MIRACLE: A prototype cloud-based reproducible data analysis and visualization platform for outputs of agent-based models

Xiongbing Jin[1], Kirsten Robinson[1], Allen Lee[2], Gary Polhill[3], Calvin Pritchard[2], and Dawn Parker[1]

[1] University of Waterloo x37jin,k4robins,dcparker@uwaterloo.ca
[2] Arizona State University allen.lee@asu.edu,pritchard.calvin@gmail.com
[3] The James Hutton Institute gary.polhill@hutton.ac.uk

Abstract Since the agent-based models we design have stochastic elements and many potential parameter combinations, multiple model runs that sweep parameters are conducted, creating large quantities of computationally generated, hyper-dimensional, "big data". Understanding the model's implications required structured exploration of these complex output data. In response to this need, the MIRACLE team has developed a cloud-based community platform that presents prototype examples of output and analysis methods for agent-based models of coupled human-natural systems (ABM-CHANS). We anticipate that the platform will facilitate improved communication within research groups, as well as increasing access and transparency for external research communities. This paper provides contextual background and a tutorial introduction to the MIRACLE data storage and analysis web tool.

1 Introduction

Pioneering work has demonstrated the scientific utility of agent-based models of coupled human-natural systems (ABM-CHANS) [36,41,42]; however, development of ABM-CHANS has posed many challenges [46]. In the last ten years, model communication and transparency have been dramatically improved in the agent based modelling community, based on improved standards for documenting, validating, verifying, and archiving models [37,40,45,51].

Yet, even with these advances, substantial challenges for the scientific replicability of ABM-CHANS remain. Replicability requires complete access to model assumptions, code, output data, and analysis methods. Platforms such as the ComSES OpenABM archive [1] provide model code archive facilities, which are increasingly being used by individuals and academic journals. However, in spite of increasing access to model code, most models are run only by the team that developed them. A preliminary analysis conducted by CoMSES Net of 199 publications has found that only 31 publications that reference computational models are properly archiving their models [39]. This means opportunities to build on existing models in order to advance research and decision making are lost. While

journal requirements for code archiving can increase the availability of code, we hypothesize that one factor leading to a lack of use of existing model code is the relative complexity and technical challenge of adapting another user's code, especially for modellers in the social and life sciences who traditionally lack access to computer science training [32].

Of equal importance for ABM-CHANS replicability is the availability of model output data and the analyses and computational workflows that transformed that data into statistically significant results. For scientific experimentation, ABM-CHANS are often run using multiple stochastic seeds and parameter sweeps to generate large output databases of virtual "big data". Given the high degree of agent decision-making and characteristic heterogeneity and complex spatial and network interactions, ABM-CHANS are capable of producing enormous data outputs. However, as the production of such output data is often time and resource-intensive, without good tools, researchers may be tempted to limit the size of their model runs or to explore only small portions of the output space, reducing the quality of their results. For other research groups, output replication is further challenged by the fact that the model's software environment can be highly specialized, and not necessarily portable [47]. Even if other researchers get the models running, replicating output data is not always feasible without the provision of all setup files that the original researcher used to generate the output on which they based a scientific article. Replicating output data may also be infeasible for pragmatic reasons: lack of access to CPU time and/or disk space. The result is that other researchers cannot test and explore the models' conclusions.

Even with good access to model output data, challenges for analysis and communication remain. Generic data analysis techniques are rarely suitable for agent based model output; rather, specialized analysis methods are needed [34]. While ABM-CHANS modellers are developing such specialized approaches [43], most are not yet part of standard statistical packages. Without appropriate protocols for even within-group sharing and archiving analysis algorithms, the opportunity for complete replicability may disappear with the departure of a research group member or a computer crash.

Increasing the utility of ABM-CHANS models, and ultimately their influence, therefore, depends upon solving the problem of effectively sharing model output data and the data analysis workflows used to transform that data into a research finding of interest. Finding solutions to store and make directly available model output data is central to improving the communication and transparency of model results and thus the impact of agent-based models.

There is currently no standard for agent based modellers to store and make available their model workflows, output data, and analysis algorithms. The MIRACLE project, funded through the international Digging into Data initiative, is an attempt to build a consistent, usable tool that is freely accessible from any computer, uses modern web standards, and makes the results of large sets of model runs immediately accessible to any interested user. It removes the need to download code, run a model, or replicate data output. Furthermore

the tool provides built-in statistical analysis techniques so other researchers can easily explore and ask new questions of the data. It thus has the potential to dramatically reduce the barriers to exploring the results from ABM-CHANS research.

This paper is a guide to using the MIRACLE tool online and also provides instructions for setting up the MIRACLE software stack on your own server. It begins with a review of existing alternative platforms and tools for managing model output data and analysis scripts, and an introduction to the MIRACLE system. We will then demonstrate a typical workflow of using MIRACLE, from uploading model information, output data, and analysis scripts, to annotating these datasets and scripts with contextual metadata, running analysis scripts with different parameterization, and sharing particular results with selected other users. Finally we comment on the current state of the prototype and discuss next steps.

2 Literature review

A number of code and data archival initiatives have sprung up recently alongside calls for greater transparency and reproducibility in the computational sciences [33,54,53]. Although these initiatives take a useful first step towards preserving the software artifacts that a given research finding depends on, they fail to comprehensively address specific challenges outlined above facing researchers that use ABMs or other complex simulation tools. MIRACLE will differentiate itself by addressing these needs while integrating with existing tools. Nothing in this space exists.

2.1 Similar data and analysis archiving platforms

A number of attempts have been made to create software platforms for preserving research data and software. There are general-purpose platforms such as Dataverse [2], CKAN [3], myExperiment [4], figshare [5], and the Open Science Framework [6], and discipline-specific platforms such as GenAP [7], CyVerse [8], and SEAD [9]. Dataverse, developed by Harvard University, is one of the oldest and most widely used generic data archive platforms today. It supports data and file management, metadata entry, basic statistical analysis and visualization tools, and an API for extension. CKAN is similar to Dataverse but adds further functionalities in geospatial data visualization and community functions (commenting on and following datasets). The myExperiment framework focuses on running and sharing workflows. The Open Science Framework supports end-to-end archival of the entire research process from inception, providing collaborative tools like wikis to be used during a project's active lifecycle, versioning for documents and other files, and integration with external data services and repositories like Dataverse and GitHub.

Data and analysis scripts in a specific research field often share common characteristics unique to that discipline that are not easily captured by general-

purpose platforms. While it is still possible to preserve such data on general-purpose platforms, it can be difficult to navigate and perform meaningful analyses on them. In light of these issues, there have been several attempts to build discipline-specific data archive and reproducibility platforms. For example, GenAP is a computing platform designed for genetics and genomics researchers to share research data and analysis pipelines. Building on top of the Compute Canada HPC infrastructure, GenAP allows research groups and users to instantiate their own analysis virtual machine and submit analysis jobs via a web interface. CyVerse aims to build a large scale cyberinfrastructure for research in the life sciences. The core components of CyVerse are Atmosphere (a cloud-computing platform that allows user to create and launch own virtual machines pre-configured with CyVerse-provided software, datasets and workflows) and a Discovery Environment that provides web access to existing bioinformatics software and analysis results. The MIRACLE project shares similar goals with these tools, but for the study of ABM-CHANS output data.

2.2 A platform designed for ABM research

With the unique characteristic of ABM research in mind, MIRACLE builds off existing work by CoMSES on building community tools for computational modelling [40,51], and by the University of Waterloo on building ABM input and output data exploration tools [50,55]. Based on the proposed metadata standard for ABM output data [49], MIRACLE automatically groups output data based on the structure of the parameter sweep data, and creates analysis run interfaces that allow users to easily explore data in the parameter space. By recording each analysis run on the platform, MIRACLE also captures the provenance of the research workflow.

MIRACLE infrastructure is based on microservices [35] implemented in lightweight Docker [10] containers, with each component running in its own isolated and secure container environment. This architecture allows MIRACLE to fully preserve the computing environment of each individual project and mitigates software conflicts and server security concerns. In a minimized setting, MIRACLE employs five containers:

- A MIRACLE app container developed using the Django Rest Framework [11,12] with a front-end built on React.js [13] and Bootstrap [14]. This front end provides web based workflows for metadata management for users and their projects, datasets, and analysis scripts as well as interfaces for uploading project archives and running analysis scripts on uploaded data with custom parameters
- A PostgreSQL [15] database container used by the Django app
- An R script execution container based on Microsoft's DeployR Open software [16] to securely run R scripts in a sandboxed environment. MIRACLE only supports R scripts as of this writing, as R is the most popular open source data analysis software in scientific research [44], but we plan to add support for Python and more languages in the future

– A Shiny apps container based on RStudio's Shiny Server open source edition [17], which runs a customized version of the open source Radiant analytics software [18] and any user-provided Shiny and RMarkdown apps
– A web proxy container that uses NGINX [19] to deliver requests securely to the MIRACLE Django app, DeployR, and the Shiny apps containers

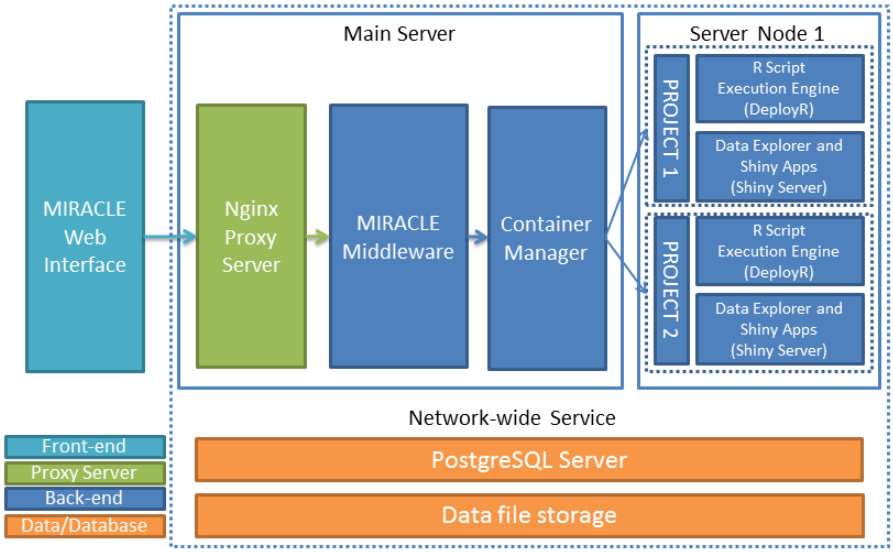

Figure 1. Architecture plan of MIRACLE

MIRACLE is designed to be elastic based on demand. The system can be configured to automatically create and spin up new containers for new projects and users, while the containers share system resources for maximum hardware utilization[4]. A detailed design of MIRACLE will be presented in an upcoming paper.

3 A user guide to the system

This section presents a step-by-step user guide for the MIRACLE platform.

3.1 Creating account and project

MIRACLE supports local and external sign-ins. You can choose to create a dedicated account, or use an existing account from an external website like GitHub, Twitter, Facebook and Google.

[4] This feature is currently under development.

After you login, you will be presented with the `Dashboard`, which lists your projects and their recent activities. You can create a new project by clicking the `NEW` button at the upper right corner, and providing a project name, a unique short name, a description of your project and a list of project members.

`Recent Activity` lists recent activities by your project members, including project creation, metadata editing, recent script runs, comments inside your project, and sharing of results.

3.2 Uploading your archive

You can upload your archive to MIRACLE using the drag-and-drop interface in the `Upload archive` tab of your project. See Sect. 3.7 for a detailed guide on how to prepare your data and scripts for use on MIRACLE.

After uploading your archive, MIRACLE will process your archive by:

- Extracting available metadata in your data files and scripts
- Categorizing your data files into data groups. Each data group contains a set of files with identical column structure. This facilitates metadata entry and management
- Making your data files and scripts accessible to the Radiant-based Data Explorer and the R script execution environment.

3.3 Metadata management

You can manage metadata for your data files using the `Data groups` tab. MIRACLE automatically extracts column names and provides best-guesses for column types to reduce your workload.

3.4 Data explorer

MIRACLE uses a modified version of the R and Shiny-based Radiant business analytics tool [18] to enable exploration of uploaded data files. Users can run exploratory data analysis including data manipulation, visualization and common statistical analyses using an intuitive user interface. Documentation and tutorial of Radiant can be found on its website [18]. Figure 2 shows an example of using the Data Explorer to explore the output data of the LUXE agent-based land market model [38] by generating a scatterplot of the relationship between open space amenity and transaction price of land.

3.5 Run scripts

All R scripts that conform to the scripts preparation guidelines (Sect. 3.7.2) will be listed in the `Analysis and scripts` tab. MIRACLE automatically manages all requirements for a script to run, including input data, parameter ranges and package dependencies. When you click `Run script`, a parameter window will pop-up, asking you to choose or enter input parameters as defined in Sect. 3.7.2. After

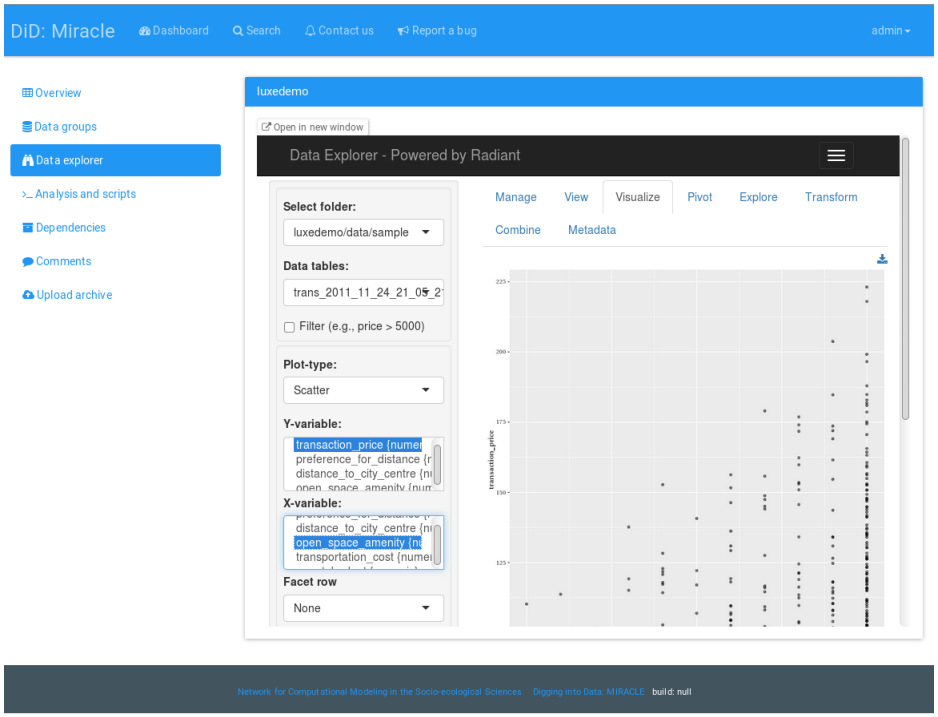

Figure 2. The data explorer based on Radiant

confirming your input parameters, an asynchronous backend task is scheduled to run the script in a sandboxed environment, and the MIRACLE web application will notify you when the task is completed and any generated results are ready to be viewed. Figure 3 shows an example of the input parameter window when running an analysis script from the LUXE model [38]. The parameter names and values are automatically extracted from the analysis script code (see Sect. 3.7.2).

All previous runs of the scripts in this project will also be listed on this tab in the `Outputs` section. You can view the details of any runs including input parameters and output files, and view the scripts themselves with syntax highlighting directly on the website. You can also download any output files and scripts to your own computer. Figure 4 shows the output from a previous run of the visualization code from the LUXE model [38].

3.6 Comment and sharing

An important feature of MIRACLE is commenting and sharing, which enables easy communication and collaboration within and among research groups. For example, if you run the visualization script from the LUXE model with a specific parameter combination and notice an interesting land use pattern (Fig. 4), you

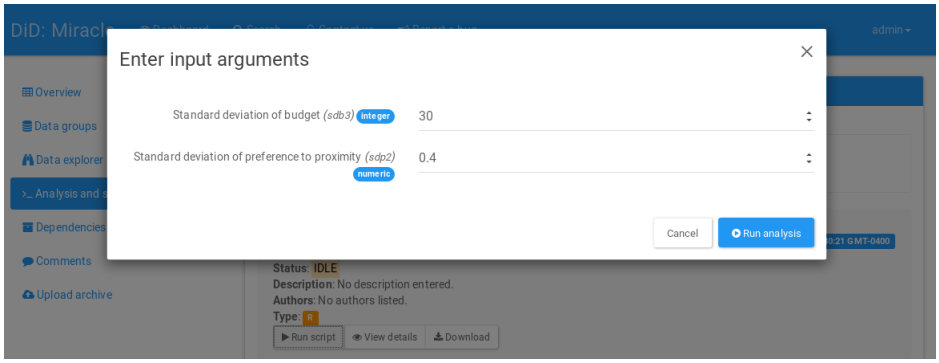

Figure 3. Run analysis script with custom input parameters

can easily share the result using the `Share` button. You can also leave comments within each project.

3.7 Project archive preparation

To use MIRACLE to its full potential, you need to arrange your project data files and scripts according to a predefined filesystem folder structure. This is necessary for MIRACLE to correctly identify and store each type of data, and to make scripts/apps runnable on the website without manual intervention.

As of this writing, the required folder structure for MIRACLE is[5]:

```
MIRACLE_ROOT
├── SHORT_NAME
│   ├── apps
│   │   ├── shiny_app
│   │   │   └── ui.R
│   │   └── rmd
│   │       └── file.rmd
│   ├── data
│   ├── docs
│   ├── output
│   └── src
└── packrat
    └── init.R
```

where

- `MIRACLE_ROOT` is the root directory of your R project[5].
- `SHORT_NAME` is the project short name that you specified when you create your project on MIRACLE[5].

[5] We plan to remove the requirements of having a top-level `MIRACLE_ROOT` folder and a `SHORT_NAME` that matches your online project, in the next release when we complete the project-specific container infrastructure.

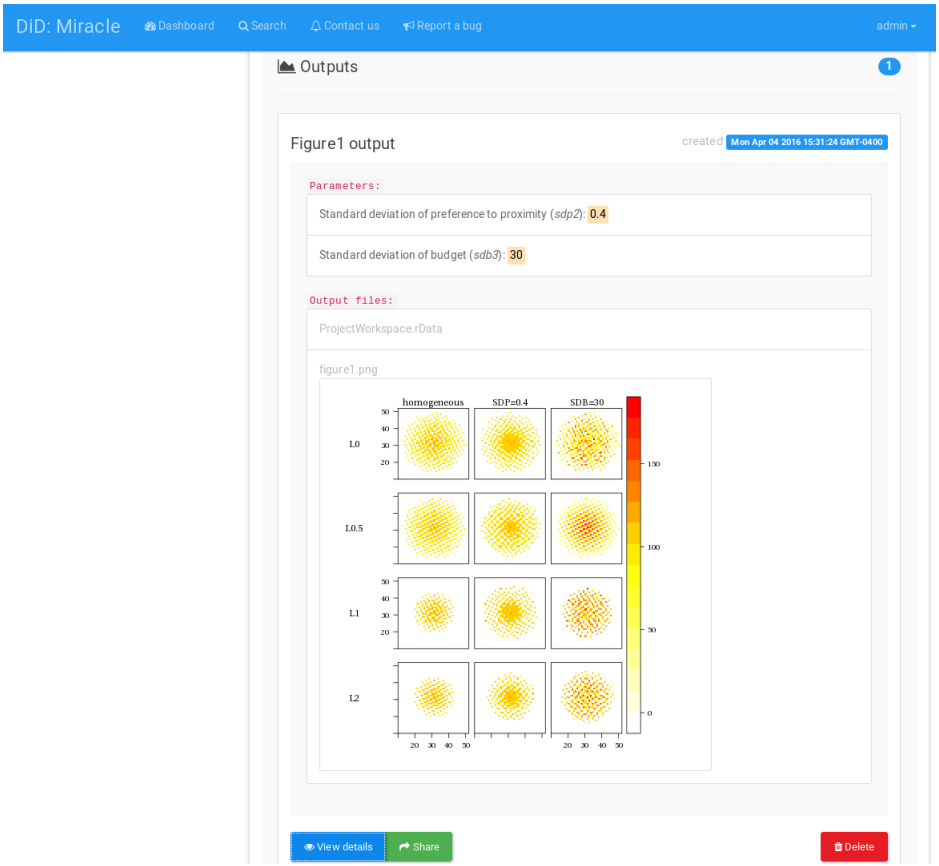

Figure 4. Results from previous analysis runs

- `apps` folder contains interactive applications such as Shiny apps and RMarkdown documents. Each Shiny app should be in its own folder, while all RMarkdown documents go to the `rmd` folder. Shiny apps and RMarkdown documents should reference data using relative paths.
- `data` folder contains all model output data. See Sect. 3.7.1 for recommendations on generating and preparing model output data.
- `docs` folder contains all documentation for the project.
- `output` folder contains all output files from scripts or apps.
- `src` folder contains R code and private R libraries not in a repository. See Sect. 3.7.2 for instructions on how to make your R scripts compatible with MIRACLE.
- `packrat` folder contains all R package dependencies used by the project. This folder is automatically generated when you use RStudio [20] and `packrat` [21] to manage your R dependencies. See Sect. 3.7.3 for preparation instructions.

3.7.1 Model output data guidelines

To make it easier for you to manage your data and metadata, and to make your scripts run more efficiently on MIRACLE, we recommend that you generate your model output data according to the following guidelines:

- Choose open file formats [52] to preserve your data. For structured data, we recommend `csv`, `dbf` or `SQL` databases. A simulation output file format survey at ESSA 2014 found that `csv` was the most popular output format for social simulation models [48]. Currently MIRACLE only supports managing metadata for `csv` files, but we plan to support more formats in the near future.
  - `csv` files must have proper headers with non-empty column names. Column names should not start with numbers.
- Avoid generating a large number of small files from your ABM model, as writing and reading small files is very inefficient and creates one of the major bottlenecks of modern computing. We recommend that individual file size be in the magnitude of megabytes (MB).
- Group data files of identical column structure into a single folder. It is recommended that each folder contain one and only one type of files.

3.7.2 Scripts preparation guidelines

A few simple modifications are recommended to make your R scripts fully compatible with MIRACLE. These modifications enable MIRACLE to automatically load package dependencies, dynamically create parameter input forms where you can run scripts with different parameters, and link your data to your scripts. MIRACLE uses Microsoft's DeployR open source edition [16] to execute R scripts and uses the `deployrUtils` R package to manage dependencies, parameters and input data. You must install `deployrUtils` before you make these modifications[6].

Dependencies [7][8]

In your R code, simply replace `library(package_name)` or `require(package_name)` with `deployrUtils::deployrPackage("package_name")`.

Input parameters To enable the parameter input interface for users to explore the parameter space of your data (see Fig. 3), you must manually mark all input parameters in your R code with `deployrUtils::deployrInput`. For example, the code below, from an analysis script from the LUXE model [38], declares that the parameter `sdp2` is a numeric value and has a default value of 0.4, with all possible values being 0, 0.1, 0.2, 0.3, 0.4 and 0.5.

[6] These modifications are not applicable to RMarkdown documents and Shiny apps.

[7] We plan to remove the requirement of dependency declaration with `deployrPackage` in the next release when we complete the project-specific container infrastructure.

[8] It is best to perform this step as the very last step of the archive preparation process, after enabling the private `packrat` dependency repository (see Sect. 3.7.3), as `packrat` does not automatically recognize dependencies declared by `deployrUtils` at the moment.

```
deployrUtils::deployrInput('{"name": "sdp2", "label": "Standard
    deviation of preference to proximity", "render": "numeric",
    "default": 0.4, "valueList": [0, 0.1, 0.2, 0.3, 0.4, 0.5]}')
```

MIRACLE supports three different methods of defining possible values for a parameter:

- `"valueList": [100, 200, 300]` lists all possible values one-by-one.
- `"valueRange": [min, max, interval]` defines possible values as all values between (including) the min and max values with the specified interval.
- `"valueSet": [0.60,0.65,0.70,0.75,0.80]` means that the value can be any subset of the provided list (`[0.60]` or `[0.60, 0.70, 0.80]` in this example).

Input data In all places that your scripts access input data files, replace the file references with `deployrUtils::deployrExternal`. For example,

```
data <- read.csv(file = deployrExternal(file_path_string))
```

where `file_path_string` is the data file path relative to the R project root folder.

3.7.3 Archive preparation guidelines

MIRACLE uses `packrat` from RStudio to preserve your R code package dependencies and make your scripts fully reproducible using the exact same version of R and R packages on the cloud-based platform. Among the several R package dependency management tools available [22,23], `packrat` was chosen as it allows maximum flexibility in preserving package dependencies from not only public CRAN repositories, but also GitHub and private repositories.

The easiest way to use `packrat` is to create your project using RStudio, or import your existing project into RStudio, then in `Project Options` of RStudio, check `Use packrat with this project`. `Packrat` will create an isolated private package library for this project, and store the source code for all dependencies in such a way that it can be easily restored on another computer or on the MIRACLE server[9]. A detailed user guide of `packrat` is available on its website [24].

Once you have organized your data and scripts according to the guidelines above and enabled `packrat` package management, you can create a project archive that contains your data, scripts, and R package dependencies using the command `packrat::bundle()`. The resulted `tar.gz` file is ready to be uploaded to MIRACLE.

We have prepared two example projects using the guidelines above, which are available in our GitHub repository [25].

[9] `packrat` restoration on MIRACLE server is part of the project-specific container infrastructure and is currently under development.

3.8 MIRACLE server setup guide

The MIRACLE server itself is also fully reproducible. You can set up the MIRACLE platform on your own server with a few simple steps. MIRACLE works on any Linux distributions that support the latest versions of Docker and `docker-compose`.

- Install Docker and `docker-compose`.
- Download the MIRACLE source code using `git clone https://github.com/comses/miracle`.
- Create configuration files based on `docker-compose.yml.example` and `django/miracle/settings/local.py.example`.
- Change all passwords, keys and secrets used in the app, and change NGINX settings according to your requirements. A detailed list of the changes required can be found on our GitHub repository homepage [26].
- Build and run the system using `./build.sh`. This will build and instantiate the containers with all required software. The first build takes approximately one hour (depending on hardware), but subsequent startup should only take about 15 seconds.

4 Ongoing work, future directions, and conclusions

For the next release of MIRACLE, we plan to add automated creation and management of project-specific containers. This feature will make it easier to prepare project archives, and make MIRACLE able to automatically preserve and restore all package dependencies of analyses, making them fully reproducible.

In the longer run, we plan to add the following features to make MIRACLE a more flexible and powerful platform for ABM researchers:

- **Platform functions**
  - Support for more data analysis and visualization languages such as Python and Scala (using Jupyter [27] or Apache Zepellin [28])
  - Improved support for provenance tracking and workflow management to track the entire user workflow and the provenance of all data files, and support for easy sharing of workflows within and beyond the platform. The work will be based on established workflow management software such as YesWorkflow [29] or Luigi [30]
  - Implementing universal search to make it easier to find information within the website
- **Integration and extension**
  - Integration of analysis methods for dimension reduction in ABM output data [43] to make it easier to explore and identify patterns in the high-dimensional output data created by parameter sweeps
  - Integration with existing data archives to make it possible for users to directly import or access data on these services, and exposing metadata to open data initiatives like DataOne [31]
  - Connection to the Compute Canada HPC infrastructure to make it more efficient to run computing-intensive analyses

  - A REST API to make it easier for users to interact with data and tools hosted by MIRACLE
- **Community building**
  - Working with developers of ABM software to incorporate standard metadata generation into Repast Simphony and Netlogo
  - Following the model established by CoMSES Net, implementing a peer-review protocol to certify model analysis archives, and forging agreements with journals that publish model-based social and ecological science to require or recommend that their authors make their model output and analysis available through MIRACLE

Ultimately, by integrating MIRACLE with the existing CoMSES OpenABM model source code archive [1], and adding support for running ABM models directly in the cloud, we plan to provide a cloud-based solution for the entire workflow of ABM research from model creation to model execution, output data generation, data analysis and visualization, and publication review and dissemination. By integrating rich metadata, provenance tracking and visualization functions with community features such as commenting, voting, sharing and forums, the integrated platform will facilitate easy reuse of models, output data and analysis tools both within the ABM community and for the general public.

We believe that sharing model output data can accomplish many of the goals of code sharing, perhaps more efficiently and with fewer barriers. It also lets other researcher explore new parameter spaces, or use different algorithms. Sharing of analysis algorithms may jump start development of complex-systems specific output analysis methods. Although MIRACLE is designed to serves the specific needs of the ABM-CHANS community, we ultimately hope it will provide a standard for all agent-based modellers to share and explore model output data. We hope thus that the tool is helpful for advancing high standards of replicability in ABM-CHANS research; and by increasing the quality of results, we hope it will increase the impact of and trust in that research.

Acknowledgments This project is supported by the third round of the Digging into Data challenge (grant title: MIning Relationships Among variables in large datasets from CompLEx systems), and by the computing infrastructure from Sharcnet and Compute Canada Cloud. The authors would like to thank participants in the iEMSs 2014 “Analyzing and Synthesizing Results from Complex Socio-ecosystem Models with High-dimensional Input, Parameter and Output Spaces” workshop and the Social Simulation Conference 2014 workshop for their inputs and feedback.

References

1. https://www.openabm.org/
2. http://dataverse.org/
3. http://ckan.org/

4. http://www.myexperiment.org
5. https://figshare.com
6. https://osf.io
7. https://genap.ca
8. http://www.cyverse.org/
9. http://sead-data.net/
10. https://www.docker.com/
11. https://www.djangoproject.com/
12. http://www.django-rest-framework.org/
13. https://facebook.github.io/react/
14. http://getbootstrap.com/
15. http://www.postgresql.org/
16. https://deployr.revolutionanalytics.com/
17. https://www.rstudio.com/products/shiny/shiny-server/
18. http://vnijs.github.io/radiant/
19. https://www.nginx.com/
20. https://www.rstudio.com/
21. https://rstudio.github.io/packrat/
22. https://cran.r-project.org/web/packages/checkpoint/index.html
23. https://github.com/gmbecker/gRAN
24. http://rstudio.github.io/packrat/rstudio.html
25. https://github.com/comses/miracle-example-projects
26. https://github.com/comses/miracle
27. http://jupyter.org/
28. https://zeppelin.incubator.apache.org/
29. https://github.com/yesworkflow-org
30. https://github.com/spotify/luigi
31. https://www.dataone.org/
32. Allesa, L.N., Laituri, M., Barton, M.: An "all hands" call to the social science community: Establishing a community framework for complexity modeling using agent based models and cyberinfrastructure (2006)
33. Collberg, C., Proebsting, T.A.: Repeatability in computer systems research. Communications of the ACM 59(3), 62–69 (2016)
34. Fagiolo, G., Moneta, A., Windrum, P.: A critical guide to empirical validation of agent-based models in economics: Methodologies, procedures, and open problems. Comput Econ 30(3), 195–226 (2007)
35. Fowler, M.: Microservices (March 2014), http://martinfowler.com/articles/microservices.html
36. Gimblett, H.R.: Integrating Geographic Information Systems and Agent-Based Modeling Techniques for Understanding Social and Ecological Processes. Oxford University Press, Oxford, U.K. (2001)
37. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., Railsback, S.F.: The odd protocol: A review and first update. Ecological Modelling 221(23), 2760–2768 (2010)
38. Huang, Q., Parker, D.C., Sun, S., Filatova, T.: Effects of agent heterogeneity in the presence of a land-market: A systematic test in an agent-based laboratory. Computers, Environment and Urban Systems 41, 188–203 (sep 2013), http://linkinghub.elsevier.com/retrieve/pii/S0198971513000616
39. Janssen, M.: Archiving practice for model code of agent-based models (December 2014), http://marco-janssen.blogspot.ca/2014/12/archiving-practice-for-model-code-of.html

40. Janssen, M.A., Alessa, L.N., Barton, M., Bergin, S., Lee, A.: Towards a Community Framework for Agent-Based Modelling. Journal of Artificial Societies and Social Simulation 11(2), 6 (2008), http://jasss.soc.surrey.ac.uk/11/2/6.html
41. Kohler, T.A.: Dynamics in Human and Primate Societies. Oxford University Press, New York and Oxford (2000)
42. Lansing, J.S., Kremer, J.N.: Emergent properties of balinese water temple networks: Coadaptation on a rugged fitness landscape. American Anthropologist 95(1), 97–114 (1993)
43. Lee, J.S., Filatova, T., Ligmann-Zilinska, A., Hassani-Mahmooei, B., Stonedahl, F., Lorscheid, I., Voinovv, A.A., Polhill, J.G., Sun, Z., Parker, D.: The complexities of agent-based modelling output analysis 18(4), – (2015)
44. Muenchen, R.A.: The popularity of data analysis software (April 2016), http://r4stats.com/articles/popularity/
45. Müller, B., Balbi, S., Buchmann, C.M., de Sousa, L., Dressler, G., Groeneveld, J., Klassert, C.J., Le, Q.B., Millington, J.D., Nolzen, H., Parker, D.C., Polhill, J.G., Schlüter, M., Schulze, J., Schwarz, N., Sun, Z., Taillandier, P., Weise, H.: Standardised and transparent model descriptions for agent-based models: Current status and prospects. Environmental Modelling & Software 55, 156 – 163 (2014), http://www.sciencedirect.com/science/article/pii/S1364815214000395
46. Parker, D.C., Manson, S.M., Janssen, M.A., Hoffmann, M.J., Deadman, P.: Multi-agent systems for the simulation of land-use and land-cover change: A review 93(2), 314–337– (2003), <GotoISI>://000184143400004
47. Pignotti, E., Edwards, P., Polhill, G.: Supporting simulations on the grid using workflows and virtual machines. In: UK e-Science All Hands Meeting 2009: Past, Present and Future, 5th IEEE International Conference. Oxford, UK (December 2009)
48. Polhill, G.: Simulation output file format survey at the Social Simulation Conference 2014 (September 2014), unpublished
49. Polhill, J.G., Dawson, T.D., Parker, D.C., Jin, X., Robinson, K., Filatova, T., Voinovv, A.A., Barton, M., Pignotti, E., Edwards, E.: Towards metadata standards for sharing simulation outputs. In: Miguel, Amblard, Barceló, Madella (eds.) Advances in Computational Social Science and Social Simulation (ESSA 2014). pp. –. Autònoma University of Barcelona, Barcelona (2014), <http://ddd.uab.cat/record/125597
50. Pritchard, C.: Documentation and justification for the sluce metadata database. Tech. rep., School of Planning, University of Waterloo (2012)
51. Rollins, N.D., Barton, C.M., Bergin, S., Janssen, M.A., Lee, A.: A computational model library for publishing model documentation and code. Environmental Modelling & Software 61, 59 – 64 (2014), http://www.sciencedirect.com/science/article/pii/S1364815214001959
52. Stanford University Libraries: Best practices for file formats, https://library.stanford.edu/research/data-management-services/data-best-practices/best-practices-file-formats
53. Stodden, V., Miguez, S., Seiler, J.: Researchcompendia. org: Cyberinfrastructure for reproducibility and collaboration in computational science. Computing in Science & Engineering 17(1), 12–19 (2015)
54. Vitek, J., Kalibera, T.: R3: Repeatability, reproducibility and rigor. ACM SIGPLAN Notices 47(4a), 30–36 (2012)
55. Yang, T.: A demonstration project for overcoming the barriers of modelling in the policy process. B.E.S Honour's Thesis, School of Planning, University of Waterloo (2011)