



breedR : An open statistical package to analyse genetic data

Facundo Munoz, Leopoldo Sanchez Rodriguez

► To cite this version:

Facundo Munoz, Leopoldo Sanchez Rodriguez. breedR : An open statistical package to analyse genetic data. 2016. hal-02800531

HAL Id: hal-02800531

<https://hal.inrae.fr/hal-02800531v1>

Submitted on 5 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

breedR

An open statistical package to analyse genetic data (WP6)

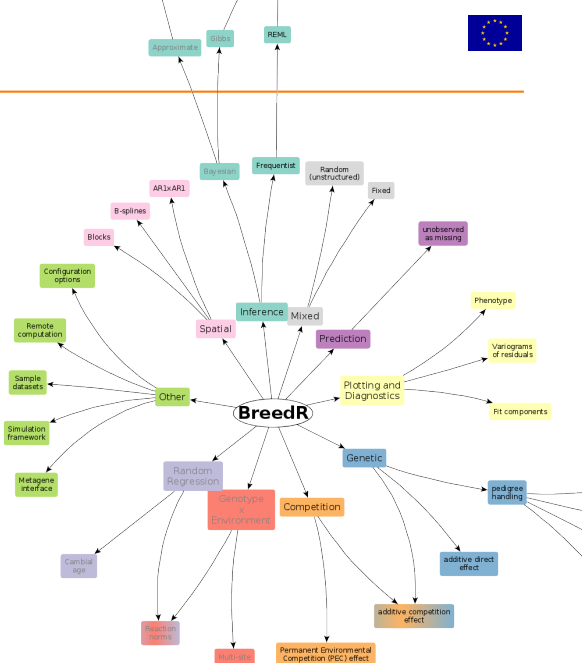
<http://famuvie.github.io/breedR/>

Facundo Muñoz

4th–6th april 2016



1. Introduction
2. Additive-genetic effects
3. Environmental effects
4. Competition effects
5. Longitudinal data
6. Genomic data
7. Multi-environment trials
8. Multi-trait models
9. Simulation framework
10. Remote computing



1 | Introduction

- R-package implementing **statistical models** specifically tailored to the analysis of forest genetic resources
- A inference tool for Linear Mixed Models, with facilities for typical needs
- **breedR** acts as an **interface** providing the means to:
 - 1 **Combine** any number of prefabricated model **components** into a larger model
 - 2 Compute automatically **incidence** and **covariance matrices** from a few input parameters
 - 3 **Fit** the model
 - 4 Plot data and results, and perform **model diagnostics**

2 alternatives:

- Project web page <http://famuvie.github.io/breedR/>
 - Set up this URL as a package repository in .Rprofile (detailed instructions on the web)
 - `install.packages('breedR')`
 - Not possible to use CRAN (or yes?) due to closed-source BLUPF90 programs
 - Stable version, with automatic updates
- GitHub dev-site <https://github.com/famuvie/breedR>
 - `if( !require(devtools) )`
`install.packages('devtools')`
 - `devtools::install_github('famuvie/breedR')`
 - Development version, latest features, more inestable, manual updates

- These slides show **WHAT** can be done with breedR
- For **HOW** to perform these analyses, refer to the website:

```
http://famuvie.github.io/breedR/
```

Where to find help

- Package's help: `help(package = breedR)`
 - Help pages `?remlf90`
 - Code demos `demo(topic, package = 'breedR')` (omit topic for a list)
 - Vignettes `vignette(package = 'breedR')` (pkg and wiki)
- Wiki pages
 - Guides, tutorials, FAQ
- Mailing list <http://groups.google.com/group/breedr>
 - Questions and debates about usage and interface
- Issues page
 - Bug reports
 - Feature requests



Figure 1: GPL-3

- **breedR** is FOSS. Licensed GPL-3
 - `RShowDoc('LICENSE', package = 'breedR')`
- You can **use** and **distribute breedR** for any purpose
- You can **modify** it to suit your needs
 - we encourage to!
 - please consider contributing your improvements
 - you can **distribute** your modified version under the GPL
- However, **breedR** makes (intensive) use of the BLUPF90 suite of Fortran programs, which are for *free* but not **free** (remember CRAN?)

$$y = X\beta + Zu + \varepsilon$$

$$u \sim \mathcal{N}(0, G)$$

$$\varepsilon \sim \mathcal{N}(0, R)$$

- A quantitative variable y is modelled as a linear function of **fixed effects** β and **random effects** u , with unaccounted residuals ε
- The function `remlf90()` yields a **REML fit** of a model to a dataset
- Additional functions (e.g. `summary()`, `fixef()`, `ranef()`, `plot()`, etc.) extract and present specific results

```
ped <- globulus[,1:3]
```

```
res <- remlf90(
  fixed = phe_X ~ gg,
  genetic = list(
    model = 'add_animal',
    pedig = ped,
    id = 'self'),
  data = globulus)
```

```
summary(res)
```

```
## Linear Mixed Model with pedigree and spatial
## effects fit by AI-REMLF90 ver. 1.122
##   Data: globulus
##   AIC   BIC logLik
##  5799 5809  -2898
##
## Variance components:
##               Estimated variances   S.E.
## genetic                      3.397 1.595
## Residual                     14.453 1.529
##
##               Estimate   S.E.
## Heritability    0.1887 0.08705
##
## Fixed effects:
##           value   s.e.
## gg.1    13.591 0.5014
## gg.2    14.085 0.7984
## ...
```

2 | Additive-genetic effects

What is an additive genetic effect

- Random effect at **individual level**
- Based on a **pedigree** (determining the *relationship* matrix A)

$$Zu, \quad u \sim \mathcal{N}(0, \sigma_a^2 A)$$

- BLUP of **Breeding Values** from own and relatives' phenotypes
- Represents the **additive component** of the genetic value
- More **general**:
 - family effect is a particular case
 - accounts for more than one generation
 - mixed relationships
- More **flexible**: allows to select individuals within families
- More **accurate**: direct inference over the **additive-genetic** variance of the base population

- A 3-column `data.frame` or `matrix` with the codes for each individual and its parents
- A **family** effect is easily translated into a pedigree:
 - use the **family code** as the identification of a fictitious **mother**
 - use 0 or NA as codes for the **unknown fathers**

self	dad	mum
69	0	64
70	0	41
71	0	56
72	0	55
73	0	22
74	0	50

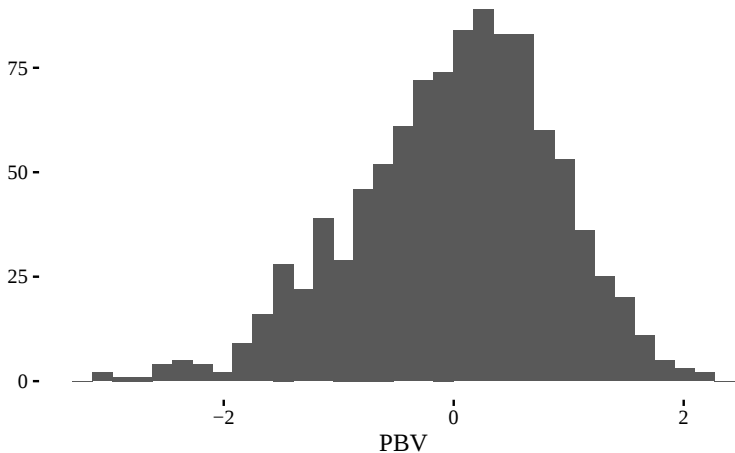


Figure 2

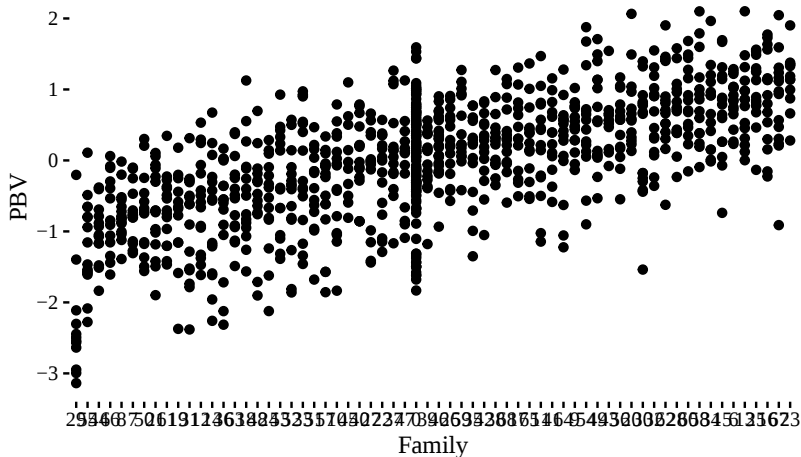


Figure 3

3 | Environmental effects

What is spatial autocorrelation

- The **residuals** of any LMM must be **noise**
- However, most times there are **environmental factors** that affect the response
- This causes that observations that are close to each other **tend** to be more similar than observations that are far away
- This is called **spatial autocorrelation**
- It may affect both the estimations and their accuracy
- This is why experiments are randomized into spatial **blocks**

- You can `plot()` the spatial arrangement of various model components (e.g. residuals)
- Look like **independent** gaussian observations (i.e. noise)?
- Do you see any **signal** in the background?

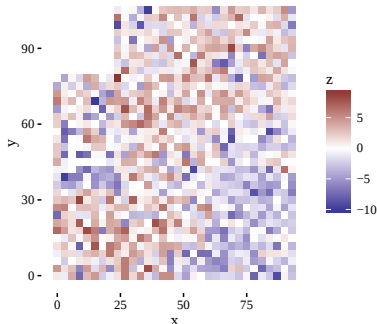


Figure 4

Plot the **variogram of residuals** with `variogram()`

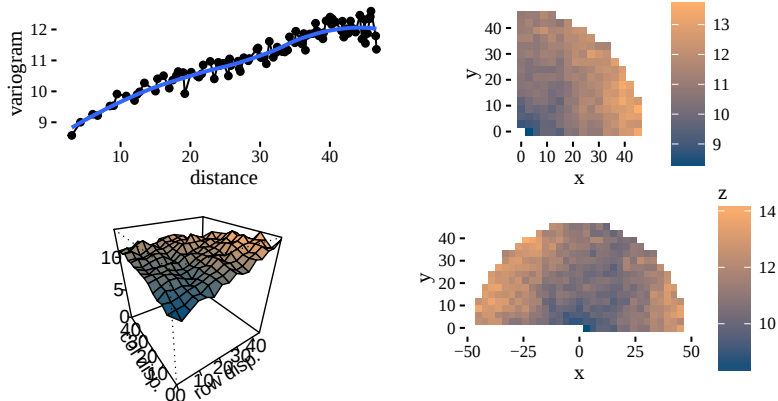


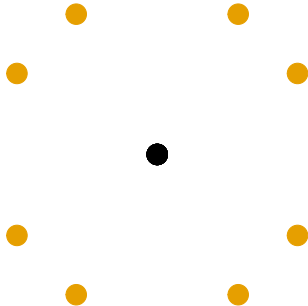
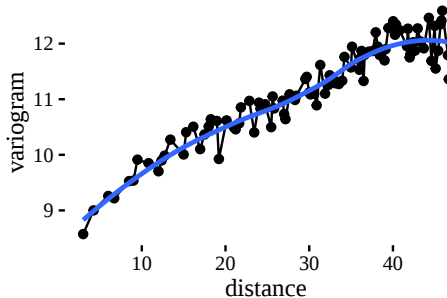
Figure 5

Interpreting the variograms

Isotropic variogram

$$\gamma(h) = \frac{1}{2} V[Z(\mathbf{u}) - Z(\mathbf{v})], \quad \text{dist}(\mathbf{u}, \mathbf{v}) = h$$

The **empirical** isotropic variogram is built by aggregating **all the pairs** of points separated by h , **no matter the direction**.

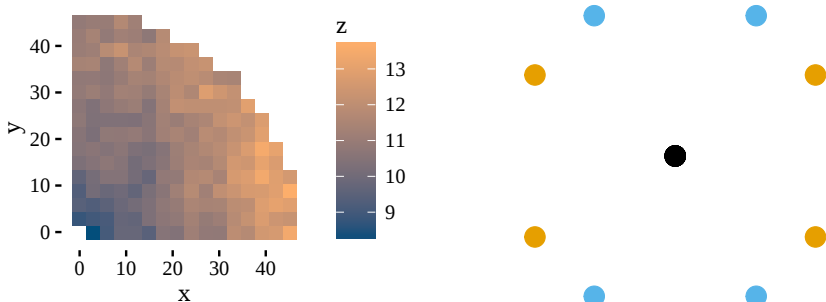


Interpreting the variograms

Row/Column variogram

$$\gamma(x, y) = \frac{1}{2} V[Z(\mathbf{u}) - Z(\mathbf{v})], \quad \text{dist}(\mathbf{u}, \mathbf{v}) = (x, y)$$

The **empirical** row/col variogram is built by aggregating **all the pairs** of points separated by exactly x rows and y columns.

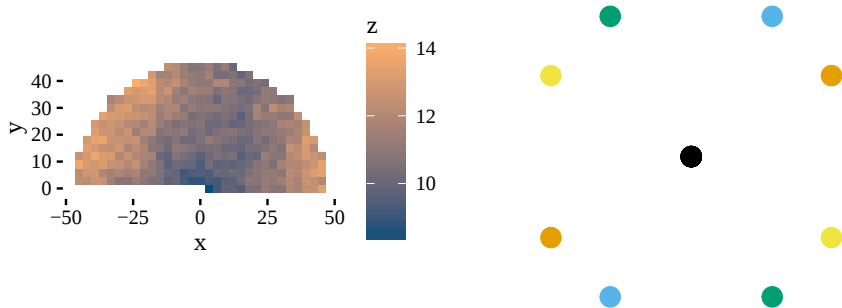


Interpreting the variograms

Anisotropic variogram

$$\gamma(\mathbf{x}) = \frac{1}{2} V[Z(\mathbf{u}) - Z(\mathbf{v})], \quad \mathbf{u} = \mathbf{v} \pm \mathbf{x}$$

The **empirical** anisotropic variogram is built by aggregating **all the pairs** of points **in the same direction** separated by $|\mathbf{x}|$.



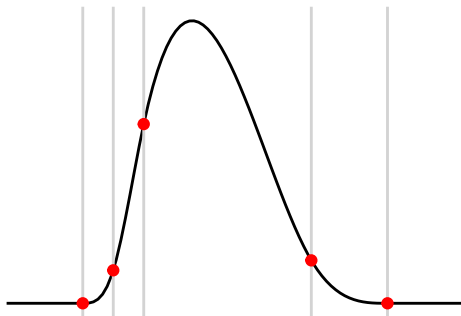
Accounting for spatial autocorrelation

- Include an explicit **spatial effect** in the model
- I.e., a **random effect** with a specific covariance structure that reflects the spatial relationship between individuals

$$Zu, \quad u \sim \mathcal{N}(0, \sigma_s^2 I)$$

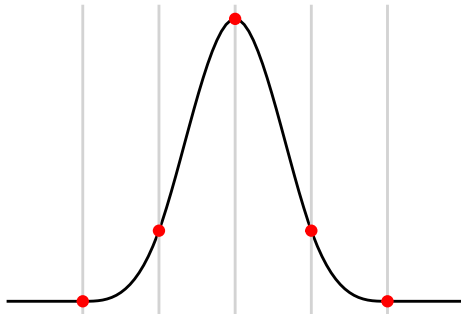
- u is the vector of random effects for the blocks
- Z is an indicator matrix such that $Z[i, j] = 1$ if the observation i belongs to block j
- σ_s^2 is the spatial variance parameter
- The **block** effect, is a very particular case of spatial effect:
 - It is designed from the beginning, possibly using prior knowledge
 - Can account for non-spatial effects (e.g. operator)
 - Introduces **independent** effects between blocks
 - Most neighbours are within the same block (i.e. share the same effect)

A **cubic B-spline** $B(x)$:



- **Piecewise** curve defined in the intervals determined by 5 **knots**
- Each *piece* is a polynomial of 3rd degree

A **cubic B-spline** $B(x)$ with regularly spaced knots:



- The curve is constrained for C^2 **continuity** at each knot
- Only 1 degree of freedom controls the **scale**

Spatial modelling

Bidimensional Splines

A number of overlapping curves form a **base** of B-splines $\{B_j(x)\}$

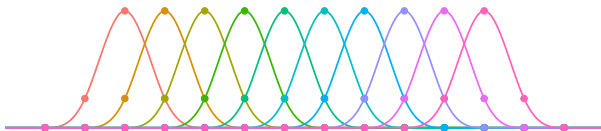


Figure 10

Each, can be **scaled** using a coefficient $\{u_j B_j(x)\}$

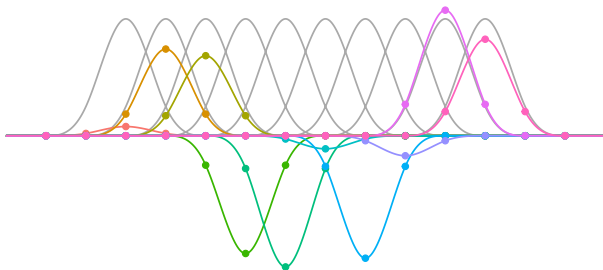


Figure 11

And **summed** to a **linear combination** $f(x) = \sum_j u_j B_j(x)$

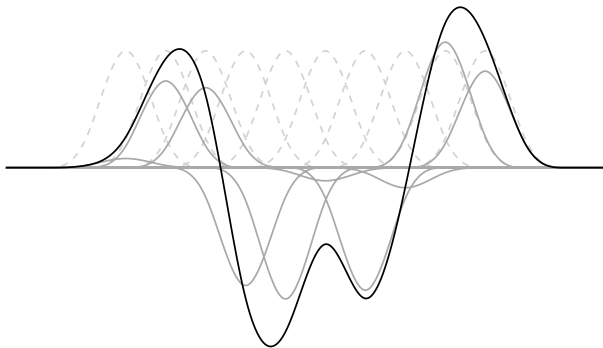


Figure 12

Spatial modelling

Bidimensional Splines in a Mixed Model

- $f(x) = \sum_j u_j B_j(x)$ provides a **spline representation** of a wide family of curves, in terms of a vector of coefficients u
- For any set of points $x = \{x_i\}$, the vector of values $f(x_i)$ can be written as a matrix operation $f = [B_j(x_i)]u$
- breedR extends this to **two dimensions** and defines a random effect

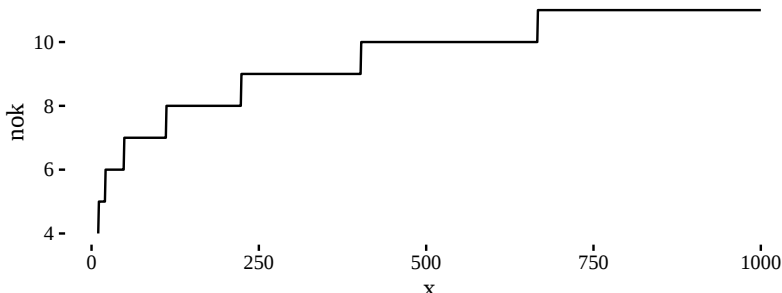
$$Bu, \quad u \sim \mathcal{N}(0, \sigma_s^2 R_s)$$

- u is the vector of spline effects
- B is the matrix of spline bases evaluated at the observations
- σ_s^2 is the spatial variance parameter
- R_s imposes a fixed positive correlation between coefficients of neighbouring spline bases

Spatial modelling

Number of knots of a splines model

- The **smoothness** of the spatial surface can be controlled modifying the number of base functions
- This is directly determined by the **number of knots** (nok) in each dimension
- If not explicitly set, it is determined heuristically by breedR as a function of the number of observations



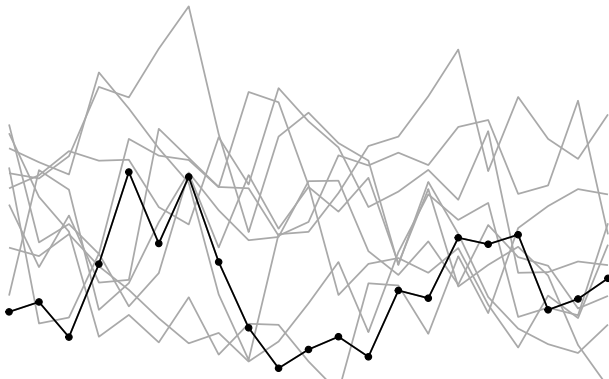
Spatial modelling

Bidimensional First-Order Autoregressive Process

- An $AR1(\rho)$ on the line is a collection of random variables $\{x_i\}$ where

$$x_t = \rho x_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, 1), |\rho| < 1$$

- A few random simulations with $\rho = 0.5$:



Spatial modelling

Bidimensional First-Order Autoregressive Process

breedR extends this model to the plane using and defines a component

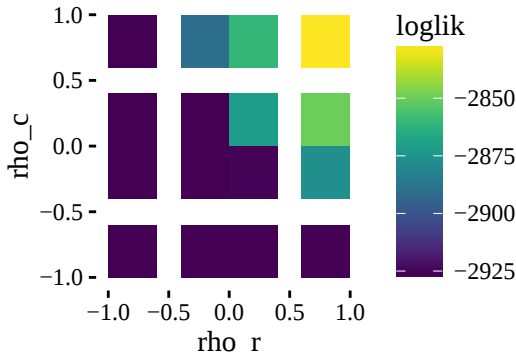
$$Zu, \quad u \sim \mathcal{N}(0, \sigma_s^2 R_{AR})$$

- u is the vector of random effects **for each individual location** on a regular grid
- Z is an **indicator matrix** such that $Z[i, j] = 1$ if the observation i is at site j
- σ_s^2 is the spatial variance parameter
- R_{AR} defines a separable correlation structure based on the kronecker product of two AR1 processes

Spatial modelling

Autoregressive parameters of a AR model

- The **smoothness** of the AR effects can be controlled by the autoregressive parameters (ρ_x, ρ_y) in each dimension
- They can be **given explicitly**
- Otherwise, breedR fits a model for each combination of parameters in a default grid and returns the most likely



Predicted spatial effects by spatial model

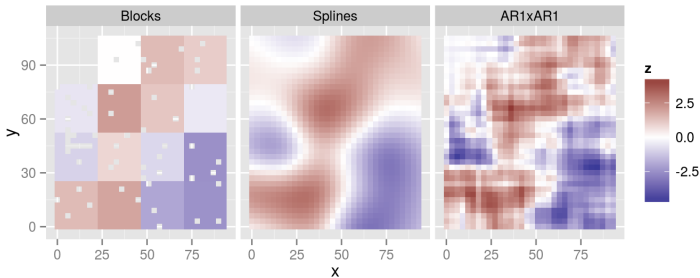


Figure 16: spatial-effects

- All capture a similar underlying **environmental pattern**
- with somewhat increasing ranges of variability

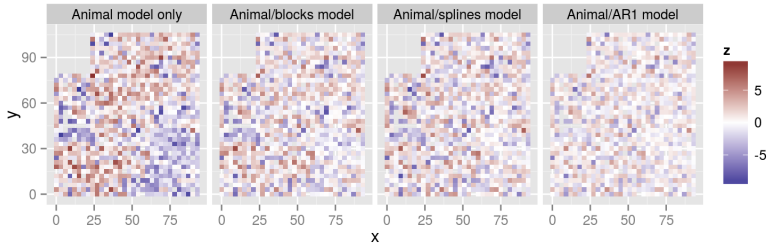


Figure 17: change residuals

- The spatial variability is taken mostly from the model residuals
- which increasingly look like pure noise

4 | Competition effects

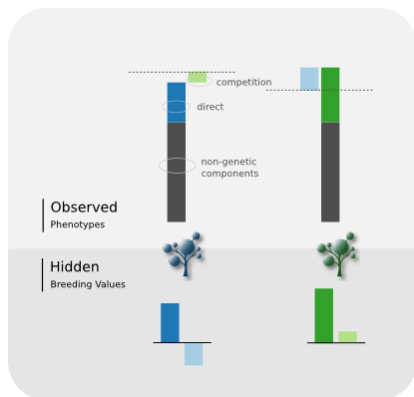


Figure 18: Competition model

- Each individual have **two** (unknown) Breeding Values (BV):
 - direct BV affects its **own** phenotype,
 - competition BV affects its **neighbours'**
- The total effect of the neighbouring competition BVs is given by their **distance-weighted sum**

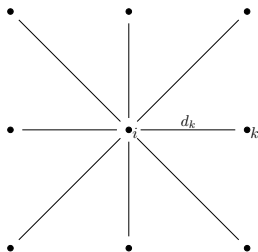


Figure 19: distance-plot

Let ∂i be the set of neighbouring locations of tree i , and $u_c = (u_{c,k})'$ the vector of competition BVs

$$\omega_i(\alpha) = \sum_{k \in \partial i} z_{ik}(\alpha) u_{c,k}$$

where $z_{ik}(\alpha) \propto 1/d_{ik}^\alpha$, such that

$$\sum_{k \in \partial i} z_{ik}(\alpha)^2 = 1.$$

This condition is **variance-stabilizer** ensuring $\forall i$:

$$\text{Var}(\omega_i) = \text{Var}(u_c) = \sigma_c^2$$

The **decay parameter** α controls the **relative intensity of competition** of the neighbours

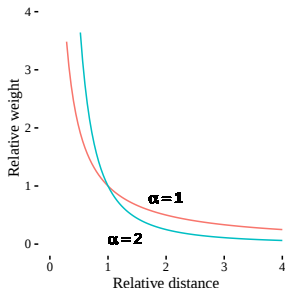


Figure 20

- The weights z_{ik} are **scale-invariant**
- e.g. a tree twice as far is weighted $1/2^\alpha$ as much
- higher values of α concentrate the weights on the closest trees

$$Z_d u_d + Z_c(\alpha) u_c, \quad \begin{pmatrix} u_d \\ u_c \end{pmatrix} \sim \mathcal{N}(0, \Sigma_a \otimes A), \quad \Sigma_a = \begin{pmatrix} \sigma_d^2 & \sigma_{dc} \\ \sigma_{dc} & \sigma_c^2 \end{pmatrix}$$

- Each set of BVs is modelled as a zero-mean **random effect** with structure matrix given by the **pedigree** and independent **variances** σ_d^2 and σ_c^2
- Both random effects are modelled jointly with **covariance** σ_{dc}
- Z_d is an indicator matrix linking observations and individuals
- $Z_c(\alpha)$ weights the competition effect of the neighbours with (fixed) **decay** parameter α

$$Z_p u_p, \quad Z_p = Z_c, u \sim \mathcal{N}(0, \sigma_p^2 I)$$

- **Optional** companion effect with **environmental** (rather than genetic) basis
- Modelled as an individual **independent** random effect that affects **neighbouring** trees in the same (weighted) way

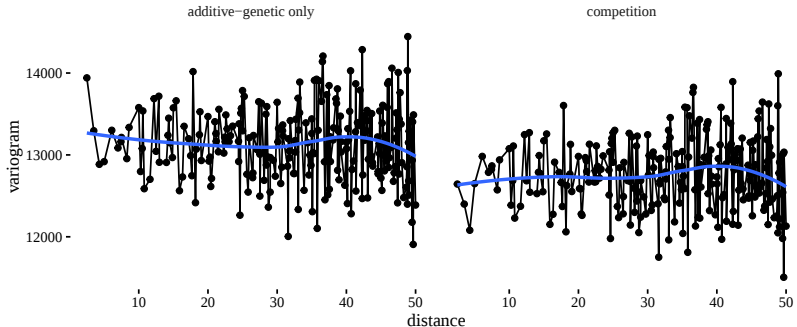


Figure 21

- Competition is often observable **in the first lag** of the variogram of residuals
 - increased antagonism between neighbouring phenotypes

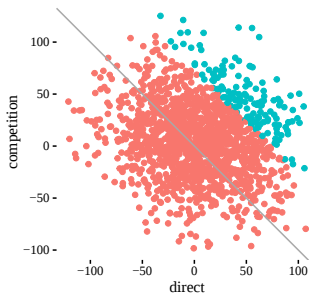


Figure 22

- direct and competition BV are usually **negatively correlated**
- selection based only on direct genetic merit tends to favour competitive individuals, hampering the **global performance**
- the competition model allows for selection based on a **joint assessment**

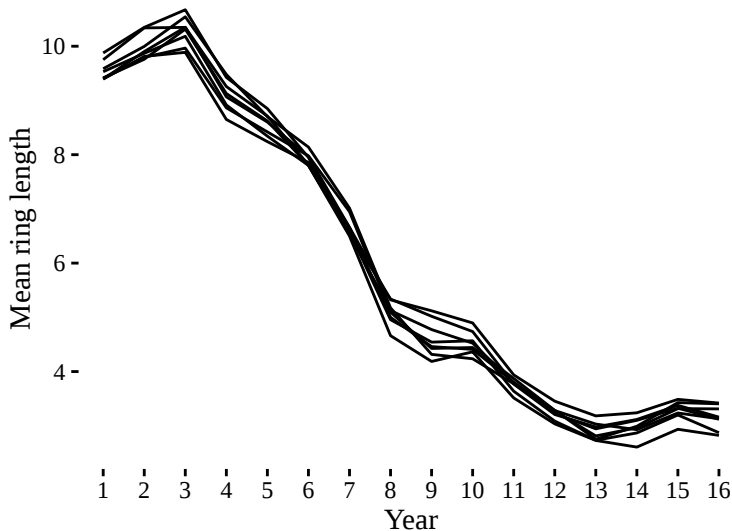
5 | Longitudinal data

What is longitudinal data

- Measurements **repeated** in **time** or along some **climatic or geographical variable** (e.g. temperature, precipitation, latitude, altitude, ...)
- All model parameters (e.g. variances, random effects) can be **functions** of the longitudinal variable
- Increased complexity: from estimating **numerical values** (dimension 0) to estimating (infinite-dimensional) **functions** (with finite data)
- Strategies:
 - assume parametric shape (e.g. linear regression)
 - nonparametric components (e.g. splines, Legendre polynomials)

Repeated measurements in time

Mean response evolution by replicate - Larix dataset



Climatic gradient

Mean ring-length by Mean Annual Precipitation

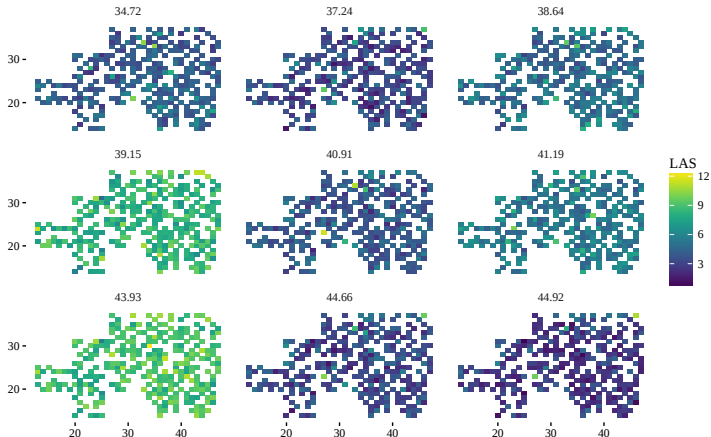
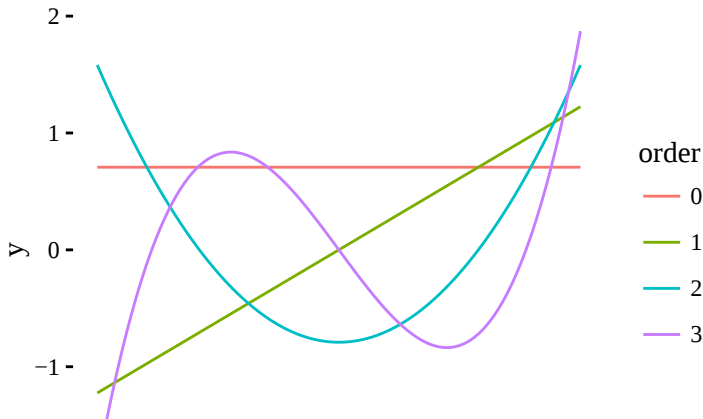


Figure 24

- Family of **orthogonal** polynomials **dense** in $\mathcal{L}_2$
 - Any regular curve can be approximated as much as needed by taking a linear combination of polynomials up to a sufficiently high order



For each observation of an individual i at j

$$y_{ij} = X_i \beta + \sum_{k=0}^{\text{ord}} a_{ik} \mathcal{L}_k(j) + \varepsilon_{ij}$$
$$(a'_0, \dots, a'_{\text{ord}})' \sim \mathcal{N}(0, \Sigma \otimes A)$$
$$\varepsilon \sim \mathcal{N}(0, \sigma_e^2)$$

- The Breeding Value of an individual is a **function** of an environmental variable
- This function is parameterised as a **linear combination** of Legendre orthogonal polynomials of order up to a fixed `ord`
- Each individual is described by `ord + 1` correlated **coefficients**

- Functional Breeding Values for each individual

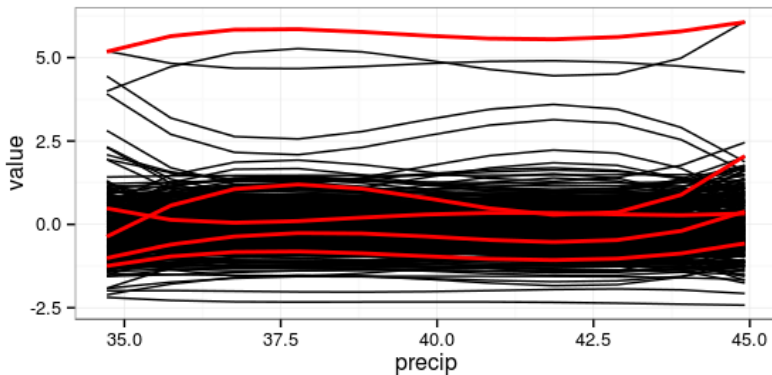


Figure 26: random-regression

6 | Genomic data

Using genomic markers

- breedR allows random effects with an **arbitrary covariance** structure (generic)
- This can be used to leverage **genomic information** (GBLUP)

This additional component allows to introduce a random effect ψ with **arbitrary** incidence and covariance matrices Z and Σ :

$$Z\psi, \quad \psi \sim \mathcal{N}(0, \sigma_{\psi}^2 \Sigma_{\psi})$$

Applications:

- include additional not-predefined components e.g. Dominance, Hybrid populations, Genomic evaluation, etc.

$$Zu, \quad u \sim \mathcal{N}(0, \sigma_G^2 G)$$

- Use markers to compute a **relationship matrix** G for individuals
 - Several methods available
 - e.g. VanRaden et al. 2009

$$G = XX' / \sum 2p(1 - p)$$

- **Replace** the additive-genetic model, which uses the pedigree-based relationship matrix A with a generic model with a genomic relationship matrix G
- Z is an **indicator** matrix linking observations with individuals
- Predicts genetic value of **individuals**, not markers
- Improved **accuracy** wrt pedigree-based evaluation

Relationship matrices pedigree-based vs. genomic

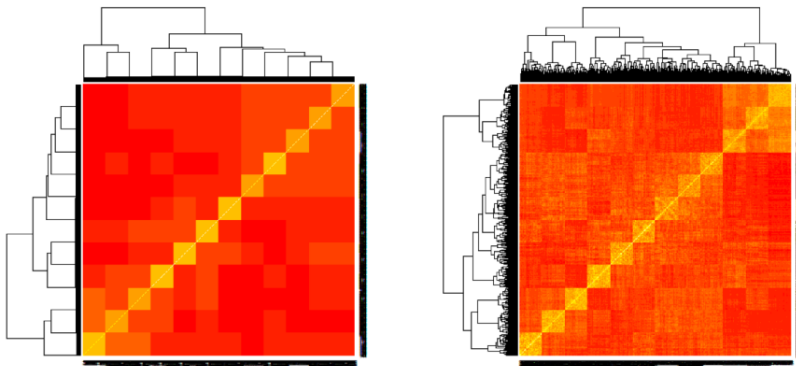


Figure 27: relationship-matrices

Note the increased level of detail in the relationship structure

7 | Multi-environment trials

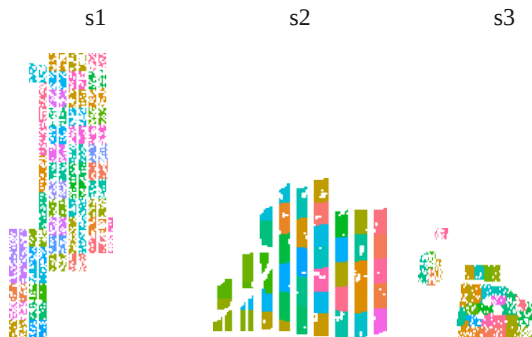


Figure 28

Fixed effects

- some can be **transversal** accross environments
 - e.g. origin, provenance
- other take different values at each environment
 - e.g. the intercept, replicate
 - modelled as a fixed **interaction**

Random effects

- some can be **transversal** accross environments
 - e.g. main effect of the genotype
- other can be **environment-specific**
 - e.g. blocks, GxE, residuals
 - modelled as a **group** of (correlated or not) random effects

$$y = X_0\beta_0 + Z_0u_0 + X \sum_e \mathbb{1}_{\{e\}}\beta_e + Z \sum_e \mathbb{1}_{\{e\}}u_e + \varepsilon$$

$$u_0 \sim \mathcal{N}(0, G_0)$$

$$(u_1, \dots)' \sim \mathcal{N}(0, \Sigma_G \otimes G)$$

$$\varepsilon \sim \mathcal{N}(0, D_R \otimes I)$$

- Particular case of the general LMM
- For each **environment** e , there is a **group** of random effects u_e , each with covariance structure G , possibly cross-correlated through Σ_G
- Independent site-specific residual variances

$$C13 = \text{orig} + \text{site} + \text{fam} + \sum_{e=1}^3 f_e 1_e + \varepsilon$$

$$\text{fam} \sim \mathcal{N}(0, \sigma_f^2 \mathbf{I})$$

$$(f_1, f_2, f_3)' \sim \mathcal{N}(0, \Sigma_{G \times E} \otimes \mathbf{I})$$

$$\varepsilon \sim \mathcal{N}(0, D_3 \otimes \mathbf{I})$$

- One **global** family effect (fam)
- One group of three **site-specific** family effects (f_i , $i = 1, 2, 3$)
- Jointly, they represent the $G \times E$ **interaction** with genetic cross-covariation $\Sigma_{G \times E}$

Predicted genetic effects by family

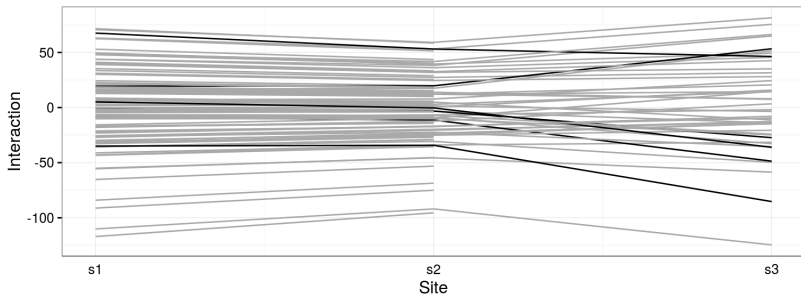


Figure 29: GxE-interaction

- Sum of main and interaction effects
- Note:
 - different **variances** per site
 - high genetic **correlation**
 - some families are more **interactive**

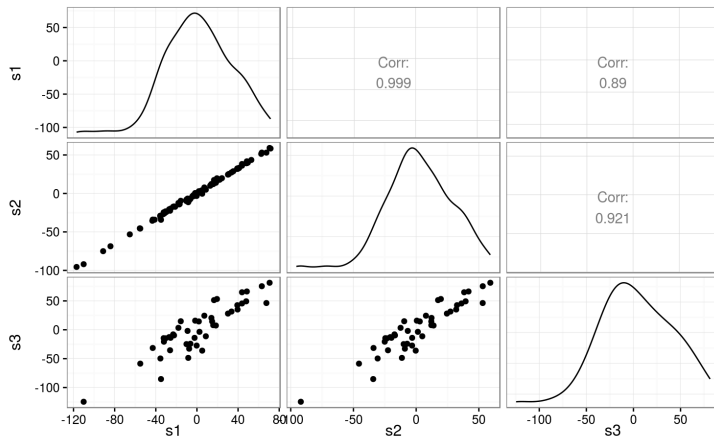


Figure 30: genetic-correlations

For each family x

$$\varphi(x) \propto \sum_{i \in x} \sum_e f_e^2$$

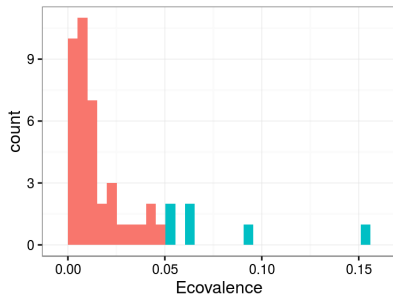


Figure 31: ecovalence

Practical case 2

Site-specific spatial effects

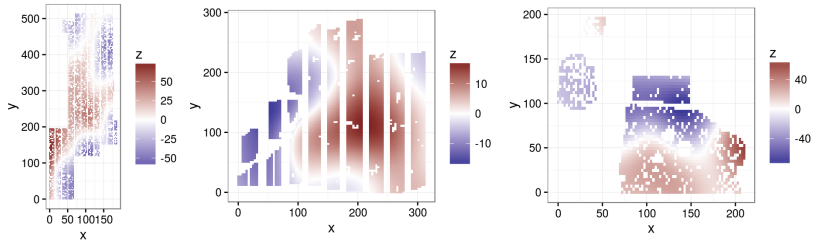


Figure 32: spatialxE

8 | Multi-trait models

Multivariate Linear Mixed Models

2-trait case

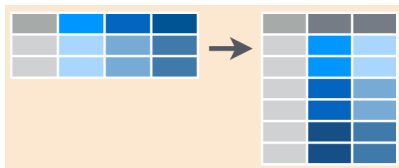
$$\begin{aligned}Y_1 &= X\beta_1 + Zu_1 + \varepsilon_1 \\Y_2 &= X\beta_2 + Zu_2 + \varepsilon_2, \\(u_1, u_2)' &\sim N(0, \Sigma_u \otimes G) \\(\varepsilon_1, \varepsilon_2)' &\sim N(0, \Sigma \otimes I_n).\end{aligned}$$

- Σ_u and Σ either **diagonal** or **fully-parameterized** 2×2 matrices
- Some of the fixed or random effects can affect only a **subset of the traits**
 - e.g. fixed effect of operator

Limitation of breedR's implementation

- All fixed and random effects are assumed to be **trait-specific**
 - **transversal effects** not directly supported (ultimately by PROGSF90)
- Simpler covariance structures **not supported**
 - e.g. independent effects with shared variance, exchangeable structure
- A workaround is to **reshape the dataset** to long-layout

Multi-trait with reshaping wide to long-layout



- Reshaping operation:
 - Stack traits into a **single variable** value
 - Additional variable trait
 - Duplicate individual information and other variables
- Use single-trait models with MET syntax
 - trait instead of site
- This overcomes the limitations breedR's multi-trait implementation
 - more complex models like multi-trait **and** multisite become cumbersome

9 | Simulation framework

breedR's simulation function

Simulating data under competition



- Simulate datasets of any size, from any most supported models
- See `?simulation` for details on the syntax

Source: local data frame [500 x 14]

	self	sire	dam	beta	x	y	spatial	BV1
	(dbl)	(dbl)	(dbl)	(dbl)	(int)	(int)	(dbl)	(dbl)
1	41	14	40	1	1	19	-1.0738194	-1.023654
2	42	18	33	1	18	22	1.6654315	2.477488
3	43	11	31	1	19	19	0.7047348	1.961305
4	44	5	38	1	1	23	0.4698724	1.207112
5	45	16	24	1	7	12	-0.7233131	-1.919742
6	46	9	38	1	3	5	-0.1197804	1.174054
..	...	...	...	...	...	...	...	...

Variables not shown: BV2 (dbl), wnc (dbl), pec (dbl), wnp (dbl), resid (dbl), phenotype (dbl)

Applications

a.k.a. what the heck I want a simulator for?

- check models under **ideal** scenarios
- **Bootstrapping:**
 - compute heritability (and its s.e.) for complex models
 - compute more accurate s.e. for fixed and random effects
 - inference on arbitrary hypotheses (involving any combination of model parameters)

Heritability of complex models

e.g. competition or splines models fitted by EM-REML (rather than AI)

- Thus, heritability **not available**
- Other methods (e.g. *Delta*) **not feasible**
- Even when available, is **approximate** (relies on asymptotic normality of parameters)

- 1 Fit the model to your data
 - 2 Write a function to **simulate data** from your fitted model parameters
 - 3 Write a function to **fit a simulated dataset** and return realised heritability
-
- Repeated calls to this function yields the **sampling distribution** of heritability
 - Compute **SE** and **CI** from numerical summaries

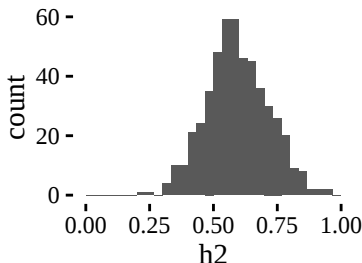


Figure 33

More accurate Standard Errors

- SE in breedR's output are **approximate**
- Rely on asymptotic normality (same as heritability)
- Same Bootstrapping procedure applies

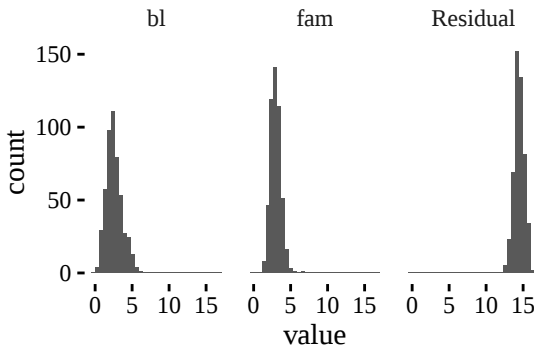


Figure 34

10 | Remote computing

If you have access to a **Linux** server through **SSH**, you can perform computations remotely

- Take advantage of more **memory** or **faster** processors
- **Parallelize** jobs
- Free **local resources** while fitting models
- See `?remote` for details

- 1 Windows users: install `cygwin` with `ssh` beforehand (<http://cygwin.org/>)
- 2 configure the client and server machines so that passwordless SSH authentication works
- 3 Set `breedR` options `remote.host`, `remote.user`, `remote.port` and `remote.bin` (see `?breedR.setOption`)
 - Optionally, set these options permanently in `$HOME/.breedRrc`

```
writeLines(  
  c("remote.host = '123.45.678.999'",  
    "remote.user = 'uname'",  
    "remote.bin  = 'remote/path/to/breedR/bin/linux'"),  
  con = file.path(Sys.getenv('HOME'), '.breedRrc')
```

Fitting models remotely

```
res <- remlf90(..., breedR.bin = "remote")
```

- Fit model **remotely**
- R-console stays in **stand-by** until job is finished
- When job finishes (provided that connection keeps alive), results are automatically **retrieved**

Identical in use to local computing, but without the processor/memory burden

```
res <- remlf90(..., breedR.bin = "submit")
```

- Fit model **remotely**
- Connection is **closed** in the meanwhile
- R-console is **active**
- Typing `res` **queries** the server for the job status (Running/Finished/Aborted)

- After you **submit** a job, you are free to submit more (specially with multiple-processor servers)
- Query the **status** of all jobs with `breedR.qstat()`
- **Kill** some job with `breedR.qdel(res)` or all jobs with `breedR.qnuke()`



breedR

<http://famuvie.github.io/breedR/>