieves the same predicted values. For example, by sampling 1000 times for a dataset with 500 observations, the same 500 predictors would be used for all 1000 samples. The other option is more in line with the Bayesian framework, in which we draw one sample from the posterior of each $\boldsymbol{X}\hat{\boldsymbol{\beta}}$ as many times as the sampling number for h^2_{Ψ}, but is also slower. Although this method is slower, it is theoretically more sound, given that we use a sufficiently large sample size. Using samples instead of the mode introduces more information, thus improving the estimate of posterior heritability. Comparing how the posteriors differ with these different approaches is relevant for understanding when they may be approximated with a Gaussian model.

3.3 Simulation setup

We lay out how we produce the simulated pedigree data and binary traits below. In short, we simulate a pedigree and binary response, from which we can fit a Gaussian model and scale the observation-scale estimated heritability back onto the liability scale.

3.3.1 Simulating a pedigree

To make a proper model for the animal model, we need to make a simulated relatedness matrix $\boldsymbol{A}$, which requires a pedigree. We can generate a simulated pedigree using the R package `GeneticsPed` (Gorjanc et al. 2023). We use the ratio N_e/N_c to determine the number of dams and sires, where N_e is the effective population size and N_c is the census population size. N_e determines the biological rates such as genetic drift, and analyzing the ratio enables us to examine the impact of different traits and factors' influence on the effective population size (Kalinowski and Waples 2002). For the purposes of this simulation, we fix N_e/N_c to $\frac{1}{2}$ and generate 100 individuals with 100 N_e/N_c dams and sires, respectively, for 9 generations providing 900 data points. We also fix the additive genetic variance σ^2_A for each simulation run. This is sufficient for generating a simulated pedigree and hence a relatedness matrix. Using the relatedness matrix, we may generate random deviates of breeding values a_i of $\boldsymbol{a} = [a_1, \dots, a_n]^\top$ with $\boldsymbol{a} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}\sigma^2_A)$. With our setup, we require a technical modification of the methods `rbv()` from the R package `MCMCglmm` (Hadfield 2010), see Appendix C for more details. The resulting pedigree also provides the generation number and sex for each individual, the latter of which can be used for the simulation of fixed-effects models.

With the defined pedigree and breeding values, we let $\tilde{\boldsymbol{\eta}}$ be the linear predictor, for instance

$$\tilde{\boldsymbol{\eta}} = \boldsymbol{a} + \boldsymbol{\varepsilon} \ , \tag{3.7}$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}\sigma_E^2)$. In most cases, σ_A^2 defaults to the arbitrarily chosen value 0.5 and σ_E^2 to 1, unless otherwise stated. The linear predictor is the true, underlying normal distribution for the simulation.

Simulating a binary response

We can use the dichotomization of $\tilde{\boldsymbol{\eta}}$ to define the simulated response variable as

$$y_i = \begin{cases} 0, & \eta_i \leq c, \\ 1, & \eta_i > c \ , \end{cases} \tag{3.8}$$

for a cutoff value c for each $\eta_i \in \tilde{\boldsymbol{\eta}}$, hereby denoted as thresholding dichotomization. When $\tilde{\boldsymbol{\eta}}$ is centered around zero, this dichotomization leads to a balanced phenotypic response ($p \approx 0.5$) with $c = 0$. Alternatively, we can generate the responses y_i as binomial realizations. Here, the probability of success p_i is based on $\tilde{\boldsymbol{\eta}}$ scaled between 0 to 1, that is,

$$y_i \sim \text{Bern}(p_i = \Phi(\eta_i)) \ . \tag{3.9}$$

We also define $\hat{p}$ as the estimator to the marginal phenotypic mean, i.e.,

$$\hat{p} = \frac{\sum_{i=1}^{N} \mathbb{1}[y_i = 1]}{N} \ . \tag{3.10}$$

This is the same as p used for the threshold model (2.8), and we use the estimator for such computations.

Fitting the model

When fitting the model, we primarily include no fixed effects or any random effects other than the breeding value, and an iid random effect in the probit models. That is, $\boldsymbol{y} = \boldsymbol{\beta}_0 + \boldsymbol{a} + \boldsymbol{\varepsilon}$ in the Gaussian case and $\boldsymbol{\eta} = \beta_0 + \boldsymbol{a} + \boldsymbol{\gamma}_0$ and $\boldsymbol{\gamma}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. When it comes to the prior choice, we will use penalized complexity priors $\text{PC}(U, 0.05)$, where U is rounded up to the closest order of magnitude when $\sigma_A^2 \geq 1$. For example, if σ_A is $3 \cdot 10^2$, U becomes 10^3, meaning that we assume $\Pr(\sigma_A \leq 10^3) = 0.95$. For $\sigma_A^2 < 1$, the prior defaults to $\text{PC}(1, 0.05)$.

3.3.2 Robustness tests

To properly evaluate the robustness of our techniques, we will look into alternative models using fixed effects and overdispersion, respectively. For the case of fixed effects, we generate simulation data using the linear predictor

$$\tilde{\boldsymbol{\eta}} = \boldsymbol{a} + \beta_{\text{sex}} \boldsymbol{x}_{\text{sex}} + \boldsymbol{\varepsilon} \ , \tag{3.11}$$

where $\boldsymbol{a}$ and $\boldsymbol{\varepsilon}$ are as in (3.7), $\boldsymbol{x}_{\text{sex}}$ are realizations of the individual's sex from the generated pedigree, and for a specified choice of β_{sex}. For a sufficiently large weighting of β_{sex} compared to σ_A^2 and σ_E^2 and a fixed variance of $\boldsymbol{x}_{\text{sex}}$, we expect the variance in the fixed effect to dominate the linear predictor. Hence, we must include the fixed effect variances for heritability estimation for such models by adding the term $\beta_{\text{sex}}^2 \sigma_{\text{sex}}^2$, where σ_{sex}^2 is the unbiased sample variance of the observations for the *sex* covariate, to the total phenotypic variance (Nakagawa and Schielzeth 2013). The underlying heritability in the simulation becomes

$$h_{\text{liab}}^2 = \frac{\sigma_A^2}{\sigma_A^2 + 1 + \boldsymbol{\beta}_{\text{sex}}^2 \sigma_{\text{sex}}^2} \ . \tag{3.12}$$

Another consequence of including a fixed effect is that η will no longer necessarily be centered around zero. Thus, we define the cutoff point in the threshold-based dichotomization based on the quantiles of the simulated $\tilde{\boldsymbol{\eta}}$. In particular, we will investigate the performance of the Gaussian model with a balanced $p \approx 0.5$ and an unbalanced phenotypic mean $p \approx 0.1$. In the runs where p is imposed balanced, the cutoff is set to $\mathrm{E}\left[\tilde{\boldsymbol{\eta}}\right]$, and in the unbalanced case, the cutoff is at the $(1-p)$th quantile. For simplicity, all runs use thresholding dichotomization. We test the models with the presence of fixed effects with $\beta_{\text{sex}} = 10$ and $\beta_{\text{sex}} = 100$. The variance explained by the fixed effect is $\beta_{\text{sex}}^2 \sigma_{\text{sex}}^2 = \beta_{\text{sex}}^2 p(1-p)$, for example 25 for a balanced phenotypic mean ($p \approx 0.5$) and $\beta_{\text{sex}} = 10$. For the case of overdispersion, we can change $\boldsymbol{\varepsilon}$ in the linear predictor for the simulation, so that $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}\sigma_E^2)$ with a chosen $\sigma_E^2 > 1$. In addition, we fit two probit models, one without the random iid effect $\boldsymbol{\gamma}_0$, and one model with the effect.

3.4 Data description

For the purposes of this thesis, the application dataset consists of observations of song sparrows from Mandarte Island. Researchers have collected data from this population since 1975, but we mainly use data from 1993-2018, as the individuals in these years have a complete pedigree (Reid et al. 2021). Song sparrows are socially monogamous and open nesting, but are known to aggressively defend territory against intrusion from neighboring males (O'Loghlen and Beecher 1999; Reid et al. 2021).

The dataset contains parental linkage information for the individuals. In the pruned dataset, that is, observations between 1994 and 2018, we have 2722 individuals with a complete pedigree. In addition to parental knowledge, the data includes the natal year, brood date, sex, coefficients of inbreeding, and proportion of genetic origin from immigrants. Finally, we have the binary trait of interest, namely, juvenile survival to adulthood, surveyed in April each year to determine if they had survived their first winter. We provide a brief exploratory data analysis in Appendix B.

For the song sparrow data, we use the same fixed and random effects as in Reid et al. (2021), and prior distributions as Rekkebo (2021), namely

$$
\begin{gathered}
\boldsymbol{\eta} = \beta_0 + \sum_{i=1}^{5} \beta_i \boldsymbol{x}_i + \sum_{j=1}^{2} \gamma_{0,j} \boldsymbol{z}_j + \boldsymbol{a} \ , \qquad (3.13) \\
\gamma_{0,1} \sim \mathcal{N}(0, \sigma_{0,1}^2) \ , \quad \gamma_{0,2} \sim \mathcal{N}(0, \sigma_{0,2}^2) \ , \quad \boldsymbol{a} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{A}\sigma_A^2) \ , \\
\mathrm{E}[y_i | \boldsymbol{X}] = \Phi(\eta_i) \ , \\
\sigma_{0,1}^2, \sigma_{0,2}^2, \sigma_A^2 \sim \mathrm{PC}(1, 0.05) \ ,
\end{gathered}
$$

where the five fixed effects include the inbreeding coefficient, immigration coefficient, natal year, brood date (in days), and sex. The three random effects (z_1, z_2, and $\boldsymbol{a}$) are the nest grouping factor, the natal year grouping factor, and the individual IDs where we encode the pedigree in the relatedness matrix $\boldsymbol{A}$. The prior sensitivity has been shown to be low in this dataset (Rekkebo 2021), so we can safely impose $\mathrm{PC}(1, 0.05)$ on all variance components.

3.4.1 Preprocessing the song sparrow data

The preprocessing step is especially important when working with real data. We start by scaling all continuous variances. Using the scaled data, we continue by constructing the relatedness matrix $\boldsymbol{A}$ using the `nadiv` package (Wolak 2012), see Appendix D for more details. Once $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$ are computed, we may fit Gaussian and binomial models to the dataset, and compute heritability on the different scales as previously described. Note that we require the inverse of $\boldsymbol{A}$ since the INLA framework operates with precision matrices.

3.4.2 Residual analysis

Lastly, we present the methods used in residual analyzes carried out on the application data. For the purposes of this thesis, we employ a cross-validated probability integral transform test (PIT test). When computing PIT values, we use a subset of the data to fit the model and evaluate the cumulative density function (CDF) of the predictions at a set of realized data that is not part of the training set used in the model. The CDF distribution should resemble the uniform distribution between 0 and 1, given that the normality assumptions in our model are correct (Conn et al. 2018; Wang et al. 2018).

3.5 Overview of methodology

This section primarily summarizes the different scales of heritability used in the thesis. Second, we summarize the methodology and thus what figures will be reported in the following chapter.

3.5.1 Overview of heritability scales

A complete overview of the heritability scales and their expressions are provided in Table 3.1. The first heritability scale is denoted h^2_{obs}, observation-level heritability, and is the heritability in a Gaussian model, computed directly as the proportion of phenotypic variance explained by additive genetic variance. Without the presence of fixed effects,

$$h^2_{\text{obs}} = \frac{\sigma^2_A}{\sigma^2_P} \ ,$$

and with a fixed effect such as *sex*, σ^2_P would also include the variance from the fixed effect, $\beta^2_{\text{sex}}\sigma^2_{\text{sex}}$. The other observation-level scale is denoted by h^2_Ψ, which is the result of passing a binomial probit model through the back-transformation algorithm. The computing scheme for h^2_Ψ is provided in Algorithm 2.2 for a binomial model with a probit link. Note that for this scale, we can pass the marginal fitted values for the model with different sampling techniques, providing us with three ways of computing h^2_Ψ, previously denoted as *Bayesian*, *Frequentist* and *No averaging*.

By directly computing heritability for a probit model, we obtain the latent scale with the notation h^2_{lat}. The latent scale is not comparable to any heritability from the Gaussian models, and consequentially will not be reported in the results or in the table overview. However, by including link variance to the denominator of the heritability expression, we get the heritability estimate of a probit model on the liability scale,

$$h^2_\Phi = \frac{\sigma^2_A}{\sum_i(\sigma^2_i) + 1} \ ,$$

where $\sum_i \sigma^2_i$ resembles the sum of all variance components from the random effects in the model. Note that the liability and observation scales are not comparable. However, recalling the threshold model (2.8), we may multiply h^2_{obs} (from a Gaussian model) by

$p(1-p)/t^2$ to obtain h^2_{liab}. Thus, we have the scales h^2_{obs} and h^2_{liab} for the Gaussian models, as well as h^2_{Φ} and h^2_{Ψ} for the binomial models.

3.5.2 Experimental overview

For the simulation study, we first fit a Gaussian model for varying σ^2_A between 10^{-3} and 10^4 and compute its heritability at the observation level (h^2_{obs}). This heritability is scaled back to the liability scale and compared to the theoretical liability scale heritability, σ^2_A/σ^2_P. We compare the rescaling using the response with both threshold and binomial dichotomization. Second, for $\sigma^2_A = 0.5$ and $\sigma^2_E = 1$, we fit a Gaussian and probit model and compute all scales of heritability. We report the mean, mode, and standard deviation, and show density on observation scales and liability scales. To effectively estimate the mode of the heritability, we use kernel density estimation with 512 equidistant points (R Core Team 2021). The deviance information criteria (DIC) are provided alongside these results.

Third, we run a probit model with $\sigma^2_A = \{0.1, 1\}$ and $\sigma^2_E = 1$ to compare the heritability densities from the probit model back-transformation algorithm with the three different sampling techniques of marginal fitted values. Subsequently, we run a simulation that includes fixed effects in the linear predictor, for varying values of σ^2_A, also between 10^{-3} and 10^4, and using rounded dichotomization. The weight β_{sex} for the fixed effect is chosen to be 10 and 100, and the phenotypic mean $\hat{p}$ is chosen to be unbalanced $\hat{p} = 0.1$, and then balanced $\hat{p} \approx 0.5$. Furthermore, for $\sigma^2_A = 10$, $\beta_{\text{sex}} = 10$, $\sigma^2_E = 1$, and $\hat{p} = 0.1$, we report the heritability density on observation scale.

Finally, we fit models without fixed effects, but with overdispersion, i.e., $\sigma^2_E > 1$. For $\sigma^2_A = 0.5$ and $\sigma^2_E = \{2, 5, 10\}$, we report the density of the heritability from a probit model with an extra iid effect. We provide one last simulation run for $\sigma^2_A = 10$ and $\sigma^2_E = 10$. For the sake of comparison, we also report a regular probit model and a Gaussian model without fixed effects and without any extra iid random effect, and report the obtained heritability densities on the observation scale alongside the probit model with an iid random effect.

For the application study, we first fit a Gaussian and probit model and compute all heritabilities on the different scales, also reporting mean, (estimated) mode, standard deviation, and plots of the heritability densities on observation and liability scales, as well as the DIC. Continuing by replicating the methodology from the simulation case, we compare the densities of the heritability obtained from the three different sampling techniques of marginal fitted values for the probit back-transformation algorithm. We also report the residual distribution and distribution of the Gaussian model's PIT values to determine model violations. The last robustness test is to report and compare histograms of off-diagonal values of the generated relatedness matrix from a real dataset, with the relatedness matrix from a simulated pedigree to determine validity of the simulation data.

Notation	h^2_{obs}	h^2_{Ψ}	h^2_{liab}	h^2_{Φ}
Scale	Observation y_i	Observation y_i	Liability $\tilde{\eta}_i$	Liability η_i and $\tilde{\eta}_i$
Model	Gaussian	Probit	Gaussian	Probit
Expression	$\sigma^2_{A,\text{obs}}/\sigma^2_{P,\text{obs}}$	Algorithm 2.1 using $\sigma^2_{i,\text{lat}}\ \forall i$	$\sigma^2_{A,\text{liab}}/\left[\sum_i(\sigma^2_{i,\text{liab}})\right]$	$\sigma^2_{A,\text{lat}}/\left[\sum_i(\sigma^2_{i,\text{lat}})+1\right]$
Interpretation	Observation-scale heritability directly computed from fitted values of variance components. Should be comparable to h^2_{Ψ}.	Observation-level heritability obtained by passing the latent fitted values to the back-transformation algorithm. Should be comparable to h^2_{obs}.	Liability scale either directly computed from an underlying continuous scale (in simulations) or by scaling the observation-level fitted values using the coefficient from the threshold model.	Liability-scale heritability from transforming the latent scale by adding variance related to the link function, here $+1$ (for a probit link). Should be comparable to h^2_{liab}.
Details	The proportion of total variance explained by genetic factors.	Estimators (originally) on latent scale	With threshold model transformation, $h^2_{\text{liab}} = p(1-p)/t^2 \cdot h^2_{\text{obs}}$, where t is the pth quantile of a $\mathcal{N}(0,1)$ density.	Estimators are on latent scale, and assuming probit link.

Table 3.1: An overview of the different heritability scales used in this thesis. For clarity, we explicitly include the scales that the estimated variance components have in the respective subscripts. For instance, $\sigma^2_{A,\text{obs}}$ is when σ^2_A is on the observation scale (that is, Gaussian model), $\sigma^2_{A,\text{liab}}$ denotes when σ^2_A has a liability scale, and $\sigma^2_{A,\text{lat}}$ denotes when it has a latent scale.

Chapter 4

Results

This chapter will present all the results of statistical models, posterior distributions, and estimated heritability, with the aim of determining whether Gaussian modeling can be sufficient to accurately estimate heritability. The results are segmented into simulation and application studies, respectively. We start by presenting the data generated by the simulation study and hence heritability estimates scaled following the threshold model for the underlying distribution. Then, we present obtained estimates of heritability for the different scales, followed by some robustness tests. For the case with the application data, we also provide posterior heritability estimates and include a residual analysis of the general Gaussian model for the application song sparrow data. The figures and data establish the basis for the discussion in the next chapter.

4.1 Simulation study

4.1.1 Gaussian model on liability scale

The main results shown related to the fitting of the Gaussian model on the simulation data are in Figures 4.1a to 4.1c, with σ_A^2, ranging from 10^{-3} to 10^4. We show the outcome for a threshold dichotomization (3.8) and binomial dichotomization (3.9). The last graph uses simulations on a finer grid of σ_A^2 ranging from 10^{-3} to 0.25. The blue line is the heritability of the fitted model on the liability scale, where we scale by the factor $p(1-p)/t^2$ to obtain the estimated h^2_{liab}. The heritabilities on the liability scale appear to generally coincide in the reported simulations.

4.1.2 Heritability on different scales

This section aims to provide an overview of the heritability estimates for its various scales using the simulated data. Initially, we compare the Gaussian and probit models on the liability scale and similarly on the observation scale. Furthermore, we specifically consider the observation-scale heritability of the binomial model (h^2_Ψ), where we investigate how changing fixed effects averaging settings may affect h^2_Ψ.

The posterior heritability density for the liability and observation scale shows similarities between the Gaussian and probit models for heritability (Figure 4.2). The notation and expressions for h^2_{obs}, h^2_{liab}, h^2_Φ and h^2_Ψ are given in Table 3.1. Finally, some statistics (the mean, estimated mode, and standard deviation) of the same posteriors are summarized in Table 4.1. Complementary to these results, we report the

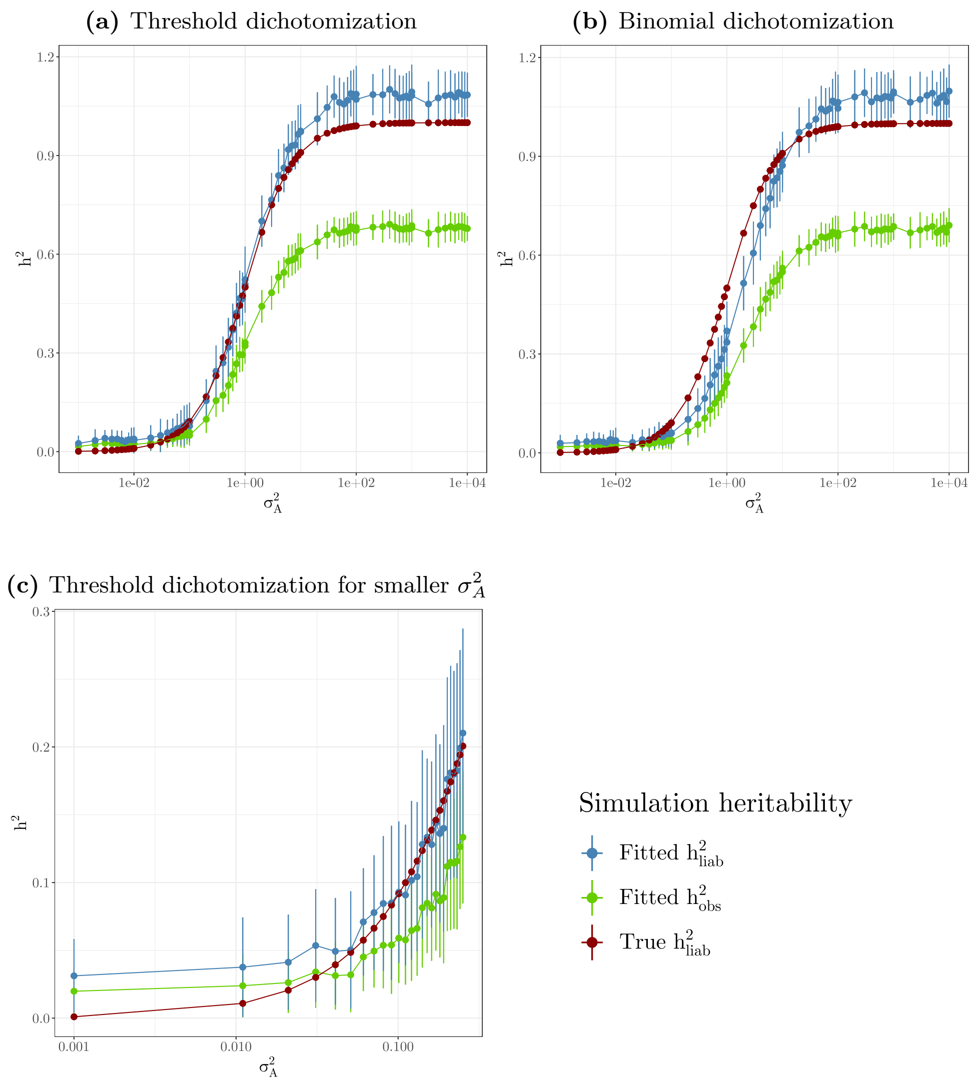


Figure 4.1: Values of heritabilities for the true, underlying continuous scale in the simulation study (red line), alongside the estimated heritability using a Gaussian model. The green line shows the heritability on the observation scale (h^2_{obs}) estimated with a Gaussian model using the dichotomized data, whereas the blue line is scaled by the threshold model formula to obtain heritability on the liability scale (h^2_{liab}). The point shows the mean of 50 runs, with each corresponding line resembling its standard deviation.

deviance information criteria for the two models (Table 4.2). In addition to the figures presented here, we refer to Figure A.1 and Figure A.2 in the appendix, providing a grid of density comparisons for the different heritability scales, for the application and simulation data, respectively.

Model	Mean	Mode	Standard deviation
Gaussian h^2_{obs}	0.178	0.179	0.046
Probit h^2_{Ψ}	0.217	0.183	0.08
Gaussian h^2_{liab}	0.281	0.282	0.073
Probit h^2_{Φ}	0.334	0.333	0.083

Table 4.1: Heritability estimates for Gaussian and probit models, in the simulation data with $\sigma^2_A = 0.5$, showing the mean, mode, and standard deviation. The first two and the last two rows provide heritability comparable to each other. We refer to Table 3.1 for a reference on how the different scales are computed.

(a) Liability scale **(b)** Observation scale

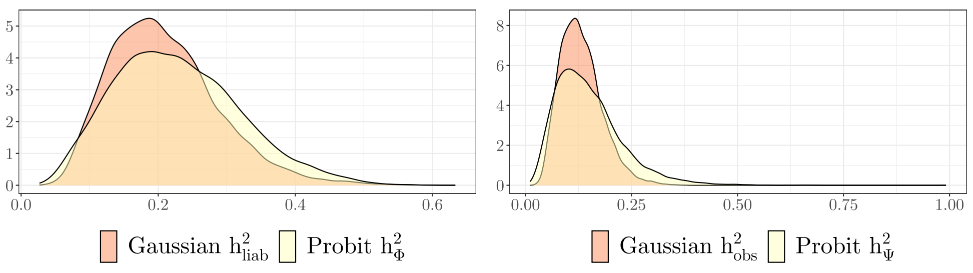


Figure 4.2: Posterior liability- and observation-scale densities of heritability for the simulation data.

Observation scale from probit model

In the framework from de Villemereuil, Schielzeth, et al. (2016), there are several parameter settings for computing observation scale heritability h^2_{Ψ} from binary data, whose difference is described in the methods chapter (section 3.2.1). As such, we first investigate to what extent the heritability densities differ from each other. The resulting posteriors for h^2_{Ψ} are provided in Figure 4.3 for the simulation data, showing little difference for a small $\sigma^2_A = 0.1$, although larger deviations for a larger σ^2_A. In

Model	DIC (Simulation data)
Binomial probit	1180.620
Gaussian	1215.321

Table 4.2: Deviance information criteria (DIC) for the two model fits, for the simulation data, with $\sigma^2_A = 0.5$ fixed.

Estimate	Runtime
h^2_Ψ Bayesian	26 sec
h^2_Ψ Frequentist	11 sec
h^2_Ψ No averaging	4 sec

Table 4.3: Runtime of computing $10,000$ samples of h^2_Ψ in seconds for the three methods provided by the `QGglmm` package, in simulation data with 900 observations.

addition to these posteriors, we show the runtime for computing $10,000$ samples in Table 4.3.

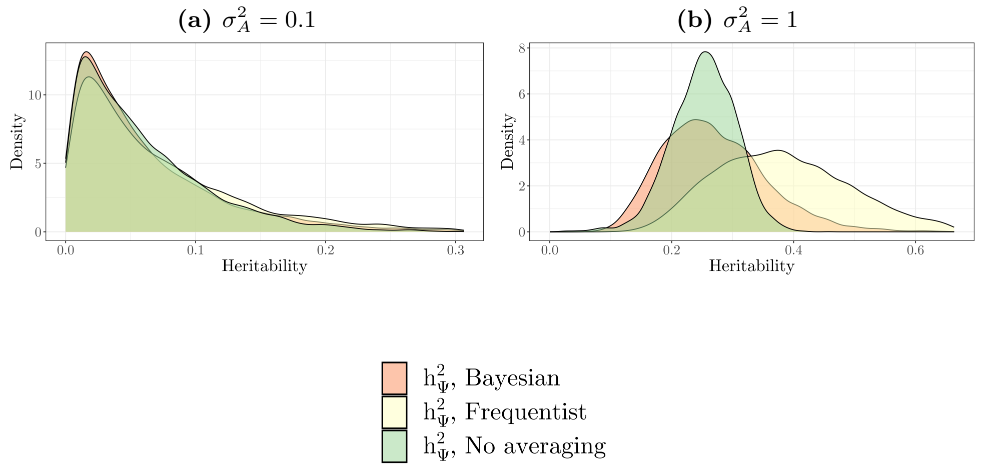


Figure 4.3: Estimated posterior heritability for the different back-transformations techniques, for a binomial probit model to simulations with small variance ($\sigma^2_A = 0.1$) and a simulation with larger variance ($\sigma^2_A = 1$), respectively.

4.1.3 Robustness tests

Lastly, we present some results exploring the robustness of the models, namely by introducing fixed effects and overdispersion. Figure 4.4 illustrates how the heritability of the Gaussian model, once transformed to its liability scale, is relative to the theoretical value for varying values of σ^2_A. In this context, the theoretical value includes fixed effects variance in the denominator, (Nakagawa and Schielzeth 2013)

$$h^2_{\text{liab}} = \frac{\sigma^2_A}{\sigma^2_A + \sigma^2_E + \beta^2_{\text{sex}}\sigma^2_{\text{sex}}} \ . \tag{4.1}$$

The plot is similar to Figure 4.1, although the subplots differ in the magnitude of the fixed effect term rather than different dichotomization methods. We can also see a significant deviation between the estimated and true h^2_{liab} for the case of fixed effects. Additionally, we provide the posterior density of heritability when including fixed effects to assess differences quantitatively (Figure 4.5). The plot shows the posterior heritabilities for a Gaussian and a probit model fitted on simulation data where $\beta_{\text{sex}} = 10$, $\sigma_E^2 = 1$, and varying σ_A^2 and $\hat{p}$. It seems like the estimate from the Gaussian and probit models differ more in the presence of fixed effects. The second robustness factor examined is overdispersion. The heritability of the resulting models for varying σ_E^2 are shown in Figure 4.6, where generally all models seem to produce similar posterior heritability estimates.

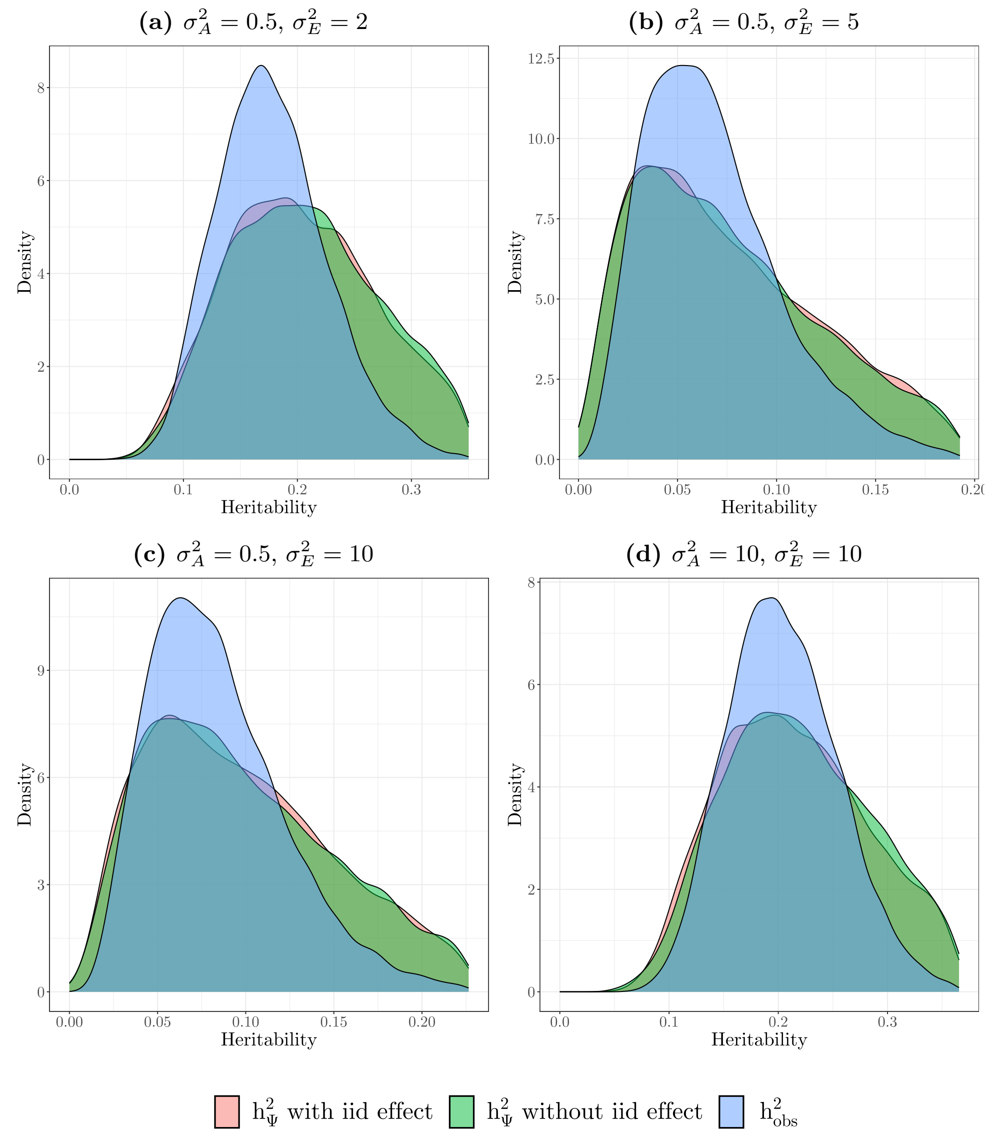


Figure 4.6: A comparison between two probit models and a Gaussian model fitted with the simulation data. One of the probit models has an additional random iid effect to account for overdispersion, whereas the other does not have the random iid effect. The heritability densities are shown for varying σ_A^2 and σ_E^2.

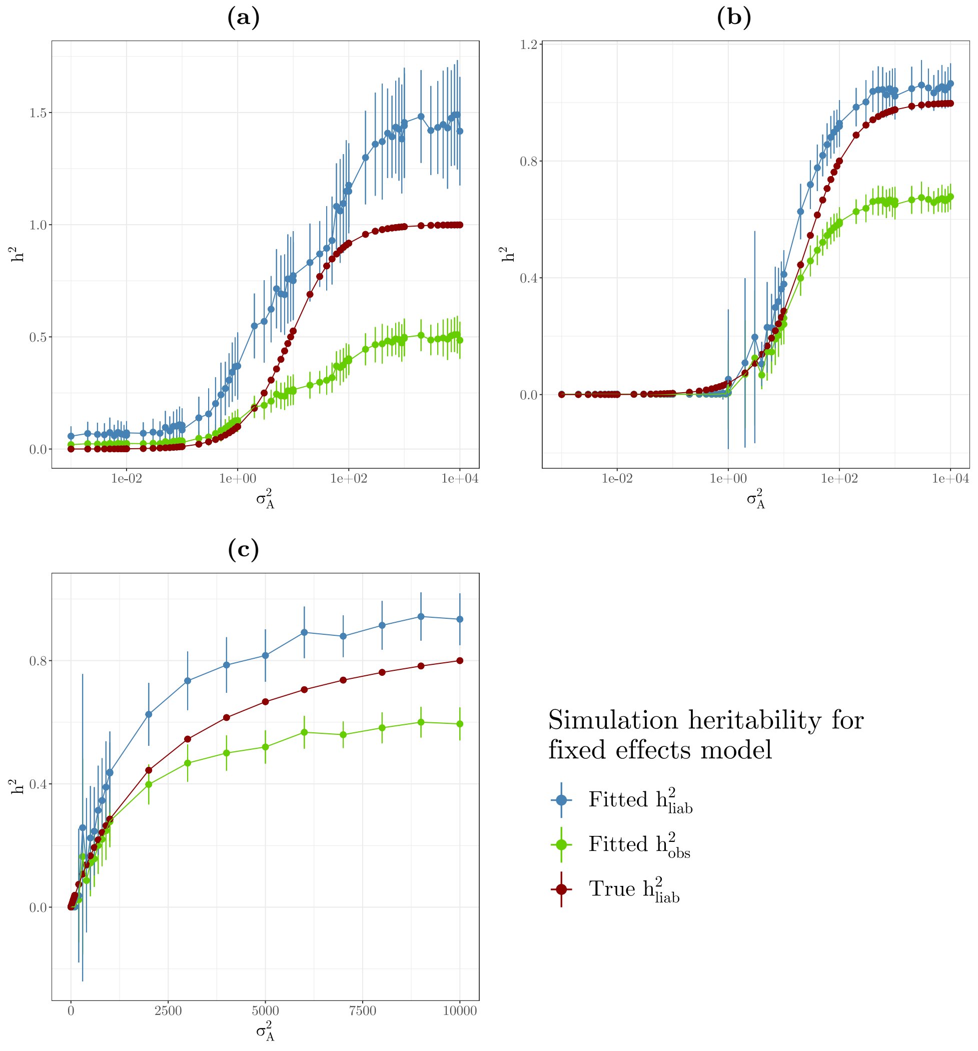


Figure 4.4: Estimates for heritability with a Gaussian model including fixed effects. The red line shows the true heritability on the liability scale and includes fixed effects variance as part of phenotypic variance. Subfigure **(a)** shows the results for an unbalanced $\hat{p} = 0.1$ and $\beta_{\text{sex}} = 10$, whereas the phenotypic mean is balanced in the other subfigures. Furthermore, subfigure **(b)** demonstrate $\beta_{\text{sex}} = 10$ and **(c)** with $\beta_{\text{sex}} = 100$. The points show the mean over 50 runs with each corresponding line resembling its standard deviation.

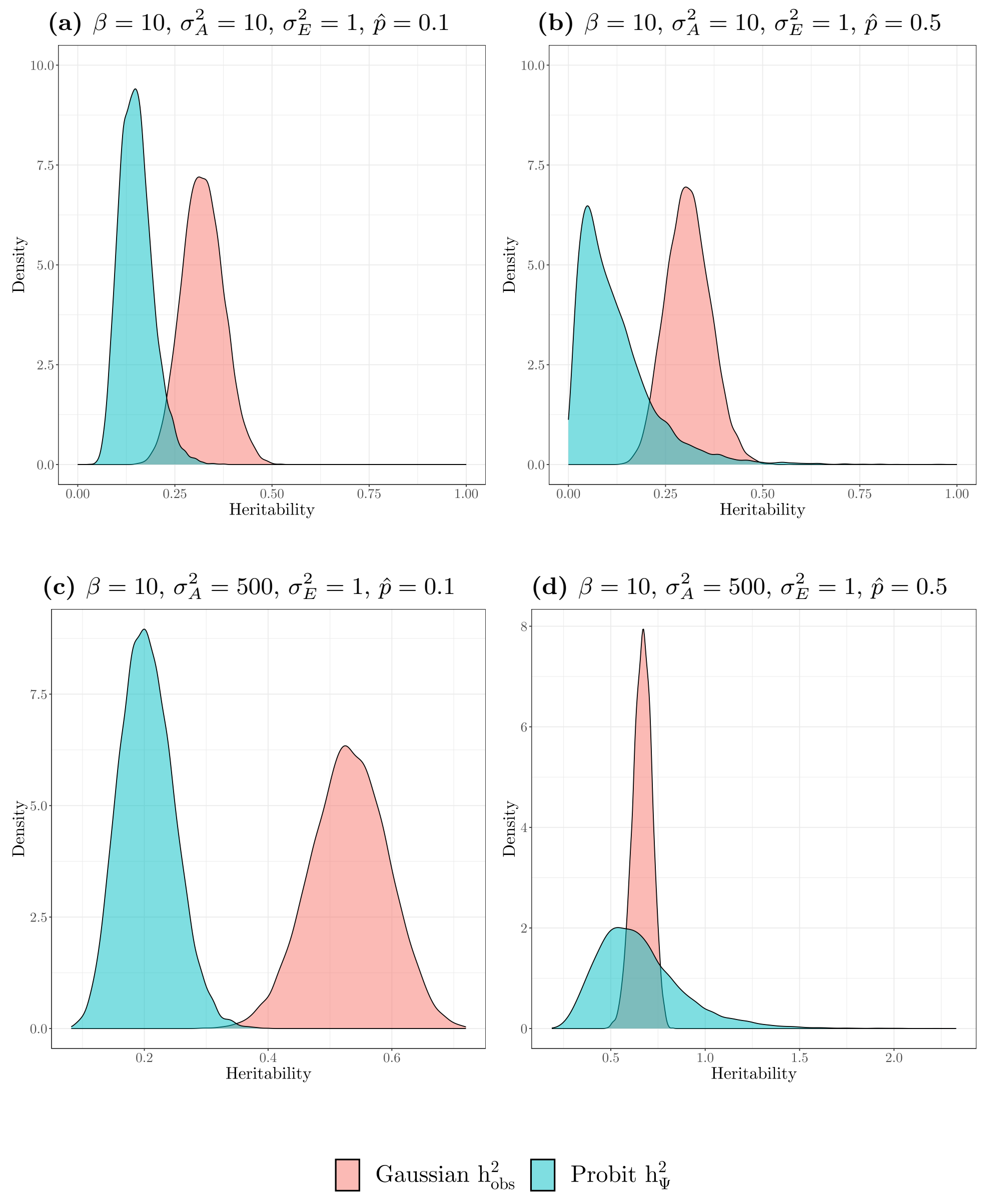


Figure 4.5: Posterior heritability for simulations with fixed effects on observation-scale. The first column shows the results with an unbalanced trait, $p = 0.1$, and the other column for a balanced trait $p = 0.5$. Furthermore, the first row show densities where $\sigma_A^2 = 10$, and the second row when $\sigma_A^2 = 500$.

4.2 Application study

4.2.1 Heritability on different scales

Replicating the method in the previous section, we present the heritability estimates for the application data. They are provided in Figure 4.7 and Table 4.4, showing similarities between the heritability of the Gaussian and probit models on the respective scales. Similarly to the simulation case, we employ the different parameter settings in the algorithm for observation-scale heritability from the probit model, yielding Figure 4.8.

Model	Mean	Mode	Standard deviation
Gaussian h^2_{obs}	0.034	0.027	0.018
Probit h^2_{Ψ}	0.03	0.022	0.016
Gaussian h^2_{liab}	0.062	0.049	0.032
Probit h^2_{Φ}	0.055	0.042	0.027

Table 4.4: Heritability estimates for the Gaussian and probit models, in the application data, showing the mean, mode, and standard deviation. The first and last two rows provide heritability comparable to one another. We refer to Table 3.1 for a reference on how the different scales are computed.

(a) Liability scale **(b)** Observation scale

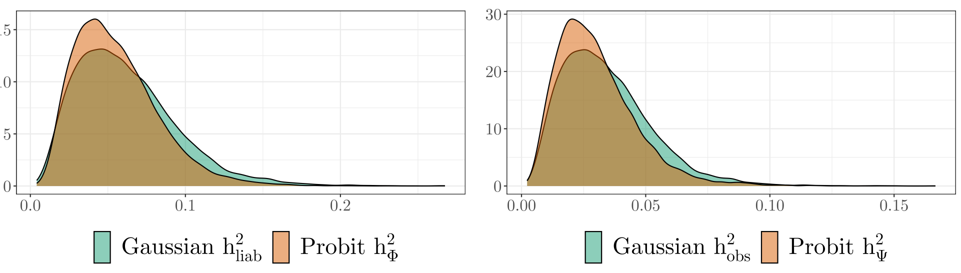


Figure 4.7: Posterior liability- and observation-scale densities of heritability for the application data.

4.2.2 Gaussian residual analysis

In addition to computing the estimated heritability, we would like to explore the residual distribution of the Gaussian model for the application dataset. We fit a Gaussian model using INLA to the application data, in which we also compute the model's PIT values to evaluate the goodness of fit. Figure 4.9 shows the sorted PIT values in ascending order relative to their quantiles, which would resemble a straight line for a uniform distribution, which is not the case for our data. Figure 4.10 shows the PIT values over the mean of the posterior fitted values, as well as a cubic spline smoothing plot. The PIT values show a clear pattern rather than being uniformly distributed, which is also the case for the plot of the residuals (Figure 4.11).

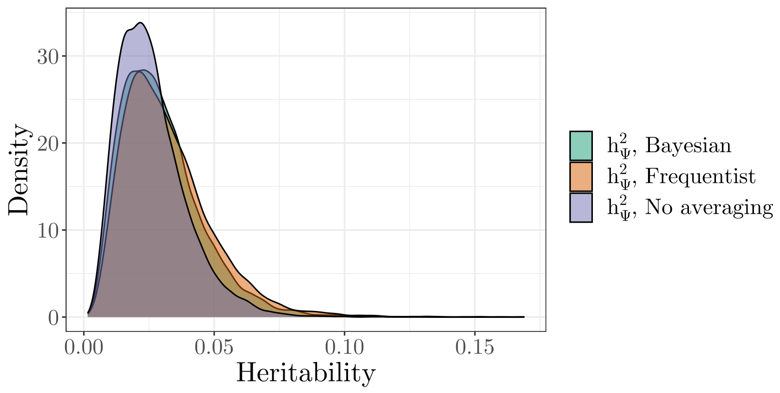


Figure 4.8: Estimated posterior heritability for the different back-transformations techniques, for a binomial probit model to the application data.

Model	DIC (Application data)
Binomial probit	2600.499
Gaussian	2712.749

Table 4.5: Deviance information criteria (DIC) for the two model fits, for the application data.

Another result that we may attribute to the residuals of the Gaussian model is that INLA seems to not always converge for the first initial values for this dataset, but succeeds all times on its second attempt. We cannot observe the same errors when fitting probit models, nor in the Gaussian simulation case. It is also relevant to report the deviance information criteria for the Gaussian and corresponding probit model (Table 4.5).

The final plot provided, Figure 4.12, is a general comparison of off-diagonal values in the relatedness matrix for both datasets. The figure showcases a relatively similar structure in both the simulation and the song sparrow data. In particular, the general relatedness in the song sparrow dataset has more tailing for larger values and a larger mode than in the simulation case. However, both datasets still provide off-diagonal relatedness values around the same area, $(0, 0.2)$, strengthening the simulated data as a viable tool for interpreting real-world data.

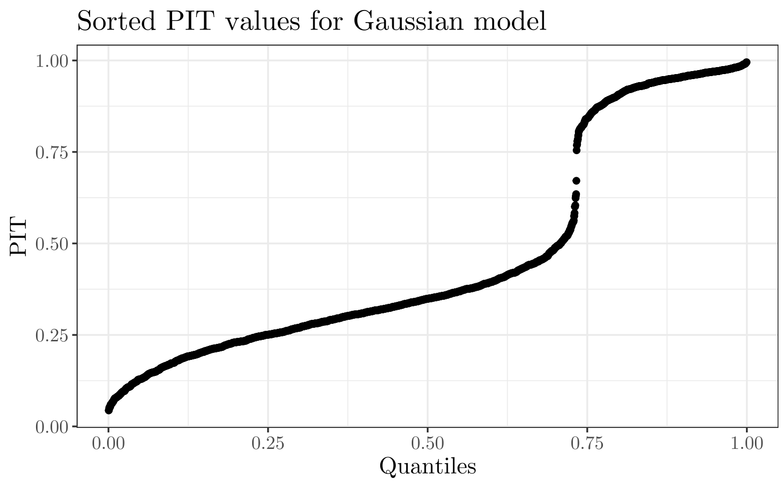


Figure 4.9: Sorted PIT values over quantiles in Gaussian application model.

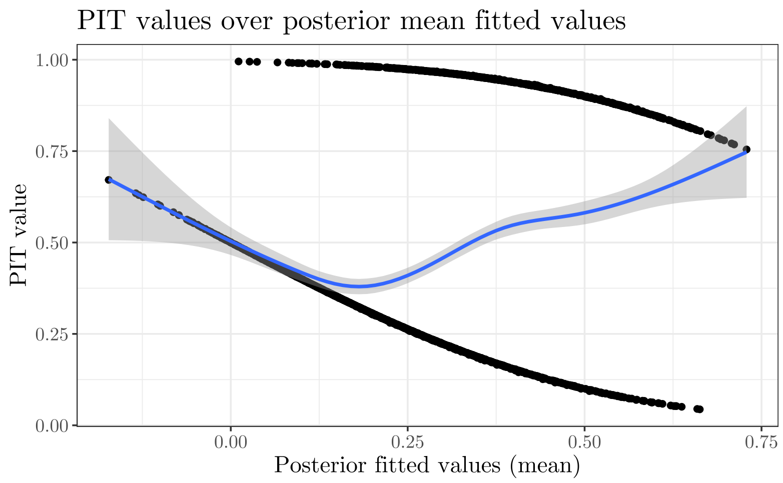


Figure 4.10: PIT values over the mean of the posterior fitted values, using application data. The black dots are the PIT values, the blue line is a cubic spline smoothing and the grey area shows its 95% confidence interval.

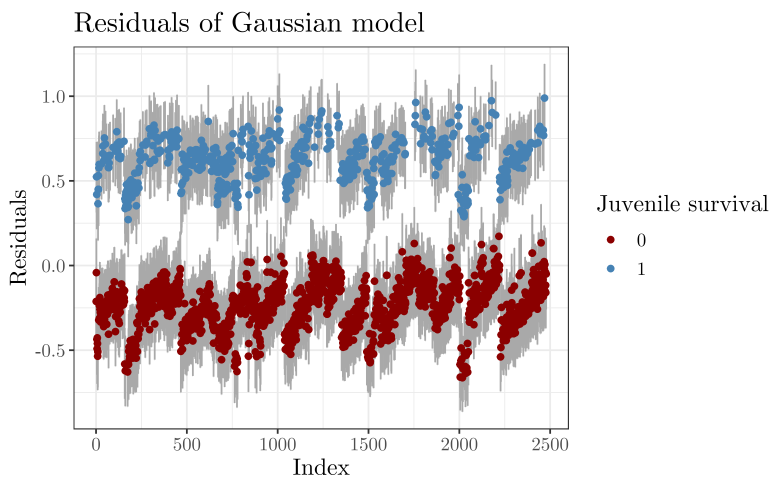
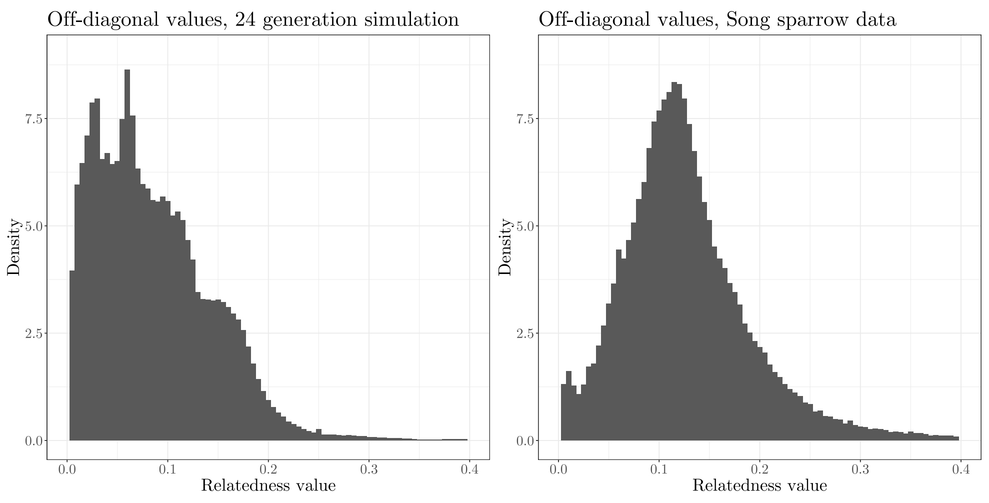


Figure 4.11: Residuals of the Gaussian model with the song sparrow data. That is, $y_i - \hat{y}_i$ for all observed i. The blue-colored dots resemble the difference between the mean of the fitted values and the true observation where juvenile survival is true. Similarly, the red dots are when the trait is false (not survived). The dark grey lines are the 95% credible intervals for each of the residuals, based on the quantiles computes from the posterior fitted values.



(a) Simulation data **(b)** Song Sparrow data

Figure 4.12: Histograms showing the density of the different relatedness values in the relatedness matrix for the simulation data, and the song sparrow data, respectively. The diagonal values (measuring relatedness to the individual itself) are not included, as self-relatedness is one by construction.

Chapter 5

Discussion

The present discussion chapter aims to provide a critical review and comparison of the different scales of heritability estimates obtained using the different models. The results in the previous chapter provide the basis for discussion, where we specifically will evaluate the validity of the Gaussian model, its computational aspects, and whether a Gaussian model can be sufficient for heritability estimation. Included in the chapter is a section on future potential work in the same research domain.

5.1 Gaussian estimates in simulation

The overall results in Figure 4.1 show that the scaled heritability from the Gaussian model coincides well with the true heritability on the underlying scale. For lower values of σ_A^2 and hence h^2 (Figure 4.1c), both the scaled and unscaled plots seem to be close. However, when h^2 increases, it becomes more apparent that we need the threshold scaling.

One potential problematic component of the model specification in the simulation case is the choice of priors. In the model, the prior is chosen based on the true value for σ_A^2, ensuring a well-suited prior estimate. However, this may not reflect the degree of prior knowledge in most datasets, especially not in wild population surveys. To challenge this prior assumption, we performed two different simulations: one with a constant PC prior $\text{PC}(1, 0.05)$ for all σ_A^2, and one case with $\text{PC}(U, 0.05)$ with $U = 2\sigma_A^2$. Both simulations yielded results very similar to the varying priors explained in the methods chapter. Furthermore, we can also note that the binomial dichotomization shown in Figure 4.1b seems to somewhat underestimate the underlying heritability, compared to the threshold dichotomization. The dichotomized values in this case are binomial realizations with success probability given by applying the normal cumulative density function (Φ) to $\tilde{\boldsymbol{\eta}}$, and this dichotomization choice introduces variance that would explain this result. One way to consider the introduced variance from binomial dichotomization is that the mapping from an underlying Gaussian distribution onto the binary traits is not exact and contributes to some unexplained variability. Thus, binomial dichotomization can represent the same as including residual variance in the denominator.

5.2 Interpretability on different scales

This section will consider how the heritabilities obtained from the different scales relate to each other, aiming to answer whether one can use Gaussian regression instead of binomial regression. Recall that a summary of the different scales is provided in Table 3.1.

The first result to consider is the mean and mode of posterior heritability (Tables 4.1 and 4.4). The means and modes have relatively little skewness, since the mean and mode are close to each other, although the mode is consistently lower than its corresponding mean. This can be attributed to some small positive skewness in all of the models.

We may also note that for the application data, the Gaussian h^2_{obs} is less than one standard deviation away from the corresponding probit models on the observation scale, h^2_{Ψ} and h^2_{Φ}. The simulation data become more challenging to interpret. The underlying heritability would in our case be $0.5/(1+0.5) = \frac{1}{3}$, assuming it follows a Gaussian distribution before dichotomization. In this case, the heritability from the Gaussian model transformed to the liability scale h^2_{liab}, along with the link variance transformation h^2_{Φ}, provides estimates closest to the true value (see Table 4.1). This is in line with what we expected, as the two heritability estimates have the same liability scale, whereas the others are on a different scale. Furthermore, solving for the theoretical h^2_{obs} using the threshold model and the true value for h^2_{liab}, we obtain with the simulated $\hat{p} = 0.46$, that the true heritability of the observation scale would be 0.211. As expected, the Gaussian h^2_{obs} and binomial back-transformed h^2_{Ψ} are the closest, where the binomial model's mode is, to a small degree, closer to the true value than the Gaussian mode (Table 4.1). Hence, it seems that the Gaussian model overestimates heritability, albeit to a small degree.

The figures supporting the heritability estimates from the tables are the figures of its posterior density (Figures 4.2 and 4.7). Recall that h^2_{Ψ} is more computationally intensive but also more precise than h^2_{Φ}, which motivates comparing the precise h^2_{Ψ} to the Gaussian model instead of h^2_{Φ}. In addition, the liability scale for the Gaussian model should be equivalent to h^2_{Φ} (de Villemereuil, Schielzeth, et al. 2016).

Observing first the observation scale (Figures 4.2b and 4.7b), we can instantly remark that h^2_{obs} and h^2_{Ψ} are very similar. The Gaussian estimate is more positively skewed for the application data, but is generally very similar to h^2_{Ψ} in terms of shape and numerical values. The simulation has a larger deviation between modes, as the probit one seems to be shifted towards the right. This is in line with our observations from the table summary. The liability scale for the Gaussian model is compared with h^2_{Φ} (Figures 4.2a and 4.7a), where we observe similar deviations, though with less skewness for the Gaussian model.

5.2.1 Robustness of the models

Robustness tests provide information on the performance of various models in estimating heritability on different scales. Based on the fixed effect Gaussian models for varying σ^2_A (Figure 4.4), the Gaussian model obtains estimates of underlying heritability that somewhat approximate the theoretical true value for a balanced phenotypic mean. However, heritability appears to be overestimated, especially in the case of an unbalanced phenotypic mean. Thus, it seems like the threshold model breaks down with a dominating fixed effect.

Additionally, the posterior density of heritability on the observation scale (Figure 4.5) suggests that both the binomial and Gaussian models struggle to accurately estimate heritability on the observation scale when a fixed effect dominates the linear predictor. In general, it is difficult to determine whether the back-transformed probit or the Gaussian model is the closest to the correct value in this result, as we have no true observation-scale heritability for reference. The true heritability (based on the chosen values for the other components of the linear predictor) is on a liability scale, and we cannot safely state that $\hat{p}$, the estimator for the marginal phenotypic mean p, is sufficient to approximate each conditional p_i. Recall that each dichotomized y_i has its underlying probability p_i and is dependent on the covariate for each individual i.

The difference between heritability from the back-transformed probit and Gaussian models appears to also be apparent in larger values of σ_A^2, though to a smaller degree with a balanced phenotypic mean (Figures 4.5c and 4.5d). When the magnitude of β_{sex} for the fixed effect decreases, the proportion of total variance from fixed effects becomes smaller, meaning that we would expect the two heritability densities to be closer to the same value, which is confirmed in Figure A.3 in the appendix.

An important aspect that may explain the results is that the simulation only has one single, binary, and dominating fixed effect. As a consequence, the individual's response is directly associated with its covariate for the fixed effect. Had we used more fixed effects, effects with a smaller weight β and perhaps continuous covariates, the inclusion of fixed effects may not lead to the same results as we have in our simulations. Supporting this idea are the results from the simulation work where the model uses several fixed effects, where the Gaussian model obtains heritability estimates close to the probit model with back-transformation (Figure 4.7).

The second aspect of robustness concerns overdispersion, and the results are provided in Figure 4.6. However, the alternative probit model without any random iid effect results in almost exactly the same posterior density as a standard probit model (with an iid effect), for all combinations tested of σ_A^2 and σ_E^2. This indicates that the model with an iid parameter sends the iid parameter towards zero and attributes the overdispersion variance to the additive genetic variance instead. In the general simulation runs, we also include an iid term, where we see that h^2_{lat} is very close to one, indicating that the iid effect goes towards zero for simulations as well. Although the probit model does not differ when including an iid term in its linear predictor, the Gaussian model generates observation-level heritability closely resembling the probit models for the four combinations of σ_A^2 and σ_E^2.

5.3 Normality of Gaussian model

The normality tests of the Gaussian model indicate that the residuals are far from normally distributed, clearly violating the model assumptions of a linear model (Figures 4.9 to 4.11). Violation of the model assumptions has consequences for further model inference, since, for example, the F-test is sensitive to the normal assumption (C. A. Markowski and E. P. Markowski 1990). In general, models that violate their assumptions can also lead to more biased estimates, since the predictor in linear regression assumes Gaussian residuals with constant variance. If the residuals no longer have constant variance, the equivalence between the closed form solution (2.3) and the maximum likelihood estimate also becomes violated. Thus, the estimates from the predictor and consequently variance and heritability estimates can also be subject to bias. This clear violation is here, however, expected, seeing as the model fits a binary phenotype as the response.

Another result in favor of using binomial regression is the deviance information criterion whose values are shown in Tables 4.2 and 4.5. In both datasets, the binomial model has a significantly lower DIC value than the Gaussian equivalent. Note that we cannot compare DIC values across different datasets, as they should only be used in model selection. Thus, varying values between probit models for the different datasets are expected, and we cannot infer any conclusions from such a result.

Despite clear violations, we still see overall good performance from the Gaussian model. Therefore, the bias to additive genetic variance introduced by normality violation may not be severe enough to bias heritability. Since heritability is a ratio between variances, another possibility would be that the bias in σ_A^2 is somewhat counteracted by taking the ratio of total phenotypic and additive variance. These hypotheses are difficult to test for but can explain why the heritability bias in the Gaussian model is not larger.

5.4 Computational aspects

An important result highlighted in the back-transformation techniques on the application data (Figure 4.8), is that the simpler, yet faster method *No averaging* yields results strikingly similar to the two slower methods averaging over fixed effects. Estimates deviate significantly more in the simulation case (Figure 4.3) when using a larger additive genetic variance.

As shown by the execution time of the methods in seconds (Table 4.3), the fastest technique (*No averaging*) is more than six times faster than the most computationally demanding (*Bayesian*). In the specific case of a binary probit model, the algorithm does not require numerical integration, meaning that all methods are sufficiently fast and we can use the method denoted by *Bayesian*. If we had considered other link functions without a closed form back-transformation such as the logit link, the fastest method might be the only viable choice.

In general, the choice of fixed effects is expected to substantially change the heritability estimate (Wilson 2008), in line with the results of the simulation with sufficiently high σ_A^2, but contradictory to our results for the application data. We fit our models in a Bayesian statistical context, and other modeling frameworks can yield different results. In addition, de Villemereuil (2020) shows a case study in which averaging changes the observation-scale heritability from about 0.3 without averaging, to 0.2 with averaging. Thus, we cannot state that one does not need the process of averaging over fixed effects in general. The results also show that for sufficiently large σ_A^2, the different densities of h_Ψ^2 will deviate more from one another (Figure 4.3b). Lastly, we have not tested different values for the hyperparameter w, the integration width in Algorithm 2.1. With a smaller w, the integrals are evaluated over a smaller space, which would yield more accurate results assuming it would not increase the numerical integration error. However, for the case of binary probit, w is not used and thus is not taken into account.

5.5 Further work

The robustness tests indicate that including fixed effects in the denominator may be appropriate. However, the tests do not clarify which cases it is necessary to include variance from fixed effects, seeing as the application data provides accurate estimates without including variance from fixed effect in the heritability estimation. Another shortcoming of the findings is to what extent we can generalize the results of the robustness tests. Although we have tested some extensions, the simulation datasets do not introduce complex animal models with several fixed effects. An alternative approach would be to fit two animal models without fixed effects, one with the data for $\boldsymbol{x}_{\text{sex}} = 0$, and the latter with $\boldsymbol{x}_{\text{sex}} = 1$. Then, one would expect, for a sufficiently large sample size, that the heritability would be similar.

Another aspect worth considering is modeling with a more unbalanced response variable. For example, looking at models whose binary trait is highly unbalanced ($p < 0.1$), could provide insight into limitations to the Gaussian model, as suggested in van Vleck (1972). Furthermore, some animal models may also have generous use of interaction terms in the model statements, which has neither been discussed nor included in this thesis. Also, note that a deliberate limit in the thesis was made by assuming that heritability is the most important metric when analyzing wild populations. However, heritability has been subject to scrutiny and some claim that other measurements related to additive genetic variance can be more appropriate, such as evolvability (Hansen et al. 2011).

Finally, in wild population studies, missing data tend to be more prevalent compared to the datasets used in this thesis. For the simulation case, we have an artificially complete pedigree. For our song sparrow data, we also have a complete pedigree that is not comparable to most wild population survey data. In a further study, it could be interesting to see how missing data and sensitivity to pedigree errors can influence Gaussian heritability estimates, for instance, by applying a capture-recapture animal model (Papaix et al. 2010).

Chapter 6

Conclusion

This thesis has explored the use of Gaussian models on binary phenotypes within the framework of the animal model and Bayesian statistics, directly obtaining observation scale heritability. The results, based on a dataset and simulation data, show that the Gaussian models can indeed be useful to easily obtain a heritability estimate, as a simpler and more direct alternative to binomial models with back-transformations. The results are limited to the datasets and models applied in the thesis and do not account for highly unbalanced phenotypes.

Furthermore, simulated data indicate that the introduction of overdispersion does not significantly challenge the Gaussian model. However, the heritability estimates from a Gaussian model diverge from the back-transformed probit model upon the introduction of fixed effects in the simulation study. Additionally, the simulation work has demonstrated that including the variance from a fixed effect in the denominator for heritability may be necessary to get an accurate heritability estimation in a Gaussian model. The models with the application data containing several fixed effects do not exhibit the same effect, and it seems like the Gaussian model provides estimates very close to the probit models' estimates. In general, using a Gaussian model also limits the use for further inference, as its general predictive abilities are significantly worse on binary phenotypes than a binomial model.

The findings show that a Gaussian mixed model is able to capture heritability on an observation scale that closely resembles the state-of-the-art back-transformation techniques for binomial models. Further investigation is required to determine exactly what effects decrease the power of a Gaussian model, although the results in this thesis indicate that it would be applicable in most practical cases. In the long term, the results can provide the foundation for developing a simpler, more effective, and easily interpretable statistical method for estimating additive genetic variance.

Bibliography

Aase, K. (2021). *Genetic group animal models in the genomics era* (Master's thesis). NTNU. https://hdl.handle.net/11250/2778383

Baker, T. A., Bell, S. P., Gann, A., Levine, M., Losick, R., & Inglis, C. (2008). *Molecular biology of the gene*. San Francisco, CA, USA:: Pearson/Benjamin Cummings.

Barton, N. H., Etheridge, A. M., & Véber, A. (2017). The infinitesimal model: Definition, derivation, and implications. *Theoretical population biology*, *118*, 50–73.

Bulmer, M. (1998). Galton's law of ancestral heredity. *Heredity*, *81*(5), 579–585.

Byers, D. (2008). Components of phenotypic variance. *Nature education*, *1*(1), 161.

Conn, P. B., Johnson, D. S., Williams, P. J., Melin, S. R., & Hooten, M. B. (2018). A guide to bayesian model checking for ecologists. *Ecological Monographs*, *88*(4), 526–542. https://doi.org/https://doi.org/10.1002/ecm.1314

Conner, J. K., Hartl, D. L., et al. (2004). *A primer of ecological genetics* (Vol. 425). Sinauer Associates Sunderland, MA.

de Villemereuil, P. (2020). How to use the qgglmm package? https://cran.r-project.org/package=QGglmm

de Villemereuil, P. (2018). Quantitative genetic methods depending on the nature of the phenotypic trait. *Annals of the New York Academy of Sciences*, *1422*(1), 29–47.

de Villemereuil, P., Gimenez, O., & Doligez, B. (2013). Comparing parent–offspring regression with frequentist and bayesian animal models to estimate heritability in wild populations: A simulation study for gaussian and binary traits. *Methods in Ecology and Evolution*, *4*(3), 260–275. https://doi.org/https://doi.org/10.1111/2041-210X.12011

de Villemereuil, P., Schielzeth, H., Nakagawa, S., & Morrissey, M. (2016). General methods for evolutionary quantitative genetic inference from generalized mixed models. *Genetics*, *204*(3), 1281–1294. https://doi.org/10.1534/genetics.115.186536

Dempster, E. R., & Lerner, I. M. (1950). Heritability of threshold characters. *Genetics*, *35*(2), 212.

Elston, R., Hill, W. G., & Smith, C. (1977). Query: Estimating "heritability" of a dichotomous trait. *Biometrics*, *33*(1), 231–236.

Fahrmeir, L., Kneib, T., Lang, S., & Marx, B. D. (2022). Regression models. In *Regression: Models, methods and applications* (pp. 23–84). Springer.

Falconer, D. S. (1996). *Introduction to quantitative genetics*. Pearson Education India.

Galton, F. (1886). Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, *15*, 246–263.

Gazzaniga, M. S., Heatherton, T. F., & Halpern, D. F. (2010). *Psychological science*. WW Norton New York.

Gorjanc, G., Henderson, D. A., with code contributions by Brian Kinghorn, & Percy, A. (2023). *Geneticsped: Pedigree and genetic relationship functions* [R package version 1.60.3]. http://rgenetics.org

Hadfield, J. D. (2010). Mcmc methods for multi-response generalized linear mixed models: The MCMCglmm R package. *Journal of Statistical Software*, *33*(2), 1–22. https://www.jstatsoft.org/v33/i02/

Hansen, T. F., Pélabon, C., & Houle, D. (2011). Heritability is not evolvability. *Evolutionary Biology*, *38*, 258–277.

Kalinowski, S. T., & Waples, R. S. (2002). Relationship of effective to census size in fluctuating populations. *Conservation Biology*, *16*(1), 129–136. https://doi.org/https://doi.org/10.1046/j.1523-1739.2002.00134.x

Kruuk, L. E. (2004). Estimating genetic parameters in natural populations using the 'animal model'. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, *359*(1446), 873–890.

Linney, Y., Murray, R., Peters, E., MacDonald, A., Rijsdijk, F., & Sham, P. (2003). A quantitative genetic analysis of schizotypal personality traits. *Psychological medicine*, *33*(5), 803–816.

Markowski, C. A., & Markowski, E. P. (1990). Conditions for the effectiveness of a preliminary test of variance. *The American Statistician*, *44*(4), 322–326. Retrieved April 7, 2023, from http://www.jstor.org/stable/2684360

Martino, S., & Paige, J. L. (2022). Lecture 11: Integrated Nested Laplace Approximation (INLA). *TMA4300*. https://www.math.ntnu.no/emner/TMA4300/2022v/Slides/Lecture11.pdf

Moore, D. S., & Shenk, D. (2017). The heritability fallacy. *Wiley Interdisciplinary Reviews: Cognitive Science*, *8*(1-2), e1400.

Morrison, K. (2017). *A gentle inla tutorial*. https://www.precision-analytics.ca/articles/a-gentle-inla-tutorial/

Nakagawa, S., Johnson, P. C., & Schielzeth, H. (2017). The coefficient of determination r 2 and intra-class correlation coefficient from generalized linear mixed-effects models revisited and expanded. *Journal of the Royal Society Interface*, *14*(134), 20170213.

Nakagawa, S., & Schielzeth, H. (2013). A general and simple method for obtaining r2 from generalized linear mixed-effects models. *Methods in ecology and evolution*, *4*(2), 133–142.

Nelson, R. M., Pettersson, M. E., & Carlborg, Ö. (2013). A century after fisher: Time for a new paradigm in quantitative genetics. *Trends in Genetics*, *29*(12), 669–676.

O'Loghlen, A. L., & Beecher, M. D. (1999). Mate, neighbour and stranger songs: A female song sparrow perspective. *Animal Behaviour*, *58*, 13–20.

Papaix, J., Cubaynes, S., Buoro, M., Charmantier, A., Perret, P., & Gimenez, O. (2010). Combining capture–recapture data and pedigree information to assess heritability of demographic parameters in the wild. *Journal of evolutionary biology*, *23*(10), 2176–2184.

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Reich, T., James, J. W., & Morris, C. A. (1972). The use of multiple thresholds in determining the mode of transmission of semi-continuous traits. *Annals of human genetics*, *36*(2), 163–184.

Reid, J. M., Arcese, P., Nietlisbach, P., Wolak, M. E., Muff, S., Dickel, L., & Keller, L. F. (2021). Immigration counter-acts local micro-evolution of a major fitness component: Migration-selection balance in free-living song sparrows. *Evolution letters*, *5*(1), 48–60.

Rekkebo, V. (2021). *Extending quantitative genetic models to estimate mutational variance* (Master's thesis). NTNU. https://hdl.handle.net/11250/2778398

Rue, H., Martino, S., & Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, *71*(2), 319–392.

Simpson, D., Rue, H., Riebler, A., Martins, T. G., & Sørbye, S. H. (2017). Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors. *Statistical Science*, *32*(1), 1–28. https://doi.org/10.1214/16-STS576

Smith, J. N., Keller, L. F., Marr, A. B., & Arcese, P. (2006). *Conservation and biology of small populations: The song sparrows of mandarte island*. Oxford University Press, USA.

Steinsland, I., & Jensen, H. (2010). Utilizing gaussian markov random field properties of bayesian animal models. *Biometrics*, *66*(3), 763–771.

Topp, C. N., Bray, A. L., Ellis, N. A., & Liu, Z. (2016). How can we harness quantitative genetic variation in crop root systems for agricultural improvement? *Journal of integrative plant biology*, *58*(3), 213–225.

van Vleck, L. D. (1972). Estimation of heritability of threshold characters. *Journal of Dairy Science*, *55*(2), 218–225.

Wang, X., Yue, Y. R., & Faraway, J. J. (2018). *Bayesian regression modeling with inla*. CRC Press.

Watson, J. D. (1970). *Molecular biology of the gene*. W.A. Benjamin.

Wilson, A. (2008). Why h2 does not always equal va/vp? *Journal of evolutionary biology*, *21*(3), 647–650.

Wolak, M. E. (2012). nadiv: An R package to create relatedness matrices for estimating non-additive genetic variances in animal models. *Methods in Ecology and Evolution*, *3*(5), 792–796.

Appendix A

Additional plots

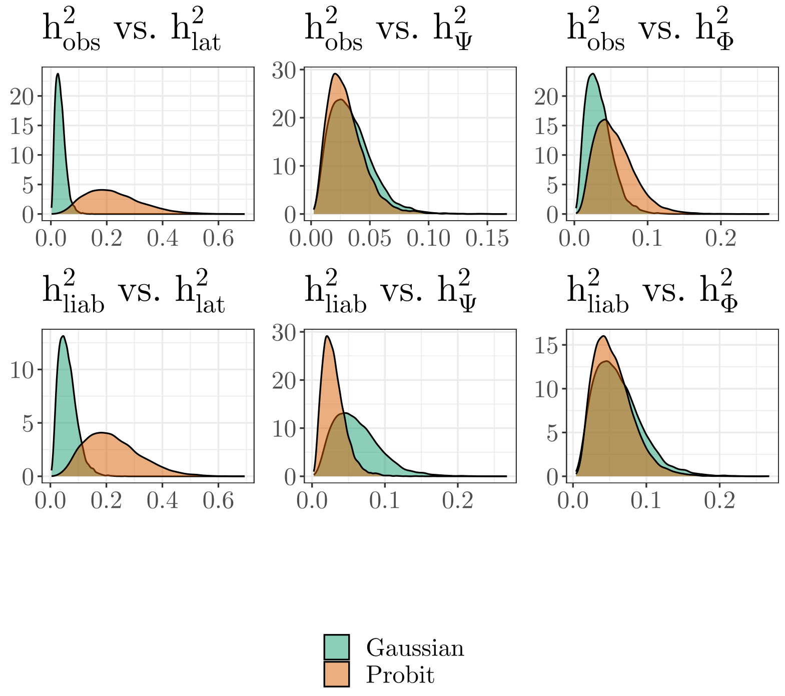


Figure A.1: Posterior distributions of estimated heritability for the Gaussian and probit model, in the application data. The densities are reported in a grid comparing each different type of scale to each other. That is, for both scales for the Gaussian model, we compare the posterior heritability with the probit model for its three different scales. The resulting 3×2 grid shows how they act comparatively.

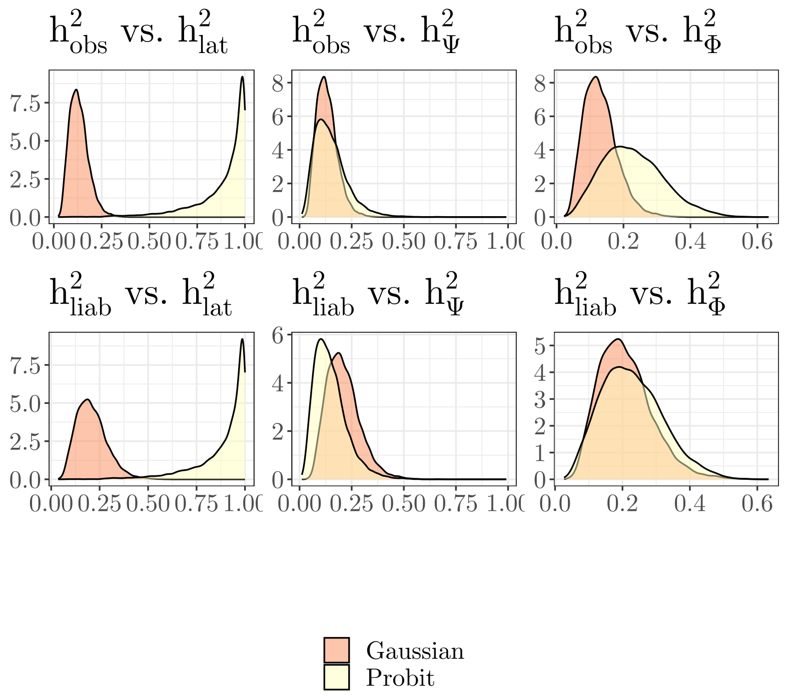


Figure A.2: Posterior distributions of estimated heritability for the Gaussian and probit model, in the simulation data with $\sigma_A^2 = 0.5$ and $\sigma_E^2 = 1$. The densities are reported in a grid comparing each different type of scale to each other. That is, for both scales for the Gaussian model, we compare the posterior heritability with the probit model for its three different scales. The resulting 3×2 grid shows how they act comparatively.

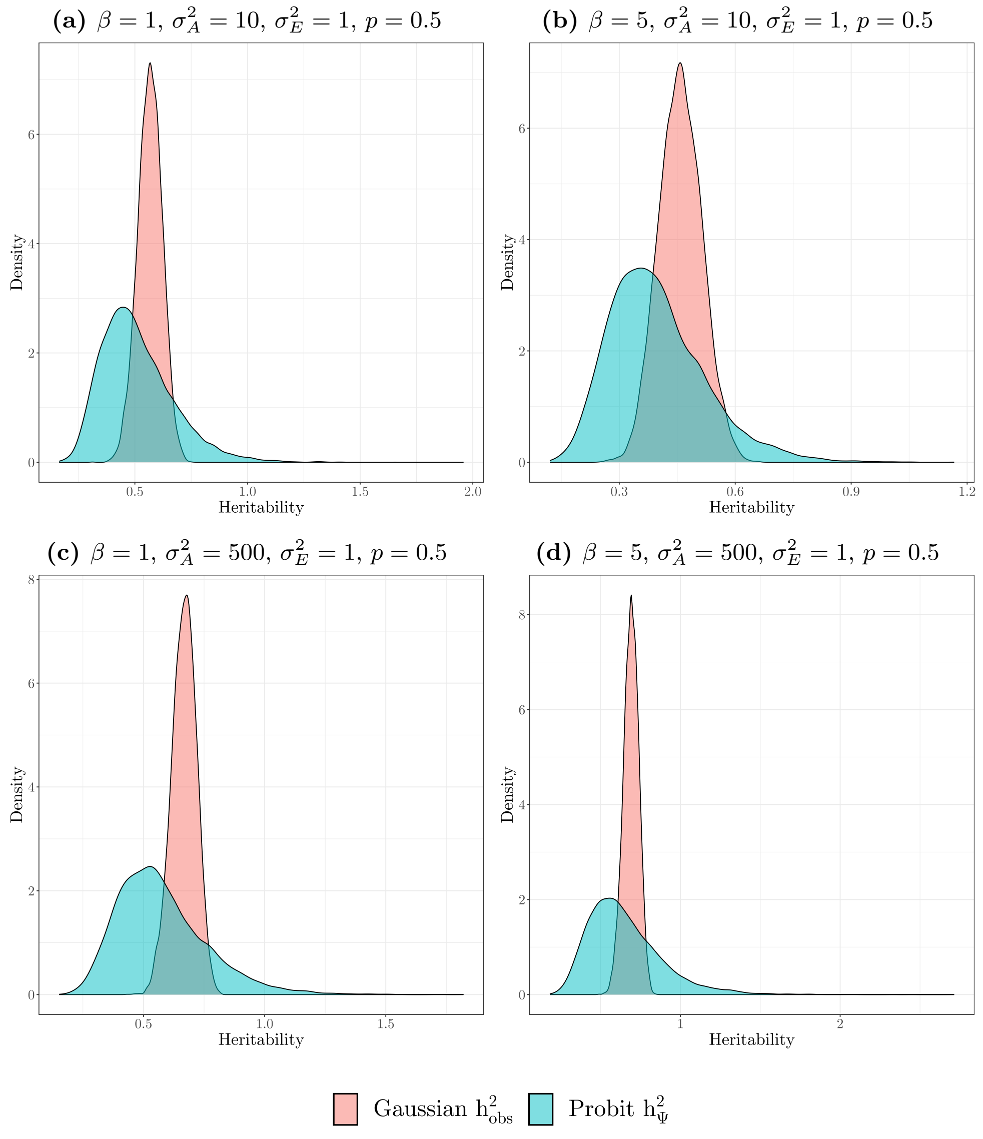


Figure A.3: Posterior heritability for simulations with fixed effects on observation-scale, varying on β between 1 and 5 across the columns, and between $\sigma_A^2 = 10$ and 500 across the rows.

Appendix B

Exploratory Data Analysis

Data frame overview

Pedigree

The first object is `d.ped` which contains the pedigree information.

```
summary(d.ped)
```

```
##     ninecode            gendam             gensire
##  Min.   :109137448   Min.   :109137468   Min.   :109137448
##  1st Qu.:146164012   1st Qu.:146130794   1st Qu.:146130313
##  Median :176124850   Median :176124382   Median :176124004
##  Mean   :196520240   Mean   :188116000   Mean   :185463038
##  3rd Qu.:243185045   3rd Qu.:226189260   3rd Qu.:226189228
##  Max.   :999999999   Max.   :999999999   Max.   :266176829
##                      NA's   :59          NA's   :59
```

It has columns *ninecode*, *gendam*, and *gensire*. The first column cannot be `NA` and is the unique identifier for an individual, whereas `gendam` and `gensire` are references (foreign keys) to the known maternal and paternal link, respectively. Both of these columns have 59 NAs. In fact, these NAs overlap completely since they are the founder population with no defined paternal or maternal link:

```
d.ped[is.na(d.ped$gendam), "gensire"]
```

```
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [51] NA NA NA NA NA NA NA NA NA
```

We see that *gensire* is NA for all instances where *gendam* is also NA. This is the founder population with no defined parental linkage.

d.Q

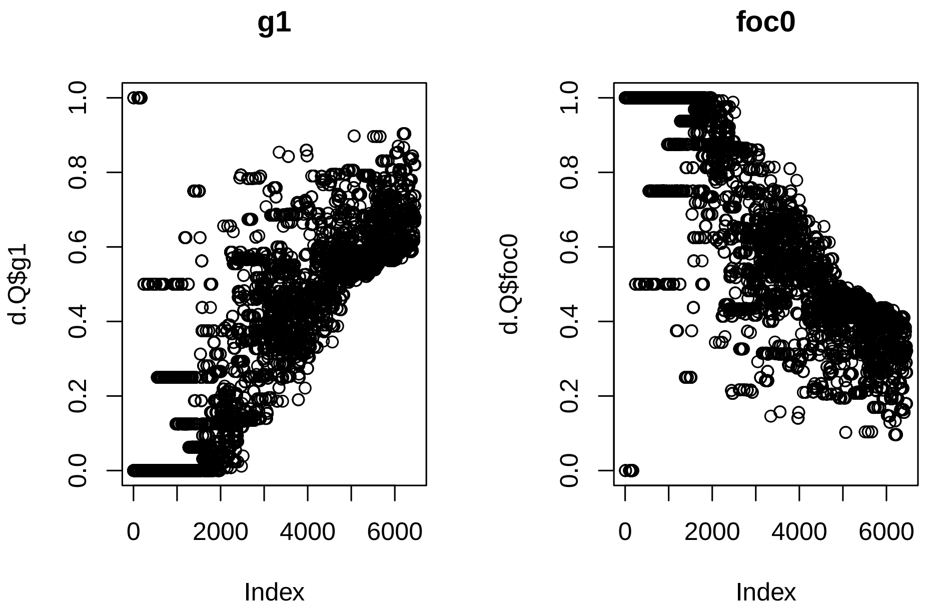
This table has the columns *g1*, *foc0* and *ninecode* (ID).

```
head(d.Q)
```

```
##   foc0 g1  ninecode
## 1    1  0 109137407
## 2    1  0 109137408
## 3    1  0 109137418
## 4    1  0 109137420
## 5    1  0 109137421
## 6    1  0 109137425
```

Considering only the first results, it might seem like `foc0` and `g1` are binary/categorical variables, but plotting the values across indices show that the order of the rows are structured so that they start at 1 and 0 respectively.

```
par(mfrow = c(1, 2))
plot(d.Q$g1, main = "g1")
plot(d.Q$foc0, main = "foc0")
```



We can also look at the correlation between these two values

```
cor(d.Q$foc0, d.Q$g1)
```

```
## [1] -1
```

Hence, we have a very strong negative correlation here. We can also look at the individuals whose ID were in the *founder population*:

```
founder_population.id <- d.ped[is.na(d.ped$gendam), "ninecode"]
table(d.Q[which(d.Q$ninecode %in% founder_population.id),
          c("foc0", "g1")])
```

```
##     g1
## foc0  0  1
##    0  0 33
##    1 26  0
```

The values seem to be relatively balanced between 0 and 1 in the founder population. This supports the idea that they measure the immigration contribution to the genetic composition of the individuals. All immigrant individuals are completely immigrant, have no pedigree and are thus part of the founder population. The latter are those who are the "initial" natives on the island, meaning that their values must be exactly zero.

ped.prune

This is a pruned pedigree, only considering the 1993-2018 observations but also combining the knowledge of the 1975-1992 observations into them.

qg.data.gg.ind

This object has the following shape:

```
head(qg.data.gg.inds)
```

```
##    ninecode natalyr sex.use nestrec surv.ind.to.ad brood.date sex.use.x1
## 1 111111112    2012       0    3086              0        120          1
## 2 111111121    2015       0    3237              0        141          1
## 3 143173366    1993       1    1838              1         96          1
## 4 143173381    1993       2    1867              1        102          2
## 5 143173382    1993       1    1867              0        102          1
## 6 143173384    1993       1    1851              0        102          1
##       f.coef      foc0        g1 natalyr.no sex
## 1 0.11155218 0.4085679 0.5914321         38   0
## 2 0.04814660 0.3299752 0.6700248         41   0
## 3 0.05108643 0.5283203 0.4716797         19   0
## 4 0.03125000 0.6250000 0.3750000         19   1
## 5 0.03417969 0.4335938 0.5664062         19   0
## 6 0.02148438 0.6328125 0.3671875         19   0
```

The response variable we will use is `surv.ind.to.ad`. Below are some elementary properties of the data.

```
## [1] "Earliest year: 1993"

## [1] "Number not survived: 1817" "Number survived: 661"

## [1] "natal year correlation: 1"
## [1] "correlation between sex and sex.x1: 0.842997540673555"
```

An overview of the columns:

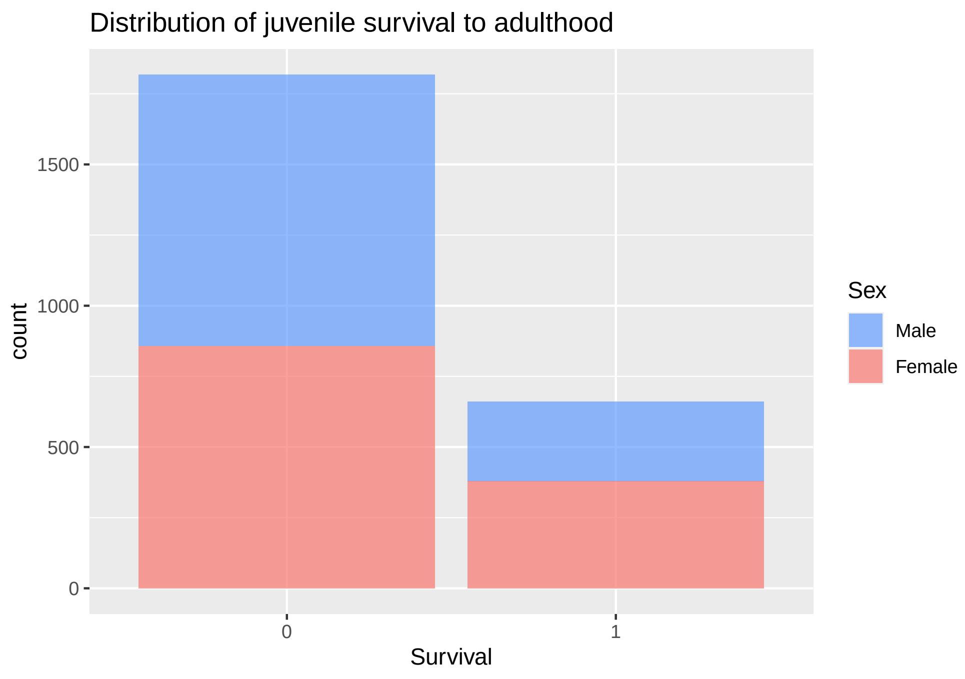
- *ninecode*: Individual ID
- *natalyr*: The year the individual was born, e.g. 2015.
- *sex.use*: **Not in use**
- *nestrec*: ID for nest number
- *brood.date*: Day of the year when the first offspring in individuals nest hatched
- *sex.use.x1*: Sex of individual, 1 or 2
- *f.coef*: Inbreeding coefficient
- *foc0*: "How foreign" individual is, related to `f.coef`
- *g1*: Inverse of *foc0*.
- *natalyr.no*: The same as the natal year, starting with 1974 as 0 (2015=41).

Vizualization of juvenile survival

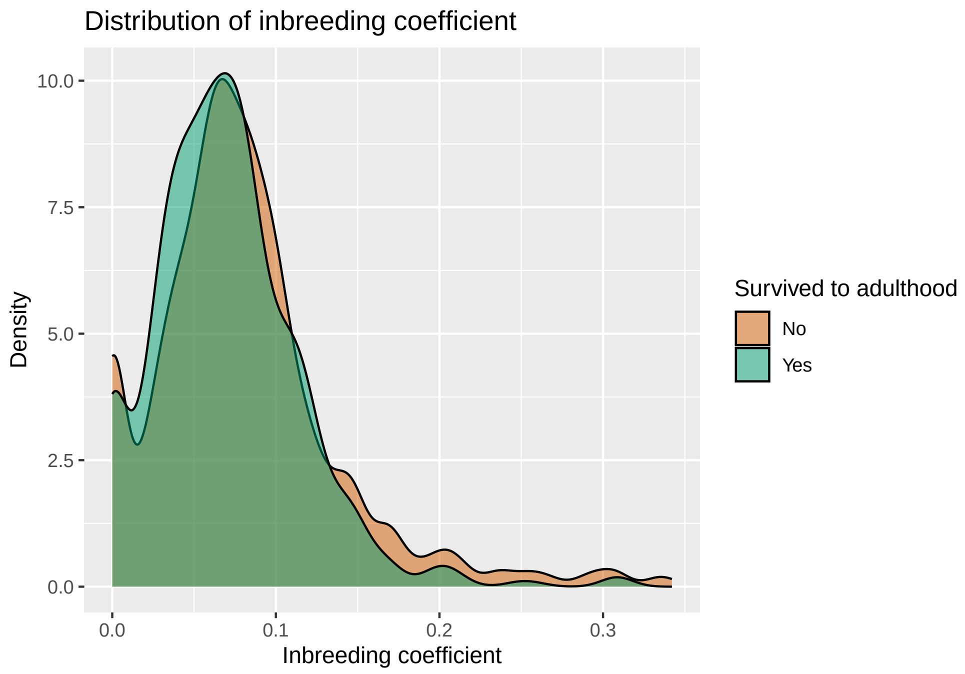
We will have a look at how the response, juvenile survival, relates to the other covariates in our data.

First, we look at sex:

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

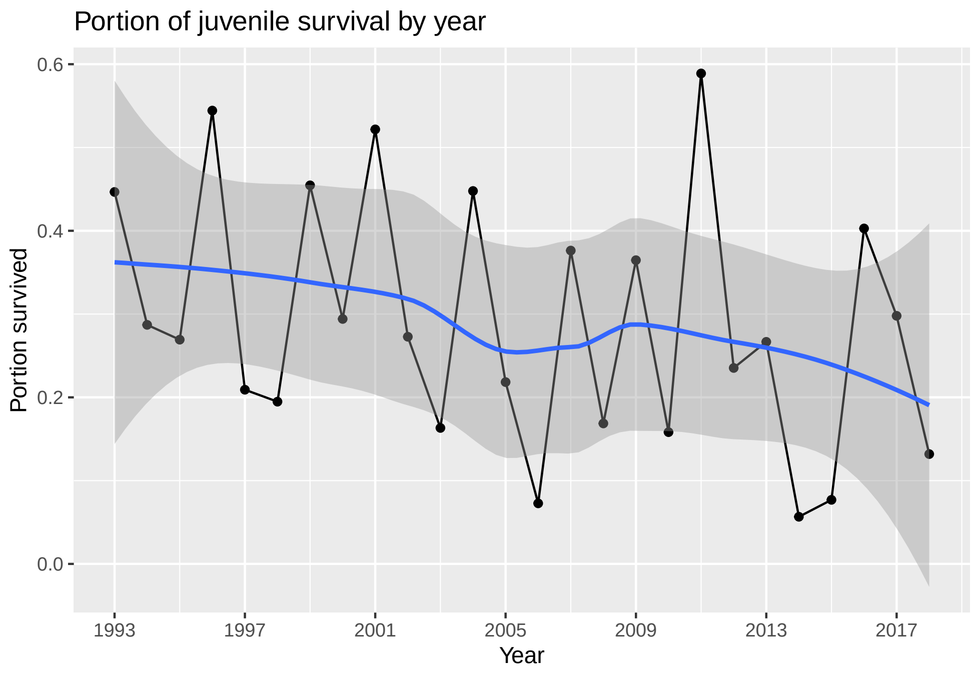


It seems like the sex in relation to survival is relatively balanced here. We can note that it seems like a larger portion of those surviving are females. Next, we examine the breeding coefficient.

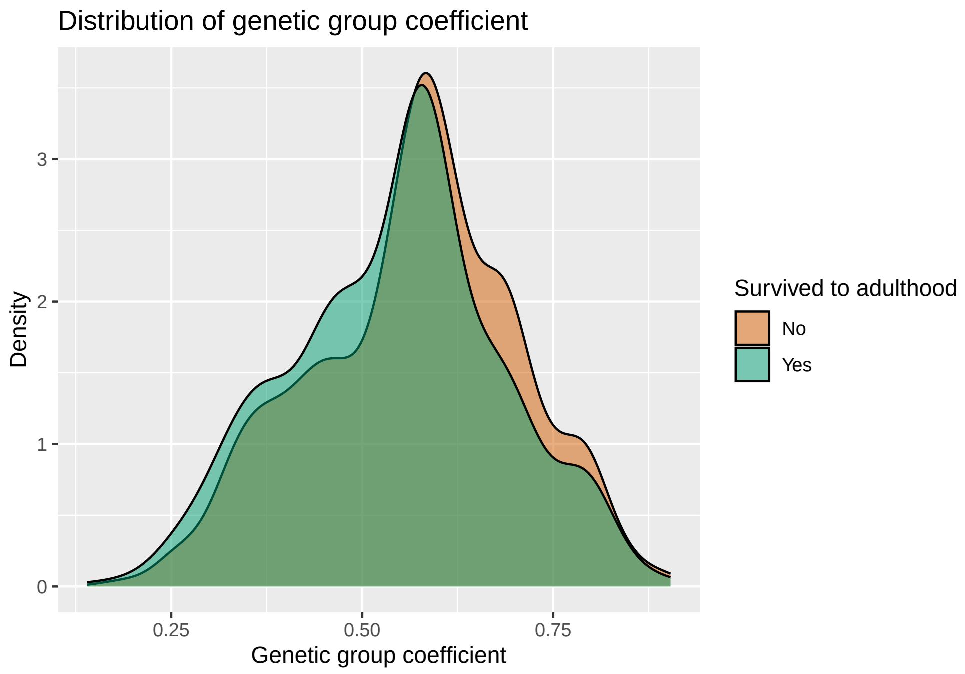


Here we see that survival is a bit more skewed toward lower inbreeding coefficients. We may also plot the proportion of individuals who survived over each year:

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

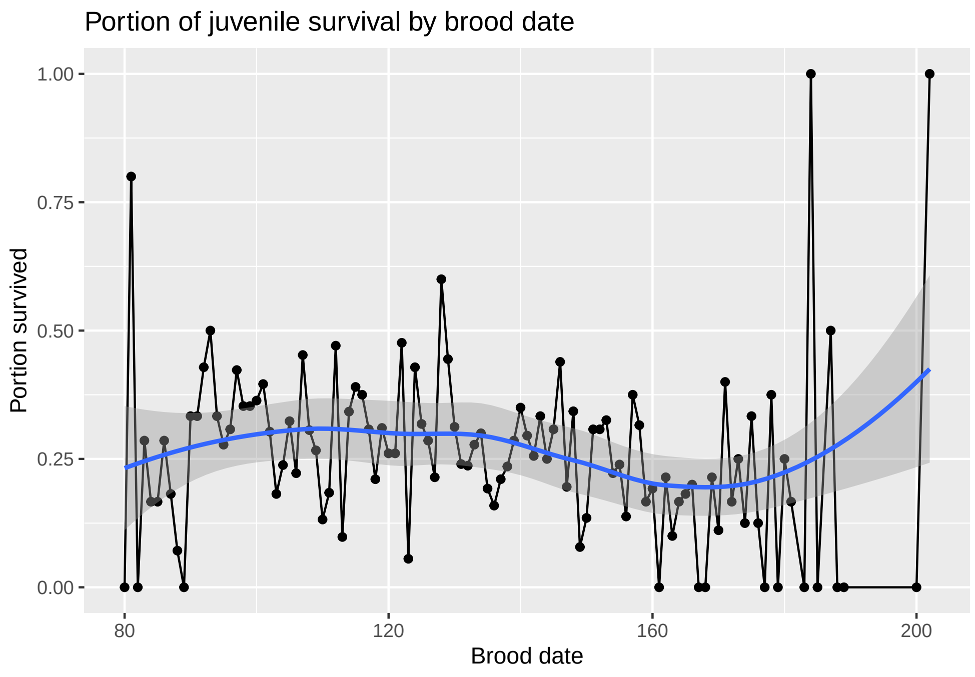


There seems to be very little trending over the years, but possibly a small negative trend. We also examine if there is some correspondence between genetic group coefficient (`g1`) and juvenile survival.



This shows a similar result to the inbreeding coefficient, namely a skew towards the right (lower values of coefficient) in the group that survived. Finally, we plot the survival probability based on brood date:

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



This last plot seems to indicate that survival is relatively stable and somewhat decreasing for those hatched relatively late. For the largest values of brood date, we get an increasing trend but also much uncertainty since not that many were hatched this late.

Appendix C

Patching `rbv`

How the bug arises

The function `rbv` generates random breeding values based on a pedigree. The pedigree must either be a data frame with columns *id, dam, sire*, or a `phylo` object. The issue arises when using GeneticsPed to generate the pedigree since this returns a multi-class object rather than just one single object.

Minimal viable product to reproduce issue

```
library(GeneticsPed)
```

```
## Warning: package 'GeneticsPed' was built under R version 4.2.2

## Loading required package: MASS

##
## Attaching package: 'GeneticsPed'

## The following object is masked from 'package:stats':
##
##     family
```

```
library(MCMCglmm)
```

```
## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 4.2.2

## Loading required package: coda

## Warning: package 'coda' was built under R version 4.2.2

## Loading required package: ape

## Warning: package 'ape' was built under R version 4.2.2
```

```r
ped0 <- generatePedigree(nId = 100, nGeneration = 9,
                         nFather = 50, nMother = 50)
pedigree <- ped0[ , c(1,3,2)]
names(pedigree) <- c("id", "dam", "sire")
attr(pedigree, "class") # Contains 2 classes
```

```
## [1] "Pedigree"   "data.frame"
```

Trying to use rbv will end in an error

```r
u <- rbv(pedigree, 0.4)
```

The error code is

```
Error in if (attr(pedigree, "class") == "phylo") { :
  the condition has length > 1
```

Patching the issue

The only line required to change is in `rbv.R`, and is the fifth line. Simply change it from

```
if(attr(pedigree, "class")=="phylo"){ped=FALSE}
```

and change it to:

```
if(any(attr(pedigree, "class")=="phylo)){ped=FALSE}
```

Then, rebuild the modified package. A patched tarball file is available on Github here.

Appendix D

R script

R code acknowledgements

The application dataset and a large portion of data preprocessing is provided by Jane Reid. The re-implementation into INLA is also largely based on the work from Stefanie Muff.

Data loading

We start by installing and importing required packages.

```r
req.packages <- c("BiocManager", "ggplot2", "latex2exp", "nadiv",
                  "QGglmm", "cowplot", "reshape2", "showtext")
to.install <- req.packages[
  is.na(match(req.packages, installed.packages()[,1]))
  ]
if(length(to.install) > 0L){
  install.packages(to.install)
}
for(pack in req.packages){
  suppressPackageStartupMessages(
    library(pack,character.only = TRUE, quietly = TRUE))
}

if (!require("GeneticsPed", quietly = TRUE)) {
  BiocManager::install("GeneticsPed")
}
if (!require("MCMCglmm", quietly = TRUE)) {
  # rbv patch
  install.packages("../MCMCglmm-rbv-patch.tar.gz")
}
if (!require("INLA", quietly = TRUE)) {
  install.packages(
    "INLA", repos=c(getOption("repos"),
                    INLA="https://inla.r-inla-download.org/R/stable"),
    dep=TRUE)

}
library(MCMCglmm)
library(MASS)
library(bdsmatrix)
library(INLA)

library(GeneticsPed) #

# Plotting libraries and settings

library(grid)
texfont <- "CMU Serif"
showtext_auto()
font.paths(file.path(
  Sys.getenv("LOCALAPPDATA"), "Microsoft",
  "Windows", "Fonts"
))
font_add(texfont, regular = "cmunrm.ttf")
theme_set(theme_bw() + theme(text = element_text(family = texfont,
                                                 size = 20)))

# Dataset import
qg.data.gg.inds <- read.table("../data/qg.data.gg.inds.steffi.txt",
  header = TRUE
)
d.ped <- read.table("../data/ped.prune.inds.steffi.txt",
  header = TRUE
)
d.Q <- read.table("../data/Q.data.steffi.txt", header = TRUE)

qg.data.gg.inds$natalyr.id <- qg.data.gg.inds$natalyr.no
```

Some global settings:

```
SAVE.PLOT <- TRUE
n.samples <- 10000
# iid N(0,1) noise in songsparrow formula:
FORMULA.EXTRA.IID.NOISE <- FALSE
```

Below we do a couple more preprocessing steps, namely:

- build a pedigree structure from the denormalized table with `prepPed`.
- assign new IDs to each individual starting at 1.
- keep a mapping record between the original IDs (ninecode) and the new 1-indexed IDs.
- use this mapping to transform the IDs for each individual's Dam and Sire to the same.
- compute the inverse of the relatedness matrix.

```
# Scale the continuous variances for stability
qg.data.gg.inds$f.coef.sc <- scale(qg.data.gg.inds$f.coef,
                                   scale = FALSE)
qg.data.gg.inds$g1.sc <- scale(qg.data.gg.inds$g1,
                               scale = FALSE)
qg.data.gg.inds$natalyr.no.sc <- scale(qg.data.gg.inds$natalyr.no,
                                       scale = FALSE)
qg.data.gg.inds$brood.date.sc <- scale(qg.data.gg.inds$brood.date,
                                       scale = FALSE)

# Binarize `sex` covariate
qg.data.gg.inds$sex <- qg.data.gg.inds$sex.use.x1 - 1
```

Deriving *A*

For INLA we need IDs that run from 1 to the number of individuals

```
d.ped <- nadiv::prepPed(d.ped)
d.ped$id <- seq_len(nrow(d.ped))

# Maps to keep track of the Ninecode to ID relations
d.map <- d.ped[, c("ninecode", "id")]
d.map$g1 <- d.Q[match(d.map$ninecode, d.Q$ninecode), "g1"]
d.map$foc0 <- d.Q[match(d.map$ninecode, d.Q$ninecode), "foc0"]

# Give mother and father the id
d.ped$mother.id <- d.map[match(d.ped$gendam, d.map$ninecode), "id"]
d.ped$father.id <- d.map[match(d.ped$gensire, d.map$ninecode), "id"]

# A can finally be constructed using `nadiv`
Cmatrix <- nadiv::makeAinv(
  d.ped[, c("id", "mother.id", "father.id")])$Ainv

# Stores ID twice (to allow for extra IID random effect)

qg.data.gg.inds$id <- d.map[
  match(qg.data.gg.inds$ninecode, d.map$ninecode),
  "id"
]
qg.data.gg.inds$u <- seq_len(nrow(qg.data.gg.inds))
```

INLA

The general INLA formula is provided below, where `f()` encode the random effect:

```
formula.inla.scaled <- surv.ind.to.ad ~ f.coef.sc + g1.sc +
  natalyr.no.sc + brood.date.sc + sex +
  f(nestrec, model = "iid", hyper = list(
    prec = list(
      initial = log(1 / 0.05),
      prior = "pc.prec",
      param = c(1, 0.05)
    ) # PC priors
  )) +
  f(natalyr.id, model = "iid", hyper = list(
    prec = list(
      initial = log(1 / 0.25),
      prior = "pc.prec",
      param = c(1, 0.05)
    ) # PC priors
  )) +
  f(id,
    model = "generic0", # Here we need to specify the covariance matrix
    Cmatrix = Cmatrix, #    via the inverse (Cmatrix)
    constr = FALSE,
    hyper = list(
      prec = list(initial = log(1 / 10), prior = "pc.prec",
                  param = c(1, 0.05))
    ) # PC priors
  )
if (FORMULA.EXTRA.IID.NOISE) {
  formula.inla.scaled <- update(
    formula.inla.scaled,
    ~ . + f(u,
      model = "iid", constr = TRUE,
      hyper = list(prec = list(
        initial = log(1),
        fixed = TRUE
      ))
    )
  )
}
```

Now we call INLA models. Note that we pass some control arguments to the function call. We compute DIC (Deviance information criterion) for all models with `dic` flag in `control.compute`. For the binomial models, we want to be able to use `QGglmm` and average over all fixed effects. This is done by supplying the *latent marginal predicted values*, which are not computed unless you pass the `return.marginals.predictor` flag set to true. We also want to set the `CPO` flag to true in the Gaussian model, so we can look a bit at its "residuals" (PIT values). Lastly, the `control.family` argument is used to pass the link functions for binomial models.

```
fit.inla.probit <- inla(
  formula = formula.inla.scaled, family = "binomial",
  data = qg.data.gg.inds,
  control.compute = list(dic = TRUE,
                         return.marginals.predictor = TRUE),
  control.family = list(link = "probit"),
)
```

```
fit.inla.gaussian <- inla(
  formula = formula.inla.scaled, family = "gaussian",
  data = qg.data.gg.inds,
  control.compute = list(dic = TRUE, cpo = TRUE)
)

data.frame(
  Gaussian = fit.inla.gaussian$dic$dic,
  Probit = fit.inla.probit$dic$dic,
  row.names = "Deviance Information Criteria"
)
```

Latent heritability

Below is a function to get h^2_{obs}, h^2_{lat} and h^2_{Φ} based on a fitted model.

```
get.h2 <- function(inla.fit, n, use.scale = FALSE, model = NA,
                   include.fixed = FALSE) {
  #' Get heritability
  #'
  #' Get n samples of heritability (h^2) from INLA object
  #' @param inla.fit fitted model
  #' @param n number of samples
  #' @param use.scale flag for adding link variance to denominator
  #' @param model string representation of model type
  #' @param include.fixed flag for including fixed effects variance
  #' @return n-sized vector of heritability samples
  samples <- inla.hyperpar.sample(n = n, inla.fit)
  denominator <- 0
  for (cname in colnames(samples)) {
    denominator <- denominator + 1 / samples[, cname]
  }
  if (include.fixed) {
    # Grab variance from summary object in INLA fit
    denominator <- denominator + sum(inla.fit$summary.fixed[, "sd"]^2)
  }

  if (use.scale) {
    scales.dictionary <- list(
      binom1.probit = 1, binom1.logit = pi^2 / 3,
      round = 0.25
    )
    scale.param <- get(model, scales.dictionary)
    denominator <- denominator + scale.param
  }

  h2.inla <- (1 / samples[, "Precision for id"]) / denominator
```

```r
  return(h2.inla)
}

threshold.scaling.param <- function(p) {
  # h^2_liab = threshold.scaling.param * h^2_obs
  p * (1 - p) / (dnorm(qnorm(p)))^2
}
```

Simulation data

We implement a general simulation method that, based on the passed arguments, generates simulation data and fits animal models onto the dichotomized response.

```r
simulated.heritability <- function(NeNc = 0.5, idgen = 100, nGen = 9,
                                   sigmaA = 0.8, linear.predictor = NA,
                                   simulated.formula = NA,
                                   dichotomize = "round",
                                   pc.prior = NA, probit.model = FALSE,
                                   simulated.formula.probit = NA,
                                   DIC = FALSE) {
  #' Simulate and fit animal model
  #'
  #' Generate pedigree, fit Gaussian (INLA) model and provide
  #' heritability estimate.
  #' @param NeNc Effective/Census population mean, used to determine
  #' number of fathers and mothers per generation,
  #' @param idgen Number of individuals per generation in pedigree
  #' @param nGen Number of generations in pedigree
  #' @param sigmaA:  Additive genetic variance
  #' @param linear.predictor Callable function of two parameters, the
  #' first 'u' (breeding values), the second for the data
  #' @param simulated.formula Formula expression using the response
  #' name `simulated.response`, param `id` and `Cmatrix`, both of
  #' which are defined locally in this method.
  #' @param dichotomize Dichotomization method (round, binomial, etc)
  #' @param pc.prior (optional) parameters for PC prior
  #' @param probit.model (optional) flag to fit binomial probit model
  #' in addition to the Gaussian model.
  #' @param simulated.formula.probit (opt) Formula for probit model
  #' @param DIC (optional) Flag for computing DIC for the models
  #'
  #' @return A list with the following items:
  #' heritability: Posterior latent heritability samples
  #' summary: List of mean, standard deviation and quantiles of h^2
  #' p: Portion of `TRUE` observations in simulated response
  #' simulated.response:  The observed values in simulation dataset
  #' fit: Gaussian fitted model
  #' fit.probit: Probit fitted model, if `probit.model=F`, is `NULL`.
  ped0 <- generatePedigree(
    nId = idgen, nGeneration = nGen,
    nFather = idgen * NeNc, nMother = idgen * NeNc
  )
  # Set correct format for pedigree
  pedigree <- ped0[, c(1, 3, 2, 5)]
  names(pedigree) <- c("id", "dam", "sire", "sex")

  # Generate random breeding values
  # The following will CRASH if you don't
  # use the patched MCMCglmm package!
  u <- rbv(pedigree[, c(1, 2, 3)], sigmaA)

  simulated.d.ped <- nadiv::prepPed(pedigree, gender = "sex")
  # Binarize from (1,2) to (0,1)
  simulated.d.ped$sex <- simulated.d.ped$sex - 1
  simulated.Cmatrix <- nadiv::makeAinv(pedigree[, c(1, 2, 3)])$Ainv
  # Make index to allow iid noise random effect
  simulated.d.ped$ind <- seq_len(nrow(simulated.d.ped))

  # Generating "true" y_i

  if (dichotomize == "binom1.logit") {
    simulated.response <- rbinom(length(u),
      size = 1,
      prob = pnorm(linear.predictor(u, simulated.d.ped))
    )
  } else if (dichotomize == "round") {
    # This assumes mean of \eta_i is 0
    simulated.response <- ifelse(
      linear.predictor(u, simulated.d.ped) <= 0, 0, 1
    )
  } else if (dichotomize == "round_balanced") {
    # Get balanced residuals for unbalanced linear predictor
    cutoff <- mean(linear.predictor(u, simulated.d.ped))
    simulated.response <- ifelse(
      linear.predictor(u, simulated.d.ped) <= cutoff, 0, 1
    )
  } else if (is.numeric(dichotomize)) {
    stopifnot(dichotomize >= 0 & dichotomize <= 1)
    eta_values <- linear.predictor(u, simulated.d.ped)
    cutoff <- quantile(eta_values, 1 - dichotomize)
    # e.g. dichotomize=0.1, p should be about 0.1
    simulated.response <- ifelse(eta_values <= cutoff, 0, 1)
  } else {
    stop(paste0(
      "Unknown dichotomization method '", dichotomize, "'. ",
      "Consider using 'binom1.logit' or 'round'."
    ))
  }
```

```r
  p <- mean(simulated.response) # portion of true responses


  # Model fitting LMM for binary trait
  # First reload formula environment to access local variables
  environment(simulated.formula) <- environment()
  environment(simulated.formula.probit) <- environment()

  simulated.fit.inla <- inla(
    formula = simulated.formula, family = "gaussian",
    data = simulated.d.ped, control.compute = list(dic = DIC)
  )

  # Checks for error status in INLA fit,
  if (simulated.fit.inla$mode$mode.status != 0) {
    cat("\n[WARNING], INLA status",
        simulated.fit.inla$mode$mode.status, "\n")
  }
  heritability <- get.h2(simulated.fit.inla, 10000)

  # Also fit probit if specified
  if (probit.model) {
    fit.probit <- inla(
      formula = simulated.formula.probit, family = "binomial",
      data = simulated.d.ped,
      control.compute = list(return.marginals.predictor = TRUE,
                             dic = DIC)
    )
  } else {
    fit.probit <- NULL
  }
  list(
    heritability = heritability,
    summary = list(
      mean = mean(heritability),
      standard.deviation = sd(heritability),
      quantiles = quantile(heritability, probs = c(0.025, 0.5, 0.975))
    ),
    p = p,
    simulated.response = simulated.response,
    fit = simulated.fit.inla,
    fit.probit = fit.probit
  )
}
```

Now we try to run it through the model pipeline. Here we try $\eta_i = u_i + e_i$ (extra iid effect) and $y_i = u_i + \varepsilon_i$.

```r
simulated.formula <- simulated.response ~  f(id,
  model = "generic0",
  Cmatrix = simulated.Cmatrix,
  constr = FALSE,
  hyper = list(
    prec = list(initial = log(1 / 10), prior = "pc.prec",
                param = c(1, 0.05))
  ))
simulated.formula.probit <- simulated.response ~  f(id,
  model = "generic0",
  Cmatrix = simulated.Cmatrix,
  constr = FALSE,
  hyper = list(
    prec = list(initial = log(1 / 10), prior = "pc.prec",
                param = c(1, 0.05))
  )) + f(ind, model="iid", hyper=list(prec=list(
    prior="pc.prec", param=c(1,0.05))))

get.simulated.threshold.value <- function(
    NeNc, idgen, nGen, sigmaA, linear.predictor, simulated.formula,
    Vp = NA) {
  #' Get h^2 statistics from simulation
  #'
  #' Makes a dataframe showing observation h^2 and
  #' liability h^2 with 95% confidence interval. Most parameters are
  #' for generating pedigree and not covered in this docstring.
  #' @param Vp Total phenotypic variance, used to get "true" h^2
  #' @return A dataframe with said statistics
  stopifnot("Vp required to get true h^2" = !is.na(Vp))
  sim.result <- simulated.heritability(
    NeNc, idgen, nGen, sigmaA,
    linear.predictor, simulated.formula
  )
  threshold.scaled.h2 <- threshold.scaling.param(sim.result$p) *
    sim.result$heritability
  simulation.h2.true <- sigmaA / (Vp)

  data.frame(
    Simulation = c(
      simulation.h2.true, mean(threshold.scaled.h2),
      paste0("(", paste(format(
        quantile(threshold.scaled.h2, probs = c(0.025, 0.975)),
        digits = 4
      ), collapse = ", "), ")"),
      mean(sim.result$summary$mean)
    ),
    row.names = c(
      "True h^2", "Estimated h^2_obs, mean",
      "95% Confidence interval", "Estimated latent mean"
    )
  )
}

get.simulated.threshold.value(
  NeNc = 0.5, idgen = 100, nGen = 9, sigmaA = 0.05,
  function(u, .) {
    u + rnorm(length(u))
  },
  simulated.formula,
  Vp = 0.05 + 1
)
```

Further, we quantitatively look into estimates for different σ_A^2:

Performance over varying V_A

```
plot.h2.deviation <- function(
    dichotomize = "round",
    title = "Simulation heritability",
    SAVE.PLOT = TRUE, plot.fn = NA, sigma.scale = "log",
    lin.pred = NULL,
    dynamic.priors = FALSE, simulated.formula = NULL, Ve = NULL,
    fixedeffects = FALSE) {
  #' Plot h^2 estimate, alongside true value, for a series of V_A
  #'
  #' For each V_A, generate simulation and fit a Gaussian model.
  #' Then, plot the obtained h^2 for observation and liability scale,
  #' alongside the true value.
  #' @param dichotomize Dichotomization method, used for simulation
  #' @param title ggplot legend title used as title for all (sub)plots
  #' @param SAVE.PLOT flag for saving plot to disk
  #' @param plot.fn (optional) String to add to the end of the
  #' filename, before file extension, when saving the plot.
  #' @param sigma.scale either "log" or "small", deciding what values,
  #' and the spacing between values of V_A to be iterated over.
  #' @param lin.pred linear predictor for simulation
  #' @param dynamic.priors Flag for changing model priors based on V_A
  #' @param simulated.formula Formula for simulatin
  #' @param Ve Residual variance, or fixed effects variance
  #' for that model
  #' @param fixedeffects Flag to determine if model has fixed effects
  #' @return List with item "p" for the ggplot object.

  if (sigma.scale == "log") {
    sigmaA.list <- c(1:10 %o% 10^(-3:3)) # Log scale[10^-3,  10^3]
  } else if (sigma.scale == "small") { # Linear scale [10^-3, 0.259]
    sigmaA.list <- seq(0.001, 0.26, by = 0.01)
  } else {
    stop("Unrecognized scale for sigmaA.")
  }
  pc.U.list <- c(rep(10, 10) %o% 10^(-3:3))
  # pc.U.list <- 2*sigmaA.list # Alternative dynamic prior

  estimates <- c()
  latent <- c()
  true.vals <- c()
  est.CI.u <- c()
  est.CI.l <- c()
  plist <- c()
  iter.num <- 0
  Ve0 <- Ve
  for (sigmaA in sigmaA.list) {
    cat(">")
    if (dynamic.priors) {
      iter.num <- iter.num + 1
      if (sigmaA < 1) {
        pc.prior <- c(1, 0.05)
      } else {
        pc.prior <- c(pc.U.list[iter.num], 0.05)
      }
    } else {
      pc.prior <- c(1, 0.05)
    }
    if (is.null(simulated.formula)) { # Defaults eta = a_i + e
      simulated.formula <- simulated.response ~  f(id,
        model = "generic0",
        Cmatrix = simulated.Cmatrix,
        constr = FALSE,
        hyper = list(
          prec = list(initial = log(sigmaA), prior = "pc.prec",
                      param = pc.prior)
        )
      )
    }
    if (is.null(lin.pred)) { # The 'usual' linear predictor
      lin.pred <- function(u, .) u + rnorm(length(u))
    }
    result <- simulated.heritability(
      NeNc = 0.5, idgen = 100, nGen = 9, sigmaA = sigmaA,
      linear.predictor = lin.pred,
      simulated.formula = simulated.formula,
      dichotomize = dichotomize,
      pc.prior = pc.prior
    )

    if (is.null(Ve)) {
      # Fallback residual variance
      Ve <- 1
    }
    if (fixedeffects) {
      # beta^2 * Var(x_fixedeffect):
      Ve <- Ve * var(result$simulated.response)
      posterior <- get.h2(result$fit, 10000, include.fixed = TRUE)
    } else {
      posterior <- result$heritability
    }

    simulation.h2.true <- sigmaA / (sigmaA + Ve)

    latent <- c(latent, mean(posterior)) # Observatoin-level
    threshold.scaled.h2 <- threshold.scaling.param(result$p)*posterior

    true.vals <- c(true.vals, simulation.h2.true)
    estimates.CI <- quantile(threshold.scaled.h2,
                             probs = c(0.025, 0.975))
    estimates <- c(estimates, mean(threshold.scaled.h2))
    est.CI.l <- c(est.CI.l, estimates.CI[1])
    est.CI.u <- c(est.CI.u, estimates.CI[2])
    plist <- c(plist, result$p)
    # Reset Ve
    Ve <- Ve0
  }
  res <- data.frame(
    estimates = estimates,
    true.vals = true.vals, latent = latent,
    est.CI.l = est.CI.l, est.CI.u = est.CI.u,
    sigmaA = sigmaA.list,
    plist = plist
  )
  # Plotting
  p <- ggplot(data = res, aes(x = sigmaA)) +
    geom_ribbon(aes(ymin = est.CI.l, ymax = est.CI.u), alpha = 0.1) +
    geom_line(aes(y = true.vals, color = "atrue"), size = 1.5) +
    geom_line(aes(y = estimates, color = "bliab"), size = 1.5) +
    geom_line(aes(y = latent, color = "obs"), size = 1.5) +
    xlab(TeX("$\\sigma_A^2$")) +
    ylab(TeX("$h^2$")) +
    scale_x_log10() +
    scale_color_manual(
      name = title,
      values = c(
        "atrue" = "darkred", "bliab" = "steelblue",
        "obs" = "chartreuse3"
      ),
      labels = c(
        expression("True " * h["liab"]^2),
        expression("Fitted " * h["liab"]^2),
        expression("Fitted " * h["obs"]^2)
      )
    ) +
    theme(text = element_text(size = 18), legend.text.align = 0)
  if (SAVE.PLOT) {
    ggsave(
      paste0(
        "../figures/simulation_deviance_",
        if (!is.na(plot.fn)) plot.fn, ".pdf"
      ), p + theme(legend.position = "none"),
      width = 20, height = 20,
      units = "cm"
    )
    # Save legend as separate plot
    p.legend <- cowplot::get_legend(p)
    pdf(paste0(
      "../figures/simulation_deviance",
      if (fixedeffects) "_fixedeffects", "_legend.pdf"
    ), width = 7.87402, height = 7.87402)
    grid.newpage()
    grid.draw(p.legend)
    dev.off()
  }
  return(list(p = p))
}
```

We also implement a method to average over several runs, reducing stochasticity. This should be ran on a remote node rather than locally, as just one run takes several minutes.

```
multiple.h2.dev <- function(sigma.scale, ntimes,
                            title = "Simulation heritability", ...){
  #' Run plot_h2_deviation `ntimes`
  #'
  #' Repeats the runs several times, and outputs error plot with
  #' mean +- SD
  #' @param sigma.scale Sigma values to test, either 'log' or 'small'
  #' @param ntimes Number of repeated runs
  #' @param title Legend title, charcter
  #' @param ... Additional parameters for plot_h2_deviation
  #' @return Dataframe with all info to plot the errorplots

  if (sigma.scale == "log") {
    sigmaA.list <- c(1:10 %o% 10^(-3:3))
  } else if (sigma.scale == "small") {
    sigmaA.list <- seq(0.001, 0.26, by = 0.01)
  } else {
    stop("Unrecognized scale for sigmaA.")
  }
  n.sigma <- length(sigmaA.list)
  # Initialize containers: each col is one run
  all.h2obs <- matrix(ncol = ntimes, nrow = n.sigma)
  all.h2liab <- matrix(ncol = ntimes, nrow = n.sigma)
  all.truevals <- matrix(ncol = ntimes, nrow = n.sigma)
  for(i in 1:ntimes){
    cat(paste0("\n [Run ", i ,"/", ntimes, "]\n"))
    res <- plot.h2.deviation(SAVE.PLOT = FALSE,
                             sigma.scale = sigma.scale, ...)
    cat("Latent:", res$p$data$latent)
    all.h2obs[, i] <- res$p$data$latent
    all.h2liab[, i] <- res$p$data$estimate
    all.truevals[, i] <- res$p$data$true.vals
  }
  plot.data <- data.frame(
    model = rep(c("h2obs", "h2liab", "true"),
                each = n.sigma, times = 2),
    top = c(rowMeans(all.h2obs) + apply(all.h2obs, 1, sd),
            rowMeans(all.h2liab) + apply(all.h2liab, 1, sd),
            rowMeans(all.truevals) + apply(all.truevals, 1, sd)
    ),
    mid = c(rowMeans(all.h2obs), rowMeans(all.h2liab),
            rowMeans(all.truevals)
    ),
    btm = c(rowMeans(all.h2obs) - apply(all.h2obs, 1, sd),
            rowMeans(all.h2liab) - apply(all.h2liab, 1, sd),
            rowMeans(all.truevals) - apply(all.truevals, 1, sd)
    ),
    xax = rep(sigmaA.list, times=3)
  )
  return(plot.data)
}
```

We test out a single run over several values below.

```
### Plots for sigmaA in (10^-3, 10^3)

simulation.res <- plot.h2.deviation(
  plot.fn = "round",
  SAVE.PLOT = TRUE, dynamic.priors = TRUE
)
simulation.res$p

### Plot for small values og sigmaA, but finer grid
simulation2.res <- plot.h2.deviation(
  SAVE.PLOT = TRUE, sigma.scale = "small",
  plot.fn = "small", dynamic.priors = TRUE
)
simulation2.res$p
```

```
### Standard plotting with different dichotomization techniques
plot.h2.deviation(
  dichotomize = "binom1.logit",
  title = "Simulation heritability",
  plot.fn = "binom", dynamic.priors = TRUE
)
```

If multiple runs have been done on remote server, loads and plots the results here,

```
# Multiple h^2 deviation plotter
markov.plotter <- function(df, legend.name=""){
  ggplot(df, aes(x=xax)) +
    geom_pointrange(aes(ymax=top, ymin=btm, y=mid, color=model)) +
    geom_line(aes(y=mid, color=model)) +
    xlab(TeX("$\\sigma_A^2$")) +
    ylab(TeX("$h^2$")) +
    scale_x_log10() +
    scale_color_manual(
      name = legend.name,
      values = c(
        "true" = "darkred", "h2liab" = "steelblue",
        "h2obs" = "chartreuse3"
      ),
      labels = c(
        expression("Fitted " * h["liab"]^2),
        expression("Fitted " * h["obs"]^2),
        expression("True " * h["liab"]^2)
      )
    ) +
    theme(text = element_text(size = 18), legend.position = "none")
}
load("markovh2dev_50_runs.Rdata")
mp1 <- markov.plotter(markov.result1)
mp2 <- markov.plotter(markov.result2)
mp3 <- markov.plotter(markov.result3,
                      legend.name="Simulation heritability")
ggsave("../figures/simulation_deviance_round.pdf", mp1)
ggsave("../figures/simulation_deviance_small.pdf", mp2)
ggsave("../figures/simulation_deviance_binom.pdf", mp3)
# Store legend separately
mp3.legend <- cowplot::get_legend(
  mp3 +theme(legend.position = "right", legend.text.align = 0))
pdf("../figures/simulation_deviance_legend.pdf",
    width = 7.87402, height = 7.87402)
grid.newpage()
grid.draw(mp3.legend)
dev.off()
```

Without residual in linear predictor

We also check what happens if we rerun with $\eta_i = a_i$, i.e. without residuals on underlying scale. Similar results.

```r
get.simulated.threshold.value(
  0.5, 100, 9, 10, function(u, .) u,
  simulated.formula, 10
)
```

Now we want to look into how **A** (the relatedness matrix) looks like.

```r
plot.A.matrix <- function(pedigree, title.append = NULL) {
  #' Plot histogram of relatedness from pedigree
  #'
  #' Compute relatedness matrix from pedigree and output
  #' histogram of its off-diagonal values.
  #' @param pedigree Pedigree dataframe/object
  #' @param title.append (optional) extra text in plot title
  #' @return ggplot object of histogram
  A.matrix <- nadiv::makeA(pedigree)
  A.diag <- diag(A.matrix)
  A.nondiag <- A.matrix
  diag(A.nondiag) <- NA
  ggplot(data = data.frame(values = A.nondiag@x)) +
    geom_histogram(aes(x = values, y = ..density..),
                   binwidth = 0.005) +
    ylim(c(0, 9)) +
    xlim(c(0, 0.4)) +
    labs(
      x = "Relatedness value", y = "Density",
      title = paste0("Off-diagonal values", title.append)
    )
}

# For the simulation data:
ped0 <- generatePedigree(
  nId = 100, nGeneration = 24,
  nFather = 0.5 * 100, nMother = 0.5 * 100
)
pedigree <- ped0[, c(1, 3, 2)]
names(pedigree) <- c("id", "dam", "sire")
simulated.d.ped <- nadiv::prepPed(pedigree)
plot.A.matrix(simulated.d.ped,
              title.append = ", 24 generation simulation")
if (SAVE.PLOT) {
  ggsave("../figures/relatedness-offdiagonal-sim.pdf",
    width = 20, height = 20, units = "cm"
  )
}
# Song sparrow data
```

```
plot.A.matrix(d.ped[, c("id", "mother.id", "father.id")],
              ", Song sparrow data")
if (SAVE.PLOT) {
  ggsave("../figures/relatedness-offdiagonal-songsparrow.pdf",
    width = 20, height = 20, units = "cm"
  )
}
```

Residual analyses on Gaussian model

- The first plot are the sorted PIT values over quantiles, analogous to a Q-Q plot in frequentist data. It shows a clear non-linear trend but rather a sigmoid-like curve.
- The second plot shows the PIT values across the different posterior fitted value means. Here we expect no clear pattern for well-behaved models, which is not the case in our model.
- The third plot is the residuals $y_i - \hat{y}_i$ with 95% credible interval. Here, we see a clear separation of those

```
pit.g <- fit.inla.gaussian$cpo$pit # PIT-values

# <Plot 1> Analogous to QQ-plot so should be linear
# --------
ggplot(data = data.frame(
  Quantiles = seq_along(1:length(pit.g)) / (length(pit.g) + 1),
  PIT = sort(pit.g)
)) +
  geom_point(aes(x = Quantiles, y = PIT)) +
  ggtitle("Sorted PIT values for Gaussian model") +
  theme(text = element_text(size = 14))
if (SAVE.PLOT) ggsave("../figures/PIT-sorted.pdf")

# <Plot 2> Posterior mean fitted values as a function of PIT values
# --------    analagous to "Residuals vs fitted"

ggplot(
  cbind(fit.inla.gaussian$summary.fitted.values, pit.g),
  aes(x = mean, y = pit.g)
) +
  geom_point() +
  geom_smooth() +
  labs(
    title = "PIT values over posterior mean fitted values",
    x = "Posterior fitted values (mean)",
    y = "PIT value"
  ) +
  theme(text = element_text(size = 14))
if (SAVE.PLOT) ggsave("../figures/PIT-over-fitted.pdf")
```

```
# <Plot 3> Plot of 'residuals', i.e. difference in true data and the
# --------    mean of the fitted values
df.resid <- qg.data.gg.inds$surv.ind.to.ad -
  fit.inla.gaussian$summary.fitted.values
rownames(df.resid) <- seq_len(nrow(df.resid))
df.resid$class <- qg.data.gg.inds$surv.ind.to.ad

ggplot(
  data = df.resid,
  aes(
    x = as.numeric(row.names(df.resid)),
    y = mean, color = factor(class)
  )
) +
  geom_errorbar(aes(ymin = `0.025quant`, ymax = `0.975quant`),
    color = "darkgrey"
  ) +
  geom_point() +
  scale_color_manual(
    name = "Juvenile survival",
    values = c("darkred", "steelblue")
  ) +
  labs(title = "Residuals of Gaussian model", x = "Index",
       y = "Residuals") + theme(text=element_text(size=14))
if (SAVE.PLOT) ggsave("../figures/Residuals-gaussian.pdf")
```

Transformations of heritability

Before developing methods for transformed heritability, we need to be able to sample from the marginal fitted values on latent scale.

```
marginal.latent.mode <- function(fit) {
  #' Helper function
  #'
  #' Get mode for each marginal linear predictor in `fit`
  #' @param fit Fitted INLA object
  #' @return vector of modes
  modes <- c()
  iter <- 1
  for (predictor in names(fit$marginals.linear.predictor)) {
    xy <- get(predictor, fit$marginals.linear.predictor)
    modes[iter] <- xy[, "x"][which.max(xy[, "y"])]
    iter <- iter + 1
  }
  modes
}
```

```r
marginal.latent.samples <- function(fit, nsamples) {
  #' Sample values from each predictor in fit
  #'
  #' Rather than only using mode for each predictor, we use samples
  #' from its posterior.
  #' @param fit Fitted INLA object
  #' @param nsamples Number of samples
  #' @return A list of `nsamples` elements, with each element in the
  #' list being a vector of the predictor size
  #' (i.e., number of observations in data)
  out.transpose <- matrix(
    nrow = nsamples,
    ncol = length(fit$marginals.linear.predictor)
  )
  for (i in seq_along(fit$marginals.linear.predictor)) {
    xy <- get(
      names(fit$marginals.linear.predictor)[i],
      fit$marginals.linear.predictor
    )
    out.transpose[, i] <- inla.rmarginal(nsamples, xy)
  }

  # We want list where each list element is one sample (transposed)
  out <- list()
  for (i in 1:nsamples) {
    out[[i]] <- out.transpose[i, ]
  }
  out
}


report.max.skewness <- function(posterior) {
  #' Get skewness for all predictors
  #'
  #' Computes skewness and prints maximum and minimum skew with index
  #' @param posterior list of predictors, assumed form [x, y]
  #' @return None (invisible `NULL`)
  library(e1071)
  iter <- 1
  posterior.skews <- c()
  for (predictor in names(posterior)) {
    posterior.skews[iter] <- skewness(get(
      predictor, posterior)[, "x"])
    iter <- iter + 1
  }
  cat(
    "Minimum skew for list no.", which.min(posterior.skews),
  "with skewness",
    min(posterior.skews), "and max for list no.",
  which.max(posterior.skews),
    "with skewness", max(posterior.skews), ".\n"
  )
}
```

We can now define methods to obtain heritability on the different scales. The first function computes h^2 on latent scale, or using the direct transformation by including link variance in denominator. The second method is more comprehensive and uses the library `QGglmm` to obtain estimates on data scale. So far we've only used `QGglmm` without averaging over fixed effects. This takes considerably more time to process, but we still do that one time to compare the results. Then we compare the heritability on four different scales

- Using 10k samples from the *marginal linear predictor*, and using this to average over
- Grabbing the mode for each *marginal linear predictor*, and passing the mode for each 10k sample
- No averaging, i.e. use intercept value instead.
- Use direct scaling method with link variance.

```
get.h2.from.qgparams <- function(inla.fit,
                                 modelname,
                                 n,
                                 averaging = FALSE,
                                 averaging.mode.only = FALSE) {
  #' Get heritability using QGglmm::QGParams()
  #'
  #' Computes a posterior of data-scale heritability using QGParams
  #' @param inla.fit Fitted INLA model
  #' @param modelname string specifying model type
  #' @param n Number of samples for posterior distribution
  #' @param averaging Flag for averaging over fixed effects
  #' @param averaging.mode.only Other flag to determine what helper
  #' to call to.
  #' @return List of n observation-scale heritabilities

  stopifnot(modelname %in% c("Gaussian", "binom1.probit",
                             "binom1.logit"))
  samples.posterior <- inla.hyperpar.sample(n = n, inla.fit)
  vp.samples <- 0
  for (cname in colnames(samples.posterior)) {
    vp.samples <- vp.samples + 1 / samples.posterior[, cname]
  }

  if (!averaging) {
    mu <- inla.fit$summary.fixed$mean[1] # Intercept
    va.samples <- 1 / samples.posterior[, "Precision for id"]
    kwargs <- list(verbose = FALSE)

    h2.getter <- function(...) {
      get("h2.obs", suppressWarnings(QGparams(...)))
    }
    posterior <- mapply(h2.getter, mu, va.samples, vp.samples,
                        modelname, MoreArgs = kwargs
    )
    return(posterior)
  } else {
    # Average over fixed effects
    vp.samples <- 0
    df <- data.frame(
      va = as.vector(1 / samples.posterior[, "Precision for id"]),
      vp = as.vector(vp.samples)
    )
    if (!averaging.mode.only) {
      # Bayesian approach
      df$predict <- marginal.latent.samples(inla.fit, n)

      posterior <- do.call("rbind", apply(df, 1, function(row) {
        QGparams(
          predict = row[["predict"]], var.a = row[["va"]],
          var.p = row[["vp"]],
          model = modelname, verbose = FALSE
        )
      }))
    } else {
      predict.argument <- marginal.latent.mode(inla.fit)
      posterior <- do.call("rbind", apply(df, 1, function(row) {
        QGparams(
          predict = predict.argument, var.a = row[["va"]],
          var.p = row[["vp"]],
          model = modelname, verbose = FALSE
        )
      }))
    }
    return(posterior$h2.obs)
  }
}
```

Compute the 3 different approaches with `QGglmm` (this is quite slow). We also include time estimates here.

```
ti <- Sys.time()
h2.psi.sparrow <- data.frame(
  bayesian =
    get.h2.from.qgparams(fit.inla.probit, "binom1.probit",
      n.samples,
      averaging = TRUE,
      averaging.mode.only = FALSE
    )
)
```

```r
cat(
  "\n-\nRuntime for Bayesian:",
  difftime(Sys.time(), ti, units = "secs"), "secs.\n"
)
ti <- Sys.time()
h2.psi.sparrow$frequentist <- get.h2.from.qgparams(fit.inla.probit,
  "binom1.probit",
  n.samples,
  averaging = TRUE,
  averaging.mode.only = TRUE
)
cat(
  "\n-\nRuntime for Frequentist:",
  difftime(Sys.time(), ti, units = "secs"), "secs.\n"
)
ti <- Sys.time()
h2.psi.sparrow$noavg <- get.h2.from.qgparams(fit.inla.probit,
  "binom1.probit",
  n.samples,
  averaging = FALSE
)
cat(
  "\n-\nRuntime for No averaging:",
  difftime(Sys.time(), ti, units = "secs"), "secs.\n"
)
```

We do the same for simulation data

```r
# We store an instance of a gaussian and probit model with V_A = 1
tmp <- simulated.heritability(
  sigmaA = 1, linear.predictor = function(u, .) u + rnorm(length(u)),
  simulated.formula = simulated.formula,
  probit.model = TRUE,
  simulated.formula.probit = simulated.formula.probit
)
fit.sim.probit <- tmp$fit.probit

ti <- Sys.time()
h2.psi.sim1 <- data.frame(
  bayesian =
    get.h2.from.qgparams(fit.sim.probit, "binom1.probit",
      n.samples,
      averaging = TRUE,
      averaging.mode.only = FALSE
    )
)
cat(
  "\n-\nRuntime for Bayesian:",
  difftime(Sys.time(), ti, units = "secs"), ".\n"
)
```

```
ti <- Sys.time()
h2.psi.sim1$frequentist <- get.h2.from.qgparams(fit.sim.probit,
  "binom1.probit",
  n.samples,
  averaging = TRUE,
  averaging.mode.only = TRUE
)
cat(
  "\n-\nRuntime for Frequentist:",
  difftime(Sys.time(), ti, units = "secs"), ".\n"
)
ti <- Sys.time()
h2.psi.sim1$noavg <- get.h2.from.qgparams(fit.sim.probit,
  "binom1.probit",
  n.samples,
  averaging = FALSE
)
cat(
  "\n-\nRuntime for No averaging:",
  difftime(Sys.time(), ti, units = "secs"), ".\n"
)

# Also fit for smaller V_A, i.e. 0.1
tmp <- simulated.heritability(
  sigmaA = 0.1,
  linear.predictor = function(u, .) u + rnorm(length(u)),
  simulated.formula = simulated.formula,
  probit.model = TRUE,
  simulated.formula.probit = simulated.formula.probit
)

fit.sim.probit <- tmp$fit.probit

h2.psi.sim2 <- data.frame(
  bayesian =
    get.h2.from.qgparams(fit.sim.probit, "binom1.probit",
      n.samples,
      averaging = TRUE,
      averaging.mode.only = FALSE
    )
)
h2.psi.sim2$frequentist <- get.h2.from.qgparams(fit.sim.probit,
  "binom1.probit",
  n.samples,
  averaging = TRUE,
  averaging.mode.only = TRUE
)
h2.psi.sim2$noavg <- get.h2.from.qgparams(fit.sim.probit,
  "binom1.probit",
  n.samples,
  averaging = FALSE
)
```

```
plot.qgglmm.heritability <- function(h2.psi, dataset, SAVE.PLOT,
                                     plot.title = NA,
                                     fn.append = NULL) {
  #' Plot heriability density
  #'
  #' Compares posterior heritability using different transformations
  #' @param h2.psi Dataframe of n rows and a column for each
  #' back-transformation technique
  #' (bayesian, frequentist, no averaging, phi).
  #' @param dataset Either 'application' or 'simulation' specifying
  #' which dataset is used
  #' @param SAVE.PLOT Flag to store plot to disk
  #' @param plot.title (optional) title for plot. No title if unused.
  #' @param fn.append (optional) string to append to filename
  #' @return ggplot object
  color.map <- c(application = "Dark2", simulation = "Spectral")
  stopifnot(dataset %in% names(color.map))
  p <- ggplot(data = melt(h2.psi)) +
    geom_density(aes(x = value, fill = variable), alpha = 0.5) +
    scale_fill_brewer(
      palette = color.map[dataset],
      labels = c(
        expression(h[Psi]^2 * ", Bayesian"),
        expression(h[Psi]^2 * ", Frequentist"),
        expression(h[Psi]^2 * ", No averaging")
      )
    ) +
    ylab("Density") +
    xlab("Heritability") +
    theme(legend.text.align = 0, legend.title = element_blank()) +
    {
      if (!is.na(plot.title)) ggtitle(plot.title)
    } +
    {
      if (dataset == "simulation") xlim(c(0,quantile(
        melt(h2.psi)$value,0.99)))
    }

  if (SAVE.PLOT) {
    set_null_device(cairo_pdf)
    p.legend <- cowplot::get_legend(p)
    pdf(paste0("../figures/qgglmm-comparison-", dataset,
               "-legend.pdf"),
      width = 3, height = 3
    )
```

```r
    grid.newpage()
    grid.draw(p.legend)
    dev.off()
    ggsave(
      paste0(
        "../figures/qgglmm-comparison-",
        dataset, fn.append, ".pdf"
      ),
      p + if (dataset == "simulation") theme(legend.position = "none"),
      width = 20, height = 10, units = "cm"
    )
  }
  p
}

plot.qgglmm.heritability(h2.psi.sparrow, "application",
  SAVE.PLOT,
  plot.title = NA
)
plot.qgglmm.heritability(h2.psi.sim1, "simulation",
  SAVE.PLOT,
  plot.title = NA,
  fn.append = "va1"
)
plot.qgglmm.heritability(h2.psi.sim2, "simulation",
  SAVE.PLOT,
  plot.title = NA,
  fn.append = "va0.1"
)
```

Compare different scales

First, we fit the simulation probit model and get simulation-based heritability on all scales. Now we compute the heritability on the different scales - for song sparrow data. First we make a function to help us obtain all heritability scales in a dataframe.

```r
get.all.heritabilities <- function(fit.gaussian, fit.probit, p, n,
                                   fixed = FALSE) {
  #' Get h^2 for all scales
  #'
  #' For a Gaussian and probit fit, computes heritability on all scales
  #' @param fit.gaussian Fitted Gaussian model
  #' @param fit.probit Fitted Probit model
  #' @param p Phenotypic mean for the data, used in threshold formula
  #' @param n Number of samples
  #' @param fixed Flag for including fixed effects variance
  #' @return Dataframe of `n` rows with columns 'gaussian',
  #' 'guassian.liability', 'probit.latent', 'probit.scaled',
  #' 'probit.qgglmm'
  out <- data.frame(gaussian = get.h2(fit.gaussian, n,
                                      include.fixed = fixed))
  out$gaussian.liability <- threshold.scaling.param(p) * out$gaussian
  out$probit.latent <- get.h2(fit.probit, n, include.fixed = fixed)
  out$probit.scaled <- get.h2(fit.probit, n,
    model = "binom1.probit",
    use.scale = TRUE, include.fixed = fixed
  )
  out$probit.qgglmm <- get.h2.from.qgparams(fit.probit,
                                            "binom1.probit", n,
    averaging = TRUE
  )
  out
}
```

```
# Application data
heritability <- get.all.heritabilities(
  fit.inla.gaussian, fit.inla.probit,
  mean(qg.data.gg.inds$surv.ind.to.ad),
  n.samples,
  fixed = FALSE
)
```

```
simulation.res2 <- simulated.heritability(0.5, 100, 9,
  sigmaA = 0.5,
  linear.predictor = function(u, .) u + rnorm(length(u)),
  simulated.formula = simulated.formula,
  probit.model = TRUE, DIC = TRUE,
  simulated.formula.probit = simulated.formula.probit
)
heritability.sim <- get.all.heritabilities(
  simulation.res2$fit, simulation.res2$fit.probit,
  simulation.res2$p, n.samples,
  fixed = FALSE
)
```

Method to export heritability estimates in a TeX table

```
get.mode <- function(vec) {
  #' General helper to get mode of a vector
  d <- density(vec)
  d$x[which.max(d$y)]
}


print.one.metric <- function(fit, param, digits) {
  #' Helper for heritability table, rounding estiamtes
  paste(
    round(mean(get(param, fit)), digits), " & ",
    round(get.mode(get(param, fit)), digits), " & ",
    round(sd(get(param, fit)), digits),
    sep = ""
  )
}


print.heritability.table <- function(digits, h2, simulation = T) {
  #' Output LaTeX table of heritability
  #'
  #' Writes table of heritability with posterior mean, posterior mode
  #' and standard deviation, to a TeX file. Works for both datasets.
  #' @param digits Number of significant digits
  #' @param h2 Heritability DF with different scales
  #' @param simulation Simulation flag for the table's filename
  filename <- ifelse(simulation, "heritability simulation",
      "heritability application")
  header <- paste(
    "% TABLE FROM R:", format(Sys.time(), "%a %b %d %X %Y"), "\n",
    "\\begin{tabular}{lccc}\n",
    "\\hline\n",
    "Model & Mean & Mode & Standard deviation  \\\\ \n",
    "\\hline \n"
  )
  main <- paste(
    " Gaussian $h^2_\\text{obs}$ &",
    print.one.metric(h2, "gaussian", digits), "\\\\ \n",
    "Probit $h^2_{\\Psi}$ &",
    print.one.metric(h2, "probit.qgglmm", digits), "\\\\ \n",
    " & & & \\\\ \n",
    "Gaussian $h^2_\\text{liab}$ &",
    print.one.metric(h2, "gaussian.liability", digits), "\\\\ \n",
    "Probit $h^2_{\\Phi}$ &",
    print.one.metric(h2, "probit.scaled", digits), "\\\\ \n",
    "\\bottomrule"
  )
  footer <- "\\end{tabular}"

  write(paste(header, main, footer, sep = "\n"),
        file=paste0("../figures/", filename, ".tex"))
}
print.heritability.table(3, heritability, FALSE)
print.heritability.table(3, heritability.sim, TRUE)
```

Code for part of discussion:

```
write(paste0("$\\hat p=", round(simulation.res2$p,2),
             "$, that the true heritability of the observation scale",
             " would be $",
             round(1/3*1/threshold.scaling.param(simulation.res2$p),3),
             "$."),
             file="../figures/trueh2discussion.tex")
```

The table gives some indication, but we also want to look qualitatively on the densities. We start by just plotting a grid to compare each Gaussian vs. Probit scale two by two.

```
plot_grid_of_heritability <- function(heritability, SAVE.PLOT,
                                      plot.fn, colorscheme = NA) {
  #' Plot 3x2 grid of h^2 comparisons
  #'
  #' Compare density of posterior heritability between all scales for
  #' Gaussian model to all scales of the probit model. First row is
  #' observation-scale compares to latent, psi and phi, respectively.
  #' Second is the same for liability scale in the Gaussian case,
  #' and the same three cases for probit.
  #' @param heritability DF of all heritability estimates
  #' @param SAVE.PLOT Flag for storing plot to disk
  #' @param plot.fn Filename, must include file extension
  #' @param colorscheme (optional) Color palette to use for density
  #' @return List of all individual plots, as well as the grid plot
  p1 <- ggplot() +
    geom_density(
      data = melt(heritability[, c("gaussian", "probit.latent")]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{obs}$ vs. $h^2_{lat}$")) +
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  p2 <- ggplot() +
    geom_density(
      data = melt(heritability[, c(
        "gaussian.liability",
        "probit.latent"
      )]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{liab}$ vs. $h^2_{lat}$")) +
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  p3 <- ggplot() +
    geom_density(
      data = melt(heritability[, c(
        "gaussian",
        "probit.qgglmm"
      )]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{obs}$ vs. $h^2_{\\Psi}$")) +
```

```r
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  p4 <- ggplot() +
    geom_density(
      data = melt(heritability[, c(
        "gaussian.liability",
        "probit.qgglmm"
      )]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{liab}$ vs. $h^2_{\\Psi}$")) +
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  p5 <- ggplot() +
    geom_density(
      data = melt(heritability[, c(
        "gaussian",
        "probit.scaled"
      )]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{obs}$ vs. $h^2_{\\Phi}$")) +
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  p6 <- ggplot() +
    geom_density(
      data = melt(heritability[, c(
        "gaussian.liability",
        "probit.scaled"
      )]),
      aes(x = value, fill = variable), alpha = 0.5
    ) +
    theme(legend.position = "none", axis.title = element_blank()) +
    labs(title = TeX("$h^2_{liab}$ vs. $h^2_{\\Phi}$")) +
    if (!is.na(colorscheme)) scale_fill_brewer(palette = colorscheme)
  set_null_device(cairo_pdf)
  p <- plot_grid(p1, p3, p5, p2, p4, p6, ggplot() +
    theme_void(),
  get_legend(
    ggplot(data.frame(v = c("Gaussian", "Probit"), x = c(0, 0))) +
      geom_density(aes(x = x, fill = v), alpha = 0.5) +
      scale_fill_discrete(breaks = c("Gaussian", "Probit")) +
      theme(legend.title = element_blank()) +
      if (!is.na(colorscheme)) scale_fill_brewer(palette =
                                                   colorscheme)
  ),
  axis = "tblr"
  )
  if (SAVE.PLOT) ggsave(paste0("../figures/", plot.fn), p)
  list(p1 = p1, p2 = p2, p3 = p3, p4 = p4, p5 = p5, p6 = p6, p = p)
```

```r
}
plot.h2.appl <- plot_grid_of_heritability(
  heritability, SAVE.PLOT, "grid_application_gaussian_vs_binom.pdf",
  "Dark2"
)
# For simulation:
plot.h2.sim <- plot_grid_of_heritability(
  heritability.sim, SAVE.PLOT,"grid_simulation_gaussian_vs_binom.pdf",
  "Spectral"
)

plot.h2.appl$p
plot.h2.sim$p
```

Key takeaways:

- The Gaussian model to liability scale doesn't fit well with the other latent models.
- The scalings from binomial latent onto data scale fit well with the Gaussian one

The ones of greatest importance are plots $(1, 2)$ (p3) and $(2, 3)$ (p6), so we extract them in particular

```r
plot.h2.appl$p3 +
  theme(legend.position = "right", legend.title = element_blank()) +
  scale_fill_brewer(palette = "Dark2",labels = c(
    TeX("Gaussian $h^2_{obs}$"), TeX("Probit $h^2_\\Psi$"))
    ) +
  theme(legend.text.align = 0, legend.position = "bottom",
        legend.text = element_text(size = 23)) + ggtitle("")

if (SAVE.PLOT) ggsave(
  "../figures/heritability_application_obsscale.pdf")

plot.h2.appl$p6 +
  theme(legend.position = "right", legend.title = element_blank()) +
  scale_fill_brewer(palette = "Dark2", labels = c(
    TeX("Gaussian $h^2_{liab}$"), TeX("Probit $h^2_\\Phi$"))) +
  theme(legend.text.align = 0, legend.position = "bottom",
        legend.text = element_text(size = 23)) + ggtitle("")
if (SAVE.PLOT) ggsave(
  "../figures/heritability_application_liabscale.pdf")

plot.h2.sim$p3 +
  theme(legend.position = "right", legend.title = element_blank()) +
  scale_fill_brewer(palette = "Spectral", labels = c(
    TeX("Gaussian $h^2_{obs}$"), TeX("Probit $h^2_\\Psi$"))
    ) +
  theme(legend.text.align = 0, legend.position = "bottom",
        legend.text = element_text(size = 23)) + ggtitle("")
if (SAVE.PLOT) ggsave(
  "../figures/heritability_simulation_obsscale.pdf")
```

```
plot.h2.sim$p6 +
  theme(legend.position = "right", legend.title = element_blank()) +
  scale_fill_brewer(palette = "Spectral", labels = c(
    TeX("Gaussian $h^2_{liab}$"),
    TeX("Probit $h^2_\\Phi$"))) +
  theme(legend.text.align = 0, legend.position = "bottom",
        legend.text = element_text(size = 23)) + ggtitle("")
if (SAVE.PLOT) ggsave(
  "../figures/heritability_simulation_liabscale.pdf")
```

Finally, we want to look at DIC values for simulation model. It can't be compared to the song sparrow data directly, but how much the Gaussian and probit differ can be compared.

```
data.frame(
  Gaussian = simulation.res2$fit$dic$dic,
  Probit = simulation.res2$fit.probit$dic$dic,
  row.names = "Deviance Information Criteria"
)
```

Fixed effects for simulation

We now add a sex covariate to the linear predictor. We use that

$$\mathrm{Var}[\beta_{\mathrm{sex}}\mathbf{x}_{\mathrm{sex}}] = \beta^2_{\mathrm{sex}}\sigma^2_{\mathrm{sex}}$$

```
linear_predictor_fixedeffects <- function(u, simulated.d.ped) {
  #' \Tilde{\eta} = a + N(0, varE) + betaSex x_{sex}
  varE <- 1
  betaSex <- 100
  out <- c()
  intercept <- 0
  residuals <- rnorm(length(u), mean = 0, sd = sqrt(varE))
  for (idx in seq_along(u)) {
    out <- c(
      out,
      intercept + betaSex * simulated.d.ped$sex[idx] + u[idx] +
        residuals[idx]
    )
  }
  out
}
simulated.formula.fixedeffects <- simulated.response ~ sex +
  f(id,
    model = "generic0",
    Cmatrix = simulated.Cmatrix,
    constr = FALSE,
    hyper = list(
      prec = list(
```

```r
        initial = log(1 / 10), prior = "pc.prec",
        param = c(1, 0.05)
      ) # PC priors
    )
  )

# u = a + 100*x_sex + N(0,1) - balanced binary trait
m <- plot.h2.deviation(
  dichotomize = "round_balanced", title = TeX("$\\beta_{sex}=100$"),
  SAVE.PLOT = SAVE.PLOT, plot.fn = "fixedeffects_beta100",
  sigma.scale = "log",
  lin.pred = linear_predictor_fixedeffects,
  simulated.formula = simulated.formula.fixedeffects,
  Ve = 100^2, fixedeffects = TRUE, dynamic.priors=TRUE
)
m$p + xlim(c(1,10^4)) + theme(legend.position = "none")
if(SAVE.PLOT){
  ggsave("../figures/simulation_deviance_fixedeffects_beta100.pdf")
  } # Re-save figure with specified x-lim.

# Re-run with smaller magnitude for fixed effect
linear_predictor_fixedeffects <- function(u, simulated.d.ped) {
  varE <- 1
  betaSex <- 10
  out <- c()
  intercept <- 0 #-4.5
  residuals <- rnorm(length(u), mean = 0, sd = sqrt(varE))
  for (idx in seq_along(u)) {
    out <- c(
      out,
      intercept + betaSex * simulated.d.ped$sex[idx] + u[idx] +
        residuals[idx]
    )
  }
  out
}

# u = a + 10*x_sex + N(0,1) - balanced binary trait
m2 <- plot.h2.deviation(
  dichotomize = "round_balanced", title = TeX("$\\beta_{sex}=10$"),
  SAVE.PLOT = SAVE.PLOT, plot.fn = "fixedeffects_beta10",
  sigma.scale = "log",
  lin.pred = linear_predictor_fixedeffects,
  simulated.formula = simulated.formula.fixedeffects,
  Ve = 10^2, fixedeffects = TRUE, dynamic.priors = TRUE
)
m2$p

# u = a + 10*x_sex + N(0,1) - somewhat unbalanced
```

```
m3 <- plot.h2.deviation(
  dichotomize = 0.1,
  title = "Simulation heritability for \nfixed effects model",
  SAVE.PLOT = SAVE.PLOT, plot.fn = "fixedeffects_beta10_unbalanced",
  sigma.scale = "log",
  lin.pred = linear_predictor_fixedeffects,
  simulated.formula = simulated.formula.fixedeffects,
  Ve = 10^2, fixedeffects = TRUE, dynamic.priors = TRUE
)
m3$p

# How unbalanced is the response?
summary(m3$p$data$plist)
```

Similar to the case without fixed effects, we provide code for plotting based on results form remote server.

```
load("markovfixed_50_runs.Rdata")
mp4 <- markov.plotter(res.fixed1)
mp5 <- markov.plotter(res.fixed2)
mp6 <- markov.plotter(res.fixed3,
  legend.name="Simulation heritability for\nfixed effects model")
fixed.fn <- "../figures/simulation_deviance_fixedeffects_"
ggsave(paste0(fixed.fn, "beta100.pdf"),
       mp4+xlim(c(1,10^4)))
ggsave(paste0(fixed.fn, "beta10.pdf"), mp5)
ggsave(paste0(fixed.fn, "beta10_unbalanced.pdf"), mp6)
mp6.legend <- cowplot::get_legend(
  mp6 +theme(legend.position = "right", legend.text.align = 0))
pdf("../figures/simulation_deviance_fixedeffects_legend.pdf",
    width = 7.87402, height = 7.87402)
grid.newpage()
grid.draw(mp6.legend)
dev.off()
```

For sufficiently large choice of β corresponding to sex, we get progressively worse results as is expected.

Fixed effect model performance

Another aspect we can examine, is how the grid plots of heritability scales would look like if we use a Gaussian and probit model with (somewhat dominating) fixed effect. This is implemented below.

```
plot.fixedeffects.h2 <- function(sA, .dichotomize, include.fixed = T,
                                 sE = 1, beta = 10, SAVE.PLOT = T,
                                 plot.legend=F) {
  #' Plot h2 density of gaussian and backtransformed probit model,
  #'
  #' Fit simulation models with fixed effects, compute h2 for
```

```r
#' Gaussian and probit case, backtransform probit h2 and plot
#' @param sA Additive genetic variance sigma^2_A
#' @param .dichotomize character denoting dichotomization method
#' @param include.fixed Wether or not to include in denom. of h2
#' @param sE Error variance sigma^2_E
#' @param beta Weight for fixed effect in linear predictor
#' @param SAVE.PLOT Flag for storing plot to disk
#' @param plot.legend Flag for including legend in saved plot
.pc.prior <- c(10^(ceiling(log10(sA))), 0.05)
.linear_predictor_fixedeffects <- function(u, simulated.d.ped) {
out <- c()
intercept <- 0
residuals <- rnorm(length(u), mean = 0, sd = sqrt(sE))
for (idx in seq_along(u)) {
  out <- c(
    out,
    intercept + beta * simulated.d.ped$sex[idx] + u[idx] +
      residuals[idx]
  )
  }
out
}
fits <- simulated.heritability(
  idgen = 100, dichotomize = .dichotomize, pc.prior = .pc.prior,
  sigmaA = sA, linear.predictor = .linear_predictor_fixedeffects,
  simulated.formula = simulated.formula.fixedeffects,
  probit.model = TRUE,
  simulated.formula.probit = simulated.formula.probit
)
h2.sim.fixed <- get.all.heritabilities(fits$fit, fits$fit.probit,
                                       fits$p, n.samples,
                                       fixed = FALSE)

p <- ggplot(melt(h2.sim.fixed[, c("gaussian", "probit.qgglmm")])) +
  geom_density(aes(x = value, fill = variable), alpha = 0.5) +
  scale_fill_discrete(
    name = "",
    labels = c(TeX("Gaussian $h^2_{obs}"),TeX("Probit $h^2_\\Psi$"))
  ) +
  theme(legend.text.align = 0) +
  xlab("Heritability") +
  ylab("Density") + theme(legend.position = "bottom")
if (SAVE.PLOT) {
  p.legend <- cowplot::get_legend(p)
  pdf("../figures/fixedeffects_gaussian_probit_legend.pdf",
      width = 5.5, height = 1)
  grid.newpage()
  grid.draw(p.legend)
  dev.off()
```

```r
    legend.pos <- if(plot.legend) "right" else "none"
    plot.height <- if(plot.legend) 10 else 20
    fn_append <- if(plot.legend) "_wide" else NULL
    fn_append <- if(beta != 10) paste0(fn_append, "_beta_", beta) else
      fn_append
    ggsave(
      paste0("../figures/fixedeffects_gaussian_probit_sA", sA,
      "_p_", 10*round(fits$p,1), fn_append, ".pdf"),
      p+theme(legend.position = legend.pos), width = 20,
      height = plot.height, units = "cm"
    )
  }
}
plot.fixedeffects.h2(10, 0.1,plot.legend=T)
plot.fixedeffects.h2(10, 0.1)
plot.fixedeffects.h2(10, "round_balanced")
plot.fixedeffects.h2(500, 0.1)
plot.fixedeffects.h2(500,"round_balanced")

# For appendix:
for(beta_sex in c(1,5)){
  plot.fixedeffects.h2(sA=10, .dichotomize = "round_balanced",
                       beta=beta_sex)
  plot.fixedeffects.h2(sA=500, .dichotomize = "round_balanced",
                       beta=beta_sex)
}
```

IID noise to probit simulation

```r
alternative.probit.sim <- function(sigmaA, linear.predictor,
                                   fit.gaussian = NULL) {
  #' Simulate and fit model with and without extra noise
  #'
  #' Modified version of `simulated_heritability()` to fit probit
  #' model, one with an extra IID noise in INLA formula,
  #' and one without.
  #' @param sigmaA Additive genetic variance
  #' @param linear.predictor Callable of two variables, for
  #' simulating response
  #' @param fit.gaussian Flag for also fitting Gaussian model.
  #' Will fit as long
  #' as it's not `NULL`.
  #' @return List of two probit fits, gaussian fit (or `NULL`) and p,
  #' the simulation's phenotypic mean.

  # Init
  idgen <- 100
  NeNc <- 0.5
```

```r
nGen <- 9

# Generate pedigree
ped <- generatePedigree(
  nId = idgen, nGeneration = nGen, nFather = idgen * NeNc,
  nMother = idgen * NeNc
)
ped <- ped[, c(1, 3, 2, 5)]
names(ped) <- c("id", "dam", "sire", "sex")
u <- rbv(ped[, c(1, 2, 3)], sigmaA)
simulated.d.ped <- nadiv::prepPed(ped, gender = "sex")
simulated.Cmatrix <- nadiv::makeAinv(ped[, c(1, 2, 3)])$Ainv
simulated.d.ped$ind <- seq_len(nrow(simulated.d.ped))

# Generate binary response
simulated.response <- ifelse(
  linear.predictor(u, simulated.d.ped) <= 0, 0, 1)
p <- mean(simulated.response)

# INLA fitting
formula.overdisp <- simulated.response ~ f(id,
  model = "generic0", Cmatrix = simulated.Cmatrix,
  constr = FALSE,
  hyper = list(prec = list(
    initial = log(1 / 10), prior = "pc.prec",
    param = c(1, 0.05)
  ))
) +
  f(ind,
    model = "iid", constr = TRUE
  )
formula.standard <- simulated.response ~ f(id,
  model = "generic0", Cmatrix = simulated.Cmatrix,
  constr = FALSE,
  hyper = list(prec = list(
    initial = log(1 / 10), prior = "pc.prec",
    param = c(1, 0.05)
  ))
)


fit.overdisp <- inla(
  formula = formula.overdisp, family = "binomial",
  data = simulated.d.ped,
  control.compute = list(return.marginals.predictor = TRUE)
)
fit.standard <- inla(
  formula = formula.standard, family = "binomial",
  data = simulated.d.ped,
    control.compute = list(return.marginals.predictor = TRUE)
  )
  if (!is.null(fit.gaussian)) {
    # Also compute Gaussian model
    fit.gaussian <- inla(
      formula = formula.standard, family = "gaussian",
      data = simulated.d.ped
    )
  }


  list(
    fit.overdisp = fit.overdisp,
    fit.standard = fit.standard,
    fit.gaussian = fit.gaussian,
    overdisp.p = p
  )
}
```

Here, we fit a probit with the formula $y_i = \beta_0 + a_i + \gamma_{0,i}$, where the last is a random iid effect. The simulated data has overdisperion in its data via the residual vector being $\mathcal{N}(0, 3)$.

```r
overdisperion_wrapper <- function(nsamps, vA, vE, SAVE.PLOT) {
  #' Wrapper for running overdispersion tests
  #'
  #' Wrapper for calling alternative model fitting with extra noise,
  #' and for
  #' plotting thee results.
  #' @param nsamps Number of samples for posterior heritability
  #' @param vA Additive genetic variance
  #' @param vE Additional noise (should be more than 1)
  #' @param SAVE.PLOT Flag for storing plot to disk

  list2env( # Loads fit.overdisp, fit.standard, fit.gaussian, p
    alternative.probit.sim(vA, function(u, .) u + rnorm(length(u),
                                                        0, sqrt(vE)),
      fit.gaussian = TRUE
    ), .GlobalEnv
  )
  df.probit.comp <- data.frame(
    Overdispersion = get.h2.from.qgparams(fit.overdisp,
                                          "binom1.probit",
      nsamps,
      averaging = TRUE
    ),
    Standard = get.h2.from.qgparams(fit.standard, "binom1.probit",
      nsamps,
      averaging = TRUE
    ),
    Gaussian = get.h2(fit.gaussian, nsamps)
  )

  curr.plot <- ggplot(data = melt(df.probit.comp)) +
    geom_density(aes(x = value, fill = variable), alpha = 0.5) +
    scale_fill_discrete(name = "", labels = c(
      TeX("$h^2_\\Psi$ with iid effect"),
      TeX("$h^2_\\Psi$ without iid effect"),
      TeX("$h^2_{obs}$")
    )) +
    xlab("Heritability") +
    ylab("Density") +
    theme(legend.position = "bottom") +
    xlim(c(0, quantile(melt(df.probit.comp)$value, 0.95)))
  if (SAVE.PLOT) {
    p.legend <- cowplot::get_legend(curr.plot)
    pdf("../figures/overdisperions_legend.pdf", width = 9, height = 1)
    grid.newpage()
    grid.draw(p.legend)
    dev.off()
    ggsave(
      paste0("../figures/overdispersion_vE-vA_", vE, "-", vA, ".pdf"),
      curr.plot + theme(legend.position = "none")
    )
  }
}

overdisperion_wrapper(10000, 0.5, 2, SAVE.PLOT)
overdisperion_wrapper(10000, 0.5, 5, SAVE.PLOT)
overdisperion_wrapper(10000, 0.5, 10, SAVE.PLOT)
overdisperion_wrapper(10000, 10, 10, SAVE.PLOT)
```

Illustrative figures

Addendum - Illustrative figure for fitted values in a probit model. This is intended to demonstrate the different scales you get when using GLMMs.

```
mod <- simulated.heritability(
  linear.predictor = function(u, .) u + rnorm(length(u)),
  simulated.formula = simulated.formula,
  probit.model = T,
  simulated.formula.probit = simulated.formula.probit
)
pmod <- mod$fit.probit
eta_samples <- marginal.latent.samples(pmod, 20)

# Plotting
sample_ids <- sort(c(LETTERS, letters))[1:40]
```

```r
eta_df <- data.frame(
  elem = c(
    rep(sample_ids[1:20], each = 900), # Latent eta
    rep(sample_ids[21:40], each = 900)), # Phi(eta)
  value = c(
    unlist(eta_samples),
    pnorm(unlist(eta_samples))
  )
)
custom_palette <- c(
  colorRampPalette(c("pink", "darkred"))(20),
  colorRampPalette(c("lightblue", "darkblue"))(20), "darkgreen"
)

ggplot(eta_df) +
  geom_line(aes(x = value, color = elem), stat = "density",
            alpha = 0.25, size = 2) +
  # True values
  geom_vline(xintercept=0,size=2,color='green4') +
  geom_vline(xintercept=1, size=2, color='green4') +
  scale_color_manual(values = custom_palette) +
  theme(legend.position = "none") +
  xlab("(Predicted) response") +
  ylab("Density")
if (SAVE.PLOT) {
  ggsave("../figures/illustration_probit_scales_fitted_values.pdf",
    width = 20, height = 10, units = "cm"
  )
  # Generate legend
  plegend <- ggpubr::get_legend(
    ggplot(melt(data.frame(r = rnorm(1), b = rnorm(1),
                           t = rnorm(1)))) +
      geom_line(aes(x = value, color = variable), stat = "density",
                size = 2, alpha = 0.9) +
      scale_color_manual(
        name = "",
        labels = c(TeX("$\\eta$"), TeX("$\\Phi(\\eta)$"),
                   "True observations"),
        values = c(r = "darkred", b = "darkblue", t = "green4")
      ) +
      theme(legend.text.align = 0, legend.position = "bottom",
            text = element_text(family = texfont)
      )
  )
  pdf("../figures/illustration_probit_scales_fitted_values_legend.pdf",
    width = 4.2, height = 1
  )
  grid.newpage()
  grid.draw(plegend)
  dev.off()
}
```

The second is an illustrative figure for binomial vs linear regression in general.

```r
data(mtcars)
library(ggplot2)
library(cowplot)
p1 <- ggplot(mtcars, aes(x = hp, y = vs)) +
  geom_point(alpha = .5) +
  ggtitle("Binomial regression") +
  stat_smooth(method = "glm", se = F,
              method.args = list(family = binomial)) +
  ylim(c(-0.2, 1.01)) +
  theme(title = element_text(size = 16))
p2 <- ggplot(mtcars, aes(x = hp, y = vs)) +
  geom_point(alpha = .5) +
  stat_smooth(method = "lm", se = F) +
  ggtitle("Linear regression") +
  ylim(c(-0.2, 1.01)) +
  theme(title = element_text(size = 16))

plot_grid(p1, p2, ncol = 2, align = "v", axis = "tb")
if (SAVE.PLOT) {
  ggsave("../figures/linear-vs-logistic-example.pdf",
    width = 20, height = 10, units = "cm"
  )
}
```

The final plot is an illustration of the threshold model.

```r
threshold.illustration <- function(){
  x <- seq(-3, 3, length.out = 100)
  threshold <- -0.4
  df <- data.frame(x = x, y = dnorm(x))
  df$samps <- c(rep(NA,10), runif(80, 0, dnorm(x[11:90])), rep(NA,10))
  df$sampscol <- ifelse(df$x >= threshold, "a", "b")
  cols <- c(hcl(h=seq(15,375,length=3), l=65, c=100)[1:2])
  # Create the ggplot
  ggplot(df, aes(x = x, y = y)) +
    geom_point(aes(x = x, y = samps, colour = sampscol)) +
    geom_line(linewidth=0.7) + ylab("") + xlab("") +
    geom_vline(aes(xintercept=threshold, linetype='Threshold M'),
               linewidth=0.7) +
    theme_classic(24) + theme(text=element_text(family=texfont)) +
    scale_color_manual(name="", values=cols,
                       labels=c("Phenotype 2","Phenotype 1")) +
    scale_linetype_manual(name="", values=c('Threshold M'=2))
  ggsave("../figures/illustration_thresholdmodel.pdf",
       width=20, height=10, units="cm")
}
threshold.illustration()
```