

**UNIVERSIDADE ESTADUAL PAULISTA (UNESP)
FACULDADE DE CIÊNCIAS AGRÁRIAS E TECNOLÓGICAS
CAMPUS DE DRACENA**

Miriam Aparecida Carvalho Basilio Antonio
Zootecnista

**DESENVOLVIMENTO DE FUNÇÕES PARA FORMATAÇÃO
DE ARQUIVOS PARA ANÁLISES GENÉTICAS NO R**

DRACENA – SP
2025

UNIVERSIDADE ESTADUAL PAULISTA (UNESP)
FACULDADE DE CIÊNCIAS AGRÁRIAS E TECNOLÓGICAS
CAMPUS DE DRACENA

Miriam Aparecida Carvalho Basilio Antonio

Zootecnista

**DESENVOLVIMENTO DE FUNÇÕES PARA FORMATAÇÃO
DE ARQUIVOS PARA ANÁLISES GENÉTICAS NO R**

Dissertação apresentado(a) à Universidade Estadual Paulista (UNESP), Faculdade de Ciências Agrárias e Tecnológicas - Câmpus de Dracena, para obtenção do título de Grau acadêmico
Mestre(a) em Ciências e Tecnologia Animal.

Orientador: Prof. Dr. Ricardo da Fonseca

DRACENA – SP

2025

FICHA CATALOGRÁFICA
Desenvolvida pela Diretoria Técnica de Biblioteca e Documentação
Campus de Dracena

A6354d

Antonio, Miriam Aparecida Carvalho Basilio

Desenvolvimento de funções para formatação de arquivos para análises genéticas no R / Miriam Aparecida Carvalho Basilio Antonio. -- Dracena: [s.n.], 2025.

75 f. : il.

Dissertação (Mestrado) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências Agrárias e Tecnológicas, Dracena. Área do conhecimento: Produção Animal, 2025.

Orientador: Ricardo da Fonseca

1. Genética animal. 2. Estatística matemática - Programas de computador. 3. Animais - Melhoramento genético. I. Título.

Bibliotecário Fábio S. Rosas
CRB 8/6665

IMPACTO POTENCIAL DESTA PESQUISA

O pacote LZFF padroniza e automatiza dados usados em softwares de avaliação genética e genômica, aumentando a reprodutibilidade. Facilita o acesso de estudantes e pesquisadores, reduzindo barreiras técnicas. Suas funções modulares em R otimizam preparo de dados, favorecendo ciência aberta e melhoramento genético animal.

POTENTIAL IMPACT OF THIS RESEARCH

The LZFF package standardizes and automates data used in genetic and genomic evaluation software, increasing reproducibility. It facilitates access for students and researchers by reducing technical barriers. Its modular R functions streamline data preparation, promoting open science and animal genetic improvement.



UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Dracena



CERTIFICADO DE APROVAÇÃO

TÍTULO DA
DISSERTAÇÃO:

Desenvolvimento de funções para formatação de arquivos para
análises genéticas no R

AUTORA: MIRIAM APARECIDA CARVALHO BASILIO ANTONIO

ORIENTADOR: RICARDO DA FONSECA

Aprovada como parte das exigências para obtenção do Título de Mestra em Ciências , pela
Comissão Examinadora:

Documento assinado digitalmente
 **RICARDO DA FONSECA**
Data: 25/07/2025 12:06:32-0300
Verifique em <https://validar.itl.gov.br>

Prof. Assoc. RICARDO DA FONSECA (Participação Presencial)
Departamento de Produção Animal - DPA / Faculdade de Ciências Agrárias e Tecnológicas - FCAT -
Unesp - Câmpus de Dracena

Documento assinado digitalmente
 **GUSTAVO DO VALLE POLYCARPO**
Data: 28/07/2025 17:04:17-0300
Verifique em <https://validar.itl.gov.br>

Prof. Assoc. GUSTAVO DO VALLE POLYCARPO (Participação Presencial)
Departamento de Produção Animal - DPA / Faculdade de Ciências Agrárias e Tecnológicas de
Dracena - FCAT
- Unesp - Câmpus de Dracena

Documento assinado digitalmente
 **LUIS ORLANDO DUITAMA CARREÑO**
Data: 29/07/2025 10:05:45-0300
Verifique em <https://validar.itl.gov.br>

Dr. LUIS ORLANDO DUITAMA CARREÑO (Participação Virtual) BRF - Brasil Foods

Dracena, 25 de julho de 2025

DADOS CURRICULARES DO AUTOR

Miriam Aparecida Carvalho Basilio Antonio, nascida em 09 de setembro de 1996, em Presidente Venceslau, SP. Concluiu a graduação em Zootecnia pela Universidade do Oeste Paulista (UNOESTE) em 2017, período em que realizou duas iniciações científicas na área de pastagens. Em 2022, finalizou a licenciatura em Ciências Biológicas e, no ano seguinte, iniciou o programa de pós-graduação em Ciência e Tecnologia Animal, em nível de mestrado, pela Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP), com ênfase em Desenvolvimento de Funções no Software R. Atualmente, atua como professora de Ciências e Biologia no Ensino Fundamental II e Médio, unindo sua experiência acadêmica à prática pedagógica para promover o aprendizado nas áreas das ciências naturais.

Dedico este trabalho aos meus pais e meu esposo, que sob muito sol, fizeram-me
chegar até aqui, na sombra.

AGRADECIMENTOS

À Deus, por me permitir chegar aonde estou e me proteger sempre.

Aos meus queridos pais José Sebastião da Silva Basilio e Jani Carvalho Basilio que são meus exemplos de vida, de caráter e luta. Obrigada por sempre me apoiar e não me deixar desistir dos meus sonhos. Sem vocês nada seria possível, agradeço por tudo, amo vocês!

Ao meu querido e amado esposo Denilson de Jesus Antônio, que sempre me apoio nas minhas decisões e esteve ao meu lado, mesmo quando o cenário não era dos melhores, segurou a minha mão e não me deixou fracassar. Sem você eu não conseguiria, obrigada por tudo, amo você!

Ao meu irmão, cunhada e sobrinha Matheus Josino Carvalho Basilio, Isabela Rodrigues Tozetti e Maitê Emanuely Rodrigues Basilio. Obrigada por estarem comigo nesta caminhada, por me preencher de amor, carinho e esperança. Principalmente a minha sobrinha que desde que nasceu me proporciona momentos de alegria, pureza e muito amor. Com vocês minha vida é mais feliz e colorida. Obrigada por tudo, amo vocês!

À minha amiga de vida e as que conheci neste trajeto, Taís Cristina Raposo Sanches, Luma Escalante e Caroline Mayume. Obrigada por me apoiar, por me incentivar e principalmente a amizade de vocês. Obrigada por cada momento compartilhado. Amo vocês!

Ao meu orientador Prof. Drº Ricardo da Fonseca agradeço imensamente. Obrigada por confiar em mim, desde o primeiro e-mail trocado. Agradeço por todo ensinamento, paciência e compreensão nos momentos difíceis desta jornada. Levarei sempre no meu coração, gratidão por tudo.

À equipe do LuCCA-Z, Isabel Oliveira Silva e Ana Laura Silva pela amizade, paciência e ajuda. Obrigada por proporcionar dias incríveis de aprendizados e amizade.

À esta universidade, seu corpo docente, direção e administração que oportunizaram o conhecimento da pós-graduação, pela confiança no mérito e ética aqui presente.

Aos membros da minha banca de qualificação e defesa, Prof. Dr. Gustavo do Valle Polycarpo, Prof. Dr. Celso Tadão Miasaki e Dr. Luis Orlando Duitama Carreño pelas considerações importantes que contribuem para a melhoria do meu trabalho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), pelo apoio financeiro, concedido por meio do Processo nº 2024/10028-4.

E a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

É justo que muito custe o que muito vale.” (Santa Teresa D’Ávila)

RESUMO

A avaliação genética animal depende da análise de grandes volumes de dados fenotípicos, de pedigree e genômicos. Nos últimos 20 anos, diversos softwares foram desenvolvidos para esse fim, sendo os mais utilizados: ASReml, VCE, DMU, Wombat e BLUPF90. O ASReml manteve-se como o software mais citado nas publicações científicas, mas sua exigência de licenciamento pago restringe o acesso para pesquisadores em ambientes acadêmicos e institucionais com recursos limitados. Por essa razão, este trabalho concentrou-se no desenvolvimento de funções voltadas aos softwares BLUPF90 e Wombat, que são gratuitos, amplamente utilizados e frequentemente atualizados. Existe um problema de falta de padronização na maneira de como os dados devem ser organizados para imputar dentro de cada software, sendo que cada software tem sua própria especificidade, ou seja, os dados devem ser organizados de maneira diferente dependendo do software que será usado nas análises genéticas, afetando a reprodutibilidade e abrindo espaço para erros. Para enfrentar esse desafio, este trabalho apresenta o desenvolvimento do pacote LZFF, que automatiza a preparação e formatação de arquivos para análises genéticas, minimizando erros e promovendo padronização. Os dados percorrem um fluxo estruturado: do formato livre (L) ao padrão (P) e, por fim, ao formato final (F), compatível com os softwares de destino. Foram desenvolvidas quatro funções principais: `rrcData`, que realiza a leitura, recodificação e verificação dos dados; `rrcPed`, voltada à estruturação de arquivos de pedigree com recodificação apropriada; `formatA`, que prepara os dados no formato exigido pelo BLUPF90; e `formatB`, que ajusta os dados para uso com o Wombat. Para validar o pacote, foram implementados testes automatizados com o uso do pacote `testthat`, garantindo a robustez do processo de leitura, recodificação e integridade estrutural dos dados processados. O pacote LZFF oferece uma solução eficiente e acessível para a preparação de dados em análises genéticas, reduzindo o tempo de formatação manual e potencializando a reprodutibilidade dos resultados.

Palavras-chave: Avaliação Genética. BlupF90. Software R. Reprodutibilidade. Wombat.

ABSTRACT

Animal genetic evaluation relies on the analysis of large volumes of phenotypic, pedigree, and genomic data. Over the past 20 years, several software packages have been developed for this purpose, with the most widely used being ASReml, VCE, DMU, Wombat, and BLUPF90. ASReml has remained the most frequently cited software in scientific publications; however, its paid licensing requirement restricts access for researchers in academic and institutional environments with limited resources. For this reason, the present work focused on developing functions tailored for BLUPF90 and Wombat, which are free, widely used, and frequently updated. A major challenge lies in the lack of standardization regarding how data must be organized for each software, as each program has its own specific requirements. Consequently, data must be formatted differently depending on the chosen software for genetic analyses, which undermines reproducibility and increases the risk of errors. To address this issue, this work introduces the development of the LZFF package, which automates the preparation and formatting of files for genetic analyses, minimizing errors and promoting standardization. The data follow a structured workflow: from free format (L) to standardized format (P), and finally to the final format (F), compatible with the target software. Four main functions were developed: `rrcData`, which performs data reading, recoding, and verification; `rrcPed`, designed for structuring pedigree files with appropriate recoding; `formatA`, which prepares the data in the format required by BLUPF90; and `formatB`, which adjusts the data for use with Wombat. To validate the package, automated tests were implemented using the `testthat` package, ensuring robustness in the processes of data reading, recoding, and structural integrity of the processed datasets. The LZFF package thus provides an efficient and accessible solution for data preparation in genetic analyses, reducing the time required for manual formatting and enhancing the reproducibility of results.

Keywords: Genetic Evaluation. BlupF90. Software R. Reproducibility. Wombat.

LISTA DE FIGURAS

Figura 1 –Representação dos softwares nos artigos pesquisados entre os anos de 1997 e 2024	19
Figura 2 –Número de publicações encontradas associada a cada software entre os anos de 1997 e 2001.....	21
Figura 3 – Número de publicações encontradas associada a cada software entre os anos de 2002 até 2006.....	23
Figura 4 – Número de publicações encontradas associada a cada software entre os anos de 2007 até 2011	24
Figura 5 – Número de publicações encontradas associada a cada software entre os anos de 2012 até 2016.....	26
Figura 6 – Número de publicações encontradas associada a cada software entre os anos de 2017 até 2021	28
Figura 7 – Número de publicações encontradas associada a cada software entre os anos de 2022 até 2024.....	29
Figura 8 – A persistência dos softwares ao longo dos anos	30
Figura 9 – Representação dos diferentes estados de um arquivo conforme a formatação aplicada	42
Figura 10 – Visão Geral do Desenvolvimento do Pacote LZFF	44
Figura 11 – Apresenta o fluxo de trabalho da função rrcData	48
Figura 12 – Exemplo de um arquivo padrão (P).....	54
Figura 13 – Data frame criado pela função rrcData a partir de um arquivo de dados no formato padrão (P)	55
Figura 14 – Arquivo formatado pela função formataA	56
Figura 15 – Arquivo formatado pela formatB	57
Figura 16 – Arquivo com dados de Pedigree	58
Figura 17 – Data frame criado pela função rrcData a partir de um arquivo de dados com Pedigree	58
Figura 18 – Recodificação com os códigos originais	59
Figura 19 – Pedigree recodificado	60
Figura 20 – Conjunto de dados original	60

LISTA DE TABELAS

Tabela 1 - Número de publicações encontradas associada a cada software durante todo o período considerado na revisão	31
--	----

LISTA DE QUADROS

Quadro 1 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 1997 até 2001 com as últimas 5 publicações encontradas na seleção	20
Quadro 2 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2002 até 2006 com as últimas 5 publicações encontradas na seleção	22
Quadro 3 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2007 até 2011 com as últimas 5 publicações encontradas na seleção	23
Quadro 4 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2012 e 2016 com as últimas 5 publicações encontradas na seleção	25
Quadro 5 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2017 e 2021 com as últimas 5 publicações encontradas na seleção	27
Quadro 6 - Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2022 até 2024 com as últimas 5 publicações encontradas na seleção	28

LISTA DE ABREVIATURAS E SIGLAS

ASReml	Average Information REML
AIREMLF90	Average Information REML for F90
BLUPF90	Best Linear Unbiased Prediction for the 90s
DFREML	Derivative-Free REML
DMU	Danish Multivariate Program
GIBBSF90	Gibbs Sampling for BLUPF90
INTERGEM	Interface for Genetic Evaluation Models
MIXBLUP	MIXed models for Best Linear Unbiased Prediction
MTDFREML	Multiple Trait Derivative-Free REML
MTGSAM	Multiple Trait Gibbs Sampling for Animal Models
REMLF90	REML Estimation for F90
THRGIBBSF90	Threshold Gibbs Sampling F90
VCE	Variance Component Estimation
WOMBAT	A program for Mixed Model Analyses by Restricted Maximum Likelihood

SUMÁRIO

CAPÍTULO 1 – ESTUDO SOBRE A UTILIZAÇÃO DE SOFTWARES DE AVALIAÇÃO GENÉTICA E GENÔMICA NO MELHORAMENTO GENÉTICO ANIMAL	18
1 INTRODUÇÃO	18
2 REVISÃO DE LITERATURA	20
2.1 Levantamento sobre a utilização de Softwares de Avaliação Genética ..	20
REFERÊNCIAS.....	34
CAPÍTULO 2 – DESENVOLVIMENTO DO PACOTE LZFF	42
1 INTRODUÇÃO	44
2 PLANEJAMENTO E DESENVOLVIMENTO	45
3 FUNÇÕES.....	48
3.1 Função rrcData (Read, Recode and Check Data)	48
3.2 Função rrcPed	52
3.3 Função formatA	53
3.4 Função formatB	55
4 TESTES	56
5 RESULTADOS E DISCUSSÕES.....	56
6 CONCLUSÃO	63
REFERÊNCIAS.....	64
ANEXO 1 – Script Testes	65
ANEXO 2 – Script Funções.....	70

CAPÍTULO 1 – ESTUDO SOBRE A UTILIZAÇÃO DE SOFTWARES DE AVALIAÇÃO GENÉTICA E GENÔMICA NO MELHORAMENTO GENÉTICO ANIMAL

1 INTRODUÇÃO

Grande parte das pesquisas em Melhoramento Genético Animal envolve a análise de grandes volumes de dados, como estimativas de parâmetros genéticos, predição de valores genéticos e genômicos e análises de associação baseadas no genoma. Para todas essas, softwares foram desenvolvidos para auxiliar os pesquisadores na execução dessas análises e interpretação dos resultados.

Entretanto, para que os softwares retornem os resultados com qualidade, os dados precisam ser trabalhados, sendo submetidos a análises de consistência e, posteriormente, formatados para o padrão exigido pelo software específico. Erros nessas etapas podem inviabilizar a análise ou produzir resultados distorcidos e, portanto, sem utilidade para o pesquisador.

Além disso, mais recentemente, o conceito de reprodutibilidade dos resultados científicos tem ganhado importância e práticas para garanti-la vem sendo implementadas nas mais importantes revistas científicas, as quais passaram a exigir o depósito, junto com o artigo, dos dados utilizados e, em alguns poucos casos, dos scripts e rotinas utilizados nos procedimentos de análise. No entanto, raramente são disponibilizados junto com os artigos as rotinas utilizadas para as etapas de consistência dos dados e principalmente para a formatação dos dados.

Especificamente na etapa de formatação dos dados para os softwares de avaliação genética, a automação é possível e contribuirá com a minimização de erros na preparação e a garantia de padronização dessa etapa, aumentando as chances da reprodutibilidade dos resultados por outros grupos de pesquisa.

A formatação dos dados de acordo com o padrão exigido pelo software pode, para alguns casos, possuir muitas regras e demandar bastante tempo e atenção para ser realizada. Pode ser especialmente complexa e desafiadora quando o software está sendo utilizado pelas primeiras vezes.

Erros nessa etapa são facilmente detectados se o software não conseguir realizar a análise. Por outro lado, em algumas situações, os erros não impedem a análise de ser executada, tendo como consequência, a existência de vários arquivos para um mesmo conjunto de dados sendo trabalhado por diferentes equipes. Nesse

cenário, a repetibilidade dos resultados está comprometida, uma vez que os resultados deverão ser diferentes para os mesmos dados e análises, não permitindo a confirmação dos resultados.

Para que os pesquisadores dediquem mais tempo à interpretação dos resultados e menos à formatação manual dos arquivos e que os resultados saiam de forma consistente e com qualidade, o desenvolvimento de funções de conversão dos arquivos de dados do formato bruto para o formato demandado pelo software de análise é muito importante. Além disso, a padronização automatizada aumenta a reprodutibilidade e o uso do software.

Assim, essas funções podem ser muito úteis para uniformizar os trabalhos de equipe, onde várias pessoas/pesquisadores estão dedicadas à utilização de softwares para a análise de dados e, principalmente, para consórcios onde diferentes grupos de pesquisa trabalham nas tarefas de análise, padronizando procedimentos dentro da equipe ou entre equipes.

Contudo, as funções devem ser facilmente acessíveis aos pesquisadores e outros profissionais da área. Dessa forma, a implementação dessas no sistema R, sob o formato de um pacote, dará rápido acesso aos interessados, uma vez que pode ser instalado de forma gratuita nos computadores para os principais sistemas operacionais em uso atualmente e, já vem sendo massivamente utilizado pelos pesquisadores da área de Melhoramento Genético Animal.

2 REVISÃO DE LITERATURA

2.1 Levantamento sobre a utilização de Softwares de Avaliação Genética

Ao longo dos anos, vários softwares para estimação de parâmetros genéticos e avaliação genética/genômica surgiram para auxiliar na pesquisa e aplicações práticas. Alguns deles já não são mais utilizados enquanto outros tiveram redução na sua utilização. Para estudar a dinâmica de utilização dos softwares durante o tempo e identificar quais os mais utilizados nos anos mais recentes, foi realizada uma avaliação dos softwares encontrados na literatura científica em intervalos de 5 anos, desde 1997 até 2024.

Nas buscas realizadas, consideraram-se os trabalhos que tinham como objetivo estimar parâmetros genéticos e/ou predizer valores genéticos aditivos ou genômicos. Não se levou em consideração a espécie trabalhada, ou seja, artigos de todas as espécies foram considerados. Foram excluídos artigos de revista sem licença de acesso. Para evitar resultados relacionados a plantas e humanos, utilizou-se o termo “animal” em nossa estratégia de busca nos periódicos CAPES e Scopus:

"genetic evaluation" OR "genomic evaluation" OR "genetic parameter" OR "genomic parameter" AND "animal"

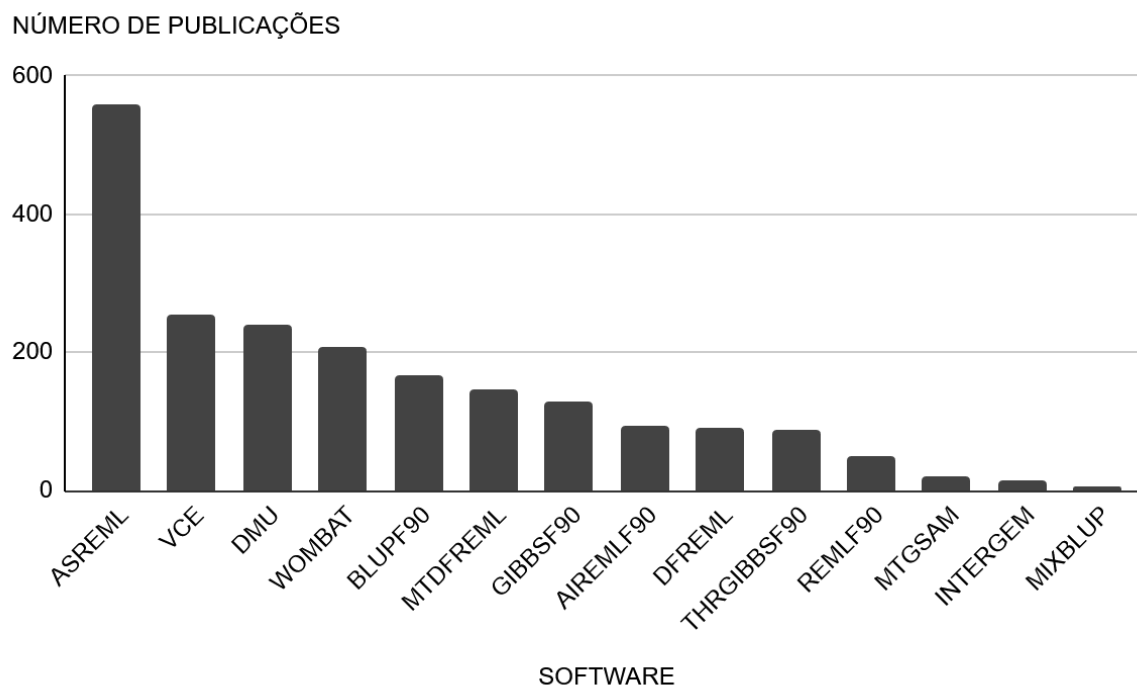
Essa expressão exige que pelo menos um dos quatro primeiros termos deva aparecer concomitantemente com o termo “animal” nos artigos para que fossem selecionados.

Os artigos retornados, posteriormente, foram filtrados para que mostrassem somente resultados contidos em periódicos revisados por pares, na língua inglesa e da área de ciência animal. Ao final 4.985 artigos foram retornados, dos quais 2.067 foram selecionados para a revisão.

O objetivo não foi comparar os softwares tecnicamente, mas sim quantificar sua frequência de uso na literatura científica.

A figura abaixo mostra a representação dos softwares no período total considerado.

Figura 1- Representação dos softwares usados nos artigos pesquisados entre os anos de 1997 e 2024.



Fonte: Elaborada pelo autor.

O gráfico mostra que, ao longo de todo o período considerado, os três softwares mais utilizados nos artigos selecionados foram ASReml, VCE e DMU. O software ASReml é um software proprietário, exigindo, portanto, o pagamento de licenças para que seja utilizado. Os outros dois softwares já não são mais mantidos pelos desenvolvedores e não exigem licença para utilização pela comunidade científica. Os três softwares foram desenvolvidos na década de 90 e intensamente utilizados nesse período pela relativa escassez de softwares da época. Entretanto, como o ASReml continuou sendo atualizado e comercializado até o presente, sua utilização foi a maior entre os demais softwares.

É interessante analisar também os principais softwares utilizados dentro de períodos de 5 anos para analisar com mais objetividade o efeito do tempo nas suas estatísticas de utilização. Os resultados estão apresentados nos quadros 1 a 6 e nas Figuras 2 a 7.

Quadro 1- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 1997 até 2001 com as últimas 5 publicações encontradas na seleção.

Software	Nº de trabalhos	Publicações ¹
VCE	45	(Johnston <i>et al.</i> , 2001). (Robinson <i>et al.</i> , 2001). (Röhe <i>et al.</i> , 2001). (Westhuizen <i>et al.</i> , 2001). (Wolf; Peškovičová; Groeneveld, 2001).
MTDFREML	37	(Amin, 2001). (Bureau <i>et al.</i> , 2001). (Meyer <i>et al.</i> , 2001). (Ruíz-Flores; Johnson, 2001). (Shanks <i>et al.</i> , 2001).
DFREML	19	(Al-Shorepy, 2001). (Dyk; Neser; Kanfer, 2001). (Hananberg; Knol; Merks, 2001). (Helmink <i>et al.</i> , 2001). (Palacios-Espinosa <i>et al.</i> , 2001).

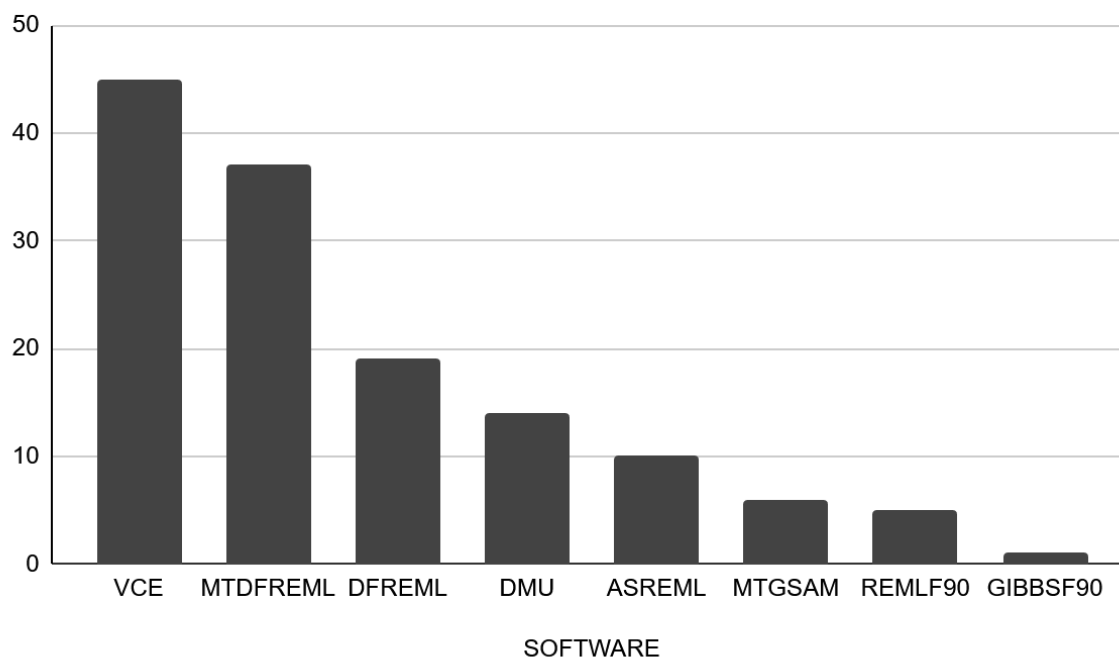
1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

A Figura 2 mostra, graficamente, os dados de frequência de utilização dos softwares mostrados no Quadro 1.

Figura 2- Número de publicações encontradas associada a cada software entre os anos de 1997 e 2001.

NÚMERO DE PUBLICAÇÕES



Fonte: Elaborada pelo autor.

A análise Do Quadro 1 e da Figura 2 mostram os softwares utilizados durante esse período. O software DFREML foi publicado em 1988 enquanto os softwares VCE e MTDFREML são do início da década (1990 e 1993, respectivamente). Os três continuaram a ser bastante utilizados até o início dos anos 2000. Nesse período o ASReml foi utilizado em um número mais modesto de trabalhos. O DMU, que também foi razoavelmente utilizado, apresentava algumas dificuldades para a confecção do arquivo de parâmetros (Miztal, 1994) e o ASReml se encontrava nos seus primeiros anos com sua utilização já sendo cobrada.

Os próximos 5 anos mostram um considerável aumento de utilização do software ASReml. Com o desenvolvimento das capacidades do software e o seu reconhecimento pela comunidade científica, o ASReml aumentou sua participação nos artigos em mais de 5 vezes. O VCE ainda continuava a ser muito utilizado para estimação de parâmetros genéticos e o MTDFREML, pela sua amplitude de modelos disponíveis e relativa facilidade de uso também continuaram com consideráveis contribuições nos trabalhos. (Quadro 2 e Figura 3).

Quadro 2- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2002 até 2006 com as últimas 5 publicações encontradas na seleção.

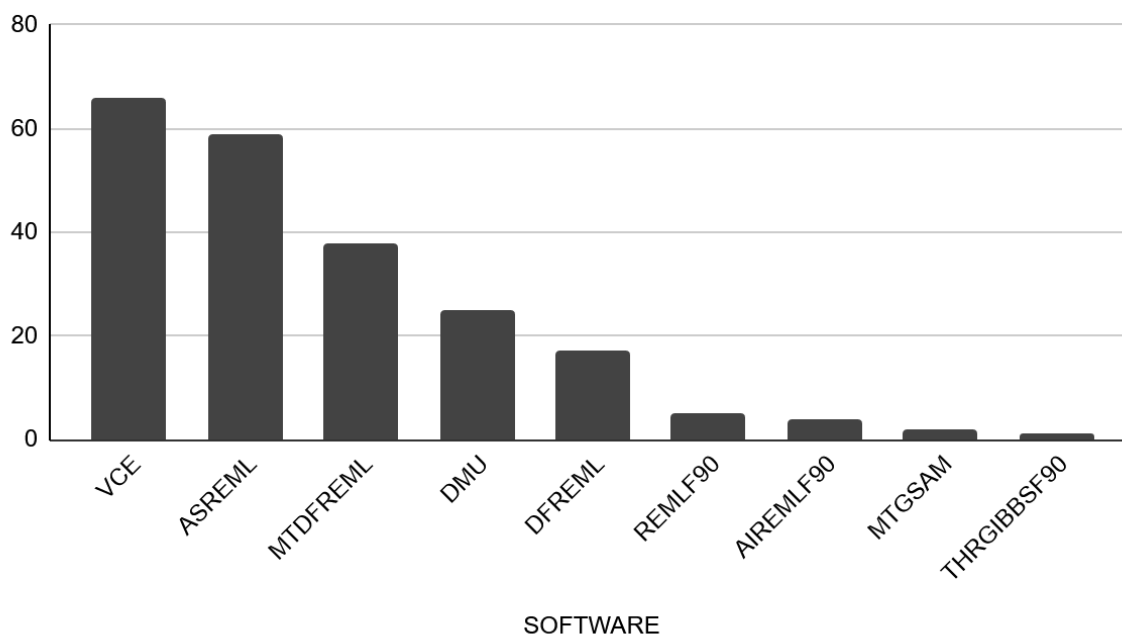
Software	Nº de trabalhos	Publicações ¹
VCE	66	(Allain; Roguet, 2006). (Gäde <i>et al.</i> , 2006). (Kenttämies; Nikkilä; Miettinen, 2006). (Norris; Ngambi, 2006). (Toth <i>et al.</i> , 2006).
ASReml	59	(Banos <i>et al.</i> , 2006). (Lambe <i>et al.</i> , 2006). (Marie-Etancelin <i>et al.</i> , 2006). (Quinn <i>et al.</i> , 2006). (Waldmann; Ericsson, 2006).
MTDFREML	38	(Angulo <i>et al.</i> , 2006). (Gaya <i>et al.</i> , 2006). (Ibi; Kahi; Hirooka, 2006). (Salces <i>et al.</i> , 2006). (Sasaki <i>et al.</i> , 2006).

1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

Figura 3- Número de publicações encontradas associada a cada software entre os anos de 2002 até 2006.

NÚMERO DE PUBLICAÇÕES



Fonte: Elaborada pelo autor.

Quadro 3- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2007 até 2011 com as últimas 5 publicações encontradas na seleção.

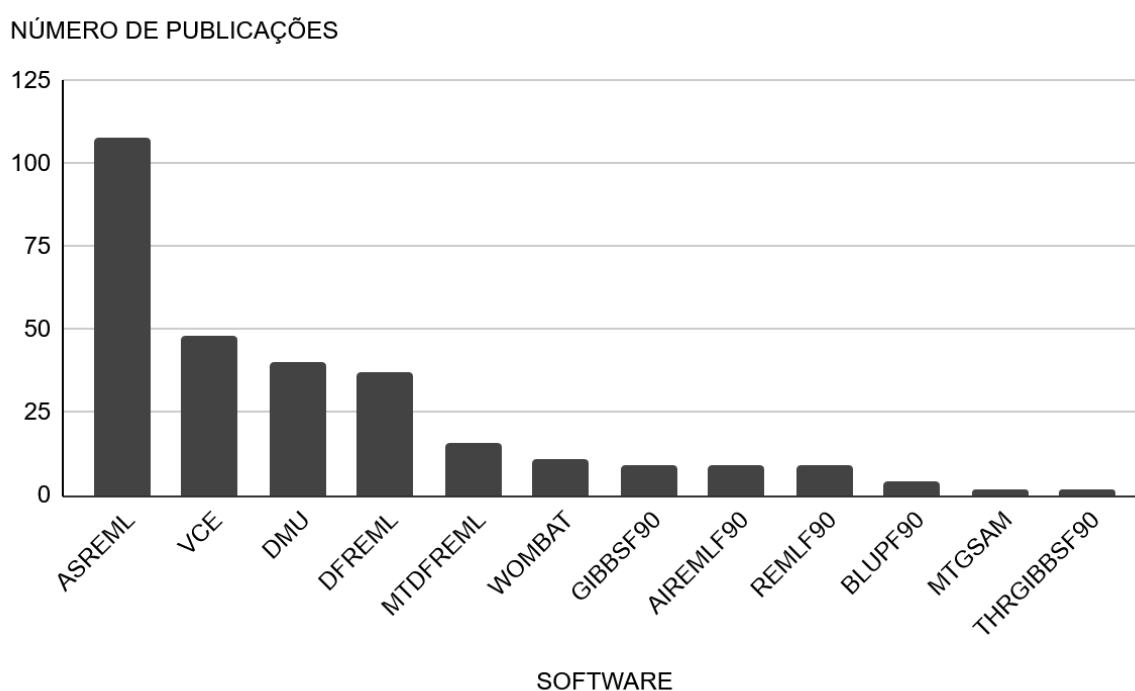
Software	Nºde trabalhos	Publicações ¹
ASReml	108	(Buttchereit <i>et al.</i> , 2011). (Ghavi Hossein-Zadeh; Ardalan, 2011). (Mc Hugh <i>et al.</i> , 2011). (Niranjan; Sharma; Gowane, 2011). (Van Hulzen <i>et al.</i> , 2011).
VCE	48	(Frank <i>et al.</i> , 2011). (Nagy <i>et al.</i> , 2011). (Ragab; Baselga, 2011). (Rupp <i>et al.</i> , 2011). (Vostrý <i>et al.</i> , 2011).
DMU	40	(Albertsdóttir <i>et al.</i> , 2011).

		(Henderson <i>et al.</i> , 2011). (Kause <i>et al.</i> , 2011). (Ødegård <i>et al.</i> , 2011). (Sewalem; Miglior; Kistemaker, 2011).
--	--	--

1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

Figura 4- Número de publicações encontradas associada a cada software entre os anos de 2007 até 2011.



Fonte: Elaborada pelo autor.

Entre 2007 e 2011 (Quadro 3 e Figura 4), observa que a utilização do ASREML praticamente triplicou em relação ao período anterior. Apesar da exigência de licença para uso, a percepção de qualidade do software entre a comunidade científica e a indústria, explicam esse aumento expressivo. O software DMU embora tenha sofrido algumas oscilações na sua participação em artigos científicos, continuou sendo amplamente utilizado nesses cinco anos, possivelmente por melhorias no software e na documentação. Nesse período, o software VCE registrou a sua menor participação desde 1997, sofrendo uma maior concorrência com o surgimento e amadurecimento de alguns softwares como Wombat e BLUPF90.

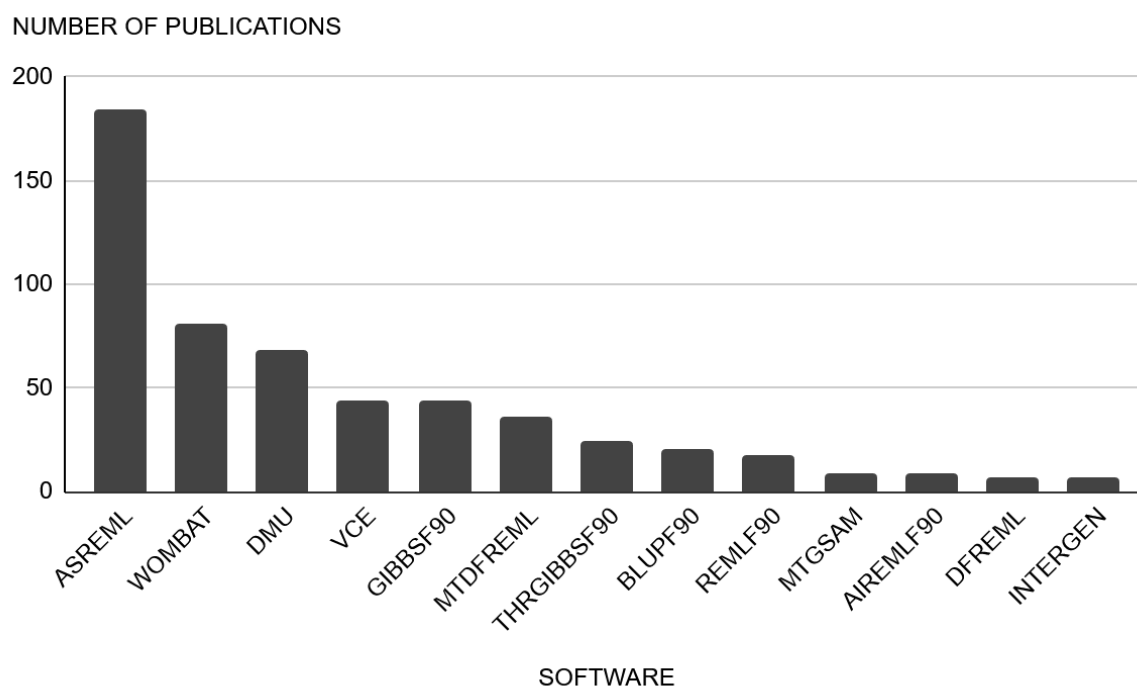
Quadro 4- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2012 e 2016 com as últimas 5 publicações encontradas na seleção.

Software	Nº de trabalhos	Publicações ¹
ASReml	184	(Mcrae <i>et al.</i> , 2016). (Sui <i>et al.</i> , 2016). (Thekkoot <i>et al.</i> , 2016). (Tsiamadis <i>et al.</i> , 2016). (Twomey <i>et al.</i> , 2016).
Wombat	81	(Do <i>et al.</i> , 2016). (Jafari; Manafiazar, 2016). (Lee <i>et al.</i> , 2016). (Meyer, 2016). (Rovadoski <i>et al.</i> , 2016).
DMU	68	(Ismael <i>et al.</i> , 2016). (Jaton <i>et al.</i> , 2016). (Martinsen <i>et al.</i> , 2016). (McIaren <i>et al.</i> , 2016). (Yang; Su, 2016).

1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

Figura 5- Número de publicações encontradas associada a cada software entre os anos de 2012 até 2016.



Fonte: Elaborada pelo autor.

Entre 2012 e 2016 (Quadro 4 e Figura 5), o ASREML manteve sua predominância nos trabalhos selecionados, entretanto o software Wombat aumentou sua participação de forma considerável, sendo aproximadamente quatro vezes mais utilizado do que nos cinco anos anteriores. Uma das principais vantagens do Wombat era sua documentação detalhada, que facilita a aplicação do software.

O DMU aumentou sua participação nos artigos analisados, de 40 para 68 comparado aos cinco anos anteriores. Pelos motivos já citados, o software conseguiu expandir significativamente sua base de usuários.

Já nos cinco anos anteriores e, mas mais notadamente, entre 2012 e 2016, os softwares da família BLUPF90 (BLUPF90, REMLF90, AIREMLF90, GIBBSF90 e THRGIBBSF90) tiveram um aumento na participação em trabalhos científicos. Mas, devido à sua estrutura, as contagens de utilização foram distribuídas entre os diferentes softwares da família. Se a contagem considerasse todos os softwares da família BLUPF90, sua utilização totalizava 115 no período atual e 33 no período anterior, posicionando-os, em conjunto, entre os mais utilizados. A grande variedade de modelos e metodologias suportadas por esses softwares, juntamente com a melhoria na documentação explica o aumento verificado nos trabalhos selecionados.

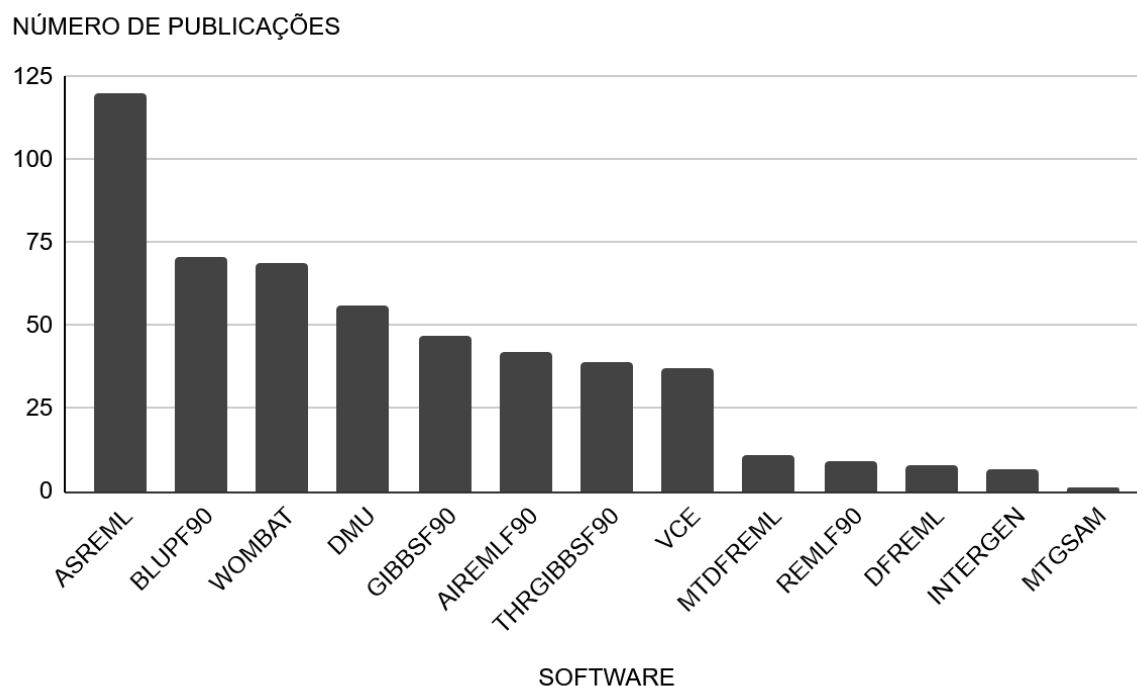
Quadro 5- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2017 e 2021 com as últimas 5 publicações encontradas na seleção.

Software	Nºde trabalhos	Publicações ¹
ASReml	120	(Cloete <i>et al.</i> , 2021). (Cobo; Raoul; Bodin, 2021). (David; Huynh Tran; Gilbert, 2021). (Duff <i>et al.</i> , 2021). (Vargovic <i>et al.</i> , 2021).
BlupF90	71	(Bermann <i>et al.</i> , 2021). (Gómez <i>et al.</i> , 2021). (Mehrban <i>et al.</i> , 2021). (Shi <i>et al.</i> , 2021). (Tarekegn <i>et al.</i> , 2021).
Wombat	69	(Ahmad <i>et al.</i> , 2021). (Al Kalaldehy <i>et al.</i> , 2021). (Bangar <i>et al.</i> , 2021). (Dominguez-Castaño <i>et al.</i> , 2021). (Salem; Nasr; Amin, 2021).

1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

Figura 6- Número de publicações encontradas associada a cada software entre os anos de 2017 até 2021.



Fonte: Elaborada pelo autor.

O Quadro 5 e a Figura 6 mostram que o software ASREML continuou sendo o mais utilizado pela comunidade científica, entretanto, o software BLUPF90 aumentou consideravelmente sua participação. Nesse período, os softwares já estavam incorporando as análises genômicas e, o BLUPF90 foi um dos primeiros a implementar esse tipo de análise, tornando-se amplamente utilizado. O software Wombat incorporou algumas análises genômicas, mas de forma mais limitada. Sua utilização foi mantida para análises de estimação de parâmetros genéticos e predição de valores genéticos com a utilização de dados fenotípicos e de pedigree.

Quadro 6- Softwares de avaliação genética/estimação de parâmetros genéticos mais utilizados entre os anos de 2022 até 2024 com as últimas 5 publicações encontradas na seleção.

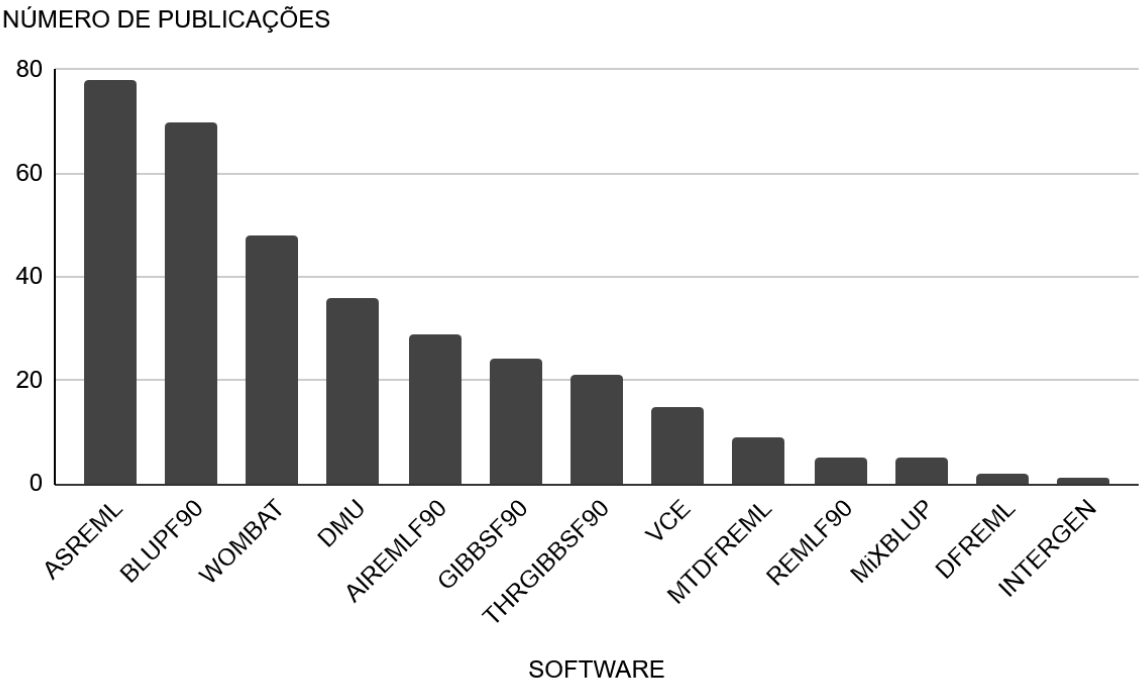
Software	N° de trabalhos	Publicações ¹
ASReml	78	(Munguía Vásquez <i>et al.</i> , 2024). (Pacheco <i>et al.</i> , 2024). (Paneru; Moghaddar; Van Der Werf, 2024). (Smith; Walkom; Clark, 2024).

		(Vargas Jurado <i>et al.</i> , 2024).
BlupF90	70	(Brunes <i>et al.</i> , 2024). (Lázaro <i>et al.</i> , 2024). (Mancin <i>et al.</i> , 2024). (Nel <i>et al.</i> , 2024). (Perdomo-González <i>et al.</i> , 2024).
Wombat	48	(Ghafouri-Kesbi; Eskandarinasab, 2024). (Kizilaslan; Arzik; Behrem, 2024). (Koçak <i>et al.</i> , 2024). (Tamilarasan <i>et al.</i> , 2024). (Miyumo <i>et al.</i> , 2023).

1 = amostra das cinco últimas publicações encontradas

Fonte: Elaborada pelo autor.

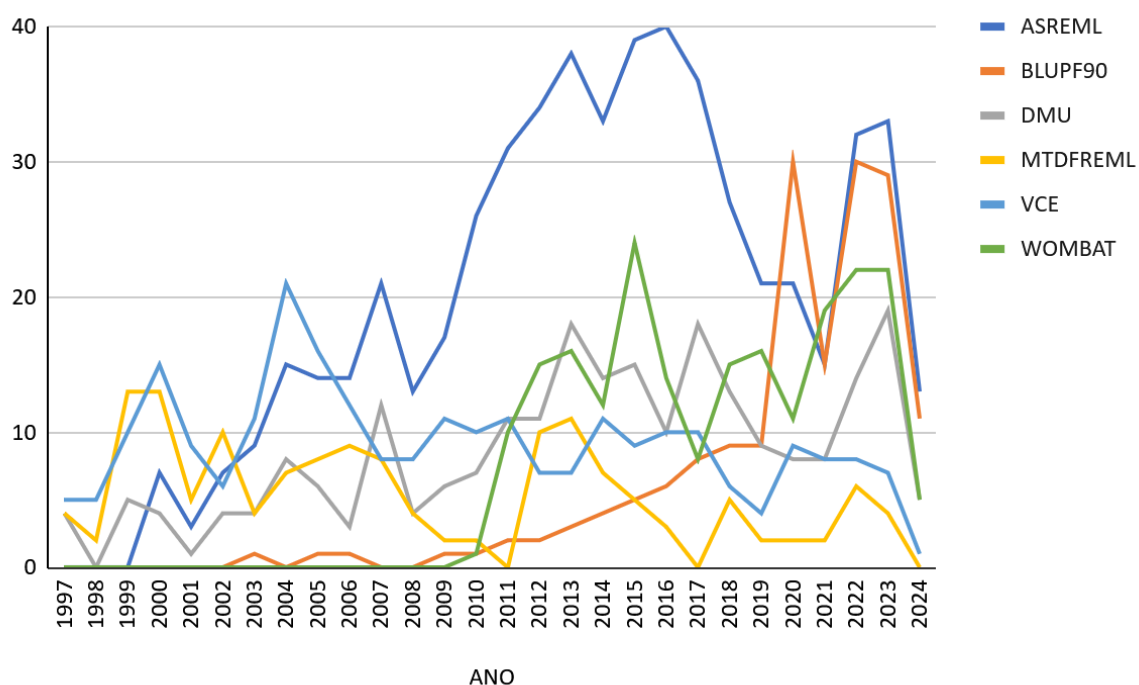
Figura 7. Número de publicações encontradas associada a cada software entre os anos de 2022 até 2024.



Fonte: Elaborada pelo autor.

No último período de dois anos (2022 a 2024), os softwares ASREML e BLUPF90 foram utilizados de forma equilibrada. A implementação de análises genômicas nesses softwares é um fator decisivo para essas estatísticas parciais do biênio. O software Wombat, ainda bastante utilizado não recebe mais atualizações e oferece poucas análises genômicas, o que deve levar à redução de sua utilização ao longo dos anos. Assim como o Wombat, o software DMU continua sendo amplamente utilizado.

Figura 8- A persistência dos softwares ao longo dos anos.



Fonte: Elaborada pelo autor.

Pela Figura 8, nota-se que o software ASREML foi amplamente utilizado entre 2012 e 2016, e sempre se entre os mais utilizados nos demais anos, quando comparado a outros softwares. Os softwares VCE e MTDFREML, foram amplamente utilizados nos anos 90 e no início dos anos 2000, mas perderam relevância a partir de 2010. O software DMU ainda apresenta uma boa persistência, embora não tenha sido o software mais utilizado em nenhum dos períodos considerados. O software Wombat passou a ser amplamente utilizado a partir de 2010 e ainda é frequentemente encontrado na literatura, mas, devido a paralização das atualizações sua

representatividade deve diminuir nos próximos anos, assim como o DMU. O software Blupf90 aumentou sua participação a partir de 2015 e continua sendo amplamente utilizado devido a sua boa documentação e atualizações constantes. Se considerarmos toda a família de softwares desenvolvidos para fins de análises genéticas e genômicas, esse conjunto é o segundo mais utilizado com 477 participações em artigos, ficando atrás somente do ASREML, com 559 citações (Tabela 1). A tendência é que o software BLUPF90 seu correlatos aumentem sua participação nos artigos científicos nos próximos anos, uma vez que oferecem muitas funcionalidades para análises genéticas e genômicas, tornando-se o principal concorrente do ASReml. Finalmente, a Figura 7, mostra o total de participações dos softwares pesquisados nesse trabalho.

Tabela 1- Número de publicações encontradas associada a cada software durante todo o período considerado na revisão.

SOFTWARE	Nº de Publicações
ASREML	559
VCE	255
DMU	239
WOMBAT	209
BLUPF90	168
MTDFREML	147
GIBBSF90	129
AIREMLF90	93
DFREML	90
THRGIBBSF90	87
REMLF90	51
MTGSAM	20
INTERGEM	15
MIXBLUP	5

Fonte: Elaborada pelo autor.

3 AGRADECIMENTO(S)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) processo 2024/10028-4.

REFERÊNCIAS

- AHMAD, S. F.; KHAN, N. N.; GANAI, N. A.; SHANAZ, S.; RATHER, M. A.; ALAM, S. Multivariate quantitative genetic analysis of body weight traits in Corriedale sheep. **Tropical Animal Health and Production**, v. 53, n. 2, p. 197, 2021.
- AL KALALDEH, M.; SWAMINATHAN, M.; GAUNDARE, Y.; JOSHI, S.; ALILOO, H.; STRUCKEN, E. M.; DUCROCQ, V.; GIBSON, J. P. Genomic evaluation of milk yield in a smallholder crossbred dairy production system in India. **Genetics Selection Evolution**, v. 53, n. 1, p. 73, 2021.
- ALBERTSDÓTTIR, E.; ERIKSSON, S.; SIGURDSSON, Á.; ÁRNASON, T. Genetic analysis of 'breeding field test status' in Icelandic horses. **Journal of Animal Breeding and Genetics**, v. 128, n. 2, p. 124–132, 2011.
- ALLAIN, D.; ROGUET, J. M. Genetic and non-genetic variability of OFDA-medullated fibre contents and other fleece traits in the French Angora goats. **Small Ruminant Research**, v. 65, n. 3, p. 217–222, 2006.
- AL-SHOREPY, S. A. Estimates of genetic parameters for direct and maternal effects on birth weight of local sheep in United Arab Emirates. **Small Ruminant Research**, v. 39, n. 3, p. 219–224, 2001.
- AMIN, A. A. Lactation and Sample Test-Day Multi-trait animal model for genetic evaluation of somatic cell scores in Hungarian Holstein-Friesian crossbreeds. **Archives Animal Breeding**, v. 44, n. 3, p. 263–276, 10 out. 2001.
- ANGULO, R.; AGUDELO-GÓMEZ, D.; CERÓN-MUÑOZ, M. F.; JARAMILLO-BOTERO, S. Genetic parameters in buffalo calves fed at full milk in beef production system in middle Magdalena region of Colombia. **Livestock Research for Rural Development**, v. 18, n. 12, 2006.
- BANGAR, Y. C.; MAGOTRA, A.; MALIK, B. S.; MALIK, Z. S.; YADAV, A. S. Evaluating advanced computing techniques for predicting breeding values in Harnali sheep. **Tropical Animal Health and Production**, v. 53, n. 2, p. 313, 2021.
- BANOS, G.; COFFEY, M. P.; WALL, E.; BROTHERSTONE, S. Genetic Relationship Between First-Lactation Body Energy and Later-Life Udder Health in Dairy Cattle. **Journal of Dairy Science**, v. 89, n. 6, p. 2222–2232, 2006.
- BERMANN, M.; LOURENCO, D.; BREEN, V.; HAWKEN, R.; BRITO LOPES, F.; MISZTAL, I. Modeling genetic differences of combined broiler chicken populations in single-step GBLUP. **Journal of Animal Science**, v. 99, n. 4, p. skab056, 2021.
- BRUNES, L. C.; BALDI, F.; COSTA, M. F. O. E.; LOBO, R. B.; LOPES, F. B.; MAGNABOSCO, C. U. Effect of age at first conception as a selection criterion on growth and carcass traits in Nellore cattle. **Ciência Animal Brasileira**, v. 25, p. e, 2024.
- BUREAU, F.; MICHAUX, C.; COGHE, J.; UYSTEPRUYST, C.; LEROY, P. L.; LEKEUX, P. Spirometric performance in Belgian Blue calves: II. Analysis of environmental factors and estimation of genetic parameters. **Journal of Animal Science**, v. 79, n. 5, p. 1162, 2001.
- BUTTCHEREIT, N.; STAMER, E.; JUNGE, W.; THALLER, G. *Short communication:* Genetic relationships among daily energy balance, feed intake, body condition score,

and fat to protein ratio of milk in dairy cows. **Journal of Dairy Science**, v. 94, n. 3, p. 1586–1591, 2011.

CLOETE, S. W. P.; THUTWA, K.; SCHOLTZ, A. J.; CLOETE, J. J. E.; DZAMA, K.; GILMOUR, A. R.; VAN WYK, J. B. Breed effects and heterosis for weight traits and tick count in a cross between an indigenous fat-tailed and a commercial sheep breed. **Tropical Animal Health and Production**, v. 53, n. 1, p. 165, 2021.

COBO, E.; RAOUL, J.; BODIN, L. Genetic parameters of litter weight, an alternative criterion to prolificacy and pre-weaning weight for selection of French meat sheep. **Livestock Science**, v. 250, p. 104596, 1 ago. 2021.

DAVID, I.; HUYNH TRAN, V.-H.; GILBERT, H. New residual feed intake criterion for longitudinal data. **Genetics Selection Evolution**, v. 53, n. 1, p. 53, 25 jun. 2021.

DO, C.; PARK, B.; KIM, S.; CHOI, T.; YANG, B.; PARK, S.; SONG, H. Genetic Parameter Estimates of Carcass Traits under National Scale Breeding Scheme for Beef Cattle. **Asian-Australasian Journal of Animal Sciences**, v. 29, n. 8, p. 1083–1094, 2016.

DOMINGUEZ-CASTAÑO, P.; TORO OSPINA, A. M.; EL FARO, L.; DE VASCONCELOS SILVA, J. A. I. Genetic principal components for reproductive and productive traits in Holstein cows reared under tropical conditions. **Tropical Animal Health and Production**, v. 53, n. 2, p. 193, 2021.

DUFF, C. J.; VAN DER WERF, J. H. J.; PARNELL, P. F.; CLARK, S. A. Comparison of two live-animal ultrasound systems for genetic evaluation of carcass traits in Angus cattle. **Translational Animal Science**, v. 5, n. 1, p. txab011, 2021.

DYK, R. V.; NESER, F. W. C.; KANFER, F. H. The effect of selection on genetic parameter estimates. **South African Journal of Animal Science**, v. 31, n. 2, p. 107–114, 2001.

FRANK, E. N.; HICK, M. V. H.; MOLINA, M. G.; CARUSO, L. M. Genetic parameters for fleece weight and fibre attributes in Argentinean Llamas reared outside the Altiplano. **Small Ruminant Research**, v. 99, n. 1, p. 54–60, 2011.

GÄDE, S.; STAMER, E.; JUNGE, W.; KALM, E. Estimates of genetic parameters for milkability from automatic milking. **Livestock Science**, v. 104, n. 1, p. 135–146, 2006.

GAYA, L. G.; FERRAZ, J. B. S.; REZENDE, F. M.; MOURÃO, G. B.; MATTOS, E. C.; ELER, J. P.; MICHELAN FILHO, T. Heritability and Genetic Correlation Estimates for Performance and Carcass and Body Composition Traits in a Male Broiler Line. **Poultry Science**, v. 85, n. 5, p. 837–843, 2006.

GHAFOURI-KESBI, F.; ESKANDARINASAB, M. Autosomal and sex-linked genetic parameters for body weight gain and its relationship with efficiency-related traits in sheep. **Small Ruminant Research**, v. 232, p. 107228, 1 mar. 2024.

GHAVI HOSSEIN-ZADEH, N.; ARDALAN, M. Genetic relationship between milk urea nitrogen and reproductive performance in Holstein dairy cows. **Animal**, v. 5, n. 1, p. 26–32, 1 jan. 2011.

GÓMEZ, M.; ROSSI, D.; CIMMINO, R.; ZULLO, G.; GOMBIA, Y.; ALTIERI, D.; DI PALO, R.; BIFFANI, S. Accounting for Genetic Differences Among Unknown Parents

in *Bubalus bubalis*: A Case Study From the Italian Mediterranean Buffalo. **Frontiers in Genetics**, v. 12, 2021.

HANENBERG, E. H. A. T.; KNOL, E. F.; MERKS, J. W. M. Estimates of genetic parameters for reproduction traits at different parities in Dutch Landrace pigs. **Livestock Production Science**, v. 69, n. 2, p. 179–186, 1 maio 2001.

HELMINK, S. K.; RODRIGUEZ-ZAS, S. L.; SHANKS, R. D.; LEIGHTON, E. A. Estimated genetic parameters for growth traits of German shepherd dog and Labrador retriever dog guides¹. **Journal of Animal Science**, v. 79, n. 6, p. 1450–1456, 2001.

HENDERSON, L.; MIGLIOR, F.; SEWALEM, A.; KELTON, D.; ROBINSON, A.; LESLIE, K. E. Estimation of genetic parameters for measures of calf survival in a population of Holstein heifer calves from a heifer-raising facility in New York State. **Journal of Dairy Science**, v. 94, n. 1, p. 461–470, 2011.

IBI, T.; KAH, A. K.; HIROOKA, H. Effect of carcass price fluctuations on genetic and economic evaluation of carcass traits in Japanese Black cattle¹. **Journal of Animal Science**, v. 84, n. 12, p. 3204–3211, 1 abr. 2006.

ISMAEL, A.; STRANDBERG, E.; BERGLUND, B.; FOGH, A.; LØVENDAHL, P. Seasonality of fertility measured by physical activity traits in Holstein cows. **Journal of Dairy Science**, v. 99, n. 4, p. 2837–2848, 2016.

JAFARI, S.; MANAFIAZAR, G. Estimates of genetic parameters for lifetime reproductive performance traits in Makuie ewes. **Small Ruminant Research**, v. 139, p. 67–72, 1 jun. 2016.

JATON, C.; KOECK, A.; SARGOLZAEI, M.; MALCHIODI, F.; PRICE, C. A.; SCHENKEL, F. S.; MIGLIOR, F. Genetic analysis of superovulatory response of Holstein cows in Canada. **Journal of Dairy Science**, v. 99, n. 5, p. 3612–3623, 2016.

JOHNSTON, D. J.; REVERTER, A.; ROBINSON, D. L.; FERGUSON, D. M. Sources of variation in mechanical shear force measures of tenderness in beef from tropically adapted genotypes, effects of data editing and their implications for genetic parameter estimation. **Australian Journal of Experimental Agriculture**, v. 41, n. 7, p. 991, 2001.

KAUSE, A.; QUINTON, C.; AIRAKSINEN, S.; RUOHONEN, K.; KOSKELA, J. Quality and production trait genetics of farmed European whitefish, *Coregonus lavaretus*¹. **Journal of Animal Science**, v. 89, n. 4, p. 959–971, 2011.

KENTTÄMIES, H.; NIKKILÄ, M.; MIETTINEN, M. Phenotypic and genetic parameters and responses in temperament of silver fox cubs in a selection experiment for confident behaviour. **Agricultural and Food Science**, v. 15, n. 3, p. 340–349, 3 jan. 2006.

KIZILASLAN, M.; ARZIK, Y.; BEHREM, S. Genetic parameters for ewe lifetime productivity traits in Central Anatolian Merino sheep. **Small Ruminant Research**, v. 233, p. 107235, 1 abr. 2024.

KOÇAK, S.; ÇINKAYA, S.; TEKERLI, M.; DEMIRTAŞ, M.; BOZKURT, Z.; ÇELIKELOĞLU, K.; HACAN, Ö.; ERDOĞAN, M. Estimation of (Co) Variance Components and Genetic Parameters for Pre- and Post-Weaning Growth Traits in Dağlıç Sheep. **Animals**, v. 14, n. 1, p. 108, 2024.

LAMBE, N. R.; NAVAJAS, E. A.; SIMM, G.; BÜNGER, L. A genetic investigation of various growth models to describe growth of lambs of two contrasting breeds¹. **Journal of Animal Science**, v. 84, n. 10, p. 2642–2654, 2006.

LÁZARO, S. F.; TONHATI, H.; OLIVEIRA, H. R.; SILVA, A. A.; SCALEZ, D. C. B.; NASCIMENTO, A. V.; SANTOS, D. J. A.; STEFANI, G.; CARVALHO, I. S.; SANDOVAL, A. F.; BRITO, L. F. Genetic parameters and genome-wide association studies for mozzarella and milk production traits, lactation length, and lactation persistency in Murrah buffaloes. **Journal of Dairy Science**, v. 107, n. 2, p. 992–1021, 2024.

LEE, S.; CHO, K.; PARK, M.; CHOI, T.; KIM, S.; DO, C. Genetic Parameters of Milk β -Hydroxybutyric Acid and Acetone and Their Genetic Association with Milk Production Traits of Holstein Cattle. **Asian-Australasian Journal of Animal Sciences**, v. 29, n. 11, p. 1530–1540, 2016.

MANCIN, E.; GOMEZ PROTO, G.; TULIOZI, B.; SCHIAVO, G.; BOVO, S.; FONTANESI, L.; SARTORI, C.; MANTOVANI, R. Uncovering genetic parameters and environmental influences on fertility, milk production, and quality in autochthonous Reggiana cattle. **Journal of Dairy Science**, v. 107, n. 2, p. 956–977, 2024.

MARIE-ETANCELIN, C.; MANFREDI, E.; AUREL, M.-R.; PAILLER, F.; ARHAINX, J.; RICARD, E.; LAGRIFFOUL, G.; GUILLOUET, P.; BIBÉ, B.; BARILLET, F. Genetic analysis of milking ability in Lacaune dairy ewes. **Genetics Selection Evolution**, v. 38, n. 2, p. 183, 2006.

MARTINSEN, K. H.; ØDEGÅRD, J.; AASMUNDSTAD, T.; OLSEN, D.; MEUWISSEN, T. H. E. Genetic relationships between boar feed efficiency and sow piglet production, body condition score, and stayability in Norwegian Landrace pigs¹. **Journal of Animal Science**, v. 94, n. 8, p. 3159–3168, 2016.

MC HUGH, N.; EVANS, R. D.; AMER, P. R.; FAHEY, A. G.; BERRY, D. P. Genetic parameters for cattle price and body weight from routinely collected data at livestock auctions and commercial farms. **Journal of Animal Science**, v. 89, n. 1, p. 29–39, 2011.

MCLAREN, A.; MUCHA, S.; MRODE, R.; COFFEY, M.; CONINGTON, J. Genetic parameters of linear conformation type traits and their relationship with milk yield throughout lactation in mixed-breed dairy goats. **Journal of Dairy Science**, v. 99, n. 7, p. 5516–5525, 2016.

MCRAE, K. M.; BAIRD, H. J.; DODDS, K. G.; BIXLEY, M. J.; CLARKE, S. M. Incidence and heritability of ovine pneumonia, and the relationship with production traits in New Zealand sheep. **Small Ruminant Research**, v. 145, p. 136–141, 2016.

MEHRBAN, H.; NASERKHEIL, M.; LEE, D.; IBÁÑEZ-ESCRICHE, N. Multi-Trait Single-Step GBLUP Improves Accuracy of Genomic Prediction for Carcass Traits Using Yearling Weight and Ultrasound Traits in Hanwoo. **Frontiers in Genetics**, v. 12, 2021.

MEYER, C. L.; BERGER, P. J.; THOMPSON, J. R.; SATTTLER, C. G. Genetic Evaluation of Holstein Sires and Maternal Grandsires in the United States for Perinatal Survival¹. **Journal of Dairy Science**, v. 84, n. 5, p. 1246–1254, 2001.

MEYER, K. Technical note: a successive over-relaxation preconditioner to solve mixed model equations for genetic evaluation¹. **Journal of Animal Science**, v. 94, n. 11, p. 4530–4535, 2016.

MISZTAL, I. Software packages in animal breeding. *In*: WORLD CONGRESS: GENETICS APPLIED TO LIVESTOCK PRODUCTION, 5., 1994, Guelph, Canadá. **Proceedings** [...]. Guelph: [s.n.], 1994.

MIYUMO, S.; WASIKE, C. B.; ILATSIA, E. D.; BENNEWITZ, J.; CHAGUNDA, M. G. Genetic and non-genetic factors influencing KLH binding natural antibodies and specific antibody response to Newcastle disease in Kenyan chicken populations. **Journal of Animal Breeding and Genetics**, v. 140, n. 1, p. 106–120, 2023.

MUNGUÍA VÁSQUEZ, M. F.; GILL, C. A.; RIGGS, P. K.; HERRING, A. D.; SANDERS, J. O.; RILEY, D. G. Genetic evaluation of crossbred Bos indicus cow temperament at parturition. **Journal of Animal Science**, v. 102, p. skae022, 2024.

NAGY, I.; FARKAS, J.; ONIKA-SZVATH, S.; RADNAI, I.; SZENDRO, Z. Genetic parameters and inbreeding depression of litter weight in Pannon White rabbits. **Agriculturae Conspectus Scientificus**, v. 76, n. 3, p. 231–233, 2011.

NEL, C.; GURMAN, P.; SWAN, A.; VAN DER WERF, J.; SNYMAN, M.; DZAMA, K.; OLIVIER, W.; SCHOLTZ, A.; CLOETE, S. Including genomic information in the genetic evaluation of production and reproduction traits in South African Merino sheep. **Journal of Animal Breeding and Genetics**, v. 141, n. 1, p. 65–82, 2024.

NIRANJAN, S. K.; SHARMA, S. R.; GOWANE, G. R. Estimation of Genetic Parameters for Wool Traits in Angora Rabbit. **Asian-Australasian Journal of Animal Sciences**, v. 24, n. 10, p. 1335–1340, 24 ago. 2011.

NORRIS, D.; NGAMBI, J. W. Genetic parameter estimates for body weight in local Venda chickens. **Tropical Animal Health and Production**, v. 38, n. 7, p. 605–609, 1 out. 2006.

ØDEGÅRD, J.; MADSEN, P.; LABOURIAU, R.; GJERDE, B.; MEUWISSEN, T. H. E. A sequential threshold cure model for genetic analysis of time-to-event data¹. **Journal of Animal Science**, v. 89, n. 4, p. 943–950, 2011.

PACHECO, A.; CONINGTON, J.; CORRIPIO-MIYAR, Y.; FREW, D.; BANOS, G.; MCNEILLY, T. N. Genetic profile of adaptive immune traits and relationships with parasite resistance and productivity in Scottish Blackface sheep. **Animal**, v. 18, n. 2, p. 101061, 2024.

PALACIOS-ESPINOSA, A.; RODRÍGUEZ-ALMEIDA, F.; JIMÉNEZ-CASTRO, J.; ESPINOZA-VILLAVICENCIO, J. L.; NÚÑEZ-DOMÍNGUEZ, R. Genetic evaluation of a holstein dairy herd in Baja California sur, utilizing an animal model with repeated measurements. **Agrocencia**, v. 35, n. 3, p. 347–353, 2001.

PANERU, U.; MOGHADDAR, N.; VAN DER WERF, J. Comparison between multiple-trait and random regression models for genetic evaluation of weight traits in Australian meat sheep. **Journal of Animal Science**, v. 102, p. skae038, 1 jan. 2024.

PERDOMO-GONZÁLEZ, D. I.; SÁNCHEZ-GUERRERO, M. J.; BARTOLOMÉ, E.; GUEDES DOS SANTOS, R.; MOLINA, A.; VALERA, M. Designing an early selection morphological traits index for reproductive efficiency in Pura Raza Española mares. **Journal of Animal Science**, v. 102, p. skad409, 2024.

QUINN, J. L.; CHARMANTIER, A.; GARANT, D.; SHELDON, B. C. Data depth, data completeness, and their influence on quantitative genetic estimation in two contrasting bird populations. **Journal of Evolutionary Biology**, v. 19, n. 3, p. 994–1002, 2006.

RAGAB, M.; BASELGA, M. A comparison of reproductive traits of four maternal lines of rabbits selected for litter size at weaning and founded on different criteria. **Livestock Science**, v. 136, n. 2, p. 201–206, 1 abr. 2011.

ROBINSON, D. L.; FERGUSON, D. M.; ODDY, V. H.; PERRY, D.; THOMPSON, J. Genetic and environmental influences on beef tenderness. **Australian Journal of Experimental Agriculture**, v. 41, n. 7, p. 997, 2001.

RÖHE, R.; SAVAS, T.; BRKA, M.; WILLMS, F.; KALM, E. Multiple-trait genetic analyses of racing Performances of German trotters with disentanglement of genetic and driver effects. **Archives Animal Breeding**, v. 44, n. 6, p. 579–588, 2001.

ROVADOSCKI, G. A.; PETRINI, J.; RAMIREZ-DIAZ, J.; PERTILE, S. F. N.; PERTILLE, F.; SALVIAN, M.; IUNG, L. H. S.; RODRIGUEZ, M. A. P.; ZAMPAR, A.; GAYA, L. G.; CARVALHO, R. S. B.; COELHO, A. A. D.; SAVINO, V. J. M.; COUTINHO, L. L.; MOURÃO, G. B. Genetic parameters for growth characteristics of free-range chickens under univariate random regression models. **Poultry Science**, v. 95, n. 9, p. 1989–1998, 2016.

RUÍZ-FLORES, A.; JOHNSON, R. K. Direct and correlated responses to two-stage selection for ovulation rate and number of fully formed pigs at birth in swine. **Journal of Animal Science**, v. 79, n. 9, p. 2286, 2001.

RUPP, R.; CLÉMENT, V.; PIACERE, A.; ROBERT-GRANIÉ, C.; MANFREDI, E. Genetic parameters for milk somatic cell score and relationship with production and udder type traits in dairy Alpine and Saanen primiparous goats. **Journal of Dairy Science**, v. 94, n. 7, p. 3629–3634, 2011.

SALCES, A. J.; SEO, K. S.; CHO, K. H.; KIM, S.; LEE, Y. C. Genetic Parameter Estimation of Carcass Traits of Duroc Predicted Using Ultrasound Scanning Modes. **Asian-Australasian Journal of Animal Sciences**, v. 19, n. 10, p. 1379–1383, 2006.

SALEM, M. M. I.; NASR, M. A. F.; AMIN, A. M. S. Principal component analysis of breeding values for birth weight milk and reproductive traits of the Egyptian buffalo. **Tropical Animal Health and Production**, v. 53, n. 1, p. 183, 26 fev. 2021.

SASAKI, Y.; MIYAKE, T.; GAILLARD, C.; OGUNI, T.; MATSUMOTO, M.; ITO, M.; KURAHARA, T.; SASAE, Y.; FUJINAKA, K.; OHTAGAKI, S.; DOUGO, T. Comparison of genetic gains per year for carcass traits among breeding programs in the Japanese Brown and the Japanese Black cattle¹. **Journal of Animal Science**, v. 84, n. 2, p. 317–323, 2006.

SEWALEM, A.; MIGLIOR, F.; KISTEMAKER, G. J. Short communication: Genetic parameters of milking temperament and milking speed in Canadian Holsteins. **Journal of Dairy Science**, v. 94, n. 1, p. 512–516, 1 jan. 2011.

SHANKS, B. C.; TESS, M. W.; KRESS, D. D.; CUNNINGHAM, B. E. Genetic evaluation of carcass traits in Simmental-sired cattle at different slaughter end points. **Journal of Animal Science**, v. 79, n. 3, p. 595–604, 2001.

SHI, R.; BRITO, L. F.; LIU, A.; LUO, H.; CHEN, Z.; LIU, L.; GUO, G.; MULDER, H.; DUCRO, B.; VAN DER LINDEN, A.; WANG, Y. Genotype-by-environment interaction in Holstein heifer fertility traits using single-step genomic reaction norm models. **BMC Genomics**, v. 22, n. 1, p. 193, 2021.

SMITH, E. G.; WALKOM, S. F.; CLARK, S. A. Exploring genetic variation in potential indicators of resilience in sheep using fibre diameter measured along the wool staple. **Animal**, v. 18, n. 2, p. 101065, 1 fev. 2024.

SUI, J.; LUAN, S.; YANG, G.; CHEN, X.; LUO, K.; GAO, Q.; WANG, J.; HU, H.; KONG, J. Impact of pedigree depth and inclusion of historical data in the estimation of variance components and breeding values in *Macrobrachium rosenbergii*. **Aquaculture**, v. 464, p. 80–86, 2016.

TAMILARASAN, K.; AHMAD, S. F.; PANDA, S.; PREETHI, A. L.; TARAFDAR, A.; PANDEY, H. O.; GAUR, G. K. Genetic analysis of first lactation and lifetime performance traits in composite Vrindavani cattle: important considerations for higher milk production. **Tropical Animal Health and Production**, v. 56, n. 1, p. 31, 2024.

TAREKEGN, G. M.; STRANDBERG, E.; ANDONOV, S.; BÅGE, R.; ASK-GULLSTRAND, P.; RIUS-VILARRASA, E.; CHRISTENSEN, J. M.; BERGLUND, B. Single-step genome-wide association study uncovers known and novel candidate genomic regions for endocrine and classical fertility traits in Swedish Red and Holstein dairy cows. **Livestock Science**, v. 253, p. 104731, 2021.

THEKKOOT, D. M.; KEMP, R. A.; ROTHSCCHILD, M. F.; PLASTOW, G. S.; DEKKERS, J. C. M. Estimation of genetic parameters for traits associated with reproduction, lactation, and efficiency in sows¹. **Journal of Animal Science**, v. 94, n. 11, p. 4516–4529, 2016.

TOTH, Z.; KAPS, M.; SÖLKNER, J.; BODO, I.; CURIK, I. Quantitative genetic aspects of coat color in horses¹. **Journal of Animal Science**, v. 84, n. 10, p. 2623–2628, 2006.

TSIAMADIS, V.; BANOS, G.; PANOUSIS, N.; KRITSEPI-KONSTANTINOOU, M.; ARSENOS, G.; VALERGAKIS, G. E. Genetic parameters of calcium, phosphorus, magnesium, and potassium serum concentrations during the first 8 days after calving in Holstein cows. **Journal of Dairy Science**, v. 99, n. 7, p. 5535–5544, 2016.

TWOMEY, A. J.; SAYERS, R. G.; CARROLL, R. I.; BYRNE, N.; BRIEN, E. O.; DOHERTY, M. L.; MCCLURE, J. C.; GRAHAM, D. A.; BERRY, D. P. Genetic parameters for both a liver damage phenotype caused by *Fasciola hepatica* and antibody response to *Fasciola hepatica* phenotype in dairy and beef cattle¹. **Journal of Animal Science**, v. 94, n. 10, p. 4109–4119, 2016.

VAN HULZEN, K. J. E.; NIELEN, M.; KOETS, A. P.; DE JONG, G.; VAN ARENDONK, J. A. M.; HEUVEN, H. C. M. Effect of herd prevalence on heritability estimates of antibody response to *Mycobacterium avium* subspecies *paratuberculosis*. **Journal of Dairy Science**, v. 94, n. 2, p. 992–997, 2011.

VARGAS JURADO, N.; NOTTER, D. R.; TAYLOR, J. B.; BROWN, D. J.; MOUSEL, M. R.; LEWIS, R. M. Combined purebred and crossbred genetic evaluation of Columbia, Suffolk, and crossbred lamb birth and weaning weights: systematic effects and heterogeneous variances. **Journal of Animal Science**, v. 102, p. skad410, 2024.

VARGOVIC, L.; HERMESCH, S.; ATHORN, R. Z.; BUNTER, K. L. Feed intake and feeding behaviour traits of gestating sows are associated with undesirable outcomes. **Livestock Science**, v. 249, p. 104526, 2021.

VOSTRÝ, L.; PŘIBYL, J.; MACH, K.; MAJZLÍK, I. Genetic parameters estimation and breeding values prediction for linear described traits in the Old Kladruber horse. **Journal of Animal and Feed Sciences**, v. 20, n. 3, p. 338–349, 2011.

WALDMANN, P.; ERICSSON, T. Comparison of REML and Gibbs sampling estimates of multi-trait genetic parameters in Scots pine. **Theoretical and Applied Genetics**, v. 112, n. 8, p. 1441–1451, 1 maio 2006.

WESTHUIZEN, R. R. V. Der; SCHOEMAN, S. J.; JORDAAN, G. F.; WYK, J. B. V. Genetic parameters for reproductive traits in a beef cattle herd estimated using multitrait analysis. **South African Journal of Animal Science**, v. 31, n. 1, p. 41–48, 2001.

WOLF, J.; PEŠKOVIČOVÁ, D.; GROENEVELD, E. Stability of genetic parameter estimates for production traits in pigs. **Journal of Animal Breeding and Genetics**, v. 118, n. 3, p. 161–172, 2001.

YANG, H.; SU, G. Impact of phenotypic information of previous generations and depth of pedigree on estimates of genetic parameters and breeding values. **Livestock Science**, v. 187, p. 61–67, 1 maio 2016.

CAPÍTULO 2 – DESENVOLVIMENTO DO PACOTE LZFF

RESUMO

A reprodutibilidade é um princípio fundamental na ciência, pois garante que os resultados possam ser verificados por outros pesquisadores. No contexto da avaliação genética animal, a preparação e a formatação de dados fenotípicos, de pedigree e de marcadores genéticos representam um desafio, especialmente para iniciantes. Este trabalho apresenta o desenvolvimento do pacote LZFF, que automatiza a formatação desses arquivos, que busca reduzir os erros e aumentar a eficiência no processamento de dados para softwares de avaliação genética animal como BLUPF90 e Wombat. O processo de desenvolvimento seguiu um fluxo estruturado, convertendo os dados de um formato livre (L) para um formato padrão (P), e, finalmente, para o formato final (F) compatível com softwares de análise genética. A função `rrcData` foi desenvolvida para leitura, recodificação e verificação dos dados. As funções `formatA` e `formatB` ajustam os arquivos para os formatos exigidos pelo BLUPF90 e Wombat, respectivamente. Para validar as funções, foram implementados testes automatizados utilizando o pacote `testthat`. Os testes avaliaram a precisão na leitura de arquivos em diferentes formatos (CSV, TXT, XLS, etc.), a fidelidade da recodificação e a preservação da estrutura dos conjuntos de dados processados.

Palavras-chave: Avaliação genética. Formatação de dados. Software R. Reprodutibilidade.

ABSTRACT

Reproducibility is a fundamental principle in science, as it ensures that results can be verified by other researchers. In the context of animal genetic evaluation, the preparation and formatting of phenotypic, pedigree, and genetic marker data represent a challenge, especially for beginners. This study presents the development of the LZFF package, which automates the formatting of these files, aiming to reduce errors and increase efficiency in data processing for animal genetic evaluation software such as BLUPF90 and Wombat. The development process followed a structured workflow, converting data from a free format (L) to a standardized format (P), and finally to a final format (F) compatible with genetic analysis software. The `rrcData` function was developed for reading, recoding, and verifying data. The `formatA` and `formatB` functions adjust the files to the formats required by BLUPF90 and Wombat, respectively. To validate the functions, automated tests were implemented using the `testthat` package. These tests assessed the accuracy of reading files in different formats (CSV, TXT, XLS, etc.), the fidelity of recoding, and the preservation of the structure of the processed datasets.

Keywords: Genetic evaluation, Data formatting, Software R, Reproducibility.

1 INTRODUÇÃO

Para que as descobertas científicas sejam confirmadas, é essencial que seus resultados sejam validados por outros grupos de pesquisa. Em outras palavras, esses resultados devem poder ser reproduzidos ou replicados quando os experimentos forem repetidos em condições semelhantes. Somente após essa validação as descobertas são amplamente aceitas pela comunidade científica.

Garantir a reprodutibilidade das pesquisas é um dos principais objetivos da ciência. Contudo, com o aumento da quantidade, da diversidade e da complexidade dos dados, ferramentas e processos envolvidos, torna-se necessário disponibilizar informações adicionais junto às descobertas científicas, garantindo que outros pesquisadores possam avaliar, reutilizar e reproduzir esses resultados (Silva, 2017).

Infelizmente, nem todos os resultados podem ser completamente reproduzidos, pois é desafiador replicar exatamente todas as condições do ambiente em que o experimento foi realizado. Mais de 70% dos pesquisadores tentaram, sem sucesso reproduzir os experimentos de outros cientistas, e mais da metade não conseguiu reproduzir seus próprios experimentos (Baker, 2016).

A reprodutibilidade é um aspecto fundamental de qualquer estudo científico, pois garante que os resultados obtidos possam ser verificados e confirmados por outros pesquisadores, sob as mesmas condições experimentais (Begley; Ellis, 2012).

Nos ensaios computacionais e de análise de dados, essa tarefa tende a ser mais simples, já que não há influências significativas do ambiente externo, e tanto o hardware quanto os softwares podem ser configurados de forma bastante similar ao experimento original (Gandrout, 2020).

No processo de avaliação genética animal, uma das etapas mais demoradas é a formatação dos arquivos de dados fenotípicos, de pedigree e, eventualmente, de marcadores genéticos. Essa tarefa pode ser especialmente desafiadora para iniciantes.

Se essa etapa não for realizada com o devido cuidado, pode introduzir erros que comprometam a análise, resultando em dados enviesados e de difícil replicação. Além disso, erros graves na formatação podem impedir a execução do software de avaliação genética.

Portanto, funções automatizadas que facilitem a formatação dos arquivos de dados e pedigree para utilização nos softwares de avaliação genética podem reduzir

significativamente o tempo necessário para o preparo. Além disso, contribuem de forma crucial para aumentar a reprodutibilidade dos resultados gerados por esses softwares.

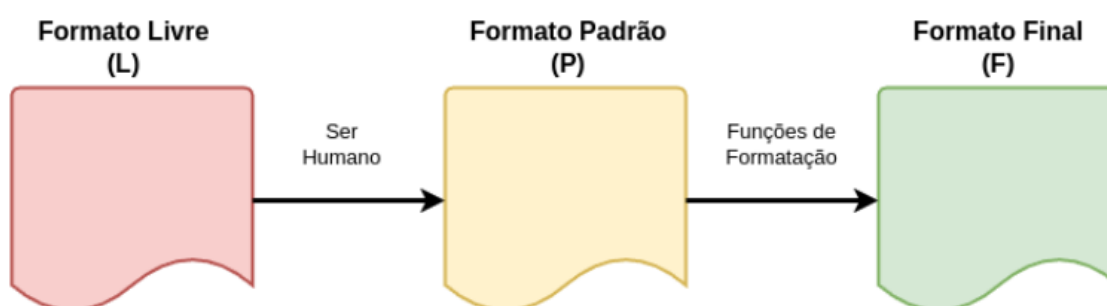
O objetivo deste trabalho é desenvolver ferramentas computacionais capazes de automatizar a preparação e a padronização de dados utilizados em avaliações genéticas animais. Busca-se, com isso, reduzir a ocorrência de erros decorrentes de procedimentos manuais, otimizar o tempo de organização dos arquivos e, principalmente, aumentar a reprodutibilidade e a confiabilidade dos resultados obtidos.

2 PLANEJAMENTO E DESENVOLVIMENTO

Durante o processo, desde a coleta até a inserção no software de análises genéticas, os arquivos de dados podem ser apresentados em três formatos: livre (L), padrão (P) e formatado (F). O formato livre representa a fase inicial dos dados, logo após serem coletados. Nessa etapa, o responsável pela coleta organiza a planilha ou o arquivo de texto conforme sua necessidade e preferência.

Dessa maneira, observa-se uma grande variação nos formatos dos arquivos, o que pode dificultar a adaptação dos dados às exigências do software de análises genéticas. Por isso, é fundamental que o usuário ajuste o arquivo para um formato padrão (P). A partir desse padrão, as funções responsáveis pela formatação podem processar os dados e gerar um arquivo no formato adequado (F) para utilização no software de análises genéticas (Figura 9).

Figura 9- Representação dos diferentes estados de um arquivo conforme a formatação aplicada.



Fonte: Elaborada pelo autor.

O formato padrão (P) organiza os dados de modo que cada variável fique disposta em uma coluna na planilha ou no arquivo de texto. Esse formato é amplamente utilizado por softwares de análise de dados, como R e SAS, para garantir uma leitura eficiente das informações. Além disso, muitas empresas já adotam esse formato diretamente no momento da coleta, o que significa que, em diversos casos, os arquivos já são criados no formato P, sem passar pelo formato livre (L) e sem exigir ajustes manuais para essa conversão.

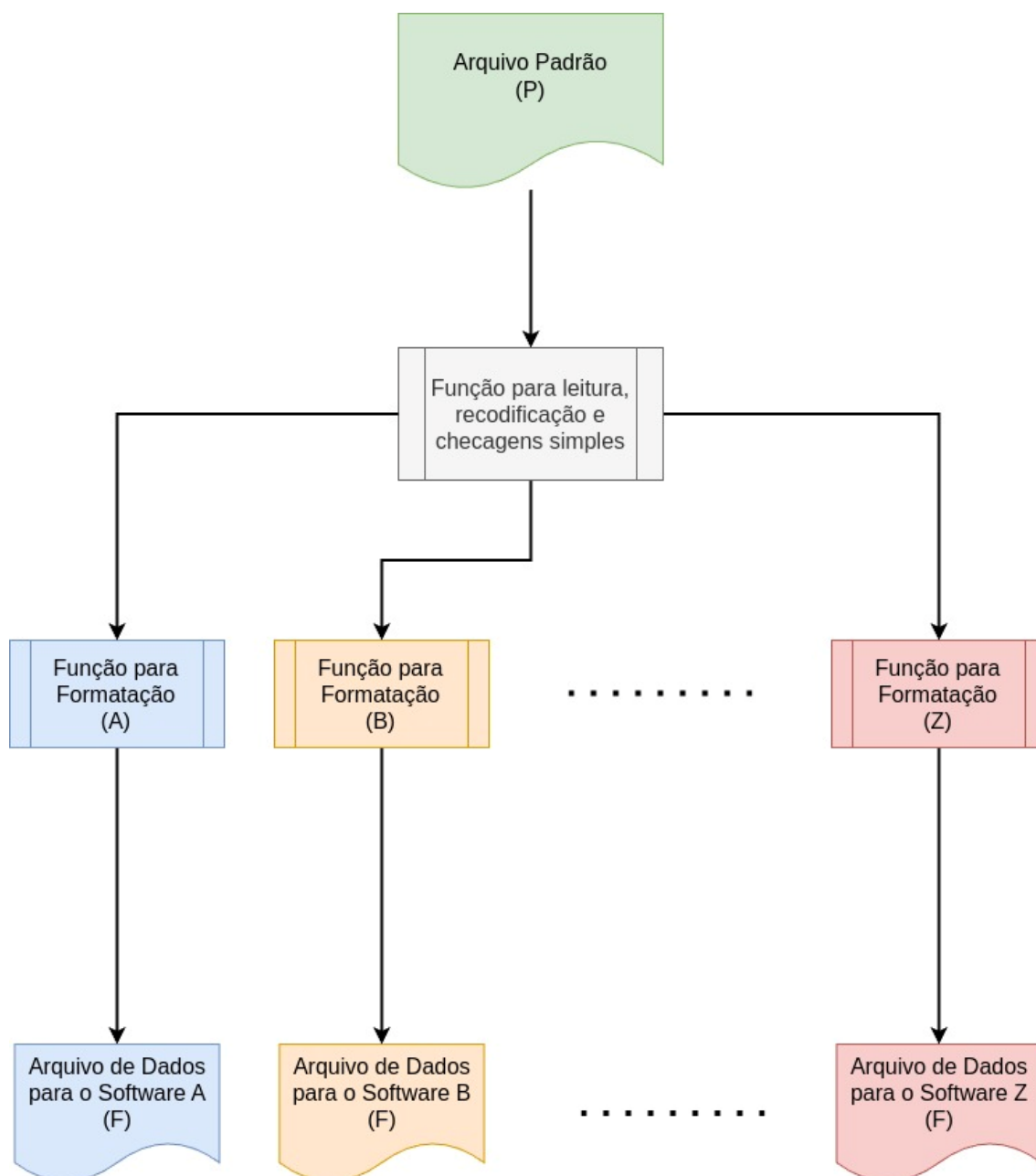
Quando os dados são registrados sem a separação das variáveis em colunas distintas, caracterizando o formato livre (L), a transformação para o formato padrão (P) tende a ser relativamente simples. Isso ocorre principalmente porque, na maioria das coletas, as informações são armazenadas em planilhas eletrônicas, facilitando a organização.

Quando os arquivos estão no formato padrão (P), as funções do pacote podem interpretá-los e convertê-los para o formato necessário para cada software de análise genética do projeto. Para alguns programas, a diferença entre o formato P e o exigido é pequena, precisando apenas de ajustes simples e algumas verificações. Para outros, o formato requerido é mais complexo, e será preciso realizar manipulações e verificações mais detalhadas para garantir que os dados estejam no formato final (F).

Na maioria das vezes, os softwares de análise genética pedem que os arquivos de dados (F) sejam numéricos, enquanto os arquivos no formato P geralmente contêm variáveis alfanuméricas. Por exemplo, arquivos usados em programas de avaliação genética costumam ter colunas com identificações de animais, pais e mães, frequentemente representadas por códigos alfanuméricos. Além disso, os efeitos de grupos contemporâneos também podem ser alfanuméricos. Por isso, seria útil que as funções de formatação realizassem a conversão desses dados, transformando os valores alfanuméricos em numéricos. Essa recodificação inicial facilitaria o desenvolvimento das funções, tornando-as mais eficazes para a tarefa específica.

Com isso, o processo de desenvolvimento do pacote poderia seguir o planejamento ilustrado na Figura 10.

Figura 10- Visão Geral do Desenvolvimento do Pacote LZFF.



Fonte: Elaborada pelo autor.

Nesse modelo, o arquivo no formato P é processado por uma função que faz a recodificação dos valores, se necessário, e verifica possíveis problemas nas etapas seguintes. Essa função evita que a leitura do arquivo seja repetida em cada uma das funções de formatação. Após o processamento, a função retorna um *data frame* (D) com os dados já lidos, recodificados e verificados, pronto para uso pelas funções de formatação.

Esse objeto D é então capturado por uma das funções de formatação, que ajusta os dados ao formato exigido para o software de análise genética.

As funções de formatação geram um *data frame* que pode ser convertido em um arquivo de texto, de acordo com os requisitos dos softwares de análise genética. Uma única função transforma os dados em arquivos no formato adequado, que recebe o *data frame* da função de formatação e o converte para o formato final (F). Essa função permite que os resultados das funções de formatação sejam utilizados em análises intermediárias antes de sua conversão em arquivos de texto.

Embora o foco principal seja em softwares de avaliação genética animal, os formatos exigidos por esses programas são semelhantes ou muito próximos dos formatos solicitados por muitos outros softwares que realizam análises em diversas áreas do conhecimento. Portanto, as funções serão nomeadas de forma genérica, como *formatA*, em vez de usar nomes específicos relacionados aos softwares que as influenciaram, como *Blupf90_F*, por exemplo.

3 FUNÇÕES

3.1 Função *rrcData* (Read, Recode and Check Data)

A função *rrcData* é responsável por três etapas principais: ler, recodificar e verificar os dados. Ela é capaz de ler arquivos de texto simples (.txt ou sem extensão), arquivos separados por vírgula (.csv), além de planilhas eletrônicas do Excel (.xls e .xlsx), do LibreOffice (.ods) e do Google Sheets (por meio de URL).

Após a leitura, a função recodifica os valores, se necessário, para garantir que as variáveis estejam no formato adequado (por exemplo, convertendo valores alfanuméricos por numéricos). A recodificação utiliza números inteiros, começando de 1 até o número necessário de níveis. Na recodificação de um arquivo de pedigree, são adotados procedimentos específicos para garantir um ajuste adequado das informações.

Em seguida, a função realiza uma série de verificações para garantir que não existam erros ou inconsistências nos dados que possam afetar as etapas posteriores do processamento. O resultado dessa função é um *data frame* pronto para ser utilizado nas funções de formatação, garantindo que os dados estejam bem estruturados e prontos para manipulação.

Conforme ilustrado na Figura 11, a função *rrcData* deve ser executada antes das funções de formatação. A implementação desse esquema previne a duplicação

de código nas funções de formatação, já que a leitura dos arquivos e outras verificações necessárias são feitas de forma centralizada pela *rrcData*. Isso garante que essas verificações sejam realizadas independentemente do software de análise genética ou genômica a ser utilizado. Além disso, esse formato facilita a manutenção do código, eliminando redundância nas funções, resultando em um código mais conciso.

Durante a leitura do arquivo de dados no formato P, a função *rrcData* realiza as seguintes verificações:

1. Não deve haver campos com valores alfanuméricos;
2. O valor indicando um dado perdido deve ser especificado (o padrão do software é ler o 0 como dado perdido);

A função foi definida na linguagem R da seguinte forma:

```
rrcData<-function(local, s, d, h = TRUE, md = c("", " ", "NA"),
                  colsPed = NULL, colsTraits = NULL){
```

local= diretório do arquivo;

s = separador de campos (valor padrão é espaço);

d = usado como separador decimal (valor padrão é “.”);

h = valor logico que indica a presença de cabeçalho (valor padrão é TRUE, porque é verdadeiro).

md = identificação de valores ausentes, pode estar como somente um espaço em branco “ ”, ou por uma virgula “,” ou somente NA.

colsPed = vetor numérico especificando as colunas correspondentes ao pedigree (normalmente, identificação do indivíduo, da mãe e do pai). O valor padrão é NULL.

colsTraits = seleção de colunas específicas para características. O valor padrão é NULL.

Se uma coluna precisar ser recodificada e não fizer parte das colunas de pedigree, a recodificação será feita de forma simples, atribuindo números inteiros sequenciais a partir de 1, substituindo todos os valores originais por valores numéricos únicos. Isso é útil, por exemplo, para converter categorias ou valores textuais em números, de forma que o software de avaliação genética processe os dados adequadamente.

Se as colunas contiverem informações de pedigree (como a identificação do indivíduo, pai e mãe), a recodificação será tratada de forma diferente. Para essas colunas, uma função específica será desenvolvida para lidar com a recodificação de maneira apropriada, considerando a natureza das informações de pedigree.

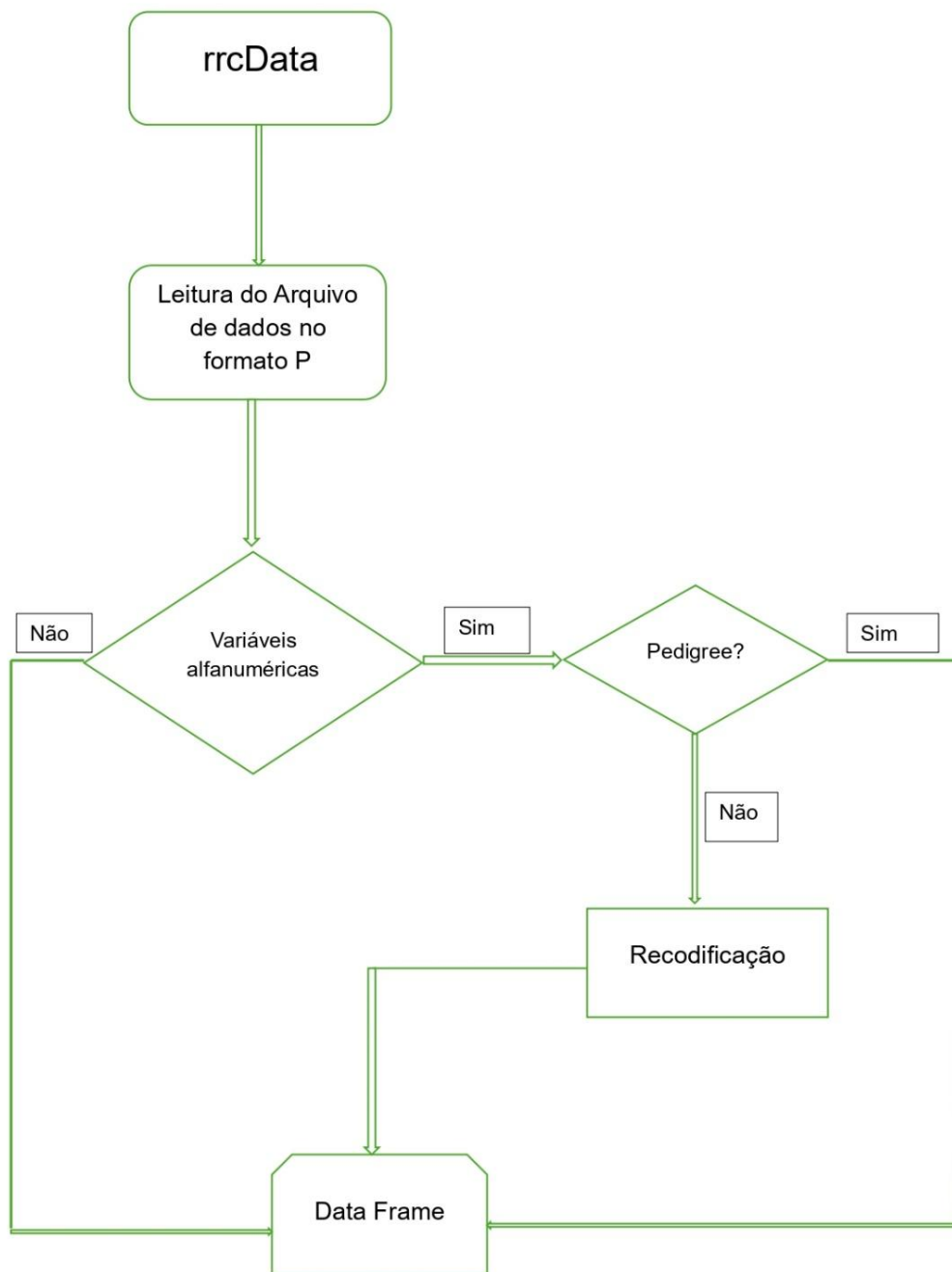
Além disso, ao utilizar essa função de recodificação, é importante fornecer argumentos que indicam quais colunas contêm dados de pedigree. Isso permite que a função ignore essas colunas na hora de determinar quais precisam ser recodificadas, evitando que as informações de pedigree sejam alteradas de maneira incorreta.

Para contemplar o item 2, uma vez que a maioria das colunas no arquivo de dados fenotípicos deve conter apenas valores numéricos, foi feita uma verificação inicial com a função `is.numeric()`. Essa função analisa cada coluna do *data frame* e retorna TRUE para as que são numéricas e FALSE para aquelas que contêm caracteres alfanuméricos. A partir dessa análise, um loop é executado exclusivamente sobre as colunas identificadas como não numéricas, realizando a recodificação dos valores. O resultado desse processo é armazenado em uma lista, que serve como um mapa de conversão, permitindo que os valores originais possam ser recuperados, se necessário.

Outra verificação realizada pela função é a de detecção de caracteres ASCII. O código verifica se todas as strings dentro do objeto dados contêm apenas caracteres ASCII imprimíveis, caso um caractere inválido seja encontrado, a execução é interrompida com uma mensagem de erro. A função também verifica os valores inteiros. Algumas colunas, como identificadores, níveis de efeitos fixos ou aleatórios precisam ser representadas por valores inteiros, que não excedam “2147483647”. Se houver valores acima desse limite, a execução será interrompida com uma mensagem de erro. A função verifica também, se as variâncias das características estão dentro de um intervalo aceitável, caso uma variância seja muito pequena ($<1e-5$) ou muito grande ($>1e5$), um aviso é emitido sugerindo o escalonamento dos dados para melhorar a análise.

Antes do processamento, é feita uma verificação utilizando a função `is.na()` para identificar quais colunas do *data frame* contêm valores ausentes. Em seguida, um loop percorre essas colunas e substitui todos os valores NA por um valor especificado pelo usuário através de um argumento da função. Caso nenhum valor seja informado, a substituição é feita pelo valor padrão, que é zero.

Figura 11- Apresenta o fluxo de trabalho da função rrcData.



Fonte: Elaborada pelo autor.

3.2 Função rrcPed

A função `rrcPed`, de forma semelhante à `rrcData`, lê o arquivo de pedigree e o recodifica, fazendo algumas checagens e recodificação. Além disso, o *data frame* gerado pela função `rrcData` deve ser passado como argumento para essa função, para que as colunas de pedigree contidas nos dados possam ser substituídas pelas colunas recodificadas pela função `rrcPed`.

A definição da função no R é dada por:

```
rrcPed<-function(local, s, h = FALSE, isdd = c(1, 2, 3, 4),
  md = c("", " ", "NA"), colsPedData = c(1, 2, 3), udata){
```

em que,

`rrcPed`= nome da função;

`local`= caminho para o arquivo de pedigree;

`s` = separador entre os campos no arquivo de pedigree. O valor padrão é espaço (" ");

`h` = valor lógico indicando a presença de cabeçalho. O valor padrão é FALSE;

`isdd` = índices das colunas no arquivo: ID, pai, mãe, data de nascimento;

`md` = valores a serem considerados como dados perdidos. Os valores padrão são: vazio (""), espaço (" ") e NA;

`colsPedData`: colunas no `udata` que contêm os mesmos IDs do pedigree (ex: ID, pai, mãe);

`udata`: objeto de dados não formatado;

Observa-se que o argumento `d`, presente na função `rrcData`, não é utilizado em `rrcPed`, uma vez que as informações de pedigree devem ser compostas exclusivamente por números inteiros.

O arquivo de pedigree deve conter exatamente quatro colunas, referentes à identificação dos animais, dos pais, das mães e à data de nascimento. A posição das colunas que contêm as identificações dos indivíduos (ID, pai e mãe) deve ser informada por meio do argumento `colsPedData`.

De maneira análoga à função `rrcData`, a `rrcPed` realiza uma checagem do tipo de arquivo fornecido e, em seguida, reorganiza as colunas de interesse. A função verifica se a coluna de data de nascimento está no formato caractere e, se necessário, realiza sua conversão para o tipo *Date*, utilizando diferentes formatos de data. Na

sequência, remove eventuais duplicatas no arquivo e verifica se algum animal aparece simultaneamente como pai e como mãe, o que representaria uma inconsistência.

Após essas validações iniciais, a função executa a etapa de recodificação, na qual os identificadores originais dos animais (ID, pai e mãe) são convertidos em códigos numéricos únicos. Para isso, é criado um mapa de recodificação que reúne todos os identificadores (indivíduos, pais e mães), ordena-os pela data de nascimento (quando disponível), remove duplicações e associa a cada identificador original um novo código numérico sequencial, armazenado no objeto `mapaCod`.

Em seguida, os identificadores no pedigree são substituídos pelos novos códigos numéricos, e os valores ausentes são substituídos por zero. A função ainda garante que os dados estejam no formato ASCII, evitando problemas com caracteres especiais. O mesmo processo de recodificação é aplicado ao objeto de dados principal (`udata`), assegurando a consistência entre os identificadores utilizados nos dois conjuntos de dados.

Ao final, a função retorna uma lista contendo três elementos:

`map`: o mapa com os códigos originais e seus respectivos valores recodificados;

`ped`: o pedigree recodificado;

`data`: contém o conjunto de dados original.

3.3 Função `formatA`

A função `formatA` recebe um objeto gerado pela função `rrcData` e tem como finalidade de preparar os dados para serem salvos no formato F, conforme especificado na documentação do software BLUPF90 (Miształ, 1999).

A função foi definida na linguagem R como:

```
formatA<-function(udata, omd = 0, of = "formatted_file", EoL = "\n"){
  if(stringr::str_detect(of, "#")){
    stop("File name cannot contain a #. Choose a name without a #")
  }
}
```

em que,

`formatA` = nome da função;

`udata` = recebe um *data frame* com dados não formatados, vindos da função `rrcData`;

`Omd` = identificação de valores ausentes a ser escrito no arquivo de saída;

`Of` = nome do arquivo de saída (valor padrão é `"formatted_file"`)

EoL = indicador de fim de linha. Unix e Linux usam "\\n", enquanto Windows usa "\\r\\n".

Foi implementada uma verificação simples utilizando a função `str_detect()` da biblioteca `stringr` para confirmar se o nome do arquivo (`of`) contém o caractere `#`. Na documentação do software BLUPF90 o caractere `#` é utilizado para indicar um comentário. Se o caractere `#` for encontrado, a função gera um erro por meio da função `stop()` e interrompe a execução, exibindo a mensagem: "O nome do arquivo não pode conter um `#`. Escolha um nome sem um `#`".

Finalmente, os dados formatados são salvos em um arquivo cujo nome é definido pelo usuário por meio do argumento `of` (output file). Na função `write.table`, utilizada dentro da `formatA` para salvar os dados, os parâmetros `col.names` e `row.names` são configurados como `FALSE`, pois, em geral, esses elementos não estão presentes nos arquivos gerados. Além disso, para garantir compatibilidade entre diferentes sistemas operacionais, como Windows, Mac e Linux, o usuário pode definir a marcação de fim de linha por meio do argumento `eol` (end of line), ajustando conforme necessário.

A função `write.table` foi definida na linguagem R como:

```
utils::write.table(udata, file = of, quote = FALSE, sep = " ",
  row.names = FALSE, col.names = FALSE, eol = EoL, na = omd)
}
```

Em que,

`udata` = data frame contendo os dados que serão salvos.

`file` = caminho e nome do arquivo de saída, definido pelo argumento `of`.

`quote` = define se os valores devem ser colocados entre aspas. O valor padrão, `FALSE`, determina que os valores não devem ser escritos entre aspas.

`sep` = define o separador entre os valores das colunas. Neste caso, é um espaço em branco.

`row.names` = exclui os nomes das linhas do arquivo de saída (valor padrão `FALSE`).

`col.names` = exclui os nomes das colunas do arquivo de saída (valor padrão `FALSE`).

`eol` = define o caractere de fim de linha, de acordo com o valor passado pelo argumento `EoL` (como "\\n", por exemplo).

`na` = define o valor que será usado para representar os dados ausentes no arquivo de saída, com base no valor fornecido pelo argumento `omd`.

Esta função salva os dados no formato desejado, respeitando os parâmetros configurados, como a representação de dados ausentes, a separação entre os valores e o formato de fim de linha.

3.4 Função formatB

A função formatB processa um objeto criado pela função rrcData, ajustando os dados para que sejam registrados no formato correto, conforme as diretrizes da documentação do software Wombat (Meyer, 2007).

A função foi definida na linguagem R como:

```
formatB<-function(udata, of = "formatB_data", omd = "-99999",
traits = NULL, widths = NULL, EoL = "\n"){
  if(stringr::str_detect(of,"#")){
    stop("File name cannot contain a #. Choose a name without a #")
  }
}
```

Os parâmetros da função são definidos da seguinte maneira:

formatB= nome da função;

udata = *data frame* com dados não formatados, vindo da função rrcData;

of = nome do arquivo de saída (valor padrão é "formatted_file");

omd= identificação de valores ausentes a ser escrito no arquivo de saída (valor padrão é -99999);

traits= vetor indicando colunas de características (valor padrão é NULL);

widths= vetor que especifica a largura das colunas no arquivo formatado (NULL);

EoL= = indicador de fim de linha. Unix e Linux usam "\n", enquanto Windows usa "\r\n".

Para verificação se o nome do arquivo contém o caracter #, foi implementado na função formatB o mesmo procedimento da função formatA.

FormatB inclui outros procedimentos de formatação, como o ajuste de colunas de dados e de efeitos. A organização correta das colunas permite que o software entenda a relação entre os diferentes efeitos, como o efeito fixo (por exemplo, grupos de contemporâneos) e o efeito aleatório (como os efeitos genéticos aditivos dos indivíduos), e consiga realizar as estimativas genéticas com precisão.

Outra verificação incluída na função incluída na função formatB é a de garantir que o nome do arquivo de saída não exceda 30 caracteres, visando compatibilidade com limitações de sistemas operacionais e do próprio software. Além disso, ao manter

os nomes dos arquivos mais curtos, o software evita problemas de leitura e escrita de arquivos.

Para finalizar a função, os dados precisam ser escritos no formato de colunas fixas, portanto, a função `gdata::write.fwf()` é utilizada para organizar os dados em colunas com uma largura predefinida. Essa função é do pacote `gdata` no R e permite escrever um *data frame* em um arquivo de texto com colunas de tamanhos fixos.

4 TESTES

Todas as funções mencionadas neste trabalho foram desenvolvidas dentro de um pacote chamado LZFF, e na estrutura deste pacote existe uma pasta chamada “test”, onde se encontram os testes que foram feitos ao longo dos desenvolvimentos das funções. Esses testes são fundamentais para garantir que o código esteja funcionando conforme esperado. De acordo com Wickham e Bryan (2019), testes ajudam a garantir que as funções do seu pacote funcionem corretamente, não apenas em condições ideais, mas também em situações de uso real. Os testes permitem a detecção rápida de problemas, especialmente quando alterações no código afetam outras partes do pacote. Além disso, um bom conjunto de testes serve como documentação adicional, ajudando outros usuários e desenvolvedores a entender como usar as funções e quais comportamentos esperar. A demonstração dos testes realizados encontra-se disponível no Anexo 1.

5 RESULTADOS E DISCUSSÕES

Para a preparação do arquivo para o software de análise genética, é necessário convertê-lo de um formato livre (L) para um formato padrão (P), conforme mencionado no item 2 desta seção. Entretanto o arquivo padrão (P) já é o mais utilizado para padronizar as planilhas, por exemplo cada variável em uma coluna (Figura 12).

Figura 12- Exemplo de um arquivo padrão (P).

	A	B	C	D	E	F	G	H
1	Ordem	Lote	Nº	DataCast	GanhoM1	UltimaP	GanhoM2	Peso
2	1	15	AB83	365	0,89	433	0,48	460
3	2	15	AC88	330	1,49	443	-0,63	408
4	3	15	AD89	356	1,14	443	0,36	463
5	4	14	AE203	360	1,04	439	0,52	468
6	5	14	AF209	352	1,22	445	0,36	465
7	6	15	AG674	355	1,08	437	0,43	461
8	7	16	AH696	365	1,08	447	0,38	468
9	8	15	AI701	363	1,03	441	0,48	468
10	9	15	AJ710	369	0,89	437	0,29	453
11	10	16	AK738	336	1,07	417	0,41	440
12	11	16	AL76	362	0,8	421	0,5	449
13	12	14	AM245	372	0,64	419	0,32	437
14	13	15	AN669	367	0,74	422	0,55	453
15	14	16	AO78	362	1,08	444	0,34	463
16	15	14	AP205	378	0,82	440	0,38	461
17	16	14	AQ247	384	0,7	437	0,23	450
18	17	18	AR248	385	0,26	405	0,79	449
19	18	15	AS671	353	0,68	405	0,79	449
20	19	16	AT682	354	0,66	404	0,41	427
21	20	18	AU727	353	1,14	440	0,48	467
22	21	16	AV731	382	0,53	422	0,61	456
23	22	16	AX734	372	0,82	434	0,43	458
24	23	16	AW748	386	0,59	431	0,32	449
25	24	13	AY199	343	1,24	437	0,52	466
26	25	18	AZ244	360				452
27	26	14	ABC217	386				467
28	27	14	ADE271	350	1,19	438		
29	28	13	AFG180	348	1,24	442		
30	29	18	AHI144	356	0,95	428		

Fonte: Elaborada pelo autor.

Neste exemplo o formato (P) está indicando que as variáveis (Lote, Número do animal, Peso da Castração, Ganho Médio 1 e 2, Ganho Última pesagem e Peso Final) estão em colunas e linhas separadas, para facilitar a identificação de cada animal. Nota-se também, que esta planilha contém dados em branco, ou seja, há dados ausentes referentes aos animais. Esse é o formato que muitas empresas já adotam durante o processo de coleta de dados.

Dessa forma, quando os arquivos no formato (P) se encontram nesta situação, as funções do pacote podem interpretá-los e convertê-los para o formato necessário para cada software de análise genética do projeto. E para que essa formatação seja correta, usaremos a função `rrcData` (Figura 13):

Figura 13- Data frame criado pela função `rrcData` a partir de um arquivo de dados no formato padrão (P).

```
> rrcData("C:/Users/miria/OneDrive/Documentos/TestesBLUPF90/XLSX/DP.CA2.xlsx", colsPed = 3, colsTraits = 4:8 )
```

	Ordem	Lote	N°	DataCast	GanhoM1	UltimaP	GanhoM2	Peso
1	1	15	AB83	365	0.89	433	0.48	460
2	2	15	AC88	330	1.49	443	-0.63	408
3	3	15	AD89	356	1.14	443	0.36	463
4	4	14	AE203	360	1.04	439	0.52	468
5	5	14	AF209	352	1.22	445	0.36	465
6	6	15	AG674	355	1.08	437	0.43	461
7	7	16	AH696	365	1.08	447	0.38	468
8	8	15	AI701	363	1.03	441	0.48	468
9	9	15	AJ710	369	0.89	437	0.29	453
10	10	16	AK738	336	1.07	417	0.41	440
11	11	16	AL76	362	0.80	421	0.50	449
12	12	14	AM245	372	0.64	419	0.32	437
13	13	15	AN669	367	0.74	422	0.55	453
14	14	16	AO78	362	1.08	444	0.34	463
15	15	14	AP205	378	0.82	440	0.38	461
16	16	14	AQ247	384	0.70	437	0.23	450
17	17	18	AR248	385	0.26	405	0.79	449
18	18	15	AS671	353	0.68	405	0.79	449
19	19	16	AT682	354	0.66	404	0.41	427
20	20	18	AU727	353	1.14	440	0.48	467
21	21	16	AV731	382	0.53	422	0.61	456
22	22	16	AX734	372	0.82	434	0.43	458
23	23	16	AW748	386	0.59	431	0.32	449
24	24	13	AY199	343	1.24	437	0.52	466
25	25	18	AZ244	360	NA	NA	NA	452
26	26	14	ABC217	386	NA	NA	NA	467
27	27	14	ADE271	350	1.19	438	NA	NA
28	28	13	AFG180	348	1.24	442	NA	NA
29	29	18	AHI144	356	0.95	428	NA	NA

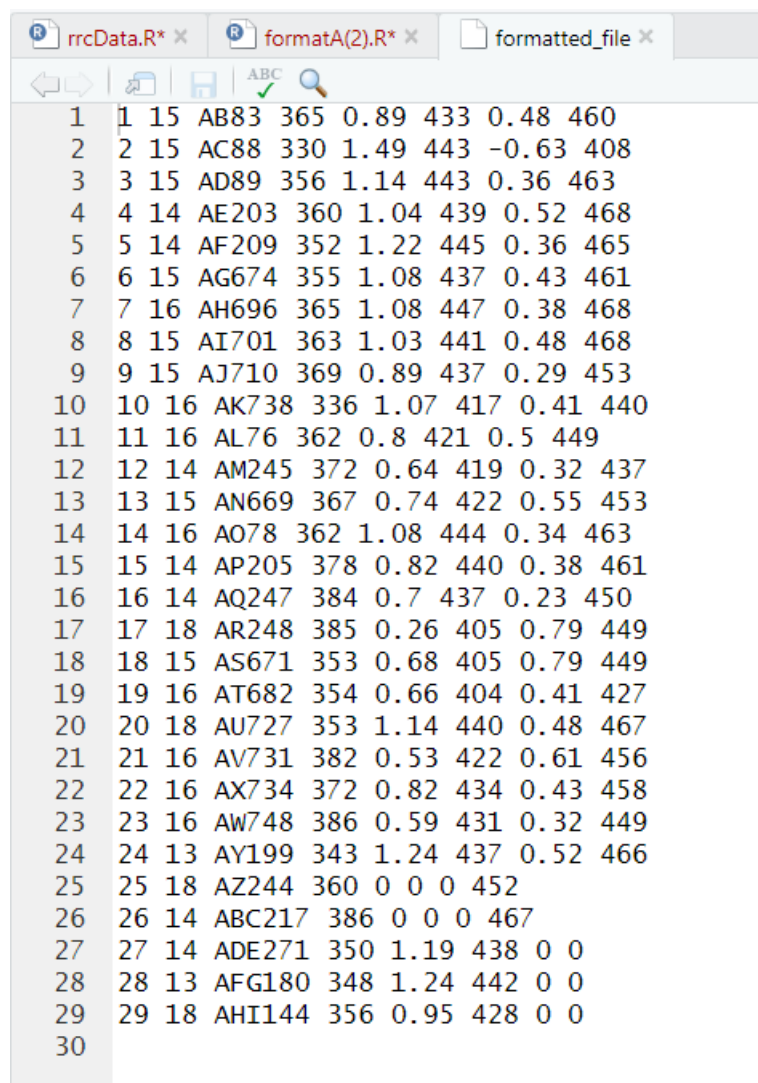
Fonte: Elaborada pelo autor.

Na chamada da função, indicamos o diretório onde o arquivo está localizado, a coluna que contém a identificação do indivíduo (`colsPed`), neste arquivo, está na coluna 3, e as colunas das características (`colsTraits`) localizadas a partir da coluna 4.

A função conforme esperada, retornou um *data frame* com dados formatados. Ela realizou a leitura do arquivo, não precisou fazer a recodificação na coluna 3, pois se trata de dados de pedigree (identificação do indivíduo). Adicionou NA para todos os dados que estavam em branco. Fez a checagem de ASCII, verificou se os numeros inteiros estão dentro dos limites e se as variâncias dos traços estão dentro de um intervalo razoável.

Após essa formatação, o data frame está pronto para ser processado pelas funções `formatA` ou `formatB`. A Figura 14 ilustra o arquivo resultante após a aplicação da função `formatA`.

Figura 14- Arquivo formatado pela função `formatA`.



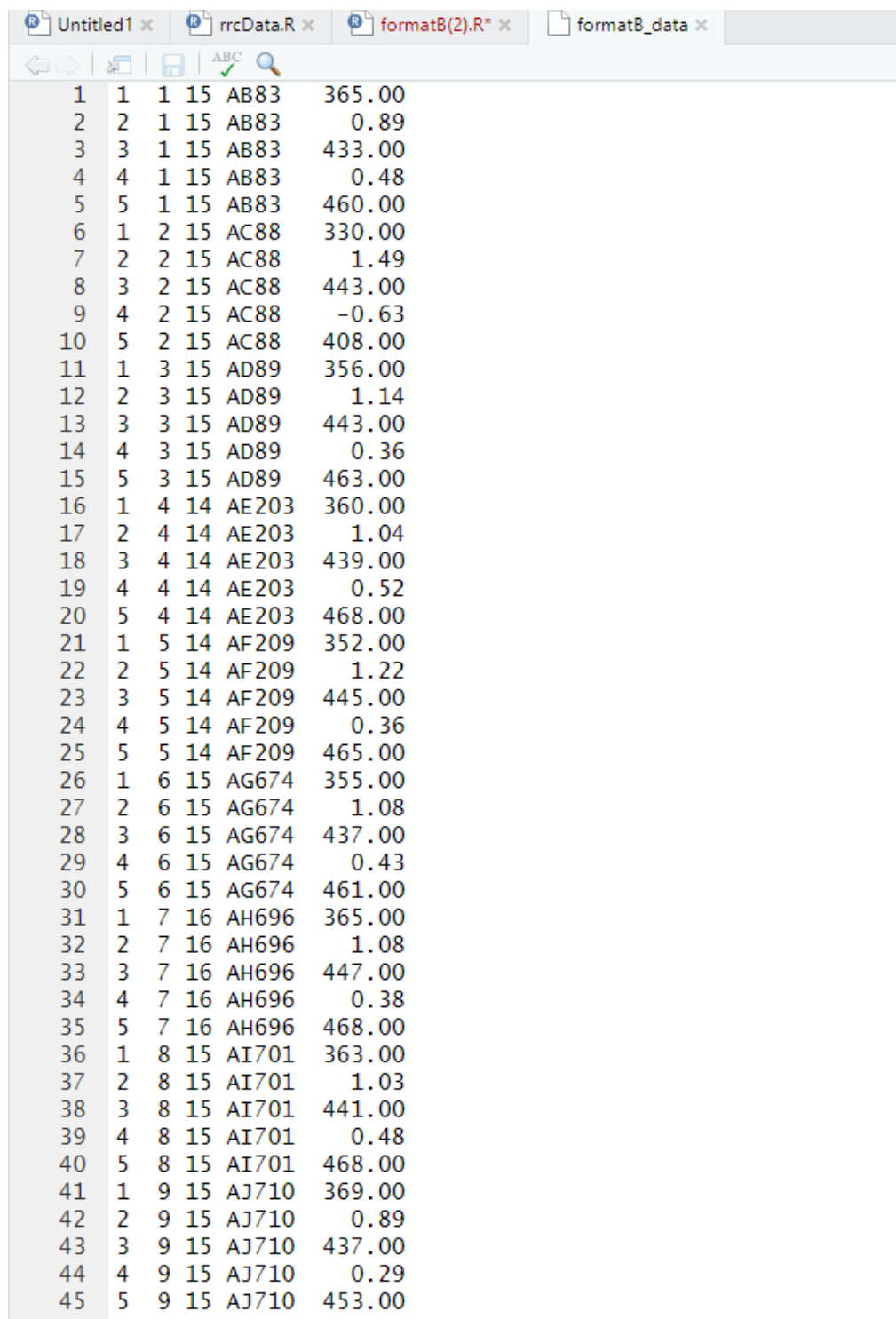
1	1	15	AB83	365	0.89	433	0.48	460
2	2	15	AC88	330	1.49	443	-0.63	408
3	3	15	AD89	356	1.14	443	0.36	463
4	4	14	AE203	360	1.04	439	0.52	468
5	5	14	AF209	352	1.22	445	0.36	465
6	6	15	AG674	355	1.08	437	0.43	461
7	7	16	AH696	365	1.08	447	0.38	468
8	8	15	AI701	363	1.03	441	0.48	468
9	9	15	AJ710	369	0.89	437	0.29	453
10	10	16	AK738	336	1.07	417	0.41	440
11	11	16	AL76	362	0.8	421	0.5	449
12	12	14	AM245	372	0.64	419	0.32	437
13	13	15	AN669	367	0.74	422	0.55	453
14	14	16	AO78	362	1.08	444	0.34	463
15	15	14	AP205	378	0.82	440	0.38	461
16	16	14	AQ247	384	0.7	437	0.23	450
17	17	18	AR248	385	0.26	405	0.79	449
18	18	15	AS671	353	0.68	405	0.79	449
19	19	16	AT682	354	0.66	404	0.41	427
20	20	18	AU727	353	1.14	440	0.48	467
21	21	16	AV731	382	0.53	422	0.61	456
22	22	16	AX734	372	0.82	434	0.43	458
23	23	16	AW748	386	0.59	431	0.32	449
24	24	13	AY199	343	1.24	437	0.52	466
25	25	18	AZ244	360	0	0	0	452
26	26	14	ABC217	386	0	0	0	467
27	27	14	ADE271	350	1.19	438	0	0
28	28	13	AFG180	348	1.24	442	0	0
29	29	18	AHI144	356	0.95	428	0	0
30								

Fonte: Elaborada pelo autor.

A `formatA` retornou um arquivo chamado `formatted_file`, com os dados formatados, pronto para o software BlupF90 ou qualquer outro software com o padrão de formatação de dados semelhante.

Desta mesma forma, após o arquivo (P) passar pela função `rrcData`, podem usar a função `formatB`. O arquivo de dados é mostrado na Figura 15.

Figura 15- Arquivo formatado pela `formatB`.



1	1	1	15	AB83	365.00
2	2	1	15	AB83	0.89
3	3	1	15	AB83	433.00
4	4	1	15	AB83	0.48
5	5	1	15	AB83	460.00
6	1	2	15	AC88	330.00
7	2	2	15	AC88	1.49
8	3	2	15	AC88	443.00
9	4	2	15	AC88	-0.63
10	5	2	15	AC88	408.00
11	1	3	15	AD89	356.00
12	2	3	15	AD89	1.14
13	3	3	15	AD89	443.00
14	4	3	15	AD89	0.36
15	5	3	15	AD89	463.00
16	1	4	14	AE203	360.00
17	2	4	14	AE203	1.04
18	3	4	14	AE203	439.00
19	4	4	14	AE203	0.52
20	5	4	14	AE203	468.00
21	1	5	14	AF209	352.00
22	2	5	14	AF209	1.22
23	3	5	14	AF209	445.00
24	4	5	14	AF209	0.36
25	5	5	14	AF209	465.00
26	1	6	15	AG674	355.00
27	2	6	15	AG674	1.08
28	3	6	15	AG674	437.00
29	4	6	15	AG674	0.43
30	5	6	15	AG674	461.00
31	1	7	16	AH696	365.00
32	2	7	16	AH696	1.08
33	3	7	16	AH696	447.00
34	4	7	16	AH696	0.38
35	5	7	16	AH696	468.00
36	1	8	15	AI701	363.00
37	2	8	15	AI701	1.03
38	3	8	15	AI701	441.00
39	4	8	15	AI701	0.48
40	5	8	15	AI701	468.00
41	1	9	15	AJ710	369.00
42	2	9	15	AJ710	0.89
43	3	9	15	AJ710	437.00
44	4	9	15	AJ710	0.29
45	5	9	15	AJ710	453.00

Fonte: Elaborada pelo autor.

O arquivo gerado pela formatB, está pronto para ser usado no software Wombat (ou qualquer outro software que exige uma formatação semelhante). A função recodificou as colunas de efeito, realizou a formatação da coluna de pedigree e dos efeitos de ambiente foi realizada pela repetição, e empilhamento dessas para cada característica presente nos dados. Os dados das características são empilhados,

gerando uma única coluna. Uma nova coluna com a identificação do número da característica é criada e inserida na posição 1 do *data frame*. Finalmente, os dados são ordenados e salvos.

Para usar a função `rrcPed`, iremos usar outro arquivo como exemplo (Figura 16). Iremos usar a função `rrcData`, como já mencionado, ela deverá ser usada inicialmente, como é mostrado na figura 17.

Figura 16- Arquivo com dados de Pedigree.

	A	B	C	D	E
1	ID	PAI	MAE	NASCIMENTO	PESO
2	A1	E1	F6	01/01/2000	456
3	A2	E2	F7	05/01/2000	587
4	A3	E3	D5	10/01/2001	475
5	A4	E4	B2	11/01/2001	488
6	A5	E5	D4	15/01/2002	501
7	A6	E6	D4	20/01/2002	503
8	A7	E7	F6	01/01/2003	452

Fonte: Elaborada pelo autor.

Figura 17- Data frame criado pela função `rrcData` a partir de um arquivo de dados com Pedigree.

```
> udata<-rrcData("C:/Users/miria/OneDrive/Documentos/TestesBLUPF90/
Pedigree Teste 1.xlsx", colsPed = 1:4, colsTraits = 5)
> udata
  ID PAI MAE NASCIMENTO PESO
1 A1  E1  F6 2000-01-01  456
2 A2  E2  F7 2000-01-05  587
3 A3  E3  D5 2001-01-10  475
4 A4  E4  B2 2001-01-11  488
5 A5  E5  D4 2002-01-15  501
6 A6  E6  D4 2002-01-20  503
7 A7  E7  F6 2003-01-01  452
```

Fonte: Elaborada pelo autor.

O arquivo utilizado possui cinco colunas: ID, PAI, MAE, NASCIMENTO e PESO. Por meio dos argumentos `colsPed = 1:4` e `colsTraits = 5`, foi informado à função que as quatro primeiras colunas correspondem às informações de pedigree (identificação do animal, pai, mãe e data de nascimento) e que a quinta coluna representa a característica fenotípica (peso).

Como as colunas de pedigree foram identificadas pelo argumento `colsPed`, a função `rrcData` não recodificou os identificadores dos animais, pois isso é responsabilidade da função `rrcPed`. O objeto resultante (`udata`) manteve os

identificadores originais nas colunas de ID, pai e mãe, preparando os dados para a próxima etapa do processo de padronização.

Após o pré-processamento realizado pela função `rrcData`, foi utilizada a função `rrcPed` com o objetivo de recodificar os identificadores do pedigree (ID, pai e mãe) em valores numéricos inteiros, conforme exigido por softwares de avaliação genética.

Foram passados como argumentos o caminho do arquivo, a indicação da presença de cabeçalho, os índices das colunas relevantes para pedigree e as colunas correspondentes no conjunto de dados `udata`. O resultado foi armazenado no objeto `resultado`, que contém: `map`: mapa com códigos originais e recodificados (Figura 18); `ped`: pedigree recodificado (Figura 19); `data`: contém o conjunto de dados original (Figura 20).

Figura 18- Recodificação com os códigos originais.

```
> resultado<-rrcPed("C:/Users/miria/OneDrive/Documentos/TestesBLUPF
90/Pedigree Teste 1.xlsx",h = TRUE,isdd = c(1, 2, 3, 4),colsPedData
= c(1, 2, 3),udata = udata)
> resultado
$map
  cod recod
1  E1      1
2  E2      2
3  E3      3
4  E4      4
5  E5      5
6  E6      6
7  E7      7
8  F7      8
9  D5      9
10 B2     10
11 D4     11
12 F6     12
13 A1     13
14 A2     14
15 A3     15
16 A4     16
17 A5     17
18 A6     18
19 A7     19
```

Fonte:

Elaborada pelo autor.

Figura 19- Pedigree recodificado.

```
$ped
  ID PAI MAE
1 13   1  12
2 14   2   8
3 15   3   9
4 16   4  10
5 17   5  11
6 18   6  11
7 19   7  12
```

Fonte: Elaborada pelo autor.

Figura 20- Conjunto de dados original.

```
$data
  ID PAI MAE NASCIMENTO PESO
1 13   1  12 2000-01-01  456
2 14   2   8 2000-01-05  587
3 15   3   9 2001-01-10  475
4 16   4  10 2001-01-11  488
5 17   5  11 2002-01-15  501
6 18   6  11 2002-01-20  503
7 19   7  12 2003-01-01  452
```

Fonte: Elaborada pelo autor.

Os dados recodificados gerados pela função `rrcPed`, especialmente o conjunto *data*, encontram-se prontos para serem utilizados em etapas posteriores de formatação. Em particular, esse resultado pode ser diretamente passado como argumento para a função `formatA` ou para `formatB`, que ajusta os arquivos ao formato exigido pelos softwares de análise genética.

6 CONCLUSÃO

As funções `rrcData`, `rrcPed`, `formatA` e `formatB` foram implementadas para padronizar e adaptar dados aos formatos exigidos pelos softwares BLUPF90 e Wombat ou qualquer outro software com o padrão de formatação de dados semelhante. As funções desenvolvidas são robustas e aplicáveis a diferentes cenários de entrada. Assim, o pacote LZFF representa uma solução prática para padronização e preparação de dados em projetos de avaliação genética animal, com o potencial para o aumento da reprodutibilidade dos resultados.

REFERÊNCIAS

BAKER, M. 1,500 scientists lift the lid on reproducibility. **Nature**, v. 533, n. 7604, p. 452–454, 1 maio 2016.

BEGLEY, C. G.; ELLIS, L. M. Drug development: Raise standards for preclinical cancer research. *Nature*, v. 483, n. 7391, p. 531–533, 28 mar. 2012.

MEYER, K. WOMBAT—A tool for mixed model analyses in quantitative genetics by restricted maximum likelihood (REML). **Journal of Zhejiang University SCIENCE B**, v. 8, n. 11, p. 815–821, out. 2007.

MISZTAL, I. BLUPF90 - a flexible mixed model program in Fortran 90. 1999.

SILVA, D. L. D. **Estratégia computacional para apoiar a reprodutibilidade e reuso de dados científicos baseado em metadados de proveniência**. Doutorado em Sistemas Digitais—São Paulo: Universidade de São Paulo, 5 set. 2017.

WICKHAM, Hadley; BRYAN, Jenny. *R packages: Organizing, testing, and releasing R code*. 2. ed. O'Reilly Media, 2019.

GANDRUD, Christopher. *Reproducible Research with R and RStudio*. 3. ed. New York: Chapman and Hall/CRC, 2019.

ANEXO 1 – SCRIPT TESTES

```
##TESTE PARA A FUNÇÃO rrcData##
#### TESTE PARA LEITURA DE ARQUIVO ####
test_that("reading files working csv", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "DP-CA.csv"),
    s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
  expect_equal(ncol(readFile), 8)
  expect_equal(nrow(readFile), 29)
})

test_that("reading files working txt", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "DP.CA2.txt"),
    s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
  expect_equal(ncol(readFile), 8)
  expect_equal(nrow(readFile), 29)
})

test_that("reading files working xls", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "SDP.SCA2.xls"),
    s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
  expect_equal(ncol(readFile), 8)
  expect_equal(nrow(readFile), 39)
})

test_that("reading files working.xlsx", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "DP.SCA2.xlsx"),
    s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
  expect_equal(ncol(readFile), 8)
  expect_equal(nrow(readFile), 29)
})

test_that("reading files working Sem extensão", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "SDPCA2"),
```

```

        s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
expect_equal(ncol(readFile), 8)
expect_equal(nrow(readFile), 39)
})

test_that("reading files working ODS", {
  readFile<-rrcData(testthat::test_path("Dados Para Teste", "TesteODS.ods"),
    s = ",", d = ".", colsPed = 1, colsTraits = 4:8)
  expect_equal(ncol(readFile), 8)
  expect_equal(nrow(readFile), 29)
})
==> Testing R file using 'testthat'

```

■ Loading LZFF

```

[ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 1 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 2 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 3 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 4 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 5 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 6 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 7 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 8 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 9 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 10 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 11 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 12 ]
Test complete

```

```

#####TESTE PARA RECODIFICAÇÃO#####
##### CRIANDO UM HELPER PARA RECODIFICAÇÃO#####
g <- sample(paste0(letters[1-4], 1:10), 10, replace = TRUE)
s<-sample(c(0, 1), 10, replace = TRUE)

```

```
testDataRecod2<-data.frame(id = 1:10, sex = s, gc = g, t1 = rnorm(10),
                           t2 = rnorm(10, 5), t3 = rnorm(10, 100, 12))
formatA(testDataRecod2)
testData1<-rrcData("formatted_file", s = " ", d = ".", h = FALSE, colsPed = 1, colsTraits
= c(4:6))
```

```
#### TESTANDO....####
```

```
test_that("Recoding is working", {
  expect_equal(nrow(testDataRecod2), nrow(testData1))
  expect_equal(nlevels(testDataRecod2$gc), nlevels(testData1$V3))
})
```

==> Testing R file using 'testthat'

i Loading LZFF

```
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 1 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 2 ]
Test complete
```

```
### TESTE PARA FUNÇÃO formatA ###
```

```
### CRIANDO UM HELPER ###
```

```
g<-sample(paste0("g", 1:3), 10, replace = TRUE)
s<-sample(c(0, 1), 10, replace = TRUE)
testData<-data.frame(id = 1:10, sex = s, gc = g, t1 = rnorm(10),
                      t2 = rnorm(10, 5), t3 = rnorm(10, 100, 12))
```

```
formatA(testData)
testData1<-rrcData("formatted_file", s = " ", d = ".", h = FALSE, colsPed = 1, colsTraits
= c(4:6))
```

```
## TESTANDO ....##
```

```
test_that("formatA is working", {
```

```

expect_equal(ncol(testData), ncol(testData1))
expect_equal(nrow(testData), nrow(testData1))
})

```

==> Testing R file using 'testthat'

i Loading LZFF

```

[ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 1 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 2 ]
Test complete

```

```

### TESTE PARA FUNÇÃO formatB ###
### CRIANDO UM HELPER ###
g<-sample(paste0("g", 1:3), 10, replace = TRUE)
s<-sample(c(0, 1), 10, replace = TRUE)
udata<-data.frame(id = 1:10, sex = s, gc = g, t1 = rnorm(10),
                  t2 = rnorm(10, 5), t3 = rnorm(10, 100, 12))
traits<-c(4:6)

```

```

d1<-udata[, -traits]
i<-2
while(i <= length(traits)){
  d1<-rbind(d1,udata[, -traits])
  i <- i + 1
}

```

```

d2<-unlist(lapply(udata[, traits], c))

```

```

d3<-rep(1:length(traits), each = length(d2)/length(traits))

```

```

## TESTANDO ....##

```

```
test_that("Effects columns are formatted correctly",{
  expect_equal(nrow(udata) * length(traits), nrow(d1))
  expect_equal(length(udata) - length(traits), length(d1))
})
```

```
test_that("Traits column is created", {
  expect_equal(nrow(udata) * length(traits), length(d2))
  expect_equal(class(d2), "numeric")
})
```

```
test_that("Number of traits column is created", {
  expect_equal(length(traits) * nrow(udata), length(d3))
  expect_equal(d3[1:(length(d2)/length(traits))], rep(1,length(d2)/length(traits)))
})
```

==> Testing R file using 'testthat'

■ Loading LZFF

```
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 1 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 2 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 3 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 4 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 5 ]
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 6 ]
Test complete
```

ANEXO 2 – SCRIPT FUNÇÕES

```
#### Função rrcData ##
Read, recode and check
#' @param local data file path
#' @param s field/column separator
#' @param d decimal point used in data file
#' @param h logical value indicating presence of header in data file
#' @param md missing data indicator
#' @param colsPed identification of columns related to pedigree data
#' @param colsTraits identification of columns related to traits
#'
#' @description
#' Simple function for read, recode and perform some checks in a data file.
#'
#' @return a data frame with data file columns read and recoded as needed. Pedigree
data are not recoded by this function
#' @export

rrcData<-function(local, s, d, h = TRUE, md = c("", " ", "NA"), colsPed = NULL,
colsTraits = NULL){
  #Data reading
  tipo<-stringr::str_extract(local,"(\\w+)$")
  if(tipo == basename(local)){
    tipo<-"txt"
  }
  if(stringr::str_detect(local,"https://docs.google.com/spreadsheets")){
    tipo<-"gsheet"
  }
  switch(tipo,
```

```

csv = dados<-utils::read.csv(local, header = h, sep = s, dec = d,
                             strip.white = FALSE, na.strings = md),

xls = dados<-as.data.frame(readxl::read_excel(local, na = md, col_names = h)),

xlsx = dados<-as.data.frame(readxl::read_excel(local, na = md, col_names = h)),

ods = dados<-as.data.frame(readODS::read_ods(local, na = md, col_names =
h)),

gsheet = {googlesheets4::gs4_auth()
          dados<-as.data.frame(googlesheets4::read_sheet(local, na = md),
                              col_names = h)},

txt = dados<-utils::read.table(local, header = h, sep = s, dec = d,
                              strip.white = FALSE, na.strings = md),

print("I did not detect the type of the file.")
)

#Checking if the file is ASCII
dadosTemp<-NULL
dadosTemp<-sapply(dados, paste0, collapse = " ")
stopifnot(all(grepl("^[-~]+$", dadosTemp)))

#####
#Recoding#
#####
#checking if all values are numeric
isnum<-sapply(dados,is.numeric)
mapList<-list()
for(i in 1:length(isnum[-colsTraits])){
  if(any(dados[,i] < 0)){
    stop("All values should be positive.")
  }
}

```

```

}
}

```

```

for(i in 1:length(isnum)){
  if(!isnum[i] && all(i != colsPed)){
    tempData<-unique(dados[,i])
    codes<-1:length(tempData)
    mapa<-data.frame("Original.Codes" = tempData, "Recodes" = codes)
    mapList<-append(mapList, list(mapa))
    index<-match(dados[,i],mapList[[length(mapList)]]$Original.Codes)
    dados[,i]<-mapList[[length(mapList)]]$Recodes[index]
  }
}

```

```

#Checking if the integers are within limits
if(any(dados[, -c(colsPed, colsTraits)] > 2147483647)){
  stop("Effects values are too big.")
}

```

```

#Checking if variances of traits are within a reasonable interval
if(!is.null((colsTraits))){
  varTemp<-sapply(dados[, colsTraits], stats::var, na.rm=TRUE)
  if(any(varTemp < 1e-5 | varTemp > 1e5)){
    warning("Variances of the traits are too small ou too big. You should scale the
data.")
  }
}
dados
}

```

```

#### Função rrcPed ####

```

```

#' Read, Recod and check Pedigree Data

```

```

#'

```

```

#' @param local data file path

```

```

#' @param s field/column separator
#' @param h logical value indicating presence of header in data file
#' @param isdd vector of columns number for individual, sire, dam and date of birth
#' @param md missing data indicator
#' @param colsPedData identification of columns with pedigree information in data
object
#' @param udata unformatted data object
#'
#' @description
#' Simple function for read, recode, format and perform checks in a pedigree file.
#'
#' @returns a list with three components: 1. a data frame with code map; 2. a data
#' frame with the formatted pedigree data; 3. a data frame with pedigree columns
#' in data recoded.
#' @export
#'
rrcPed<-function(local, s, h = FALSE, isdd = c(1, 2, 3, 4), md = c("", " ", "NA"),
                 colsPedData = c(1, 2, 3), udata){
  #Data reading
  tipo<-stringr::str_extract(local,"(\\w+)$")
  if(tipo == basename(local)){
    tipo<-"txt"
  }
  if(stringr::str_detect(local,"https://docs.google.com/spreadsheets")){
    tipo<-"gsheet"
  }
  switch(tipo,
         csv = dadosPed<-utils::read.csv(local, header = h, sep = s,
                                           strip.white = FALSE, fill = TRUE,
                                           na.strings = md),

         xls = dadosPed<-as.data.frame(readxl::read_excel(local, na = md, col_names =
h)),

```

```
xlsx = dadosPed<-as.data.frame(readxl::read_excel(local, na = md, col_names =
h)),
```

```
ods = dadosPed<-as.data.frame(readODS::read_ods(local, na = md, col_names
= h)),
```

```
gsheet = {googlesheets4::gs4_auth()
  dadosPed<-as.data.frame(googlesheets4::read_sheet(local, na = md),
    col_names = h)},
```

```
txt = dadosPed<-utils::read.table(local, header = h, sep = s,
  strip.white = FALSE, fill = TRUE,
  na.strings = md),
```

```
print("I did not detect the type of the file.")
)
```

```
#reorganizing columns
```

```
dadosPed<-data.frame(dadosPed[,isdd])
```

```
#Checking if date of birth column is character
```

```
if (inherits(dadosPed[, 4], "character")) {
```

```
  dadosPed[,4]<-as.Date(dadosPed[,4], tryFormats = c("%d/%m/%Y", "%d/%m/%y",
    "%d/%B/%Y", "%d/%B/%y",
    "%d/%b/%Y", "%d/%b/%y",
    "%m/%d/%Y", "%m/%d/%y",
    "%B/%d/%Y", "%B/%d/%y",
    "%Y-%m-%d", "%y-%m-%d"))
```

```
}
```

```
#checking if there are duplicate data
```

```
dadosPed<-unique(dadosPed)
```

```
#Checking if there are animal who appear in both columns: sire and dam
```

```

bad<-match(dadosPed[,2], dadosPed[,3], incomparables = NA)
if (!all(is.na(bad))){
  stop("Error: An animal appears in both columns: sire and dam. Please, check your
pedigree.
      Function aborted!")
}

```

#Recoding the Pedigree

```

dfPai<-data.frame(id = dadosPed[,2], datanas = as.Date(NA))
dfMae<-data.frame(id = dadosPed[,3], datanas = as.Date(NA))
dfId<-data.frame(id = dadosPed[,1], datanas = dadosPed[,4])

```

#Ordering individuals with date of birth

```

dfId<-dfId[order(dfId$datanas, na.last = FALSE), ]

```

```

mapaCod<-rbind(dfPai, dfMae, dfId) #order of the argument is important

```

#Removing duplications

```

mapaCod<-mapaCod[!duplicated(mapaCod[,1], fromLast = TRUE), ]
mapaCod<-mapaCod[!is.na(mapaCod[,1]),]

```

#Finally recoding

```

mapaCod<-data.frame(cod = mapaCod$id, recod = 1: nrow(mapaCod))

```

#Removing date of birth column

```

dadosPed<-dadosPed[,-4]

```

#Replacing the original codes in the pedigree object

```

for(j in 1:3){
  i<-match(dadosPed[,j], mapaCod[,1])
  dadosPed[,j]<-mapaCod[,2][i]
}

```

#Replacing absent parents in the pedigree object

```

dadosPed<-replace(dadosPed, list = is.na(dadosPed), values = 0)

#Checking if the file is ASCII
dadosPedTemp<-NULL
dadosPedTemp<-sapply(dadosPed, paste0, collapse = " ")
stopifnot(all(grepl("^[-~]+$", dadosPedTemp)))

#Replacing the original codes in data object
for(j in colsPedData){
  i<-match(udata[,j], mapaCod[,1])
  udata[,j]<-mapaCod[,2][i]
}

#Replacing absent parents in the data object
udata[,colsPedData]<-replace(udata[,colsPedData], list = is.na(udata[,colsPedData]),
values = 0)

pedDataList<-list(map = mapaCod, ped = dadosPed, data = udata)
pedDataList
}

#### Função formatA ####

#' Data Formatting
#'
#' @description
#' Function for formatting phenotypic data following the rules of the BLUPF90 software.
#'
#' @param udata data frame received from rrcData function with unformatted data.
#' @param omd missing data value to be written in the output file.
#' @param of output file name.
#' @param EoL end of line indicator. Unix and Linux uses "\\n", while Windows uses
"\\r\\n".
#'

```

```
#' @return a formatted file.
```

```
#'
```

```
#' @export
```

```
#'
```

```
formatA<-function(udata, omd = "0", of = "formatted_file", EoL = "\n"){
  if(stringr::str_detect(of,"#")){
    stop("File name cannot contain a #. Choose a name without a #")
  }
}
```

```
utils::write.table(udata, file = of, quote = FALSE, sep = " ",
  row.names = FALSE, col.names = FALSE, eol = EoL, na = omd)
}
```

```
##Função formatB ##
```

```
#' Formatting Data
```

```
#'
```

```
#' @param udata data frame received from rrcData function with unformatted data.
```

```
#' @param of output file name.
```

```
#' @param omd missing data value to be written in the output file.
```

```
#' @param traits vector indicating traits columns.
```

```
#' @param EoL end of line indicator. Unix and Linux uses "\n", while Windows uses
"\r\n".
```

```
#' @param width vector specifying the width of columns in the formatted file.
```

```
#'
```

```
#' @returns a formatted file.
```

```
#' @export
```

```
#'
```

```
formatB<-function(udata, of = "formatB_data", omd = "-99999", traits = NULL,
  widths = NULL, EoL = "\n"){
  if(stringr::str_detect(of,"#")){
    stop("File name cannot contain a #. Choose a name without a #")
  }
}
```

```

#Formatting columns of data
#Effects columns
d1<-udata[, -traits]
i<-2
while(i <= length(traits)){
  d1<-rbind(d1,udata[, -traits])
  i <- i + 1
}
#Traits columns
d2<-unlist(lapply(udata[, traits], c))
#Adding trait number column
d3<-rep(1:length(traits), each = length(d2)/length(traits))

udata<-data.frame(d3,d1,d2)

#Ordering
fdata<-udata[order(udata[, 2], udata[, 1], na.last = FALSE), ]

#Ensuring that output file's name is no longer than 30 characters
length_of<-nchar(of)
if(length_of > 30){
  of<-stringr::str_sub(of,-30)
}

fnames<-paste0("x",1:length(fdata))
colnames(fdata)<-fnames
rownames(fdata)<-NULL

#Writing data with fixed columns format
gdata::write.fwf(fdata, file = of, width = widths, sep = " ", na = omd,
                rownames = FALSE, colnames = FALSE, eol = EoL, scientific = FALSE)
}

```