# Microxanox: an R package for simulating an aquatic *MICR*obial ecosystem that can occupy *OX*ic or *ANOX*ic states.

Rainer M Krug[a,1], Owen L. Petchey[a]

[a] *Department of Evolutionary Biology and Environmental Studies Winterthurerstrasse 190 8057 Zurich*

---

## Abstract

*Microxanox* is an R package to simulate a three functional group ecosystem (cyanobacteria, phototrophic sulfur bacteria, and sulfate-reducing bacteria) with four chemical substrates (phosphorus, oxygen, reduced sulfur, and oxidized sulfur) using a set of ordinary differential equations. Simulations can be run individually or over a parameter range. The model can be implemented with different numbers of species per functional group. The package is constructed in such a way that the results contain the input parameter used, so that a saved results can be loaded again and the simulation be repeated. Furthermore, the package framework and code should serve as a useful starting point for making simulation models of other types of ecosystems.

*Keywords:* reproducibility, regime shift, final state, ordinary differential equations

---

*Corresponding author

*Email addresses:* `Rainer.Krug@uzh.ch; Rainer@krugs.de` (Rainer M Krug), `Owen.Petchey@ieu.uzh.ch` (Owen L. Petchey)

[1]Corresponding Author

[2]Equal contribution

## 1. Required Metadata

### *1.1. Current code version*

Ancillary data table required for subversion of the codebase.

| Nr. | Code metadata description | Please fill in this column |
|---|---|---|
| C1 | Current code version | v0.9.1 |
| C2 | Permanent links to code/repository used for this code version | `https://github.com/UZH-PEG/microxanox` |
| | | 10.5281/zenodo.7148667 |
| C3 | Code Ocean compute capsule | |
| C4 | Legal Code License | MIT |
| C5 | Code versioning system used | git |
| C6 | Software code languages, tools, and services used | R |
| C7 | Compilation requirements, operating environments | R (>= 4.1.0) |
| | | magrittr |
| | | tibble |
| | | ggplot2 |
| | | patchwork |
| | | grDevices |
| | | stats |
| | | mgcv |
| | | deSolve |
| | | dplyr |
| | | tidyr |
| | | stringr |
| | | multidplyr |
| C8 | If available Link to developer documentation/manual | User Guide |
| C9 | Support email for questions | `Rainer.Krug@uzh.ch`; `Owen.Petchey@ieu.uzh` |

## 2. Motivation and significance

Mathematical models play a key role in the development of ecosystem models and in increasing the understanding of how and why ecosystems change when the environment changes [1, 3, 17]. They are also important for developing hypotheses to test in empirical studies. One area of ecology in which models have been influential is the understanding of ecosystem responses to gradual change in an environmental driver [12]. An environmental driver is an environmental condition that affects an ecosystem, but which is not itself affected by the ecosystem. An example driver would be the amount of nutrients entering a lake from run-off from surrounding agricultural land.

It is conceivable that an ecosystem state, such as the total biomass of a particular type of bacteria, may remain unchanged when an environmental driver changes. It is also possible that the ecosystem state changes gradually. It is also possible that the ecosystem state changes abruptly to a new state that is difficult to recover from [12]. This possibility for abrupt, perhaps catastrophic changes, which are difficult to reverse, causes considerable concern [4, 9, 16].

An example where a gradual change of an environmental variable causes an abrupt change in the system is the switch from an aerobic (oxygen is available for metabolism) to an anaerobic (oxygen generally unavailable) state in a microbial ecosystem. This system has been investigated by Bush et al. [2] in a simulation study of a mathematical model. Three types of microbes occur in the model: cyanobacteria (CB) dominating the oxic state, and two types of sulfur bacteria that dominate the anoxic state (sulfate reducing bacteria (SB) and phototrophic sulfur bacteria (PB)). The model shows that gradual change in the rate at which oxygen can diffuse into the ecosystem (termed oxygen diffusivity) could cause catastrophic changes in the ecosystem state that would be difficult to reverse.

One feature of the study by Bush et al. [2] was limited biodiversity. Specifically, there was no biodiversity within each of the three types (i.e., functional groups) of bacteria. This leaves the question of if and how biodiversity within these functional groups might affect the ecosystem response to environmental change. This limitation is not specific to the study of Bush et al. [2]. There are few if any studies of the effects of biodiversity on abrupt transitions between ecosystem states.

We decided to fill this research gap by making a simulation study of how within functional group biodiversity affects ecosystem responses to environmental change Limberger et al. [8], and to base our work on the work and model of Bush et al. [2]. With this goal in mind, we developed the `microxanox` package [7]. The first stage of development was to write code from scratch (as there was no available code to start from) and to confirm that this new implementation would reproduce the previously published results. The resulting reproduction is available as one of the package vignettes: vignette Partial reproduction of Bush et al.

The second stage was to add functionality that would be necessary to answer our research question. Most importantly, we made it possible to have multiple species of bacteria within each of the three functional groups, for the multiple species to differ in their characteristics, and to vary the number of species and amount of variability among them. We also added functionality that allowed: temporally varying environmental conditions, the addition of random noise to state variables, and immigration. In addition to the model itself, the package includes some functions to analyse the results and visualize these to provide a starting point for customized visualizations based on own requirements. The basic and additional functionality is described in the package User Guide.

## 3. Software description

The *microxanox* package is for simulating a three functional group system (*CB*: cyanobacteria, *PB*: phototrophic sulfur bacteria, and *SB*: sulfate-reducing bacteria) with four chemical substrates (*P*: phosphorus, *O*: oxygen, *SR*: reduced sulfur, *SO*: oxidized sulfur). It includes feedback between organisms and biogeochemical processes and is based on Bush et al. [2] (See Bush et al. [2] for a detailed discussion of the model). At the core of the simulations is a set of ordinary differential equations (specified in the function `bushplus_dynamic_model()`, though this function need not be directly called). There are functions for running individual simulations and a set of simulations across, for example, a range of environmental conditions.

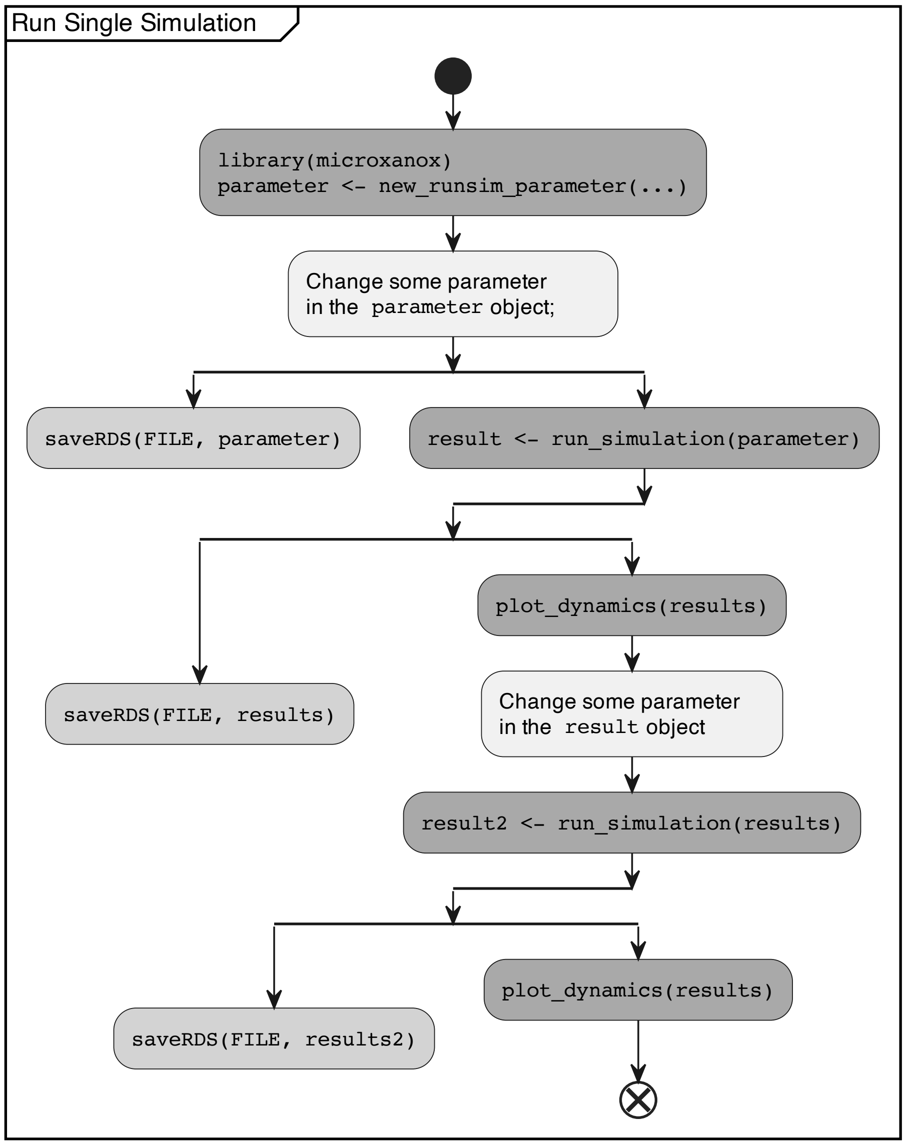
To make the simulation run with multiple species per functional group, we expressed different species' characteristics in the elements of vectors and matrices. We also coded the ordinary differential equations in matrix form to use matrix mathematics. In this way, we made it possible to run simulations with different numbers of species without having to change the underlying code.

The package functions and code have a modular structure so that new functionality can be easily added. E.g., temporally defined events of any type could be specified. Further, all parameter values required to run a simulation are stored in one object. Lastly, the general structure of the code should make it straightforward to adapt the model to other similar systems (described in more detail in the Impact section).

### *3.1. Software architecture*

The framework used when writing this package aimed to maximise simplicity for the user and to make it straightforward to reproduce results (see the supplement [10] to Limberger et al. [8] for an example of how this is used). As such, all the parameters needed to run a simulation or a set of simulations are contained in a single object (which can easily be created using included functions). This parameter object is given to a function that runs the simulations and returns the results. The returned results object is identical to the parameter object but with an additional slot named `results`, which contains the simulation results. Thus the returned results object contains the simulation conditions (parameters) as well as the results, and can be used to run the simulation again. This promotes reproducibility and makes incremental changes of individual parameters with a consecutive re-running of the simulations straightforward.

In the following sections we describe how to use the package to run one simulation and to find final states across an environmental gradient.



Run Single Simulation
library(microxanox)
parameter <- new_runsim_parameter(...)
Change some parameter in the parameter object;
saveRDS(FILE, parameter)
result <- run_simulation(parameter)
plot_dynamics(results)
saveRDS(FILE, results)
Change some parameter in the result object
result2 <- run_simulation(results)
plot_dynamics(results)
saveRDS(FILE, results2)

Figure 1: Typical flow of a simulation. Dark Grey boxes: commands necessary for simulation; Light Grey:Saving of parameter and results; Lightest Grey: Different non specified commands.

### 3.2. *Running one simulation*

A typical simulation would look as shown in Figure 1.

A simulation is run using the `run_simulation()` function. In this function, the ODEs are solved using the function `ode()` in the package *deSolve* [14]. The `run_simulation()` function needs only one argument - an object as created by the function `new_runsim_parameter()`. The parameter object returned by `new_runsim_parameter()` contains, among other things, the `strain_parameter` object, which can be created by the function `new_strain_parameter()`. For a detailed description of the parameter objects, their meaning and how they are created and have values set and changed, please see the *User Guide* which accompanies the package or is available at User Guide.

After the parameter object has been defined, it can be used in the `run_simulation()` function. The function returns an object identical to the parameter object, except of an additional slot containing the results. This design produces a fully reproducible object as it can be used as a parameter object to be fed back into the `run_simulation()` function to rerun the simulation again from the parameter used to generate the results before.

### 3.3. *Examining ecosystem responses to environmental change*

One approach to finding how the ecosystem responds to environmental driver change is to examine the relationship between the environmental driver value and the numerical value of a system's state. The first component of this approach is to run the simulation for a long time and record the final state (i.e., the state of the system at the end of a simulation). When one does this across a range of environmental conditions, one discovers how the final state of the system changes with the environmental conditions. The package contains functionality for finding final states that correspond to values of one environmental driver, namely the value of oxygen diffusivity. In terms of non-linear systems analysis, this would be termed a *parametric sensitivity analysis* being conducted by running an open loop of the dynamic system under a set of initial conditions and a subset of parameter values (here oxygen diffusivity).

When one wishes to be able to make conclusions about how the *steady state* (or an equilibrium point) of the system is affected by the environmental driver, it is very important to note that the *final state* (provided by the simulation) is not guaranteed to be a *steady state*. The software does not provide the user with a steady state. In order to somewhat safely assume that the final state is a steady state, the user must ensure that the simulation is run for sufficiently long time for any transient dynamics to disappear, and must also check the type of long-term dynamics occurring. In the results presented here, and in the paper Limberger et al. [8], this was performed by visual inspection, and by checking the sensitivity of conclusions to the length of the simulation. Furthermore, the package does not include methods for a formal analysis of the stability of the system and users should take care to assess if steady states are unique.

Two numerical strategies for finding final states and their sensitivity to parameters are implemented. The first method runs a independent simulation for each combination of initial conditions and oxygen diffusivity (we term this the *Replication method*). This is the method used in the Bush et al. [2] study and was used to obtain the results in figures 3 and 4 of that article.

The second method runs two simulations, one with step-wise and slowly temporally *increasing* oxygen diffusivity, and the other with step-wise and slowly *decreasing* oxygen diffusivity. (Put another way, to explore the sensitivity of final states of the dynamic model under oxygen diffusivity variation, subset values of this parameter are numerically evaluated. Two value vectors of oxygen diffusivity are presented step-wise, gradually increasing and decreasing.) During this temporal environmental change, the state of the system is recorded just before change to a new oxygen diffusivity (we term this the *Temporal method*).

A potentially important difference between the two methods is in the system state when a new value of oxygen diffusivity is set. In the replication method, the system state when a new value of oxygen diffusivity is set is always the same. Whereas in the temporal method, the system state when a new value of oxygen diffusivity is set is the final state of the system for the previously set value of oxygen diffusivity. Since some modellers prefer one approach and others another, we decided to implement both.

The replication method is implemented in the function `run_replication_ssfind()` which takes a parameter object as returned by the function `new_replication_ssfind_parameter()` and the number of cores for multithreading the simulation. As the multithreading uses the package function `mclapply()` from the package `parallel` [11], the multithreading only works on Linux and Mac. It is planned to move to `parLapply()` [11] in a future release.

The temporal method is implemented in the function `run_temporal_ssfind()`, which takes a parameter object as created by the function `new_temporal_ssfind_parameter()`. It is planned for a later release, to run these two simulations in parallel.

For a more detailed walk-through of these two approaches and explanation please see the User Guide.

### *3.4. Analysing and visualising results*

From the results returned, summary measures about how the ecosystem final states changes with environmental change can be extracted. The function `get_stability_measures()` returns quantities such as the amount of environmental change required for the system to abruptly change to a different state.

The function `plot_dynamics()` plots a single simulation run, as returned from the `run_simulation()` function. This function is only provided as a convenience function to provide a way to easily see the results of a simulation run. An example plot resulting from this function is shown in Figure 2.

## **4. Impact**

The open source implementation and extension of the model used in Bush et al. [2] provides the means of reproducing the results published while at the same time provides the means of making unique, innovative, and important

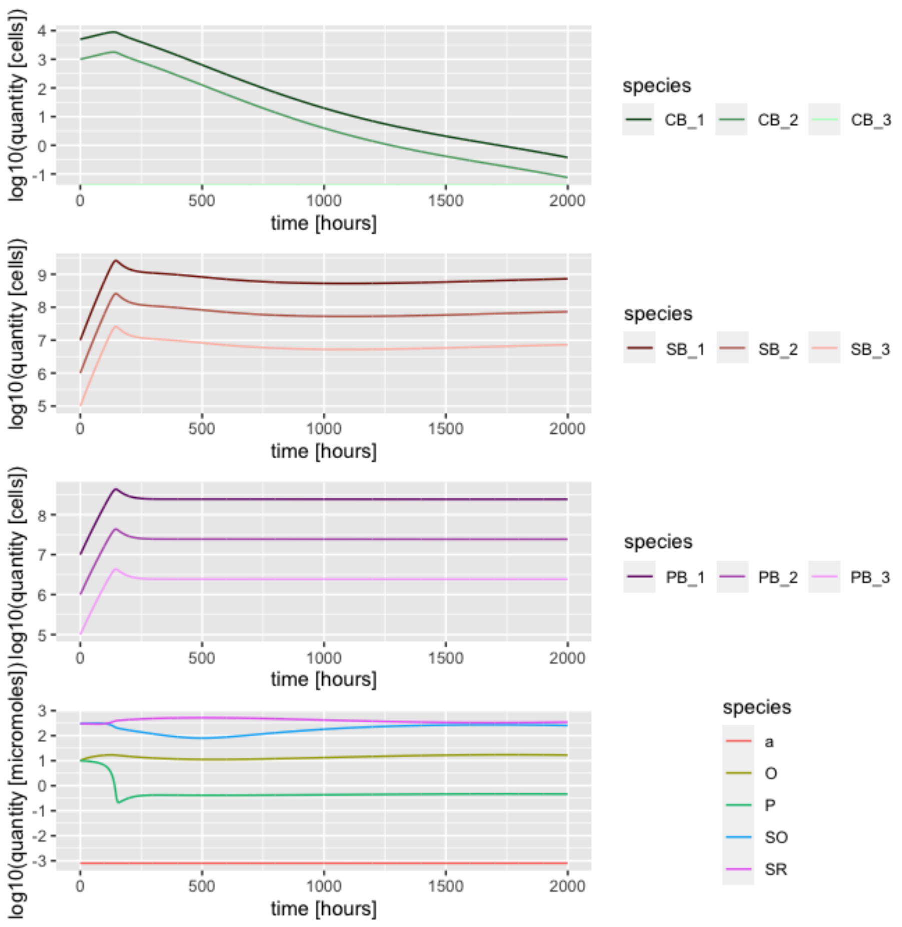


Figure 2: Results of a simulation run shown using the function *plot_dynamics*(). In this case, there were three strains per functional group, though strains within functional groups had identical properties in this example. CB_1 = cyanobacteria strain 1; SB_1 = sulfur reducing bacteria stain 1; PB_1 = phototrophic sulfur bacteria strain 1.

investigations of how ecosystems change as the environment changes, and how biodiversity may modulate this.

The design of the package code and functionality is with reproducibility in mind (for other example of reproducibility directed software, see e.g. Correndo et al. [5], Spillner et al. [15] or Fox et al. [6]): the combination of all parameters being in a single parameter object and the return of the simulation as a result object which inherits the parameter values from the parameter object, provides a relatively easy-to-use framework to implement reproducible experiments.

Here we evidence the impact of the *microxanox* package by describing three use-cases and then by describing how the package can be a starting point for models of other ecosystems. The first two use cases are described in detail (including the code for reproducing them) in the User Guide and the *Partial Reproduction* vignettes. The third is taken from Limberger et al. [8] and Petchey et al. [10].

### *4.1. Use case 1: Regime shifts during temporal environmental change*

The study of Bush et al. [2] includes simulations of the effect of oxygen diffusivity (an environmental driver, in the sense that it affects the ecosystem but is not affected by it) on the ecosystem state (oxic or anoxic). The `microxonox` package contains functionality to make a specific temporal pattern of change in the oxygen diffusivity. As well as allowing individual simulations during which oxygen diffusivity varies, this functionality forms the basis of the temporal method for finding final states.

An example of this functionality is given in the *Partial Reproduction* vignette, which we briefly describe and show here (Figure 3). The example is composed of a single simulation, at the beginning of which the system is in the oxic state with high abundance of cyanobacteria. Oyxgen diffusivity is then slowly decreased and eventually, around hour 30'000 the system switches to the anoxic state, with high abundance of both sulfur bacteria types. The oxygen diffusivity is then increased and at around hour 38'000 the system abruptly switches back to the oxic state.

Also visible in the results are thick lines showing abundances of bacteria when abundances are low. This is due to the implementation of a function that at regular intervals, increases the abundance to a preset level. This prevents abundances reducing to very small numbers. The function that implements this increase abundance can also be made to add a certain abundance to each strain at regular intervals, thus simulating immigration in to the system.

### *4.2. Use case 2: The extent of hysteresis depends on community composition*

The package contains a function to extract summary features of ecosystem responses to environmental change, such as the amount of hysteresis displayed by the ecosystem. Hysteresis is a key feature of ecosystem responses to environmental change, because it is related to how difficult it can be to reserve the effects of environmental change [12]. The amount of hysteresis is measured as the extent of the environmental condition (here oxygen diffusivity) for which there were

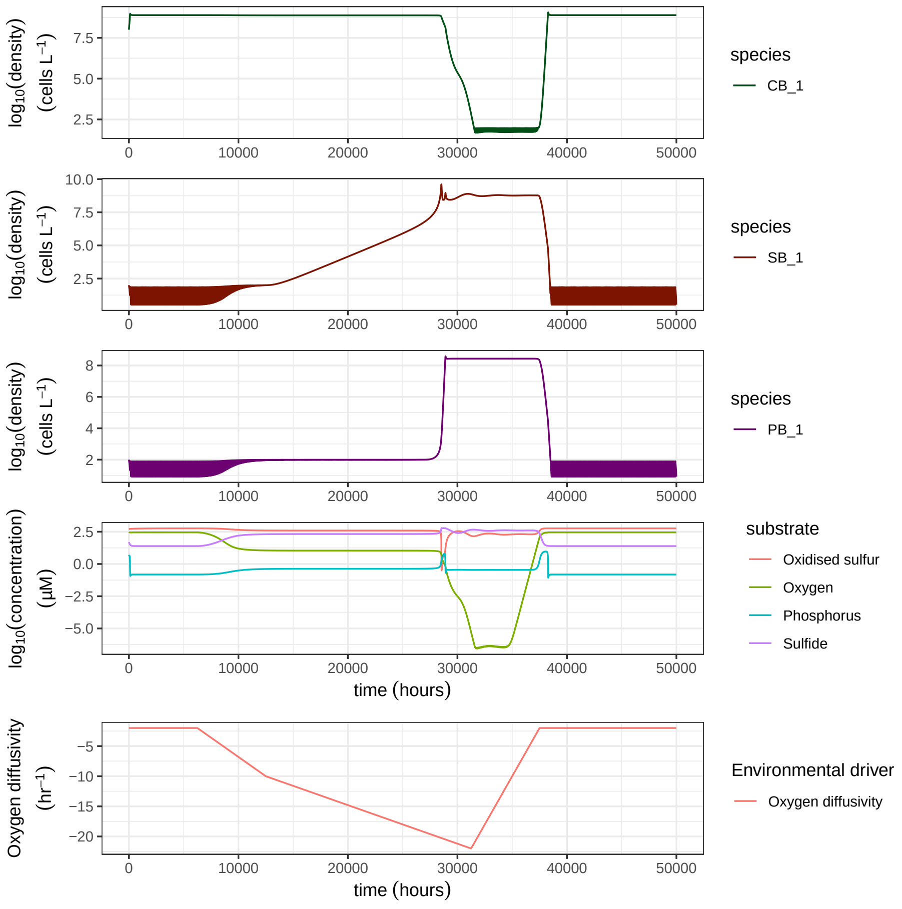


Figure 3: The temporal dynamics of the ecosystem model when an environmental condition (here parameter $a$, the oxygen diffusivity) changes. Plot of the final states of the simulation runs under different oxygen diffusivity. In this simulation there is only one strain in each functional group. CB extunderscore1 = cyanobacteria strain 1; SB extunderscore1 = sulfur reducing bacteria stain 1; PB = phototrophic sulfur bacteria strain 1. Here we show a figure adapted from the output of the $plot\_dynamics()$ function.

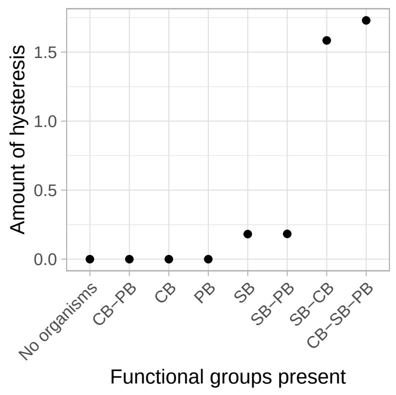


Figure 4: The amount of hysteresis depends on the combination of types of organisms present. The model is entirely deterministic, hence there are no error bars.

two distinct final states. I.e., it is the extent of the environmental conditions for which historical conditions play an important role in determining the current system state.

Using the package to calculate the extent of hysteresis involves setting ecosystem and simulation parameters, including parameters for the finding of final states across an environmental gradient, running the final state finding function, and analysing the results with the function that calculates extent of hysteresis. The code for this is provided in the User Guide.

The results show that the amount of hysteresis depends greatly on the combinations of organisms present (Figure 4). For example, with only the CB (cyanobacteria) present, there was no hysteresis. In contrast, the presence of both CB and SB (sulfate reducing bacteria) led to a large amount of hysteresis. (These results are also given in the *Partial Reproduction* vignette.)

### *4.3. Use case 3: Effects of functional diversity on regime shifts*

As discussed in the Introduction section, the package was motivated by the question of how biodiversity influences ecosystem responses to environmental change. Extensive results concerning this question are given in a separate publication Limberger et al. [8]. Here we describe one of the results, which is that having biodiversity in a functional group can allow state changes to occur that otherwise would not have, i.e. biodiversity can qualitatively change the state of the ecosystem.

Biodiversity was added to the functional groups using the `new_strain_parameter()` function to create a parameter set with multiple species per functional group (albeit all with identical features) and then to add variability among the species by calling the `add_strain_var()` function. This function takes an already existing parameter set and adds the specified about of variation. The new parameter object is then used as before.

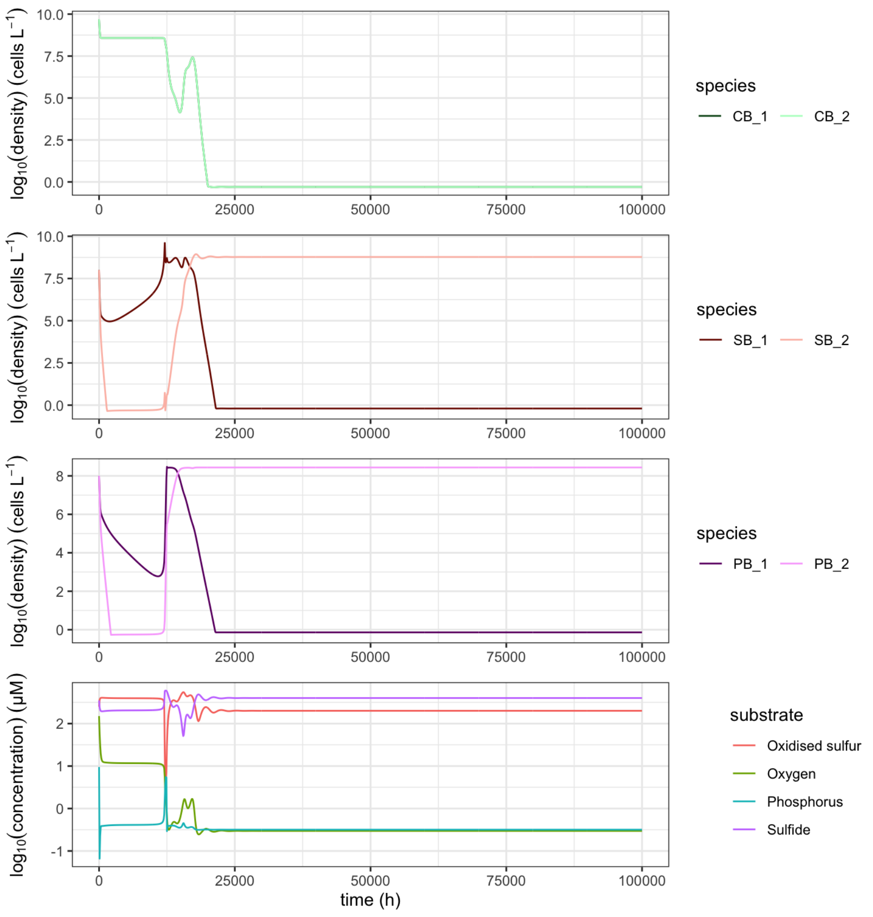


Figure 5: The dynamics of the ecosystem when there are two species in each functional group, and some variation (diversity) in species parameters.

Figure 5 shows a simulation with two species (strains) in each of the three functional groups. The ecosystem starts in the oxic state, though with relatively high abundance of each functional group. The strain of SB that is more tolerant to oxygen (SB_1) initially decreases in abundance, but then increases, and the other (SB_2) strain then becomes abundance and SB_1 declines. Furthermore, the cyanobacteria crash in abundance, and the system switches to the anoxic state. In contrast, if there are two identical strains with tolerance half way between those in Figure 5 the ecosystem remains in the oxic state.

## *4.4. Adapting for other ecosystems and organisms*

We anticipate that the package can be a useful starting point for investigating models of other types of ecosystem and how biodiversity in them affects responses to environmental change. The overall framework of the package, the purpose of each function, and the objects used for storing parameters and results could be retained. E.g., all such models would have parameters that differ among species and need to be described in an object, and studies will often need to run simulations and sets of simulations across environmental conditions.

Researchers wanting to model a new ecosystem do not, therefore, have to start from scratch. This will relieve researchers from needing to make software design decisions, and rather focus on appropriately representing their ecosystem and finding the results that interest them. Nevertheless, adaptation of the code in the package will require a person / persons that can take a conceptual model of an ecosystem and then represent that in terms of parameters and rate equations, and a person or persons relatively proficient in R programming.

## 5. Conclusions

The *microxanox* R package allows the simulation, visualisation, and analysis of a model of a microbial ecosystem while allowing variation in the amount of diversity present in each of the functional groups of organisms present. It has been used for the research described in another paper that provides one of the first investigations of the effects of diversity on ecosystem resilience Limberger et al. [8]. In that paper, we show that diversity can have large and important effects on ecosystem responses, highlighting the need for models such as ours, with which one can easily manipulate the amount of biodiversity. The *microxanox* package has also been used to reproduce the results of the paper that inspired the package development [2].

The package greatly lowers the amount of work required in further investigations of the specific ecosystem modelled. There has, for example, been quite limited investigation of how biodiversity influences the short-term responses of the modelled ecosystem to environmental change. Likewise, the package could be used to power an investigation of the effects of biodiversity on the usefulness of early warning signals of abrupt ecosystem change [13]. In addition this package could be used as a template for developing models of other types of ecosystems and organism. By doing so, other models can profit from the overall framework used, and the reproducibility aspects as well as the flexibility implemented.

## 6. Conflict of Interest

The authors declare no known conflicting or competing interests associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

## 7. Acknowledgements

This project was part of SNF Project 310030_188431. The project was also supported by the University of Zurich Research Priority Programme in Global Change and Biodiversity.

## References

[1] Amrei Binzer, Christian Guill, Björn C. Rall, and Ulrich Brose. Interactive effects of warming, eutrophication and size structure: Impacts on biodiversity and food-web structure. *Global Change Biology*, 22(1):220–227, January 2016. ISSN 13541013. doi: 10.1111/gcb.13086.

[2] Timothy Bush, Muhe Diao, Rosalind J. Allen, Ruben Sinnige, Gerard Muyzer, and Jef Huisman. Oxic-anoxic regime shifts mediated by feedbacks between biogeochemical processes and microbial community dynamics. *Nature Communications*, 8(1):789, October 2017. ISSN 2041-1723. doi: 10.1038/s41467-017-00912-x.

[3] P. Catalina Chaparro-Pedraza. Fast environmental change and eco-evolutionary feedbacks can drive regime shifts in ecosystems before tipping points are crossed. *Proceedings of the Royal Society B: Biological Sciences*, 288(1955):20211192, July 2021. ISSN 0962-8452, 1471-2954. doi: 10.1098/rspb.2021.1192.

[4] Scott L. Collins, Jesse B. Nippert, John M. Blair, John M. Briggs, Pamela Blackmore, and Zak Ratajczak. Fire frequency, state change and hysteresis in tallgrass prairie. *Ecology Letters*, 24(4):636–647, April 2021. ISSN 1461-023X, 1461-0248. doi: 10.1111/ele.13676.

[5] Adrian A. Correndo, Austin Pearce, Carl H. Bolster, John T. Spargo, Deanna Osmond, and Ignacio A. Ciampitti. The soiltestcorr R package: An accessible framework for reproducible correlation analysis of crop yield and soil test data. *SoftwareX*, 21:101275, February 2023. ISSN 23527110. doi: 10.1016/j.softx.2022.101275.

[6] Nathan Fox, Tom August, Francesca Mancini, Katherine E. Parks, Felix Eigenbrod, James M. Bullock, Louis Sutter, and Laura J. Graham. “photosearcher” package in R: An accessible and reproducible method for harvesting large datasets from Flickr. *SoftwareX*, 12:100624, July 2020. ISSN 23527110. doi: 10.1016/j.softx.2020.100624.

[7] Rainer M Krug and Owen L Petchey. Microxanox: Model to simulate three functional group system. June 2022. doi: 10.5281/ZENODO.6624125.

[8] Romana Limberger, Uriah Daugaard, Anubhav Gupta, Rainer M. Krug, Kimberley D. Lemmen, Sofia J. Van Moorsel, Marcel Suleiman, Debra Zuppinger-Dingley, and Owen L. Petchey. Functional diversity can facilitate the collapse of an undesirable ecosystem state. *Ecology Letters*, 26(6): 883–895, June 2023. ISSN 1461-023X, 1461-0248. doi: 10.1111/ele.14217.

[9] Amanda C. Northrop, Vanessa Avalone, Aaron M. Ellison, Bryan A. Ballif, and Nicholas J. Gotelli. Clockwise and counterclockwise hysteresis characterize state changes in the same aquatic ecosystem. *Ecology Letters*, 24(1): 94–101, January 2021. ISSN 1461-023X, 1461-0248. doi: 10.1111/ele.13625.

[10] Owen L Petchey, Uriah Daugaard, Anubhav Gupta, Rainer M Krug, Kimberley D. Lemmen, van Moorsel, Sofia J., Marcel Suleiman, Zuppinger-Dingley, Debra, and Romana Limberger. Code package for: "Functional diversity can facilitate the collapse of an undesirable ecosystem state". Zenodo, June 2022.

[11] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022.

[12] Marten Scheffer, Steve Carpenter, Jonathan A. Foley, Carl Folke, and Brian Walker. Catastrophic shifts in ecosystems. *Nature*, 413(6856):591–596, October 2001. ISSN 0028-0836, 1476-4687. doi: 10.1038/35098000.

[13] Marten Scheffer, Jordi Bascompte, William A. Brock, Victor Brovkin, Stephen R. Carpenter, Vasilis Dakos, Hermann Held, Egbert H. van Nes, Max Rietkerk, and George Sugihara. Early-warning signals for critical transitions. *Nature*, 461(7260):53–59, September 2009. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature08227.

[14] Karline Soetaert, Thomas Petzoldt, and R. Woodrow Setzer. Solving differential equations in R: Package deSolve. *Journal of Statistical Software*, 33(9):1–25, 2010. doi: 10.18637/jss.v033.i09.

[15] Josef Spillner, Panagiotis Gkikopoulos, Pamela Delgado, and Christine Choirat. Towards reproducible software studies with MAO and Renku. *SoftwareX*, 17:100947, January 2022. ISSN 23527110. doi: 10.1016/j.softx.2021.100947.

[16] John Vandermeer and Ivette Perfecto. Hysteresis and critical transitions in a coffee agroecosystem. *Proceedings of the National Academy of Sciences*, 116(30):15074–15079, July 2019. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1902773116.

[17] David A. Vasseur and Kevin S. McCann. A Mechanistic Approach for Modeling Temperature-Dependent Consumer-Resource Dynamics. *The American Naturalist*, 166(2):184–198, August 2005. ISSN 0003-0147, 1537-5323. doi: 10.1086/431285.