lpjmlkit: A toolkit for operating LPJmL and model-specific data processing

Jannes Breier [1*¶]**, Sebastian Ostberg** [1*¶]**, Stephen Wirth** [1,2]**, Sara Minoli** [1,3]**, Fabian Stenzel** [1]**, and Christoph Müller** [1]

1 Potsdam Institute for Climate Impact Research (PIK), Member of the Leibniz Association, P.O. Box 60 12 03, 14412 Potsdam, Germany **2** Institute of Crop Science and Plant Breeding, Grass and Forage Science/Organic Agriculture, Kiel University, Hermann-Rodewald-Str. 9, 24118, Kiel, Germany **3** Climate Focus, Germany **¶** Corresponding author ***** These authors contributed equally.

Summary

The *lpjmlkit* R package (Breier et al., 2023) is an open source software that is developed for handling the open source dynamic global vegetation model (DGVM) LPJmL. It contains two main modules. One, *LPJmL Runner*, provides the functionality to create multiple model configurations and start the corresponding simulations either on a personal computer or on an HPC (High Perfomance Computing) cluster with SLURM (Simple Linux Utility for Resource Management) support, in both cases requiring a working LPJmL installation. The other, *LPJmL Data*, offers a generic function that supports reading both simulation output and model input data in multiple file formats used by LPJmL. The associated data class LPJmLData contains both the data and the corresponding metadata to ensure data integrity within a single instance. LPJmLData objects act as data containers that provide modification functions such as subsetting or transformations of the data. LPJmLData objects can be exported into various other common R data formats. In addition to these modules, other functions are included to facilitate common use cases of LPJmL. This article introduces *lpjmlkit*, an R package that serves as an interface to LPJmL to simplify direct work with the model and to enable new generic software developments based on LPJmL simulations or data.

Statement of need

A simple interface facilitates the use of software and improves accessibility, user experience, and overall adoption. Scientific software should also be findable, accessible, interoperable and reusable, according to the FAIR principles for research software (Barker et al., 2022). Numerical models based on process-based approaches and implemented in low-level programming languages are often characterised by a long history and long development cycles, that affect operability. These models therefore often lack simple and reliable interfaces that are also FAIR (Barker et al., 2022; Wilson et al., 2014). LPJmL is a well-established dynamic global vegetation model

34 (DGVM), widely used in the scientific community. A DGVM simulates potential vegetation
35 shifts and associated biogeochemical and hydrological cycles in response to changes in climate.
36 (Cramer et al., 2001) DGVMs are typically used to simulate the effects of future climate
37 change on natural vegetation and its carbon and water cycles. In this context LPJmL has
38 been used for more than a decade and was employed by researchers to conduct numerous
39 studies in various research areas related to the terrestrial biosphere. To this end, the original
40 DGVM LPJ (Sitch et al., 2003) was extended by adding an improved representation of the
41 hydrological cycle (Gerten et al., 2004), by implementing managed land components, forming
42 "LPJmL" (LPJ with *m*anaged *L*and)) (Bondeau et al., 2007; Lutz et al., 2019; Rolinski et al.,
43 2018; Schaphoff et al., 2018), and by including the nitrogen cycle (Bloh et al., 2018). This
44 facilitated broader, interdisciplinary studies such as the work of Gerten et al. (2020), which
45 answered the question of whether it is possible to feed ten billion people within four planetary
46 boundaries, or studies that implemented features of sustainable agriculture in LPJmL (Herzfeld
47 et al., 2021; Porwollik et al., 2022) that had not been simulated before within a DGVM. There
48 are many other examples of different scientific studies based on LPJmL simulations, all using
49 individual scripts to create inputs for the model or analyze outputs.

50 Unlike its sister model LPJ-GUESS (Bagnara et al., 2019) or other models, such as MAgPIE
51 (Dietrich et al., 2019), LPJmL was never equipped with standardised interfaces for higher level
52 programming languages to either run simulations or read and process input or output data.
53 The lack of a standardized interface means that both beginners and experienced users need
54 to constantly develop their own custom scripts and tools for what should be routine (data)
55 processing steps. Tools shared informally between individual users often have limited scope
56 and as such limited re-usability. They lack documentation and are often not well tested or
57 maintained, nor released to a wide user base in a transparent manner.

58 *lpjmlkit* was developed to address these problems and at the same time create a standard in
59 the handling of LPJmL to improve the management of simulation experiments and allow for
60 better documentation and reproducibility of studies with LPJmL. By using the *LPJmL Runner*
61 functionality, model configurations are stored in a single and unique configuration file that
62 references the exact model version of LPJmL used for the simulations to achieve reproducible
63 results. *LPJmL Data* subsequently ensures a generic standard for version-independent processing
64 of LPJmL (output) data. This way, *lpjmlkit* serves as a user and programming interface to
65 LPJmL and provides an easy-to-use basis for interaction with LPJmL in a simple R script
66 as well as for further software development based on LPJmL, for example model calibration,
67 benchmarking or indicator development.

Package features

69 *lpjmlkit* is an R package that contains two main modules, each of which contains supplementary
70 documentation in form of a guide in addition to the package documentation:

71 - *LPJmL Runner* vignette
72 - *LPJmL Data* vignette

73 Together, these modules can be used to configure and run LPJmL model simulations as well
74 as read and process the resulting data.

LPJmL Runner

76 The Runner module is designed to operate LPJmL on Unix-based operating systems that have
77 a working LPJmL installation (version 4 or higher) and includes four key functions.

78 The basic idea is to create tables for the parameters and settings to be changed from a default
79 configuration file for single runs or any number of related simulations, e.g. multi-scenario or
80 uncertainty/sensitivity analyses. The creation of these tables is based on the concept of `tidy`
81 data with columns as parameters/settings (variables) and rows as simulations (observations)

⁸² ([Wickham, 2014](#)). In order to facitlitate the execution of these simulations, the function
⁸³ `write_config()` handles the formatting of the corresponding `tidy` data tables into LPJmL
⁸⁴ compatible JSON configuration files.

⁸⁵ LPJmL simulations can either be run sequentially on single CPUs with `run_lpjml()`, which
⁸⁶ is particularly suitable for testing the model or for focusing on small regions. In addition,
⁸⁷ simulations can be submitted to an HPC cluster equipped with the SLURM workload manager
⁸⁸ using `submit_lpjml()`, which is particularly useful for computing multiple global simulations
⁸⁹ in parallel mode using multiple CPUs.

⁹⁰ LPJmL Data

⁹¹ While the module *LPJmL Runner* covers the whole range of running LPJmL simulations, `LPJmL`
⁹² `Data` provides the tools for the subsequent data analysis part. LPJmL simulations output data
⁹³ in a raw binary data format designed for memory efficient data structures. The LPJmL model
⁹⁴ can write metadata for each output file, either as a file header or as an additional metadata file
⁹⁵ in JSON format. Using `read_io()`, LPJmL raw output data can be read into a standardised
⁹⁶ data format called `LPJmLData`, which makes use of the corresponding metadata to correctly
⁹⁷ represent the data arrays (dimensions, units, etc.).

⁹⁸ This data format is designed to faciliitate the functionalities commonly used by LPJmL users
⁹⁹ for data analysis. In addition to the descriptive statistics that an LPJmLData object displays
¹⁰⁰ by default, there is a `plot()` method to easily display the the data for visual inspection.
¹⁰¹ LPJmLData objects can be transformed into various common spatial and temporal formats with
¹⁰² the funciton `transform()`. For example, the one-dimensional cell vector can be transformed
¹⁰³ into a longitude-latitude matrix to allow further spatial operations such as "nearest neighbour"
¹⁰⁴ subsetting, using coordinate pairs. The temporal dimension can be decomposed into further
¹⁰⁵ sub-time dimensions, which is particularly useful for high temporal resolutions such as monthly
¹⁰⁶ or daily data covering several years. Several export functions allow for easy conversion of the
¹⁰⁷ LPJmLData format to other common data formats such as `SpatRaster` (`as_terra()`), `tibble`
¹⁰⁸ (`as_tibble()`) or simply array (`as_array()`). Here, in addition to subsetting, the data can
¹⁰⁹ also be aggregated using common functions such as `mean()` or `sum()`.

¹¹⁰ Especially in combination, these different functions can greatly increases the utility for data
¹¹¹ analysis. For example, a combination of the transformation of the read-in discharge into sub-
¹¹² time dimensions and latitude-longitude matrix and the subsequent subsetting of the summer
¹¹³ months of the northern hemisphere and the aggregation over years and months allows the
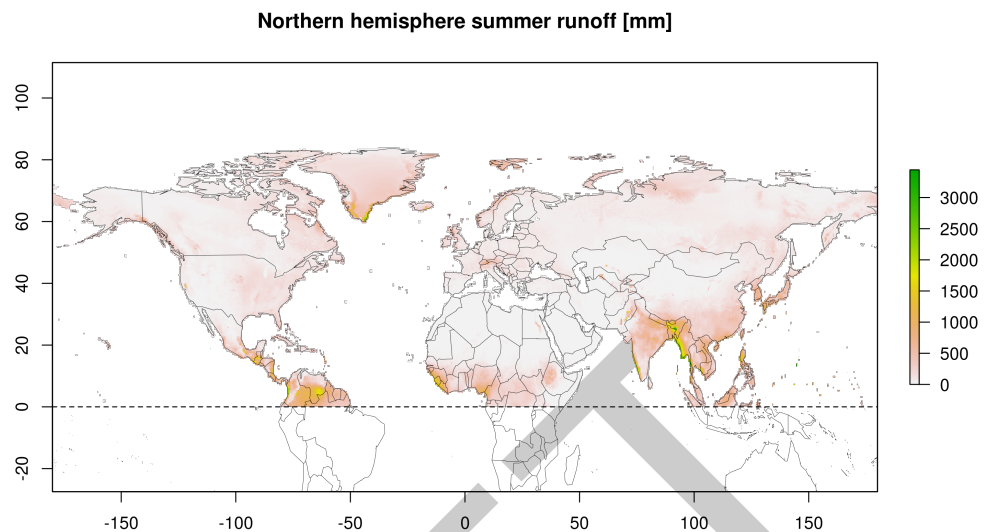¹¹⁴ following presentation.

Northern hemisphere summer runoff [mm]

Figure 1: Exemplary LPJmLData plot of variable runoff generated by functions read_io, transform, subset and plot

Miscellaneos

The *lpjmlkit* package also includes additional functions to support different applications that involve handling LPJmL or related data.

For example, the function `calc_cellarea()` calculates the area of the grid cells in an `LPJmLData` object or for any vector of latitude coordinates representing a regular grid with a set spatial resolution.

While `read_io()` is the main function to access LPJmL input and output files, *lpjmlkit* also includes a number of functions providing low-level access to the binary LPJmL file format: `read_header()`, `create_header()`, and `write_header()` allow to retrieve/generate/save file headers, while `get_headersize()` and `get_datatype()` return information about the header and about the data type used in different LPJmL files, respectively. These low-level access functions can form the basis for new applications, for example, to generate new LPJmL input files.

Documentation & License

lpjmlkit is an open-source software package for operating LPJmL and processing related data that provides comprehensive documentation and vignettes. The documentation is available online at https://pik-piam.r-universe.dev/lpjmlkit and includes instructions for installation, usage, and examples. The package is licensed under the GNU Affero General Public License (AGPL-3.0), which grants users the freedom to access, use, and modify the code, and ensures that any modifications or derivative works are also available under the same license, allowing for continued collaboration and development within the community. The source code is available on GitHub at https://github.com/PIK-LPJmL/lpjmlkit, where users can also report issues and suggest improvements.

Acknowledgements

References

Bagnara, M., Silveyra Gonzalez, R., Reifenberg, S., Steinkamp, J., Hickler, T., Werner, C., Dormann, C. F., & Hartig, F. (2019). An R package facilitating sensitivity analysis, calibration and forward simulations with the LPJ-GUESS dynamic vegetation model. *Environmental Modelling & Software*, *111*, 55–60. https://doi.org/10.1016/j.envsoft.2018.09.004

Barker, M., Chue Hong, N. P., Katz, D. S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., & Honeyman, T. (2022). Introducing the FAIR principles for research software. *Scientific Data*, *9*(1), 622. https://doi.org/10.1038/s41597-022-01710-x

Bloh, W. von, Schaphoff, S., Müller, C., Rolinski, S., Waha, K., & Zaehle, S. (2018). Implementing the nitrogen cycle into the dynamic global vegetation, hydrology, and crop growth model LPJmL (version 5.0). *Geoscientific Model Development*, *11*(7), 2789–2812. https://doi.org/10.5194/gmd-11-2789-2018

Bondeau, A., Smith, P. C., Zaehle, S., Schaphoff, S., Lucht, W., Cramer, W., Gerten, D., Lotze-Campen, H., Müller, C., Reichstein, M., & Smith, B. (2007). Modelling the role of agriculture for the 20th century global terrestrial carbon balance. *Global Change Biology*, *13*(3), 679–706. https://doi.org/10.1111/j.1365-2486.2006.01305.x

Breier, J., Ostberg, S., Wirth, S. B., Minoli, S., Stenzel, F., & Müller, C. (2023). *lpjmlkit: Toolkit for basic LPJmL handling*. https://CRAN.R-project.org/package=lpjmlkit

Cramer, W., Bondeau, A., Woodward, F. I., Prentice, I. C., Betts, R. A., Brovkin, V., Cox, P. M., Fisher, V., Foley, J. A., Friend, A. D., Kucharik, C., Lomas, M. R., Ramankutty, N., Sitch, S., Smith, B., White, A., & Young-Molling, C. (2001). Global response of terrestrial ecosystem structure and function to CO2 and climate change: Results from six dynamic global vegetation models. *Global Change Biology*, *7*(4), 357–373. https://doi.org/10.1046/j.1365-2486.2001.00383.x

Dietrich, J. P., Bodirsky, B. L., Humpenöder, F., Weindl, I., Stevanović, M., Karstens, K., Kreidenweis, U., Wang, X., Mishra, A., Klein, D., Ambrósio, G., Araujo, E., Yalew, A. W., Baumstark, L., Wirth, S., Giannousakis, A., Beier, F., Chen, D. M.-C., Lotze-Campen, H., & Popp, A. (2019). MAgPIE 4 – a modular open-source framework for modeling global land systems. *Geoscientific Model Development*, *12*(4), 1299–1317. https://doi.org/10.5194/gmd-12-1299-2019

Gerten, D., Heck, V., Jägermeyr, J., Bodirsky, B. L., Fetzer, I., Jalava, M., Kummu, M., Lucht, W., Rockström, J., Schaphoff, S., & Schellnhuber, H. J. (2020). Feeding ten billion people is possible within four terrestrial planetary boundaries. *Nature Sustainability*, *3*(3), 200–208. https://doi.org/10.1038/s41893-019-0465-1

Gerten, D., Schaphoff, S., Haberlandt, U., Lucht, W., & Sitch, S. (2004). Terrestrial vegetation and water balance—hydrological evaluation of a dynamic global vegetation model. *Journal of Hydrology*, *286*(1), 249–270. https://doi.org/10.1016/j.jhydrol.2003.09.029

185 Herzfeld, T., Heinke, J., Rolinski, S., & Müller, C. (2021). Soil organic carbon dynamics from
186 agricultural management practices under climate change. *Earth System Dynamics*, *12*(4),
187 1037–1055. https://doi.org/10.5194/esd-12-1037-2021

188 Lutz, F., Herzfeld, T., Heinke, J., Rolinski, S., Schaphoff, S., Bloh, W. von, Stoorvogel, J. J.,
189 & Müller, C. (2019). Simulating the effect of tillage practices with the global ecosystem
190 model LPJmL (version 5.0-tillage). *Geoscientific Model Development*, *12*(6), 2419–2440.
191 https://doi.org/10.5194/gmd-12-2419-2019

192 Porwollik, V., Rolinski, S., Heinke, J., Bloh, W. von, Schaphoff, S., & Müller, C. (2022). The
193 role of cover crops for cropland soil carbon, nitrogen leaching, and agricultural yields – a
194 global simulation study with LPJmL (V. 5.0-tillage-cc). *Biogeosciences*, *19*(3), 957–977.
195 https://doi.org/10.5194/bg-19-957-2022

196 Rolinski, S., Müller, C., Heinke, J., Weindl, I., Biewald, A., Bodirsky, B. L., Bondeau, A.,
197 Boons-Prins, E. R., Bouwman, A. F., Leffelaar, P. A., Roller, J. A. te, Schaphoff, S., &
198 Thonicke, K. (2018). Modeling vegetation and carbon dynamics of managed grasslands
199 at the global scale with LPJmL 3.6. *Geoscientific Model Development*, *11*(1), 429–451.
200 https://doi.org/10.5194/gmd-11-429-2018

201 Schaphoff, S., Bloh, W. von, Rammig, A., Thonicke, K., Biemans, H., Forkel, M., Gerten, D.,
202 Heinke, J., Jägermeyr, J., Knauer, J., Langerwisch, F., Lucht, W., Müller, C., Rolinski,
203 S., & Waha, K. (2018). LPJmL4 – a dynamic global vegetation model with managed
204 land – part 1: Model description. *Geoscientific Model Development*, *11*(4), 1343–1375.
205 https://doi.org/10.5194/gmd-11-1343-2018

206 Sitch, S., Smith, B., Prentice, I. C., Arneth, A., Bondeau, A., Cramer, W., Kaplan, J. O.,
207 Levis, S., Lucht, W., Sykes, M. T., Thonicke, K., & Venevsky, S. (2003). Evaluation of
208 ecosystem dynamics, plant geography and terrestrial carbon cycling in the LPJ dynamic
209 global vegetation model. *Global Change Biology*, *9*(2), 161–185. https://doi.org/10.1046/
210 j.1365-2486.2003.00569.x

211 Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, *59*, 1–23. https://doi.org/10.
212 18637/jss.v059.i10

213 Wilson, G., Aruliah, D. A., Brown, C. T., Hong, N. P. C., Davis, M., Guy, R. T., Haddock,
214 S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., Waugh, B., White, E. P., &
215 Wilson, P. (2014). Best practices for scientific computing. *PLOS Biology*, *12*(1), e1001745.
216 https://doi.org/10.1371/journal.pbio.1001745