The Journal of Open Source Software

ggpedigree: Visualizing Pedigrees with 'ggplot2' and 'plotly'

S. Mason Garrison [1¶]

1 Department of Psychology, Wake Forest University, North Carolina, USA ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Pedigree diagrams underpin research and practice across genetics, animal breeding, genealogy, forensics, and counseling. They help medical geneticists trace Mendelian disease inheritance and identify at-risk relatives; enable dairy breeders to plan matings that improve milk yield; support genealogists in reconstructing ancestry; assist forensic scientists establish familial connections in criminal investigations; and facilitate family therapists and counselors in understanding their clients' relationships (McGoldrick, Gerson, & Petry, 2020). Early R tools such as kinship2 (Sinnwell, Therneau, & Schaid, 2014) plot simple nuclear families effectively, but they do not scale to today's pedigrees that can exceed 1,000s of individuals. As datasets have grown, researchers now work with increasingly complex family structures, including large-scale plant breeding pedigrees (Shaw, Graham, Kennedy, Milne, & Marshall, 2014), web-based pedigree management systems (Ranaweera, Makalic, Hopper, & Bickerstaffe, 2018), interactive pedigree editors (Carver et al., 2018), and behavior genetic studies of extended family structures (Garrison et al., 2023; M. Hunter, Garrison, Burt, & Rodgers, 2021). Such complexity exposes limits in existing tools, which often struggle to handle large datasets and complex relationships. ggpedigree addresses this need by combining a vectorised layout algorithm, ggplot2 output, and optional plotly interactivity.

Statement of need

Pedigree visualization has traditionally relied on specialized proprietary software (e.g., Progeny, GenoPro) or R packages like kinship2 (Sinnwell et al., 2014) or pedtricks (J. Martin, Wolak, Johnston, & Morrissey, 2025), which generate static plots using base graphics and minimal layout control. While these tools are functional for small, nuclear pedigrees, their limitations become pronounced when working with complex, modern pedigree datasets. First, base R graphics lack the modular design and extensibility needed for generating publication-quality pedigree figures. Second, most existing R-based tools offer no interactivity, making it difficult to explore large pedigrees dynamically. Third, current solutions are poorly integrated with tidyverse workflows and do not produce ggplot2-based objects that users can customize or extend (Hadley Wickham, 2016).

Recent advances in behavior genetics and genetic epidemiology (Garrison et al., 2023; McArdle & McDonald, 1984) have introduced new visualization demands, particularly for extended families and highly interconnected pedigrees. Modern pedigree datasets can include hundreds or thousands of individuals across multiple generations, with overlapping relationships, consanguinity, remarriages, and twin sets that strain existing layout algorithms. Additionally, researchers increasingly need to overlay phenotypic information, genetic relatedness values, and model outputs onto pedigree plots to support interpretation and communication of their findings.

Several R packages attempt to address these challenges with built-in pedigree plotting functions.

kinship2 (Sinnwell et al., 2014) remains widely used but produces static base graphics and relies on non-vectorized recursive layout functions that does not scale well to large families. A partial ggplot2 implementation exists but is incomplete, non-vectorized, and not actively maintained. The geneHapR (Zhang, Jia, & Diao, 2023) package focuses on haplotype visualization rather than general pedigree structure. The pedgene package provides some plotting functions but is primarily designed for association testing. None of these packages offers the combination of modern ggplot2 integration, interactive capabilities, and extensibility that ggpedigree provides. ggpedigree addresses these limitations by providing a comprehensive visualization framework built on modern R graphics infrastructure. It leverages the extensive customization capabilities of ggplot2 while adding specialized functionality for pedigree-specific visualization challenges.

Software Architecture

ggpedigree is built on a modular architecture that separates data processing, layout calculation, and visualization layers. The core workflow involves: (1) data standardization and family structure analysis using BGmisc functions, (2) coordinate calculation using algorithms adapted from kinship2, (3) relationship connection mapping, and (4) layer-based plot construction using ggplot2 geometry functions. This design allows users to customize any aspect of the visualization while maintaining computational efficiency for large pedigrees. The package integrates tightly with the broader R ecosystem, particularly the tidyverse (H. Wickham et al., 2019) and BGmisc (Garrison, S. M., Hunter, M. D., Lyu, X., Trattner, J. D., & Burt, S. A., 2024). All functions return standard R objects (ggplot or plotly) that can be further customized using familiar syntax, ensuring accessibility for users already comfortable with modern R data science workflows.

Features

We briefly describe the main features of the ggpedigree package below. A more detailed description of the package's features and usage is available from the package vignettes, including examples of how to create static and interactive pedigree plots, customize aesthetics, and visualize relatedness matrices. Additional example data include squirrel data from the Kluane Red Squirrel Project (McFarlane et al., 2014, 2015) and Targaryen family data from the Song of Ice and Fire universe(G. R. Martin, 1997; G. R. R. Martin, 2018; G. R. R. Martin, Jr, & Antonsson, 2014). These datasets demonstrate the package's capabilities for both empirical and fictional pedigrees.

Core Functions

- Data Standardization and Family Structure Analysis: `ggPedigree()` integrates with BGmisc functions like `ped2fam()` to organize individuals by family, `recodeSex()` to standardize sex coding, and `checkParentIDs()` to validate pedigree structures. The function handles complex scenarios including consanguineous relationships and individuals appearing in multiple pedigree positions.
- Coordinate Calculation: `calculateCoordinates()` computes optimal positioning for individuals using algorithms adapted from `kinship2::align.pedigree`, with enhancements for large multi-generational pedigrees and complex family structures. The function returns coordinate grids that optimize spacing and minimize visual overlap. These steps are vectorized as much as possible to ensure computational efficiency, especially for large pedigrees.
- Relationship Connection Mapping: `calculateConnections()` generates connection paths between family members, mapping parent-child, sibling, spousal, and twin relationships. The function determines midpoints for line intersections and handles overlapping connections with specialized curved segments. These calculations are optimized for large

datasets by using vectorized operations rather than the loop-based approaches used in kinship2.

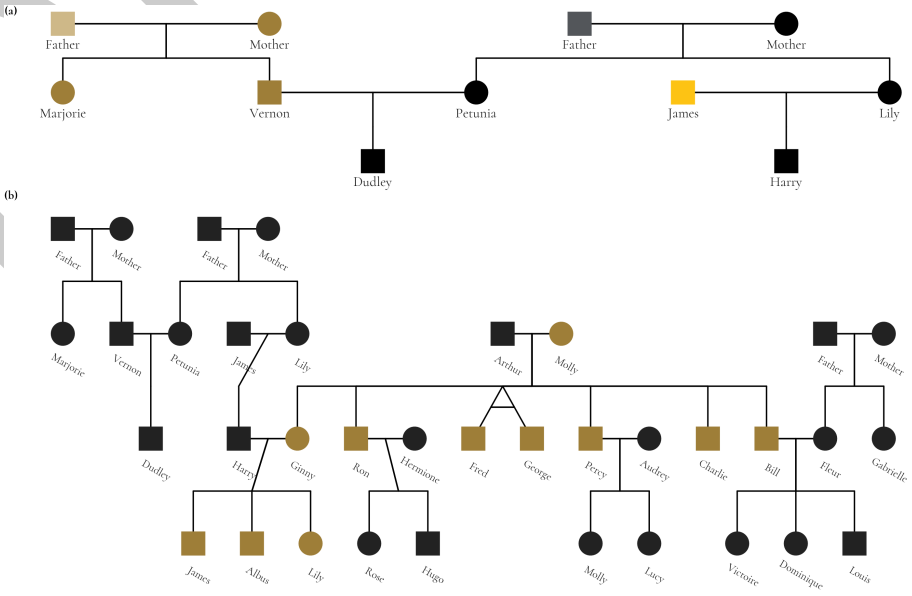
- Layer-based Plot Construction: `ggPedigree()` constructs plots using ggplot2 geometry functions, returning standard ggplot2 objects that integrate with existing R workflows. ggPedigreeInteractive() extends plots into interactive plotly widgets with hover tooltips and dynamic exploration capabilities. A comprehensive config system allows customization of over 100 aesthetic and layout parameters.
- Focal Individual Highlighting: Advanced functionality to highlight specific individuals and their relatives based on additive genetic, mitochondrial, or other relationship matrices.

Specific Visualization Functions

- Pedigree Plotting: `ggPedigree()` creates static pedigree plots using ggplot2, supporting complex family structures, multiple generations, and customizable aesthetics. It can handle large pedigrees with thousands of individuals while maintaining clarity and readability.
- Interactive Pedigree Visualization: `ggPedigreeInteractive()` generates interactive pedigree plots using plotly, allowing users to explore large pedigrees dynamically. Users can configure tooltip content to display individual IDs, names, phenotypic information, and other relevant data. It supports tooltips, zooming, and panning for detailed exploration of family structures.
- Relatedness Matrix Heatmaps: `ggRelatednessMatrix()` creates customizable heatmap visualizations for genetic relatedness matrices with support for hierarchical clustering, interactive exploration, and seamless integration with BGmisc relatedness calculations.
- Phenotype-Relatedness Analysis: `ggPhenotypeByDegree()` produces specialized plots for visualizing phenotypic correlations as a function of genetic relatedness, including confidence intervals and statistical summaries for quantitative genetic analysis.

Illustrative Example of ggPedigree

Here, we illustrate some of the features by reproducing the figure from (M. D. Hunter et al., 2025) but restyled to conform to the Wake Forest Style Guidelines. As you can see, the ggpedigree package allows for complex family structures to be visualized in a clear and aesthetically pleasing manner.



We have combined two figures using `patchwork` (Pedersen, 2025) to show the pedigree of the

Potter family from the Harry Potter series. The first plot (a) highlights the individual mtdna lines in the Dursley and Evans families, while the second plot (b) shows the entire family structure with mitochondrial relationships highlighted in a gradient color scale. Features such as focal individual highlighting, custom color scales, text labels, fonts, are easily implemented. The package supports both static and interactive plots, allowing users to explore large pedigrees dynamically.

The plot includes individuals' names, relationships, and phenotypic information, all while maintaining clarity even with overlapping relationships. Although pleasing, this level of customization is not required, as default settings produce clear and informative pedigree plots.

Collectively, these tools provide a valuable resource for behavior geneticists and others who work with extended family data. They were developed as part of a grant and have been used in several ongoing projects, presentations (Garrison, 2024; M. D. Hunter, Garrison, Lyu, Good, & Burt, 2024), and forthcoming papers (Burt et al., 2025; M. D. Hunter et al., 2025; Lyu, Hunter, Good, Carroll, & Garrison, 2025).

Availability

The `ggpedigree` package is open-source and available on both GitHub at https://github.com/R-Computing-Lab/ggpedigree and the Comprehensive R Archive Network (CRAN) at https://cran.r-project.org/package=ggpedigree. It is licensed under the GNU General Public License.

Acknowledgments

The current research is indirectly supported by the National Institute on Aging (NIA), RF1-AG073189. The authors would like to thank Michael Hunter for their support and enthusiasm to the development of this package.

References

Burt, S. A., Garrison, S. M., Lyu, X., Rodgers, J. L., Carroll, S. L., Smith, K. R., & Hunter, M. D. (2025). *Inherited mtDNA contributes to longevity: Evidence from extended pedigrees with 176 million kinship pairs. Revise and resubmit at* eBioMedicine.

Carver, T., Cunningham, A. P., Babb de Villiers, C., Lee, A., Hartley, S., Tischkowitz, M., Walter, F. M., et al. (2018). Pedigreejs: A web-based graphical pedigree editor. *Bioinformatics*, *34*(6), 1069–1071. doi:10.1093/bioinformatics/btx705

Garrison, S. M. (2024). Charting new paths in behavior genetics: Developing a comprehensive r visualization atlas. *Behavior Genetics* (Vol. 54, p. 488488). SPRINGER. Retrieved from https://scholar.google.com/scholar?cluster=4710819130852314260&hl=en&oi=scholarr

Garrison, S. M., Hunter, M. D., Lyu, X., Trattner, J. D., & Burt, S. A. (2024). BGmisc: An r package for extended behavior genetics analysis. *Journal of Open Source Software*, *9*(94). doi:10.21105/joss.06203

Garrison, S. M., Lyu, X., Hunter, M. D., Rodgers, J. L., Smith, K. R., Coon, H., & Burt, S. A. (2023). Analyzing extended cousin similarity to unravel the mystery of mtDNA and longevity. *Behavior Genetics*. doi:10.1007/s10519-023-10156-9

Hunter, M. D., Garrison, S. M., Lyu, X., Good, R., & Burt, S. A. (2024). Tools for biometric modeling in large population databases. *Behavior Genetics* (Vol. 54, p. 531531). SPRINGER. Retrieved from https://scholar.google.com/scholar?cluster=16878992125450714238&hl=en&oi=scholarr

Hunter, M. D., Garrison, S. M., Lyu, X., Good, R., Carroll, S. L., & Burt, S. A. (2025). *Tracing the right path: Determination of large pedigree segmentation and relatedness. Revise and Resubmit at Behavior Genetics*.

Hunter, M., Garrison, S. M., Burt, S. A., & Rodgers, J. L. (2021). The Analytic Identification of Variance Component Models Common to Behavior Genetics. *Behavior Genetics*, *51*(4), 425–437. doi:10.1007/s10519-021-10055-x

Lyu, S. A., X.and Burt, Hunter, M. D., Good, R., Carroll, S. L., & Garrison, S. M. (2025). Detecting mtDNA effects with an extended pedigree model: An analysis of statistical power and estimation bias. *Behavior Genetics*. doi:10.1101/2024.12.19.629449

Martin, G. R. (1997). *A game of thrones* (Vol. 1). Bantam.

Martin, G. R. R. (2018). *Fire & blood: 300 years before a game of thrones*. New York: Random House Worlds.

Martin, G. R. R., Jr, E. M. G., & Antonsson, L. (2014). *The world of ice & fire: The untold history of westeros and the game of thrones*. New York: Random House Worlds.

Martin, J., Wolak, M., Johnston, S., & Morrissey, M. (2025). *Pedtricks: Visualize, summarize and simulate data from pedigrees*. doi:10.32614/CRAN.package.pedtricks

McArdle, J. J., & McDonald, R. P. (1984). Some algebraic properties of the reticular action model for moment structures. *British Journal of Mathematical and Statistical Psychology*, *37*, 234–251. doi:10.1111/j.2044-8317.1984.tb00802.x

McFarlane, S. E., Boutin, S., Humphries, M. M., McAdam, A. G., Gorrell, J. C., & Coltman, D. W. (2015, January 21). Data from: Very low levels of direct additive genetic variance in fitness and fitness components in a red squirrel population. Dryad. doi:10.5061/DRYAD.N5Q05

McFarlane, S. E., Gorrell, J. C., Coltman, D. W., Humphries, M. M., Boutin, S., & McAdam, A. G. (2014). Very low levels of direct additive genetic variance in fitness and fitness components in a red squirrel population. *Ecology and Evolution*, *4*(10), 1729–1738. doi:10.1002/ece3.982

McGoldrick, M., Gerson, R., & Petry, S. (2020). *Genograms: Assessment and Treatment*. Erscheinungsort nicht ermittelbar: W. W. Norton & Company.

Pedersen, T. L. (2025). *Patchwork: The composer of plots*. doi:10.32614/CRAN.package.patchwork

Ranaweera, T., Makalic, E., Hopper, J. L., & Bickerstaffe, A. (2018). An open-source, integrated pedigree data management and visualization tool for genetic epidemiology. *International Journal of Epidemiology*, *47*(4), 1034–1039. doi:10.1093/ije/dyy049

Shaw, P. D., Graham, M., Kennedy, J., Milne, I., & Marshall, D. F. (2014). Helium: Visualization of large scale plant pedigrees. *BMC Bioinformatics*, *15*(1), 259. doi:10.1186/1471-2105-15-259

Sinnwell, J. P., Therneau, T. M., & Schaid, D. J. (2014). The kinship2 r package for pedigree data. *Human Heredity*, *78*, 91–93. doi:10.1159/000363105

Wickham, Hadley. (2016). *ggplot2*. Use R! Cham: Springer International Publishing. doi:10.1007/978-3-319-24277-4

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Grolemund, G., et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. doi:10.21105/joss.01686

Zhang, R., Jia, G., & Diao, X. (2023). geneHapR: An r package for gene haplotypic statistics and visualization. *BMC Bioinformatics*, *24*(1), 199. doi:10.1186/s12859-023-05318-9

Appendix

Source code for the figures above is available in the `vignettes/articles/_paper.Rmd` file. The code below reproduces the figure using the ggpedigree package.

```
library(ggpedigree) # ggPedigree lives here
library(BGmisc) # helper utilities & example data
library(tidyverse) # for data manipulation and plotting
library(showtext)
library(sysfonts)
library(patchwork) # for combining plots

# Load the potter pedigree data
data("potter")
# Removing Last names
df_potter <- potter %>%
  mutate(
    name = case_when(
      personID == 1 ~ "Vernon",
      personID == 2 ~ "Marjorie",
      personID == 3 ~ "Petunia",
      personID == 4 ~ "Lily",
      personID == 5 ~ "James",
      personID == 6 ~ "Dudley",
      personID == 7 ~ "Harry",
      personID == 8 ~ "Ginny",
      personID == 9 ~ "Arthur",
      personID == 10 ~ "Molly",
      personID == 11 ~ "Ron",
      personID == 12 ~ "Fred",
      personID == 13 ~ "George",
      personID == 14 ~ "Percy",
      personID == 15 ~ "Charlie",
      personID == 16 ~ "Bill",
      personID == 17 ~ "Hermione",
      personID == 18 ~ "Fleur",
      personID == 19 ~ "Gabrielle",
      personID == 20 ~ "Audrey",
      personID == 21 ~ "James",
      personID == 22 ~ "Albus",
      personID == 23 ~ "Lily",
      personID == 24 ~ "Rose",
      personID == 25 ~ "Hugo",
      personID == 26 ~ "Victoire",
      personID == 27 ~ "Dominique",
      personID == 28 ~ "Louis",
      personID == 29 ~ "Molly",
      personID == 30 ~ "Lucy",
      personID == 101 ~ "Mother",
      personID == 102 ~ "Father",
      personID == 103 ~ "Mother",
      personID == 104 ~ "Father",
      personID == 105 ~ "Mother",
      personID == 106 ~ "Father"
    )
```

```
  )
# Load Google fonts for styling
font_add_google(name = "Cormorant", family = "cormorant")
showtext_auto() # Load the showtext package to render Google fonts

# Set the WFU style guidelines for the plot
text_color_wfu <- "#222222" # dark grey for text labels
focal_fill_color_values_wfu <- c(
  "#9E7E38", "#000000", "#FDC314", "#CEB888", "#53565A")
family_wfu <- "cormorant"
text_size_wfu <- 14

# Create Panel A
m1 <- ggPedigree(df_potter %>% filter(personID %in% c(1:7, 101:104)),
  famID = "famID",
  personID = "personID",
  config = list(
    label_include = TRUE,
    label_column = "name",
    point_size = 8,
    # outline_include = TRUE,
    focal_fill_personID = 8,
    outline_multiplier = 1.5,
    segment_linewidth = 0.5,
    label_text_size = 11, # 3,
    label_text_color = text_color_wfu,
    axis_text_color = text_color_wfu,
    label_text_family = family_wfu,
    focal_fill_include = TRUE,
    label_nudge_y = -0.30,
    focal_fill_method = "manual",
    focal_fill_color_values = focal_fill_color_values_wfu,
    focal_fill_force_zero = TRUE,
    label_method = "geom_text",
    focal_fill_na_value = text_color_wfu,
    focal_fill_scale_midpoint = 0.40,
    focal_fill_component = "matID",
    focal_fill_labels = NULL,
    sex_legend_show = FALSE,
    sex_color_include = FALSE
  )
) + guides(shape = "none") + theme(
  plot.title = element_blank(),
  plot.title.position = "plot",
  text = element_text(family = family_wfu,
                      size = 14)
) + coord_cartesian(ylim = c(3.25, 1),
                    clip = "off")
# Create Panel B
m2 <- ggPedigree(df_potter,
  famID = "famID",
  personID = "personID",
  config = list(
    label_include = TRUE,
    label_column = "name",
```

```
    point_size = 8,
    focal_fill_personID = 8, # Molly Weasley
    outline_multiplier = 1.5,
    segment_linewidth = 0.5,
    label_text_size = 9,
    label_text_family = family_wfu,
    label_text_color = text_color_wfu,
    axis_text_color = text_color_wfu,
    label_nudge_y = -0.3,
    label_nudge_x = .1,
    focal_fill_include = TRUE,
    focal_fill_method = "gradient2",
    focal_fill_high_color = "#9E7E38",
    focal_fill_mid_color = "#9E7E38",
    focal_fill_low_color = text_color_wfu[2],
    focal_fill_scale_midpoint = 0.85,
    focal_fill_component = "mitochondrial",
    focal_fill_force_zero = TRUE,
    label_method = "geom_text",
    focal_fill_na_value = text_color_wfu,
    label_text_angle = -30,
    sex_legend_show = FALSE,
    sex_color_include = FALSE
  )
) + theme(
  legend.position = "none",
  plot.title = element_blank(),
  plot.title.position = "plot",
  text = element_text(family = family_wfu,
                      size = text_size_wfu, face = "bold")
) + coord_cartesian(ylim = c(4.25, .9), clip = "off")

# Combine the two plots using patchwork
showtext_auto()
result <- m1 + m2 +
  plot_layout(
    ncol = 1, heights = c(1.1, 2.5),
    guides = "collect", tag_level = "new"
  ) +
  plot_annotation(
    tag_levels = list(c("(a)", "(b)")),
    theme = theme(plot.margin = margin(0, 0, 0, 0), )
  ) +
  guides(shape = "none") &
  theme(
    legend.position = "none",
    plot.margin = unit(c(0, 0, 0.0, 0), "lines"),
    plot.tag = element_text(family = family_wfu,
                            size = 2*text_size_wfu, face = "bold")
  )

# save as a png
ggsave(
  filename = "wfu_potter_pedigree.png",
  plot = result,
```

```
  width = 9.5, height = 6, dpi = 300, units = "in"
)
```