

¹ daltoolbox: Leveraging Experiment Lines for Modular and Reproducible Data Analytics

³ **Eduardo Ogasawara**  ¹, **Ana Carolina Sá**  ¹, **Antonio Castro**  ¹, **Caio Santos**  ¹, **Diego Carvalho**  ¹, **Diego Salles** ¹, **Eduardo Bezerra** ¹, **Esther Pacitti** ³, **Fabio Porto** ², **Janio Lima** ¹, **Lucas Tavares** ¹, **Rafaelli Coutinho** ¹, **Rebecca Salles** ⁴, and **Vinicius Saidy** ¹

⁷ 1 Federal Center for Technological Education of Rio de Janeiro (CEFET/RJ), Brazil 2 National
⁸ Laboratory of Scientific Computing (LNCC), Brazil 3 University of Montpellier, LIRMM, France 4
⁹ National Institute for Research in Digital Science and Technology (INRIA), France 5 Petróleo Brasileiro
¹⁰ S.A. (Petrobras), Brazil

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#))

¹¹ Summary

The **daltoolbox** package provides an open-source framework for constructing modular and reproducible data analytics workflows in R. Built upon the concept of *Experiment Lines* (EL) (Marinho et al., 2017), **daltoolbox** enables the definition of flexible experiment families through the composition of alternative preprocessing, modeling, and evaluation steps. This design allows researchers and practitioners to create, compare, and evolve analytical workflows with minimal code modification. The package integrates with external R and Python libraries, fostering interoperability and transparency in experimental data analysis.

¹⁹ Background

The rapid expansion of data-driven research across domains such as finance, healthcare, and environmental sciences has increased the need for tools that support reproducibility, modularity, and flexibility in data analytics. Researchers often need to construct and compare multiple workflows, each differing in transformation methods, learning algorithms, or evaluation criteria. However, managing this variability is time-consuming and error-prone when using traditional scripting or static pipeline tools. Scientific workflow systems have advanced reproducibility but often lack the flexibility required for experimentation.

The concept of *Experiment Lines* (EL) (Marinho et al., 2017), derived from software product line engineering, extends workflow design by introducing **variability** (alternative components) and **optionality** (configurable presence or absence of steps). **daltoolbox** operationalizes EL principles for data analytics, providing a practical, code-based framework for managing experimental diversity.

³² Statement of Need

Data analytics workflows frequently require the exploration of multiple preprocessing, modeling, and evaluation alternatives. Managing these alternatives often leads to repetitive code, fragmented design, and limited traceability, which hinder reproducibility across experiments. **daltoolbox** was developed to address this challenge by enabling modular and flexible experiment definition through a unified interface.

The **target audience** includes researchers, educators, and data practitioners who require

39 transparent, reproducible workflows for experimentation in classification, regression, clustering,
40 and time series prediction. The package is particularly valuable in academic and applied
41 research contexts, where multiple analytical alternatives must be compared under controlled
42 conditions.

43 daltoolbox provides a consistent syntax and modular architecture that facilitate systematic
44 experimentation. Users can easily modify, replace, or omit workflow components, allowing
45 efficient exploration of design alternatives while preserving reproducibility and transparency.

46 State of the Field

47 Several tools exist for designing machine learning workflows. Visual environments such as
48 **WEKA** (Witten et al., 2016), **Orange** (Demsar et al., 2013), and **KNIME** (Berthold et al.,
49 2009) are widely used for education and prototyping but offer limited flexibility for dynamic
50 reconfiguration. Frameworks such as **Scikit-learn** (Pedregosa et al., 2011) and **MLlib** (Meng et
51 al., 2016) provide robust APIs but focus on static pipelines rather than structured workflow
52 variability. AutoML systems like **Auto-WEKA** (Kotthoff et al., 2017) and **Auto-sklearn** (Feurer
53 et al., 2015) automate model selection but reduce user control and transparency.

54 daltoolbox differentiates itself by offering explicit modeling of variability and optionality,
55 allowing controlled exploration of alternatives. This focus on transparency and user-driven
56 design complements rather than replaces existing ML frameworks, positioning daltoolbox as an
57 intermediary layer for reproducible experimentation.

58 Main Features

- 59 ▪ Unified API for **transformation**, **classification**, **regression**, and **clustering**.
- 60 ▪ Explicit modeling of *optional* and *variable* workflow components.
- 61 ▪ Modular operators for scaling, normalization, and dimensionality reduction.
- 62 ▪ Easy substitution of models and preprocessing steps without code refactoring.
- 63 ▪ Visualization utilities for model comparison and interpretation.
- 64 ▪ Interoperability with external R and Python libraries.
- 65 ▪ Comprehensive documentation and testing, distributed under the MIT license.

66 Example Usage

```
# Define a tiny workflow runner once
DemoWorkflow <- function(model, prep, train, test) {
  prep <- fit(prep, train)
  train <- transform(prep, train)
  model <- fit(model, train)
  predict(model, test)
}

# Scenario A: skip transformation (no-op) + KNN
prep_a <- dal_transform() # no-op transformer
model_a <- cla_knn("rain", levels = c("yes", "no"), k = 3)
preds_a <- DemoWorkflow(model_a, prep_a, train, test)

# Scenario B: min-max normalization + Random Forest
prep_b <- minmax()
model_b <- cla_rf("rain", levels = c("yes", "no"))
preds_b <- DemoWorkflow(model_b, prep_b, train, test)
```

⁶⁷ This pattern shows how a single workflow function enables testing alternative pipelines by
⁶⁸ switching only the prep or model component, without refactoring code.

⁶⁹ Acknowledgements

⁷⁰ This work was partially supported by **CNPq**, **CAPES**, and **FAPERJ**. The authors acknowledge
⁷¹ the DAL community and institutional partners for their support.

⁷² References

- ⁷³ See paper.bib for the complete list of references.
- ⁷⁴ Berthold, M. R., Cebron, N., Dill, F., Gabriel, T. R., Kötter, T., Meinl, T., Ohl, P., Thiel, K.,
⁷⁵ & Wiswedel, B. (2009). KNIME - the konstanz information miner: Version 2.0 and beyond.
⁷⁶ *Journal of Machine Learning Research*, 11, 3191–3195.
- ⁷⁷ Demsar, J., Curk, T., Erjavec, A., Gorup, C., Hocevar, T., Milutinovic, M., Mozina, M.,
⁷⁸ Polajnar, M., Toplak, M., Staric, A., Stajdohar, M., Umek, L., Zagar, L., Zbontar, J.,
⁷⁹ Zitnik, M., & Zupan, B. (2013). Orange: Data mining toolbox in python. *Journal of*
⁸⁰ *Machine Learning Research*, 14(71), 2349–2353.
- ⁸¹ Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M., & Hutter, F.
⁸² (2015). Efficient and robust automated machine learning. *Advances in Neural Information*
⁸³ *Processing Systems*, 2962–2970.
- ⁸⁴ Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., & Leyton-Brown, K. (2017). Auto-WEKA
⁸⁵ 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of*
⁸⁶ *Machine Learning Research*, 18, 1–5.
- ⁸⁷ Marinho, A., Oliveira, D. de, Ogasawara, E., Silva, V., Ocana, K., Murta, L., Braganholo,
⁸⁸ V., & Mattoso, M. (2017). Deriving scientific workflows from algebraic experiment
⁸⁹ lines: A practical approach. *Future Generation Computer Systems*, 68, 111–127. <https://doi.org/10.1016/j.future.2016.08.016>
- ⁹¹ Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D.
⁹² B., Amde, M., Owen, S., Xin, D., Franklin, M. J., Zadeh, R., Zaharia, M., & Talwalkar, A.
⁹³ (2016). MLLib: Machine learning in apache spark. *Journal of Machine Learning Research*,
⁹⁴ 17, 1–7.
- ⁹⁵ Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M.,
⁹⁶ Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D.,
⁹⁷ Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python.
⁹⁸ *Journal of Machine Learning Research*, 12, 2825–2830.
- ⁹⁹ Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data mining: Practical machine*
¹⁰⁰ *learning tools and techniques* (4th ed.). Morgan Kaufmann. ISBN: 978-0-12-804357-8