Journal of Simulation

ISSN: 1747-7778 (Print) 1747-7786 (Online) Journal homepage: www.tandfonline.com/journals/tjsm20

On the reproducibility of discrete-event simulation studies in health research: an empirical study using open models

Amy Heather, Thomas Monks, Alison Harper, Navonil Mustafee & Andrew Mayne

To cite this article: Amy Heather, Thomas Monks, Alison Harper, Navonil Mustafee & Andrew Mayne (09 Sep 2025): On the reproducibility of discrete-event simulation studies in health research: an empirical study using open models, Journal of Simulation, DOI: 10.1080/17477778.2025.2552177

To link to this article: https://doi.org/10.1080/17477778.2025.2552177

JOURNAL OF SIMULATION
https://doi.org/10.1080/17477778.2025.2552177

RESEARCH ARTICLE

On the reproducibility of discrete-event simulation studies in health research: an empirical study using open models

Amy Heather[a], Thomas Monks[a], Alison Harper[b], Navonil Mustafee[b] and Andrew Mayne[c]

[a]Department of Health and Community Sciences, University of Exeter Medical School, Exeter, UK; [b]Centre for Simulation, Analytics and Modelling (CSAM), University of Exeter Business School, Exeter, UK; [c]Somerset NHS Foundation Trust, Taunton, UK

ABSTRACT
Reproducibility of computational research is critical for ensuring transparency, reliability, and reusability. Challenges with computational reproducibility have been documented in several fields, but healthcare discrete-event simulation (DES) models have not been thoroughly examined in this context. This study assessed the computational reproducibility of eight published healthcare DES models (Python or R), selected to represent diverse contexts, complexities, and years of publication. Repositories and articles were also assessed against guidelines and reporting standards, offering insights into their relationship with reproducibility success. Reproducing results required up to 28 hours of troubleshooting per model, with 50% fully reproduced and 50% partially reproduced (12.5% to 94.1% of reported outcomes). Key barriers included the absence of open licences, discrepancies between reported and coded parameters, and missing code to produce model outputs, run scenarios, and generate tables and figures. Addressing these issues would often require relatively little effort from authors: adding an open licence and sharing all materials used to produce the article. Actionable recommendations are proposed to enhance reproducibility practices for simulation modellers and reviewers.

ARTICLE HISTORY
Received 22 January 2025
Accepted 20 August 2025

KEYWORDS
Discrete-event simulation; health; reproducibility; open science; open models

1. Introduction

The reproducibility and replicability of published research are widely recognised as cornerstones of rigorous science and have been investigated across numerous disciplines (Baker et al., 2020). Within the field of modelling and simulation, the importance of reproducibility has gained increasing attention and is recognised as essential for ensuring the reliability and impact of simulation-based research (Fišar et al., 2024; Monks et al., 2019; Wilsdorf et al., 2023). Despite this growing emphasis, no study has empirically assessed the reproducibility of results generated by healthcare discrete-event simulation (DES) models. These models are important tools in healthcare decision-making, aiming to enhance patient outcomes and optimise health service delivery by addressing challenges such as patient flow management, resource allocation, and cost-effectiveness analyses (Salleh et al., 2017). This study investigates the reproducibility of open DES models—those with code made publicly available for others to run and reuse—with an emphasis on identifying barriers and facilitators to reproduction. The study identifies the work required to reuse the author-supplied model code to reproduce the simulation results presented in the tables, charts, and text of journal articles, including any troubleshooting required during the process. The study aims to provide actionable guidance to simulation modellers to improve the reproducibility of their work.

1.1. Why reproducibility matters

Reproducibility is the ability to regenerate published results using the provided code and data. Failures in reproducibility may indicate underlying issues, such as incorrect parameters, missing or outdated code, or unexpected changes in software behaviour due to updates. Achieving reproducibility fosters trust and transparency, building confidence that the model behaves consistently with the original implementation. It is vital to establish reproducibility before a model is reused (Alston & Rick, 2021; Harper et al., 2021; Sandve et al., 2013)

Reuse is the adaptation and application of a model in new contexts, amplifying the potential real-world impact of the model. Reuse is in high demand; for example, 37% of Wellcome Trust-funded researchers report using external code, often from colleagues or community repositories (Van Den Eynden et al., 2016). As shown in Figure 1, reproducibility exists within a continuum of code attributes, including being re-runnable, repeatable, reusable, and replicable. Together, these characteristics underpin the validity and reliability of simulation models (Benureau & Rougier, 2018).

CONTACT Thomas Monks t.m.w.monks@exeter.ac.uk South Cloisters, St Luke's Campus, University of Exeter, Exeter, EX1 2LU, UK

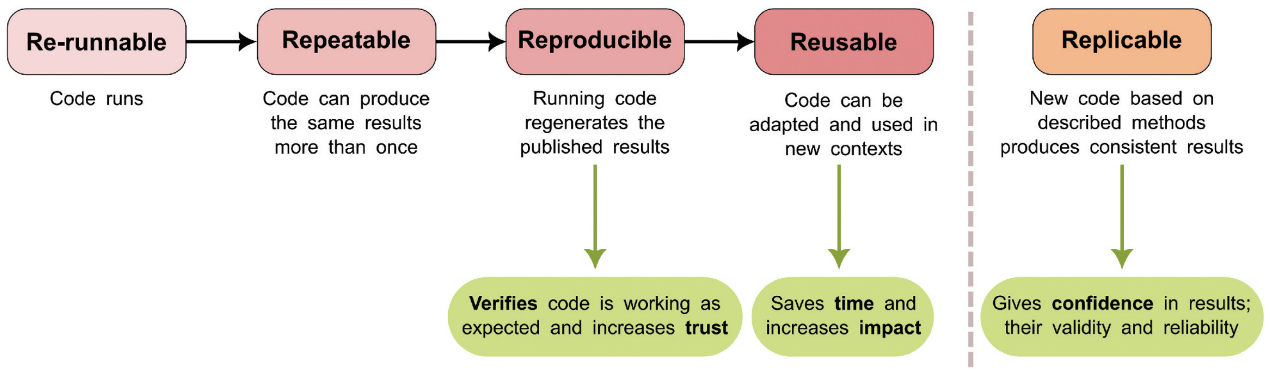
Re-runnable
Code runs
Repeatable
Code can produce the same results more than once
Reproducible
Running code regenerates the published results
Reusable
Code can be adapted and used in new contexts
Replicable
New code based on described methods produces consistent results
Verifies code is working as expected and increases trust
Saves time and increases impact
Gives confidence in results; their validity and reliability

Figure 1. Five standards for scientific code, as described in Benureau & Rougier, (2018).

Reproducibility also benefits authors by making it easier to revisit and reuse their own code, such as for updating outputs or conducting new analyses (Alston & Rick, 2021). Troubleshooting non-reproducible code can be time-consuming or even impossible if required information is now lost (Alston & Rick, 2021; Sandve et al., 2013). For instance, code may need to be re-run following peer review (Alston & Rick, 2021). Given that the mean time from submission to publication in biomedical research ranges from 27 to 639 days (Andersen et al., 2021), it is important that code remains functional and reproducible long after its initial execution. Even if authors retain access to their code, failure to record exact parameters for every scenario or document the computational environment used, for example, could make the code non-reproducible when revisited, even if only a few months later. Finally, reproducibility often improves overall code quality. It encourages clear structure and documentation, reducing errors and ambiguities (Alston & Rick, 2021). These advantages apply regardless of whether code is shared publicly or remains proprietary.

1.2. *Addressing challenges in reproducibility*

The computational reproducibility of published research has been investigated across numerous disciplines. These studies, which focus on the reproducibility of results using shared code and data, vary in scope and methodology—for example, the extent of troubleshooting allowed when running code. Across many of these studies though, a consistent finding is that the majority of attempts to reproduce results fail, with examples in epidemiology (Henderson et al., 2024), public health (B. D. K. Wood et al., 2018), economics (Galiani et al., 2017; McCullough et al., 2006), political science (Eubank, 2016), physics (Krafczyk et al., 2021; Stodden, Krafczyk, et al., 2018), and hydrology (Stagge et al., 2019)

However, some studies do observe higher rates of reproducibility, often under specific conditions. Investigations in political science (Stockemer et al., 2018), psychology (Obels et al., 2020), and articles published in *Science* (Stodden, Seiler, et al., 2018) found that while many papers lack accessible data or code, the majority of results could be reproduced in those that provide these resources, in these cases. Similarly, studies in geoscience (Konkol et al., 2019) and a large-scale analysis of articles with Jupyter notebooks (Samuel & Mietchen, 2024) found that artefacts may often fail to execute, but when they do, the majority of results could be reproduced. High reproducibility was observed in management science (Fišar et al., 2024) where 95% of studies were fully or largely reproduced—though this dropped to 68% when including studies with inaccessible datasets.

Journals can play a pivotal role in addressing reproducibility issues, as summarised in Table 1. Simple initiatives include requiring code availability statements or requiring that code is shared upon request. However, authors may simply state that code is unavailable or choose not to share it (Janssen et al., 2020; Stodden, Seiler, et al., 2018). For example, in a study of individual- and agent-based simulation models, researchers emailed authors who had stated in journal articles that their code was available upon request. However, they received responses from less than 1% of authors—some provided links to the code, but others stated they no longer had access to it (Janssen et al., 2020). To address this, journals may mandate code sharing via public repositories or archives (Association for Computing Machinery [ACM], 2020; Blohowiak et al., 2023; Cadwallader et al., 2021; Fišar et al., 2024; Hardwicke & Vazire, 2024; Institute of Electrical and Electronics Engineers [IEEE], 2024; Nature Computational Science, 2023; Loder et al., 2024). However, code sharing alone does not guarantee reproducibility, so journals may also review repositories to check that code is documented, complete, and/or likely to be executable (ACM, 2020; Hardwicke & Vazire, 2024; Nature Computational Science, 2023; IEEE, 2024; Management Science, 2019). Some journals attempt to run the provided code during peer review to assess its computational

Table 1. Cross-disciplinary journal initiatives to improve computational transparency and reproducibility.

Initiatives	Examples
Mandatory code sharing	Journals incentivise or require code to be shared publicly through: • Policies for mandatory code sharing (Cadwallader et al., 2021; Fišar et al., 2024; Hardwicke & Vazire, 2024; Nature Computational Science, 2023; Loder et al., 2024; Management Science, 2019). • Badges awarded to articles with shared code (Association for Computing Machinery ACM, 2020; Blohowiak et al., 2023; Institute of Electrical and Electronics Engineers IEEE, 2024).
Review of shared artefacts	Journals evaluate the shared artefacts with: • Policies requiring details on the language and packages, commented code, and test data/problems sufficient for replication (Management Science, 2019). • Policies requiring open licenses and clear documentation (Hardwicke & Vazire, 2024). • Integration with sharing platforms which streamline submission and peer review of code within the publication workflow (Nature Computational Science, 2023). • Badges awarded following evaluation of artefacts (e.g., checking they are complete, executable and documented) (Association for Computing Machinery ACM, 2020; Institute of Electrical and Electronics Engineers IEEE, 2024).
Assess computational reproducibility	Journals actively test whether results can be reproduced through: • Collaboration with independent institutions to reproduce and/or replicate a sample of published articles from the journal (Hardwicke & Vazire, 2024, Nature Human Behaviour, 2024). • Independent reproduction attempts during peer review, with badges awarded for successful reproductions (Association for Computing Machinery ACM, 2020; Hardwicke & Vazire, 2024; Institute of Electrical and Electronics Engineers IEEE, 2024).
Accept reproductions as formal publications	Journals recognise reproduction work as scholarly contribution by publishing: • Independent reproduction studies from independent groups reproducing, reusing, or extending published code (Nature Machine Intelligence, 2022). • Reports documenting reproduction attempts during peer review as supplementary material (Association for Computing Machinery ACM, 2020).

reproducibility (ACM, 2020; Hardwicke & Vazire, 2024; IEEE, 2024; Nature Human Behaviour, 2024) or accept reproductions of published work by other groups as formal publications (Nature Machine Intelligence, 2022). When code or data cannot be shared for ethical or legal reasons, at least providing the reviewer access for reproducibility checks is valuable, as others will not be able to interrogate it and assess validity after publication (Eubank, 2016). Alternatively, authors could share code with synthetic data for validation (Monks et al., 2019). To provide a clear overview and comparison of journal policies, the Centre for Open Science (COS) reviews them on their website (https://topfactor.org/) against their Transparency and Openness Promotion (TOP) Guidelines (Centre for Open Science, 2024).

Several conferences have also taken similar actions. For example, the *Association for Computing Machinery (ACM) Special Interest Group on Simulation and Modelling (SIGSIM) Principles of Advanced Discrete Simulation (PADS)* conference had a reproducibility committee to assess papers against the ACM badges (ACM SIGSIM, 2025).

1.3. Healthcare simulation models

This study focuses on healthcare simulation models, specifically DES. DES has a wide range of applications, including healthcare operations and system design, medical decision-making, and infectious disease modelling (Salleh et al., 2017). It is the most commonly used simulation method in healthcare (Philip et al., 2023; Roy et al., 2021; Salleh et al., 2017; Salmon et al., 2018).

In recent years, efforts have been made to enhance the sharing and reuse of healthcare simulation models. The *Strengthening The Reporting of Empirical Simulation Studies-DES* (STRESS-DES) guidelines from Monks et al. (2019) were developed to facilitate replication by providing technical details necessary to recreate a model. Zhang et al. (2020) developed a "generic reporting checklist" focused on model quality, derived from the *International Society for Pharmacoeconomics and Outcomes Research (ISPOR)—Society for Medical Decision Making (SMDM) Modelling Good Research Practices Task Force* reports. In 2024, the *Sharing Tools and Artefacts for Reusable Simulations* (STARS) framework was introduced to provide guidance on which artefacts should be shared in repositories to facilitate model reuse (Monks et al., 2024)

These guidelines and frameworks (fully detailed in Appendix A) were developed because many healthcare DES models are not shared in ways that facilitate understanding, use, and reproducibility. A 2023 review (Monks & Harper, 2025) of 564 healthcare DES papers (2019–2022) revealed only 8.3% shared model code, although this was improving over time. Among shared models, only 60% included a README, with just 32% explaining how to run the model. Less than half provided an open licence, and only 10.6% archived their models. Across all papers, only 12.8% used reporting guidelines (Monks & Harper, 2025).

There are no studies specifically focused on the reproducibility of healthcare DES models. However, several studies have included simulation models in their sample, though these were in other domains (Eubank, 2016; Fišar et al., 2024; Krafczyk et al., 2021; Stodden, Seiler, et al., 2018). One healthcare-specific example is Henderson et al. (2024), which examined the reproducibility of infectious disease models and included some simulation models. They analysed two samples of 100 articles and found that 21% to 23% were fully reproducible, 42% to 46% were partially reproducible, and 31% to 37% were not reproducible, with minimal troubleshooting allowed (Henderson et al., 2024). However, their study did not explore the barriers and facilitators of reproducibility within this context, an aspect that would have required more extensive troubleshooting. While the non-healthcare simulation studies have explored barriers and facilitators (Eubank, 2016; Fišar et al., 2024; Krafczyk, 2021; Stodden, Seiler, et al., 2018), they were not limited to simulation models or healthcare contexts.

This study aims to assess the reproducibility of published healthcare DES models by examining a sample of eight studies. It will identify factors that facilitate or hinder each reproduction and will evaluate adherence to relevant frameworks, badges, and reporting guidelines, to understand how these influence reproducibility. By understanding which elements are most important when reporting results in articles and sharing code, the study seeks to identify key factors that can improve reproducibility in healthcare DES models. Rather than providing a theoretical list of best practices, this empirical approach focuses on understanding what modellers actually do in practice and what specific barriers arise when attempting reproduction, enabling more targeted and actionable guidance than general principles alone. While not all studies may have been designed with computational reproducibility in mind, it is increasingly recognised as an important consideration for any research involving code (Benureau & Rougier, 2018) as it helps verify the integrity of findings. Adopting basic practices to improve reproducibility can enhance scientific rigour and is achievable without requiring advanced technical skills.

2. Methods

The study protocol (as summarised in Figure 2 and described below) was pre-registered on June 20 2024 (Heather et al., 2024). It was informed by prior reproduction studies (Konkol et al., 2019; Krafczyk et al., 2021; Laurinavichyute et al., 2022; Schwander et al., 2021; B. D. K. Wood et al., 2018) and refined through a pilot on a Python DES model by Allen et al. (2020).

2.1. Sample

This study included eight models which were selected through purposive sampling. Of these, seven models were drawn from the sample of an existing systematic scoping review (2018–2022) (Monks & Harper, 2025), whilst one additional model was identified through an informal exploratory search of earlier literature. Models were developed in Python (Python Core Team, 2024) or R (R Core Team, 2024), the most

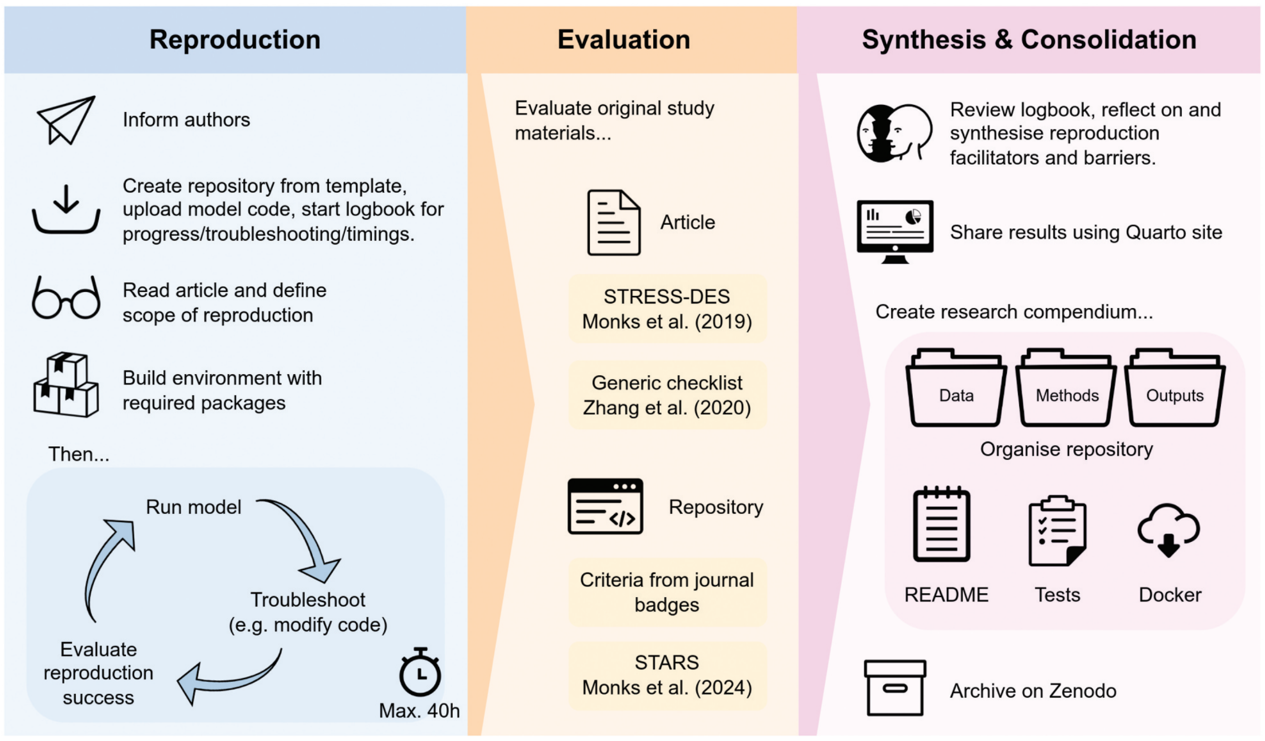
Reproduction
Inform authors
Create repository from template, upload model code, start logbook for progress/troubleshooting/timings.
Read article and define scope of reproduction
Build environment with required packages
Then...
Run model
Troubleshoot (e.g. modify code)
Evaluate reproduction success
Max. 40h
Evaluation
Evaluate original study materials...
Article
STRESS-DES Monks et al. (2019)
Generic checklist Zhang et al. (2020)
Repository
Criteria from journal badges
STARS Monks et al. (2024)
Synthesis & Consolidation
Review logbook, reflect on and synthesise reproduction facilitators and barriers.
Share results using Quarto site
Create research compendium...
Data
Methods
Outputs
Organise repository
README
Tests
Docker
Archive on Zenodo

Figure 2. Study methodology. Abbreviations: STARS, *Sharing Tools and Artefacts for Reusable Simulations*; STRESS-DES, *Strengthening The Reporting of Empirical Simulation Studies-Discrete-Event Simulation*.

Table 2. Description of the included studies.

Study	Journal/Conference	Model	Geographical context	Scope*	Language**
Hernandez et al. (2015)	*Computers & Industrial Engineering*	Optimal staff numbers at sites dispensing counter-measures in a public health emergency	New York, US	8	Python (*SimPy*) (Team SimPy, 2024) & R
Huang et al. (2019)	*Frontiers in Neurology*	Wait time and resource utilisation of an endovascular clot retrieval service under different configurations	Australia	8	R (*simmer*) (Ucar et al., 2019)
Lim et al. (2020)	*Clinical Biochemistry*	Impact of staff configurations and safety measures on COVID-19 transmission in a laboratory	Not specified	9	Python
Kim et al. (2021)	*PLOS ONE*	Impact of COVID-19-related disruption on outcomes of an abdominal aortic aneurysm screening programme	England	10	R
Johnson et al. (2021)	*Applied Health Economics and Health Policy*	Cost-effectiveness of different primary care-based case detection strategies for COPD	Canada	5	R (interface for C++ model)
R. M. Wood et al. (2021)	*Medical Decision Making*	Deaths and life years lost under different triage strategies for an intensive care unit during COVID-19	UK	5	R
Shoaib & Ramamohan (2022)	*Simulation*	Resource utilisation in primary health centres in with differing resources, services and operational patterns	India	17	Python (*salabim*) (Van Der Ham, 2018)
Anagnostou et al. (2022)	*Winter Simulation Conference*	Intensive care unit capacity during COVID-19	Not specified	1	Python (*SimPy*) (Team SimPy, 2024)

* Scope refers to the number of items to be reproduced: figures, tables, and/or results described in the text.
** Language includes simulation package, where applicable; some studies instead developed the model from scratch.
Abbreviations: COPD, chronic obstructive pulmonary disease; COVID-19, coronavirus disease 2019; UK, United Kingdom; US, United States.

popular Free and Open Source Software (FOSS) languages for healthcare DES (Monks & Harper, 2025). Models were selected to ensure diversity across a range of factors including the health focus (e.g., healthcare condition, specific system), geographical context, and model complexity—as in Table 2.

The protocol initially planned to include six studies, but this was later expanded to eight during the research process. This allowed the inclusion of another "medium-sized" model (R. M. Wood et al., 2021) that was felt to better represent typical operational research applications of DES in health compared to some larger-scale models. An older model (Hernandez et al., 2015) was also included, enabling consideration of reproducibility challenges from an older code base (approximately a decade ago), as opposed to more recent studies (with all others published within the last 5 years).

2.2. Reproduction

Before starting, the authors of the eight studies were informed, and four were requested to add an open licence to their repository. A maximum of 40 hours was allowed for each reproduction (excluding computation time). The reproduction steps were as follows:

(1) **Set-up**. A repository was created, including a logbook to track time and record progress and any barriers to running the code and reproducing results.
(2) **Scope**. At least two team members reviewed the article and agreed on which experimental results (e.g., tables, figures, results described in the text) to reproduce. The repository was then archived on Zenodo (European Organization For Nuclear Research and OpenAIRE, 2013).
(3) **Running and troubleshooting**. A researcher set up an environment with the necessary software and packages. When creating a software environment for each reproduction, the protocol originally planned to backdate software and package versions to those specified or available at the time of publication. This approach was successfully implemented for Python studies. However, for studies using R, there were substantial challenges in backdating R and its packages. Hence, the latest versions were instead used, with code modifications applied as needed. Code was run and troubleshooted—typically by modifying or writing code, and consulting the article. Unresolved issues were discussed with the team, and then with the original authors, for whom response was optional. This approach differs from typical journal reproducibility assessments, as much more extensive troubleshooting was allowed in this research.
(4) **Reproduction success**. A consensus decision was made as to whether each item had been successfully reproduced or not. This was a subjective decision that allowed some expected deviation due to model stochasticity.

2.3. Evaluation

Articles were evaluated using two DES reporting guidelines: STRESS-DES from Monks et al. (2019), and the "generic reporting checklist" from Zhang et al. (2020). Repositories were evaluated against the STARS framework (Monks et al., 2024) and criteria for journal badges from: ACM (Association for Computing Machinery ACM, 2020), National Information Standards Organisation (NISO) (NISO Reproducibility Badging and Definitions Working Group, 2021), Centre for Open Science (COS) (Blohowiak et al., 2023), Institute of Electrical and Electronics Engineers (IEEE) (IEEE, 2024), and *Psychological Science* ([APS], 2024; Hardwicke & Vazire, 2024). The evaluation was performed by one researcher, with consensus from a second for uncertain or unmet criteria. For reference, full guidelines and badge criteria are provided in Appendix A.

2.4. Synthesis and consolidation

Results from the reproduction, evaluation, and reflections on the facilitators and barriers encountered were shared via a Quarto website (Allaire et al., 2024) hosted on GitHub Pages (GitHub, 2024). The repository was organised into a research compendium (Gentleman & Lang, 2007), with a README, conventional file organisation (e.g., data, methods, outputs), tests, and a Docker environment (Merkel, 2014). These tests, specifically written for this study, run the model to confirm it produces consistent results, either reproducing full paper outputs or, for long-running models, executing smaller verification cases. A second researcher verified execution and reproducibility by running the model and/or tests (locally and via Docker). The final repository was archived on Zenodo (European Organization For Nuclear Research and OpenAIRE, 2013), and results were shared with the original authors.

2.5. Citations and visualisations

Studies are not referenced by name in the results, to keep the emphasis on overarching trends rather than individual reproduction success. For transparency, full results are available, with links to the reproduction and original study repositories provided in Appendix C. For this article, plots were created using Python 3.10.14 (Python Core Team, 2024), *Plotly* 5.23.0 (Plotly Technologies Inc, 2015) *Matplotlib* 3.9.2 (Hunter, 2007) and *pandas* 2.2.2 (The pandas development team, 2024; McKinney, 2010), with a full list of dependencies in the research repository (Heather et al., 2025a). Other visualisations were created using Inkscape 1.3.2 (https://inkscape.org/) and Sketchpad v2022.2.21.0 (https://sketch.io/sketchpad/).

3. Results

3.1. Reproduction

Half of the studies were fully reproduced, with two completed in under 4 hours and two requiring 12–15 hours. The remaining four were partially reproduced (12.5% to 94.1%) and took 18–28 hours (Figure 3). These times exclude model computation, which ranged widely: one model ran in seconds, three in under an hour, and four required several hours to days. Times were influenced by optimisations like parallel processing, and the machine used. Models were run on an Intel Core i7 -12,700 H with 32GB RAM (Ubuntu 22.04.4), or an Intel Core i9-13900K with 81GB RAM (Pop!_OS 22.04), which was able to run for longer amounts of time, and accommodated models with high memory demands.

3.2. Evaluation of the repository

Repositories were assessed using the STARS framework, with the results of the assessment in Figure 4. All models were FOSS and hosted on a remote repository, though only one was archived. Licensing was inconsistent, with half requiring follow-up with authors. Documentation, dependency management, citation information, and ORCID were minimal.

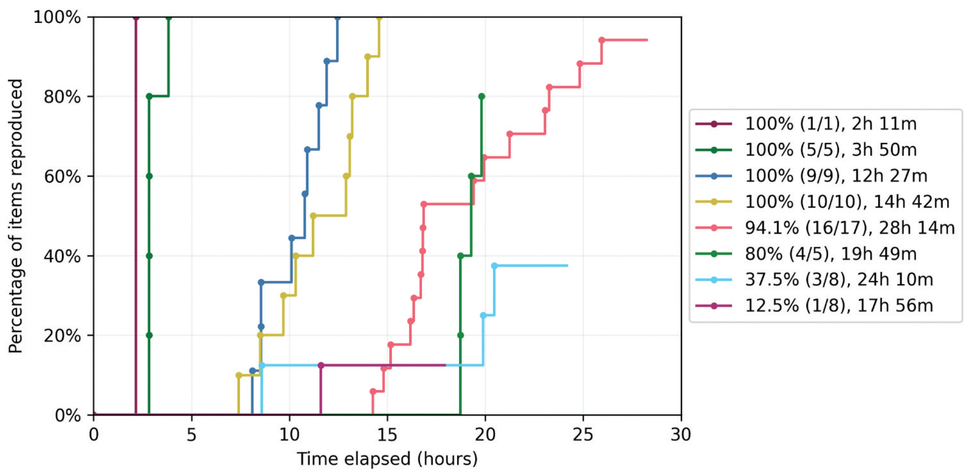

Figure 3. Count, proportion, and time to reproduce items within the scope of each study. Inspired by a figure in Krafczyk et al. (2021).

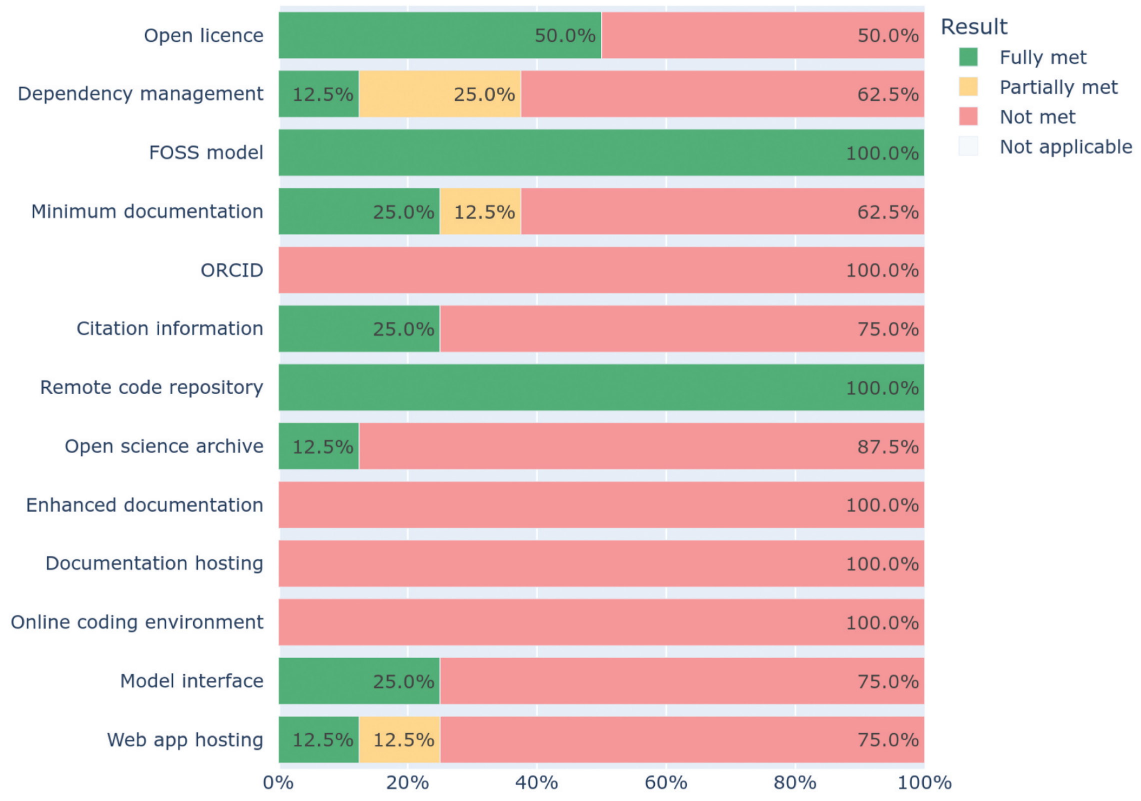

Figure 4. Of the eight healthcare DES studies evaluated, the proportion that met each recommendation in the current STARS framework. For a full description of each criterion, see Appendix A. Abbreviations: DES, discrete-event simulation; STARS, *Sharing Tools and Artefacts for Reusable Simulations*.

Table 3. Evaluation of repositories against ACM badge criteria.

Badge	Criteria	Studies that met criteria
ACM "Artefacts Available"	• Artefacts are archived in a repository that is: (a) public (b) guarantees persistence (c) gives a unique identifier (e.g., DOI)	1/8 (12.5%)
ACM "Artefacts Evaluated - Functional"	• Documents (a) inventory of artefacts (b) sufficient description for artefacts to be exercised • Artefacts relevant to the paper • Complete (all relevant artefacts available) • Scripts can be successfully executed	0/8 (0.0%)
ACM "Artefacts Evaluated - Reusable"	Criteria for "Functional" badge plus: • Artefacts are carefully documented and well-structured to the extent that reuse and repurposing is facilitated, adhering to norms and standards	0/8 (0.0%)
ACM "Results Reproduced"	• Reproduced results (assuming (a) acceptably similar (b) reasonable time frame (c) only minor troubleshooting)	1/8 (12.5%)

Abbreviations: ACM, Association for Computing Machinery; DOI, digital object identifier.

With regards to optional STARS components, two studies had applications, though none had enhanced documentation or online coding environments.

The repositories were also evaluated against journal badge criteria, with results for the ACM badges summarised here (Table 3) and the full evaluation in Appendix B. Only one repository met the standards for the "Artefacts Available" badge. None satisfied "Artefacts Functional" requirements, typically due to incomplete code (missing scenarios, outputs, etc.). Whilst those evaluations used the original artefacts as shared by the study authors, the "Results Reproduced" assessment was based on reproduction within a reasonable time allowing minimal troubleshooting, as per ACM criteria. This differs from the extended reproduction attempts described above, which allowed many hours of extensive troubleshooting. Under the stricter ACM criteria, only one study was eligible for "Results Reproduced".

The proportion of STARS criteria met by each individual study ranged from 25% to 88% for essential components and 0% to 40% for optional components. These metrics showed no relationship with reproduction success (Table 4). However, some individual STARS components and badge criteria were critical to reproduction. For example, identifying the packages and versions necessary to run the code without error can be time-consuming if not stated by the authors, and dependency management is a component of both STARS and several badges. A complete set of materials, required for the ACM "Artefacts Evaluated" badge (though absent in STARS), was also essential. Missing materials—such as parameters, scenarios, or code—were a major barrier to reproduction in most studies. Documentation is required by STARS and several

Table 4. The proportion of applicable criteria that were fully met, from evaluation of repository or article, alongside the proportion of items reproduced from each study.

Items reproduced	STARS (essential)	STARS (optional)	STRESS-DES	Generic checklist
Fully reproduced				
100% (10/10)	50%	0%	68%	71%
100% (9/9)	25%	0%	71%	75%
100% (5/5)	25%	0%	92%	76%
100% (1/1)	88%	20%	67%	53%
Partially reproduced				
94.1% (16/17)	25%	0%	71%	73%
80% (4/5)	50%	0%	84%	88%
37.5% (3/8)	25%	40%	61%	44%
12.5% (1/8)	25%	0%	78%	59%

Abbreviations: STARS, *Sharing Tools and Artefacts for Reusable Simulations*; STRESS-DES, *Strengthening The Reporting of Empirical Simulation Studies-Discrete-Event Simulation*.

badges, and were helpful if provided as they can reduce time spent deciphering how to run the code. Finally, an open licence, as specified in both STARS and for some badges, is fundamental to enable legal access and use of the code.

3.3. Evaluation of the article

Studies were evaluated against STRESS-DES (Monks et al., 2019) and the generic reporting checklist (Zhang et al., 2020)—with the proportion of studies meeting each individual criterion presented in Figures 5 and 6, respectively. Most met criteria related to purpose and design. However, technical details in STRESS-DES—such as input parameters, initialisation, random sampling, and execution—were often incomplete or unclear. While all studies mentioned the software or programming language used, these descriptions were minimal—for example, often mentioning the programming language and main simulation package, but not the operating system or versions used. System specifications were rarely fully described, and

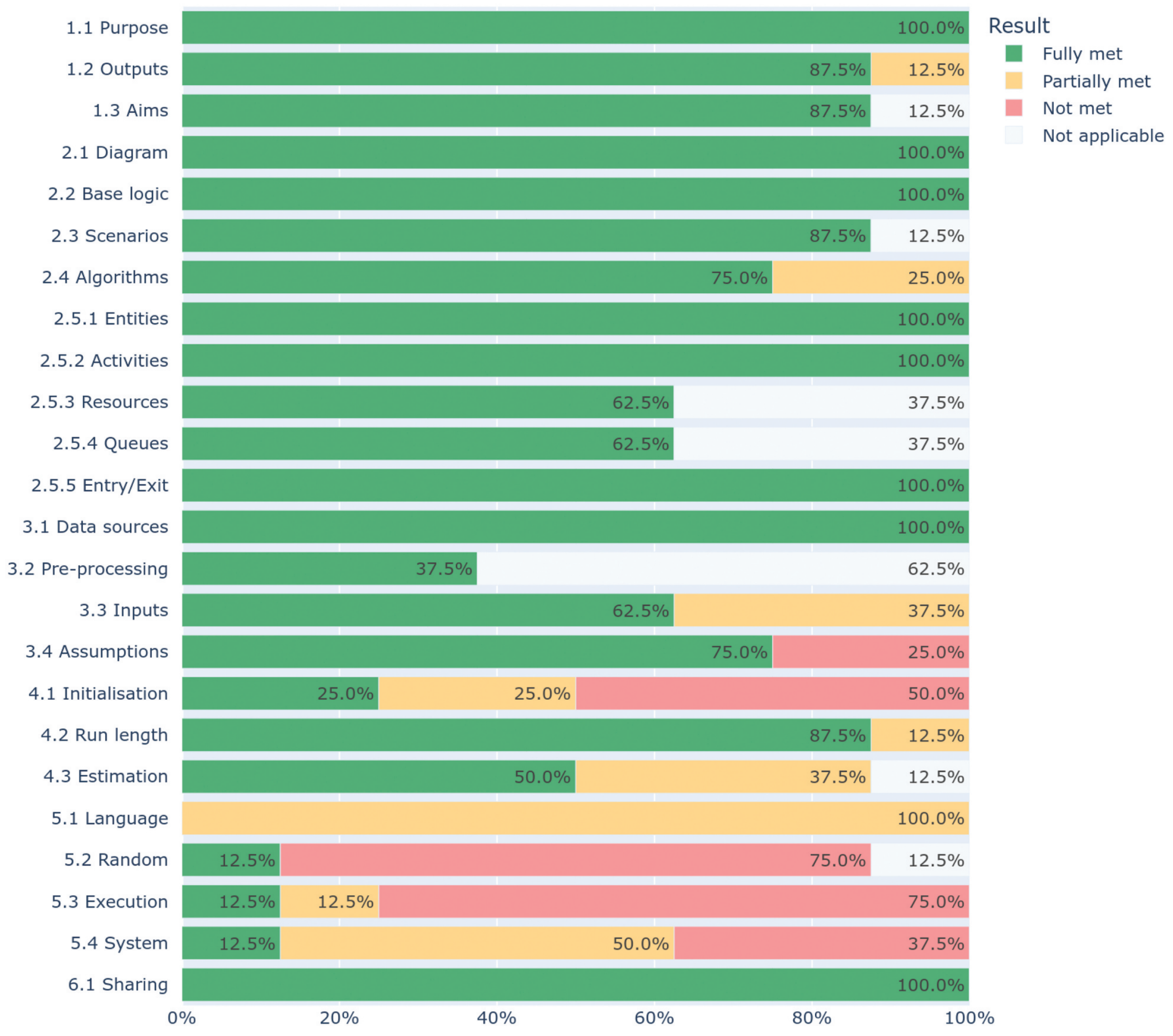

Figure 5. Of the eight healthcare DES studies evaluated, the proportion that met each item in the current STRESS-DES criteria (Monks et al., 2019). For a full description of each criterion, see Appendix A. Abbreviations: DES, discrete-event simulation; STRESS-DES, *Strengthening The Reporting of Empirical Simulation Studies-Discrete-Event Simulation*.

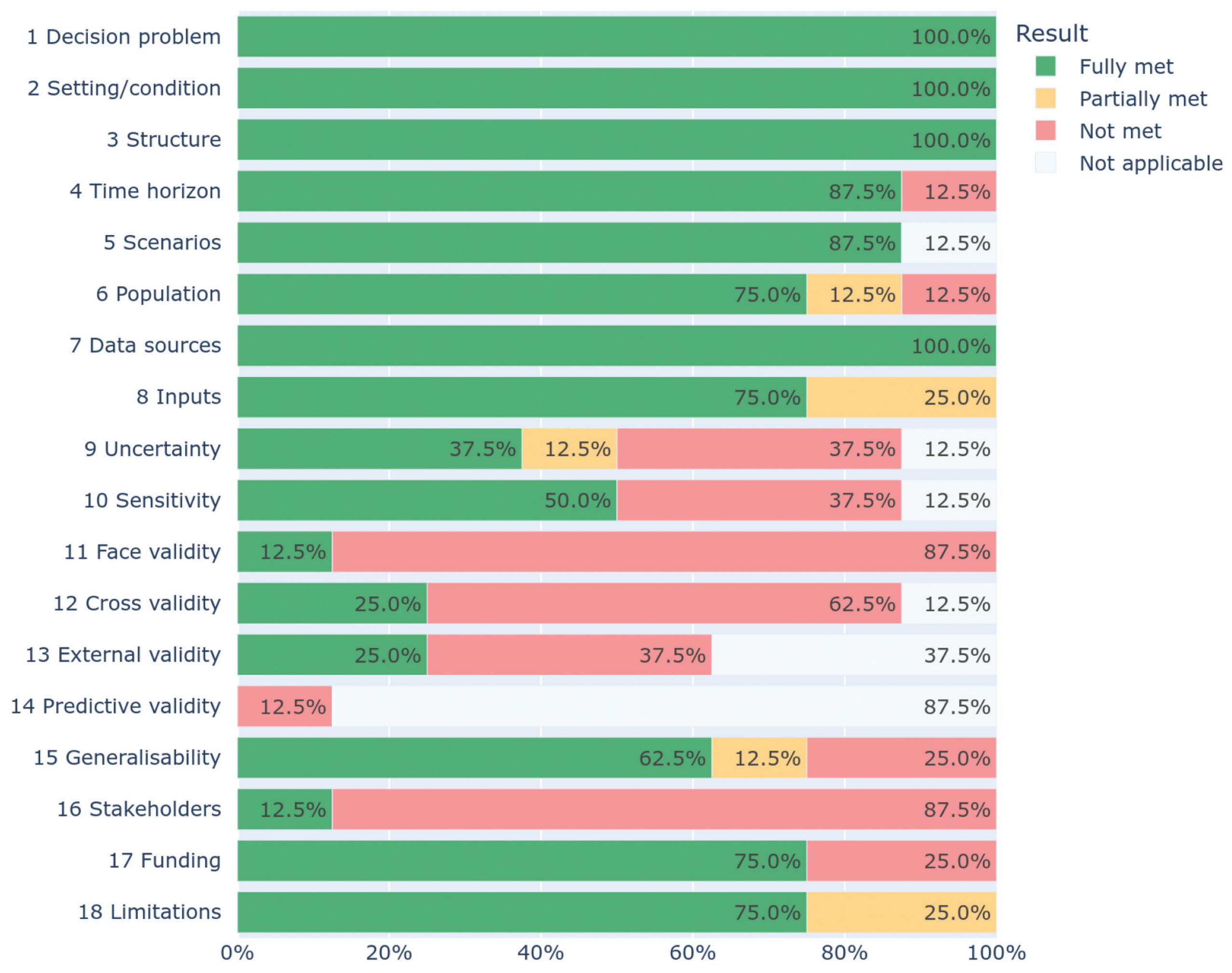

Figure 6. Of the eight healthcare DES studies evaluated, the proportion that met each criteria in the general reporting checklist for DES (Zhang et al., 2020). For full description of each criteria, see Appendix A. Abbreviations: DES, discrete-event simulation.

the requirement for pre-processing was unclear, and often marked as non-applicable. For the generic checklist, few studies addressed model uncertainties or performed sensitivity analyses, and validation was rarely discussed. Generalisability and stakeholder involvement were also seldom reported.

The proportion of applicable criteria met by each individual study is presented in Table 4. This ranged from 61% to 92% of the criteria per study for STRESS-DES and from 44% to 88% for the generic checklist. Although these values again showed no relationship with reproduction success (Table 4), certain elements were important to reproducibility. Complete and clear provision of input parameters (STRESS-DES 3.3, generic checklist 8) enabled verification and correction of parameters in the code. When code for the scenarios was missing, a clear description in the article was essential (STRESS-DES 2.3, generic checklist 5)—ideally specifying input parameters for all scenarios, as made explicit in STRESS-DES. Clear documentation of software and package versions (STRESS-DES 5.1) was helpful when dependency management was lacking. Run-time details (STRESS-DES 5.4) were also valuable, especially when studies had long execution times. When code for output calculations was missing, clear descriptions in the article were essential for understanding.

4. Recommendations

The facilitators and barriers encountered during the reproducibility assessments are presented as a series of recommendations, grouped into two themes: factors that primarily supported reproduction itself, ensuring the provided code could be run and yielded the same results as reported in the articles (Figure 7), and factors that facilitated troubleshooting, enhancing the code's adaptability for new contexts (Figure 8).

4.1. Recommendations to support reproducibility

4.1.1. Set-up

4.1.1.1. Share code with an open licence. Half of the models were initially unlicensed, effectively preventing the code from being downloaded or run. Guidance on selecting an appropriate licence can be found in

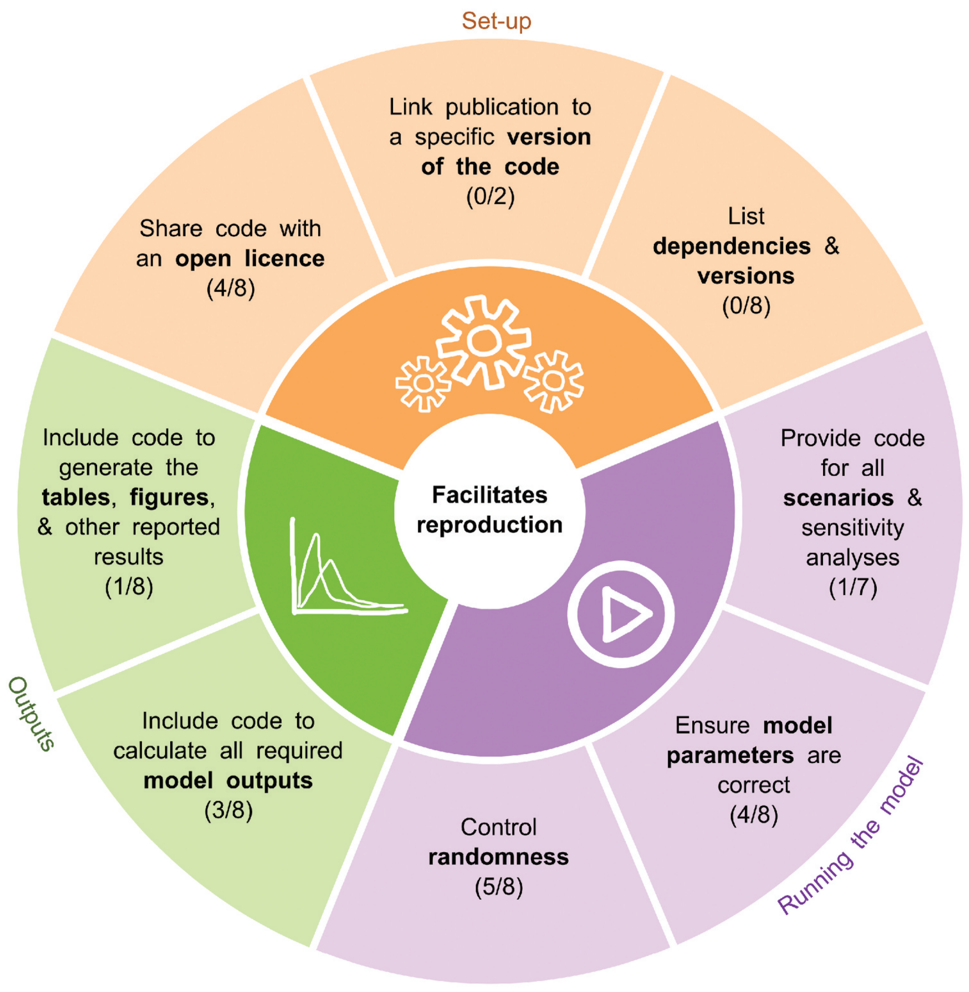
Set-up
Link publication to a specific version of the code (0/2)
List dependencies & versions (0/8)
Share code with an open licence (4/8)
Provide code for all scenarios & sensitivity analyses (1/7)
Facilitates reproduction
Include code to generate the tables, figures, & other reported results (1/8)
Ensure model parameters are correct (4/8)
Include code to calculate all required model outputs (3/8)
Control randomness (5/8)
Outputs
Running the model

Figure 7. Recommendations to support reproducibility. Below each recommendation, a count of studies that fully met it is provided. The total may fall below eight if the criteria were not applicable to a given study (e.g., If they didn't perform scenario analysis, or only provided one version of the code).

Morin et al. (2012) (Morin et al., 2012) and on https://choosealicense.com/.

4.1.1.2. Link publication to a specific version of the code. In one study, updates to the code after publication meant it was unclear which version produced the reported results. Journal articles can cite the exact version of code used to generate results if it is deposited in an open science archive. These archives, such as Figshare (Digital Science, 2025) or Zenodo (European Organization For Nuclear Research and OpenAIRE, 2013), provide a digital object identifier (DOI) for citation (Van Gulick & Curtin, 2024). Modified versions of code can be deposited at key stages during the publication process (e.g., submission, after revision, and acceptance) to obtain a new DOI. Code repositories can also contain a changelog (e.g., https://keepachangelog.com) that can help track changes between versions.

4.1.1.3. List all dependencies and versions. Only two studies provided a complete list of required packages, and none gave all versions. Environment managers like *Conda* (conda contributors, 2023) or *renv* (Ushey & Wickham, 2024) are recommended for creating isolated environments and tracking dependencies. This includes recording the specific versions used, as code may break with package updates.

4.1.2. Running the model

4.1.2.1. Provide code for all scenarios and sensitivity analyses. Six of the seven studies with scenarios omitted code for these, often just providing the base case. Without code, it was challenging and time-consuming to implement the scenarios—especially as descriptions of the scenarios were often ambiguous, with the required parameters sometimes unclear or not stated.

4.1.2.2. Ensure model parameters are correct. In half the studies, code parameters did not align with those in the articles, often causing large differences in the model outcomes. It is important to ensure that the correct parameters are provided in the code and clearly documented in the article (which can be used to verify or correct the code).

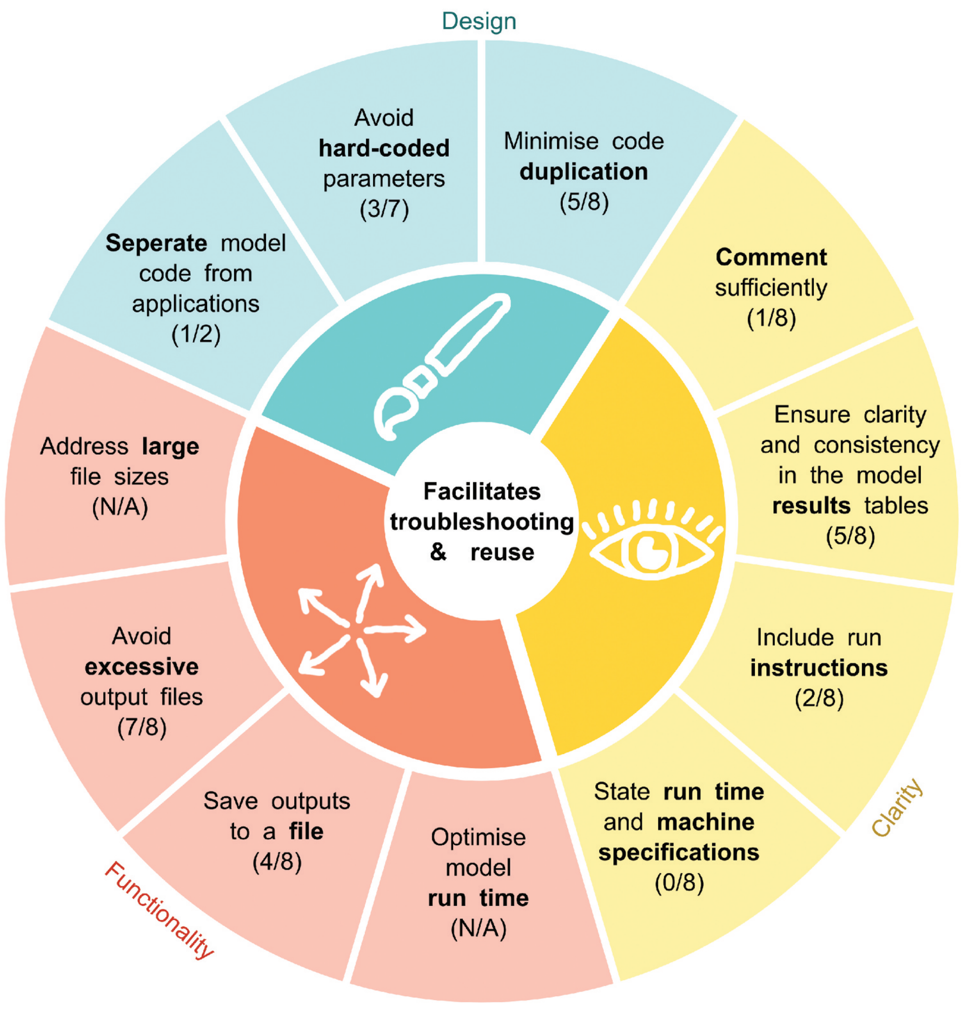
Design
Avoid **hard-coded** parameters (3/7)
Minimise code **duplication** (5/8)
Seperate model code from applications (1/2)
Comment sufficiently (1/8)
Ensure clarity and consistency in the model **results** tables (5/8)
Address **large** file sizes (N/A)
Facilitates troubleshooting & reuse
Include run **instructions** (2/8)
Avoid **excessive** output files (7/8)
Save outputs to a **file** (4/8)
Optimise model **run time** (N/A)
State **run time** and **machine specifications** (0/8)
Clarity
Functionality

Figure 8. Recommendations to support troubleshooting and reuse. Below each recommendation, a count of studies that fully met it is provided. The total may fall below eight if the criteria were not applicable to a given study (e.g., If they didn't have a web application, or didn't have scenarios to vary parameters). Some recommendations were marked as "N/A" where it was not felt appropriate or feasible to count/assess their inclusion. Abbreviations: N/A, not applicable.

4.1.2.3. Control randomness. Random seeds were not included in three studies. This made it impossible to determine whether differences between the results obtained from running the code and the results reported in the articles were due to variability in the simulation or issues in the code. Including a random seed is crucial for ensuring consistent results from stochastic models like DES. Tools like set.seed() in R (R Core Team, 2024) or *NumPy's* (Harris et al., 2020) SeedSequence() and default_rng() are recommended.

4.1.3. Outputs

4.1.3.1. Include code to calculate all required model outputs. In five studies, some outputs were not generated by the provided code or included in the model's output tables. These included basic measures (e.g., outcome counts), results from additional time points, and complex transformations of the provided outputs.

4.1.3.2. Include code to generate the tables, figures, and other reported results. Only one study provided code to generate all results in the article. Without this code, recreating outputs is time-consuming and challenging, requiring the identification of relevant results tables and columns, appropriate pre-processing steps, and the creation of the desired plots or tables.

4.2. Recommendations to support troubleshooting and reuse

4.2.1. Design

4.2.1.1. Separate model code from applications. One study embedded the model within the code for a web application (which itself did not produce the paper outputs). To execute it, the model had to be extracted from the app—so sharing standalone model scripts would be preferable.

4.2.1.2. Avoid hard-coded parameters. In four studies, parameters that varied between scenarios were hard-coded into model functions, making them difficult to adjust. Parameters should be defined as adjustable inputs, allowing the same code to run both base case and scenario analyses without modifications. This avoids duplicating files and ensures clear documentation of changes between scenarios.

4.2.1.3. Minimise code duplication.
Three studies contained repeated sections of code, such as for each scenario, or model warm-up and execution. This duplication reduced code readability and increased the risk of errors—for example, when modifying parameters that were defined in multiple locations, and so had to be updated in each instance. It can be mitigated by reusing code through functions or classes.

4.2.2. Clarity

4.2.2.1. Comment sufficiently.
Most studies were felt to have insufficient comments, hindering understanding of the parameters and code functionality when troubleshooting. Researchers should follow best practices, such as using docstrings for all functions and classes and adding concise yet informative comments elsewhere. Style guides, like PEP-8 (https://peps.python.org/pep-0008/) and PEP-257 (https://peps.python.org/pep-0257/) in Python, and the tidyverse style guide (https://style.tidyverse.org/) in R, offer recommendations on writing code documentation.

4.2.2.2. Ensure clarity and consistency in the model results tables.
In some studies, there was uncertainty regarding which tables, columns, or scenarios to use from the model results. For instance, one model produced two alternative spreadsheets with overlapping metrics but occasionally differing results, making it unclear which set of results was correct. In another case, it was not initially evident that a reported result was derived from a combination of columns, rather than from a single column. To address these issues, columns should be clearly labelled, results should be consistent across tables, and data dictionaries should be provided for complex outputs. Additionally, it should be explicitly clear which elements contribute to the results, ideally by including the relevant code.

4.2.2.3. Include run instructions.
Only two studies provided guidance. Clear instructions, including script run order and any necessary commands, are essential—especially in complex analyses with several scripts.

4.2.2.4. State run times and machine specifications.
Clearly stating the expected run times and specifications of the machines used helps users understand the resource demands and assess if their machine can handle the workload, as they may not have the capacity for long or memory-intensive runs. Times can be easily recorded such as by using the *time* module from Python's standard library (Python Core Team, 2024), or the Sys.time() function from base R (R Core Team, 2024). It can also be helpful to record the memory usage when running the model.

4.2.3. Functionality

4.2.3.1. Optimise model run time.
Long run times were a significant challenge in this research. Reducing run times is valuable both during initial testing, where quicker runs aid troubleshooting and debugging, and for full-scale runs, which may be impractical if requiring machines be left running continuously for extended periods. Options such as simplified configurations designed for model verification (e.g., reducing entities or replications) and parallel processing could significantly reduce run times or enable minimal test runs for troubleshooting and/or development.

4.2.3.2. Save outputs to a file.
In four studies, results were output to dataframes within the Python/R environment but not saved to files. Without saving, the entire model must be re-executed to modify analyses, which is inefficient, especially for long-running models. Saving outputs allows for easier manipulation and analysis in separate scripts.

4.2.3.3. Avoid excessive output files.
In one study, the model generated numerous files which were not used in analysis. Researchers should implement a run mode that limits output to essential files for analysis, while still allowing additional files for verification or debugging. Use a.gitignore file to prevent uploading unnecessary files to GitHub.

4.2.3.4. Address large file sizes.
Output files sometimes exceeded GitHub's 100MB threshold. This can be addressed by compressing files (e.g., csv.gz) or using GitHub Large File Storage—although that has limits under GitHub's free tier.

4.3. Key recommendations

From the complete set of recommendations, five were identified as having the greatest impact on the reproductions, based on both fundamental enablers of reproduction and observed patterns of what consistently created barriers.

(1) Share code with an open licence.
(2) Ensure model parameters are correct.
(3) Include code to calculate all required model outputs.
(4) Provide code for all scenarios and sensitivity analyses.
(5) Include code to generate the tables, figures, and other reported results.

The first recommendation, sharing code with an open licence, is fundamental because, without it, others cannot legally use, modify, or share the code, making reproduction otherwise impossible. The remaining recommendations (items 2 to 5) have been prioritised because they consistently posed significant barriers to achieving similar results to the original article, and/or required extensive time to troubleshoot and resolve.

5. Discussion

This research identified factors affecting reproducibility in healthcare DES models and proposed five key recommendations to address these issues. These recommendations were highlighted based on their critical impact, either because they were essential prerequisites for reproduction, caused significant discrepancies in reproduced results, or required extensive troubleshooting. In this discussion, these recommendations are reflected upon in the context of evidence from other studies and established best practices.

First, sharing models with an open licence. Half of the models in this research (50%, $n = 4/8$) initially lacked a licence, similar to findings in broader healthcare DES reviews (65%, $n = 26/47$) (Monks & Harper, 2025) and other fields, such as computational physics (71%, $n = 39/55$) (Stodden, Krafczyk, et al., 2018) and *Science* articles (34%, $n = 19/56$) (Stodden, Seiler, et al., 2018). Open licensing is recommended by initiatives like the NHS "Goldacre review" (Goldacre et al., 2022), journal badge programmes (ACM, 2020; Blohowiak et al., 2023; NISO Reproducibility Badging and Definitions Working Group, 2021) and the STARS framework (Monks et al., 2024). While private code can support internal reproducibility, it limits broader validation and reuse.

Second, maintaining an accurate and consistent record of the model parameters between the article and code was critical, as incorrect parameters often had a large impact on the observed outcomes. Third, sharing code for scenarios and sensitivity analyses—not just the base case—was vital, as reconstructing these without access to the original code was time-consuming and error-prone. These recommendations align with requirements from frameworks like STRESS-DES (Monks et al., 2019) and the generic reporting checklist (Zhang et al., 2020), as well as prior research identifying missing data, parameters, and scripts as major barriers to reproducibility across disciplines (Fišar et al., 2024; Konkol et al., 2019; Krafczyk et al., 2021; McCullough et al., 2006; Obels et al., 2020; Stodden, Krafczyk, et al., 2018; Stodden, Seiler, et al., 2018)

The fourth and fifth recommendations involve sharing code to calculate all required model outputs and to then generate the results presented in the article (e.g., tables, figures). A lack of visualisation code or reliance on proprietary visualisation tools has similarly hindered reproduction in computational physics (Krafczyk et al., 2021; Stodden, Krafczyk, et al., 2018). These recommendations, along with others, can help produce a reproducible analytical pipeline (RAP), wherein executing the code will run the model and produce all results from the paper. Tools like Python and R facilitate RAPs by generating figures and tables directly from code, avoiding the manual input required in drag-and-drop tools. RAPs have been highlighted as a priority by initiatives like the NHS RAP Community of Practice (NHS England RAP Community of Practice, 2024) and the UK Government Analysis Function RAP Strategy (Reproducible Analytical Pipelines RAP Team, 2022). More widely, initiatives such as the European Commission's exploration of reproducibility challenges in scientific research (Baker et al., 2020) and a European Union project focused on RAPs for monitoring plastic pollution (Community Research and Development Information Service CORDIS, 2024) reflect the growing international emphasis on reproducibility and RAPs in diverse contexts.

In addition to the five key recommendations, further recommendations—such as setting seeds, linking to specific code versions, listing dependencies, avoiding hard coding, adding comments, providing run instructions, minimising unnecessary outputs, clarifying results tables, and considering runtime and hardware needs—are also backed by the literature as important reproducibility factors across various fields (Eubank, 2016; Konkol et al., 2019; Krafczyk et al., 2021; McCullough et al., 2006; Obels et al., 2020; Stodden, Seiler, et al., 2018)

Some reproducibility issues are difficult to avoid, even with thorough preparation, as they may depend on future software updates or system differences. For example, one of the models used an older, now unsupported Python version (2.7.12), which may not work with some code editors/interpreters. Creating environments with an older language was straightforward in Python via *Conda* (conda contributors, 2023) but was more challenging and not successful in R in this research. R packages are intended to be forward compatible, although this cannot be guaranteed, with newer packages breaking the code in some studies (Konkol et al., 2019; Trisovic & Lau, 2022). Similarly, one study shared a web application, but this was no longer accessible remotely after its hosting site shut down. System-level code dependencies can also vary depending on the operating system used, and though they can be identified by testing the code on fresh builds, this may be beyond the scope of many projects.

5.1. Implications

5.1.1. For model developers

Model developers, including researchers and practitioners, should consider adopting the outlined recommendations to improve reproducibility, which can, in turn, help facilitate model reuse. The relevance of this is highlighted by the fact that three of the eight articles analysed in this article reused models described in prior studies:

- Kim et al. (2021) used a model described in Glover et al. (2018) and Thompson et al. (2018).
- Johnson et al. (2021) used a model described in Sadatsafavi et al. (2019).
- R. M. Wood et al., (2021) used a model described in R. M. Wood et al. (2020).

Common concerns about the skills (Hrynaszkiewicz et al., 2021; Van Den Eynden et al., 2016) or time (Borghi & Van Gulick, 2021; Cadwallader & Hrynaszkiewicz, 2022; Hrynaszkiewicz et al., 2021; Van Den Eynden et al., 2016) required to prepare before sharing code can be mitigated by making a few incremental improvements throughout the project. Four of the key recommendations in this research—using correct parameters, running scenarios, producing outputs, and creating tables and figures—will already have been performed by researchers throughout the research process, and so the primary action is simply to share these complete artefacts. The fifth recommendation, adding an open licence, is a simple but essential step, enabling others to download, run, and use the code. Other recommendations from this article may require a greater time investment, though this can be minimised if considered from the outset of the project. The more recommendations that are implemented, the greater the anticipated improvement in the reproducibility of the research.

5.1.2. For peer reviewers

Regardless of whether journal or conference policies are in place, reviewers can promote reproducibility by incorporating a "reproducibility review" (Table 5). This proposed review incorporates three of the five main recommendations from this research: ensuring open licences, availability of scenario and sensitivity analysis code, and providing result-processing scripts. These checks are straightforward to identify from any provided code or artefacts. Alternatively, they could be posed as peer review questions that an author could self-certify. The remaining two recommendations—verifying consistent parameters and confirming that the model generates all outputs—were excluded, as they require a much greater time investment to investigate, including potentially needing to run the code. This review is consistent with the requirements for openly licensed and complete materials from ACM's "Artefacts Evaluated" Artefacts Evaluated and IEEE's "Code Reviewed" (IEEE, 2024) badges. The review complements other existing suggestions, such as "open scholarship review" and "longevity review" (Table 5) from Monks & Harper (2025).

Table 5. Simple checklists to assist reviewers in assessing the openness, longevity, and reproducibility of DES models during peer review.

Review	Checklist	Proposer
Open scholarship review	• Used a reporting checklist? • Deposited model in a public archive? • Included ORCID in model metadata? • Shared model under an open licence? • Provided basic instructions to run and use model?	Monks & Harper (2025)
Longevity review	• Used a reporting checklist? • Explained how dependencies are managed?	Monks & Harper (2025)
Reproducibility review	• Shared model under an open licence? • Provided code for all scenarios and sensitivity analyses? • Provided code for tables, figures and other reported results?	Present study

Abbreviations: ORCID, Open Researcher and Contributor Identifier.

5.1.3. For reporting guidelines

Levels of adherence to the reporting guidelines were similar to those observed in prior studies. The mean proportion of fully met items on the generic reporting checklist (Zhang et al., 2020) in this research—either calculated out of applicable criteria (67.4%) or all criteria (61.1%)—was similar to previous evaluations of 211 healthcare DES articles (63.7%) (Zhang et al., 2020) and 18 radiotherapy pathway simulation articles (63.6%) (Robinson et al., 2023). Gaps in uncertainty assessment, sensitivity analysis, generalisability, and validity were common across the studies (Robinson et al., 2023; Zhang et al., 2020). Although they do not report on individual items, a previous study did assess adherence to STRESS-DES for 13 pharmacoeconomic models, finding higher mean adherence (80.4%) (Nwanosike, 2023) than in this study—as calculated out of applicable criteria (74.0%) or all criteria (68.2%).

In this research, adherence to the reporting checklists did not translate to reproducibility; as observed in prior studies with other checklists (McManus et al., 2019; Schwander et al., 2021; Stodden, Krafczyk, et al., 2018). This is unsurprising as the generic checklist emphasises model quality whilst STRESS-DES is focused on reporting all necessary information to facilitate replication of simulations. However, some items on these checklists—like describing input parameters, scenarios, and dependencies—were helpful in the reproductions. Focusing on STRESS-DES, this could be adapted to further facilitate reproducibility:

(1) **Clarification**: Provide suggestions to improve clarity in the reporting of model outputs, scenarios, and input parameters (e.g., using tables) —since these were often partially or ambiguously described.
(2) **Simplification**: Break down complex sections into smaller, more manageable items for easier reporting and review. For example, section 5.3

"Model execution" (Monks et al., 2019) covers several distinct topics in only one item: the event processing mechanism; priority rules for entities competing for resources; use of parallel, distributed, and cloud computing; time management algorithms; and details about high-level architecture (version, run-time, supporting documents).

(3) **Redundancy**: Reconsider sections on random number generation and event-processing mechanisms, as modern tools may not need this level of detail.

(4) **Sharing**: Recommend including the completed checklist as supplementary material, which would provide direct access to relevant information (which can be tricky to find in complex articles—up to 2 hours per evaluation in this research).

Completion and sharing of reporting checklists as supplementary material—though not guaranteed to facilitate reproducibility—certainly increase the transparency of reported models. Alongside the checklists in this research, others exist for different model types—for example, STRESS-ABS for agent-based simulation and STRESS-SD for system dynamics models (Monks et al., 2019), as well as the ODD protocol (*Overview, Design concepts and Details protocol for describing Individual and Agent-Based Models*) (Grimm et al., 2020). These checklists are complemented by TRACE (*TRAnsparent and Comprehensive Ecological modelling documentation*), a framework for documenting model development, testing, and analysis (Grimm et al., 2014)—supported by maintaing modelling notebooks through the development process (Ayllón et al., 2021).

5.1.4. For the STARS framework

The STARS framework was designed to support healthcare DES model reuse. As it was published in 2024 (Monks et al., 2024), it was not used in the reviewed studies. This research examined whether incidental adherence to its recommendations improved reproducibility; no clear link was found. This is similar to findings in other fields where practices like documentation and archiving do not guarantee reproducibility (Henderson et al., 2024; Stodden, Krafczyk, et al., 2018). This suggests that, while these practices are valuable, other factors contribute to reproducibility challenges. Modifications to STARS that may better facilitate reproducibility include requiring a complete set of materials (i.e., parameters, scenarios, results processing), which aligns with key recommendations from this study and existing badge expectations. Additionally, incorporating guidelines on runtime/hardware, as well as code design (e.g., seeds, running scenarios programmatically, docstrings/comments, output saving).

5.2. Strengths and limitations

This study is the first to systematically assess the computational reproducibility of healthcare DES models, providing insights that are broadly relevant to other contexts and modelling approaches. A key strength is the transparency of the research, with all code openly licensed, available on GitHub, archived on Zenodo (Appendix C), and supported by a pre-registered protocol (Heather et al., 2024). Additionally, a complementary website summarises the results and provides further details (https://pythonhealthdatascience.github.io/stars_wp1_summary/) (Heather et al., 2025a). The study benefits from a diverse selection of eight studies, although they are biased toward models with shared licensing and resources, which are uncommon in healthcare DES (Monks & Harper, 2025). The reproducibility process was time-intensive, requiring up to 28 hours per study, which may not reflect typical researcher resources—but this time investment was crucial to understanding the factors influencing reproducibility, requiring a close examination of the troubleshooting for each case.

Reproductions were conducted by a single researcher, occasionally consulting others when troubleshooting. If the aim of this research had been to determine the exact level of reproducibility, it would have been necessary for multiple researchers to independently attempt to reproduce the studies, to ensure reliable results. A prior study of psychology articles found low inter-rater reliability in such assessments (75% agreement on the executability of R scripts and 56% agreement on reproducibility) (Obels et al., 2020). However, it was felt suitable for a single researcher to conduct the reproductions, given the study's focus on identifying factors impacting reproducibility, rather than definitive reproduction rates. Barriers and facilitators encountered are expected to be relatively consistent between researchers—as evidenced by the complementary recommendations from reproductions in other fields. Determining successful reproduction involved subjective judgement—acknowledged in the protocol (Heather et al., 2024)—so decisions were made by consensus between at least two team members.

The study focused on models implemented in Python and R, excluding those developed using commercial DES software. Although most DES studies use commercial off-the-shelf software, the majority (62%) of open DES models use FOSS (Monks & Harper, 2025). Our expectation is that reproduction of results generated by commercial software may present some differences—particularly in troubleshooting the unique nuances of commercial DES software as well as access to a compatible and suitably licensed version. However, we also expect the top five study recommendations to remain broadly applicable. For example, a simulation model will still need a licence (CC-BY may be best suited to a simulation file); and correct parameters will still be required.

While absolute and percentage differences between the reproductions and original results were considered, these could vary significantly based on scale. For example, percentage differences are greater for smaller values (e.g., 0.1, 0.2) than for larger values (e. g., 10, 15), even if the latter are more practically significant. The nature of the metrics analysed could also impact outcomes. For example, the costs and Quality-Adjusted Life Years from a health economic model in this study appeared similar, but even small differences led to substantial variations in derived calculations (e. g., Incremental Cost–Effectiveness Ratio).

Evaluations were conducted by one researcher, with a second checking uncertain or unmet criteria, and all evaluations were revisited side-by-side for consistency. The potential for mistakes in the evaluations highlights the value of attaching completed reporting guidelines to articles to make technical details more accessible—particularly for longer papers or when information is spread across multiple appendices or prior publications.

6. Conclusion

This study identifies critical factors impacting the reproducibility of healthcare DES models. It offers actionable recommendations to improve the reproducibility of shared models, emphasising five key recommendations: adopting an open licence, ensuring model parameters are correct, ensuring the model outputs all required metrics, including code to run all scenarios and/or sensitivity analyses, and including code to generate all tables, figures, and other reported results. While adherence to guidelines like STRESS-DES and STARS did not translate to successful reproductions, suggested improvements to these frameworks could help facilitate reproducibility. Further work could include refining these frameworks and providing practical examples of RAP in DES that meet these criteria, as examples could help address concerns about the skills and time investment needed for implementation.

Acknowledgments

We would like to thank the authors who made their code available under open licences, facilitating our research. We are especially grateful to the following authors for their helpful communication during the project: Ivan Hernandez, Tze Ping Loh, Chun Yee Lim, Lois Kim, Kate Johnson, Richard Wood, Mohd Shoaib, and Anastasia Anagnostou. The circular design of Figures 7 and 8 was inspired by Gomes et al. (2022).

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the Medical Research Council under grant number [MR/Z503915/1].

ORCID

Amy Heather http://orcid.org/0000-0002-6596-3479
Thomas Monks http://orcid.org/0000-0003-2631-4481
Alison Harper http://orcid.org/0000-0001-5274-5037
Navonil Mustafee http://orcid.org/0000-0002-2204-8924
Andrew Mayne http://orcid.org/0000-0003-1263-2286

Data availability statement

The records supporting the findings of this study—including code, models, results, and other relevant materials—are synthesised and available at https://pythonhealthdatascience.github.io/stars_wp1_summary/ archived on Zenodo (Heather et al., 2025a). Links to the individual repositories for the reproduced studies and the original studies are provided in Appendix C.

References

ACM SIGSIM. (2025). Reproducibility and artifact evaluation. *ACM SIGSIM - PADS 2025*. https://sigsim.acm.org/conf/pads/2025/blog/artifact-evaluation/

Allaire, J. J., Teague, C., Scheidegger, C., Xie, Y., & Dervieux, C. (2024). Quarto. *Zenodo*. https://doi.org/10.5281/zenodo.5960048

Allen, M., Bhanji, A., Willemsen, J., Dudfield, S., Logan, S., & Monks, T. (2020, August). A simulation modelling toolkit for organising outpatient dialysis services during the COVID-19 pandemic. *PLOS ONE*, *15*(8), 0 e0237628. https://doi.org/10.1371/journal.pone.0237628

Alston, J. M., & Rick, J. A. (2021). A beginner's guide to conducting reproducible research. *Bulletin of the Ecological Society of America*, *102*(2), 0 e01801. https://doi.org/10.1002/bes2.1801

Anagnostou, A. (2022, January). *CHARM: Dynamic hospital ward management*. Brunel University London. https://doi.org/10.17633/rd.brunel.18517892.v1

Anagnostou, A., Groen, D., Taylor, S. J. E., Suleimenova, D., Abubakar, N., Saha, A., Mintram, K., Ghorbani, M., Daroge, H., Islam, T., Xue, Y., Okine, E., & Anokye, N. (2022, December). FACS-CHARM: A hybrid agent-based and discrete-event simulation approach for COVID-19 management at regional level. In *2022 Winter Simulation Conference (WSC)* (pp. 1223–1234). https://doi.org/10.1109/WSC57314.2022.10015462

Andersen, M. Z., Fonnes, S., & Rosenberg, J. (2021, June). Time from submission to publication varied widely for biomedical journals: A systematic review. *Current Medical Research and Opinion*, *37*(6), 985–993. https://doi.org/10.1080/03007995.2021.1905622

Association for Computing Machinery. (2020, August). *Artifact review and badging (Version 1.1)*. Association for Computing Machinery. https://www.acm.org/publications/policies/artifact-review-and-badging-current

Association for Psychological Science. (2024, September). *Psychological science submission guidelines*. APS. https://www.psychologicalscience.org/publications/psychological_science/ps-submissions

Ayllón, D., Railsback, S. F., Gallagher, C., Augusiak, J., Baveco, H., Berger, U., Charles, S., Martin, R., Focks, A., Galic, N., van Loon Chun Liu, E. E., Nabe-Nielsen, J., Cyril Piou, J. G. P., Preuss, T. G., Radchuk, V., Schmolke, A., Stadnicka-Michalak, J., Thorbek, P., & Grimm, V. (2021, February). Keeping modelling notebooks with TRACE: Good for you and good for environmental research and management support. *Environmental Modelling & Software*, *136*, 104932. https://doi.org/10.1016/j.envsoft.2020.104932

Baker, L., Cristea, I. A., Errington, T. M., Jaśko, K., Lusoli, W., MacCallum, C. J., Parry, V., Pérignon, C., Šimko, T., & Winchester, C. (2020, December). *Reproducibility of scientific results in the EU: Scoping report*. European commission. https://www.ouvrirlascience.fr/wp-content/uploads/2020/12/Reproducibility-of-scientific-results-in-the-EU.pdf

Benureau, F. C. Y., & Rougier, N. P. (2018, January). Re-run, repeat, reproduce, reuse replicate: Transforming code into scientific contributions. *Frontiers in Neuroinformatics*, *11*. https://doi.org/10.3389/fninf.2017.00069

Blohowiak, B. B., Cohoon, J., de Wit, L., Eich, E., Farach, F. J., Hasselman, F., Holcombe, A. O., Humphreys, M., Lewis, M., & Nosek, B. A. (2023, September). *Badges to acknowledge open practices*. Osf. https://osf.io/tvyxz/

Borghi, J. A., & Van Gulick, A. E. (2021, May). Data management and sharing: Practices and perceptions of psychology researchers. *PLOS ONE*, *16*(5), e0252047. https://doi.org/10.1371/journal.pone.0252047

Cadwallader, L., & Hrynaszkiewicz, I. (2022, August). A survey of researchers' code sharing and code reuse practices, and assessment of interactive notebook prototypes. *PeerJ*, *10*, e13933. https://doi.org/10.7717/peerj.13933

Cadwallader, L., Papin, J. A., Gabhann, F. M., & Kirk, R. (2021, March). Collaborating with our community to increase code sharing. *PLOS Computational Biology*, *17*(3), e1008867. https://doi.org/10.1371/journal.pcbi.1008867

Centre for Open Science. (2024). *Top guidelines*. https://www.cos.io/initiatives/top-guidelines

Community Research and Development Information Service. (2024, July). *European quality controlled harmonization assuring reproducible monitoring and assessment of plastic pollution*. European Commission. https://doi.org/10.3030/101003805

conda contributors. (2023, July). Conda: A system-level, binary package and environment manager running on all major operating systems and platforms. *Conda*. https://docs.conda.io/projects/conda/

Digital Science. (2025). Figshare - credit for all your research. *Figshare*. https://figshare.com/

Eubank, N. (2016, April). Lessons from a decade of replications at the Quarterly Journal of Political Science. *PS, Political Science & Politics*, *49*(2), 273–276. https://doi.org/10.1017/S1049096516000196

European Organization For Nuclear Research and OpenAIRE. (2013). Zenodo. *CERN*. https://doi.org/10.25495/7gxk-rd71

Fišar, M., Greiner, B., Huber, C., Katok, E., & Ozkes, A. I. (2024, March). Reproducibility in management science. *Management Science*, *70*(3), 1343–1356. https://doi.org/10.1287/mnsc.2023.03556

Galiani, S., Gertler, P., & Romero, M. (2017, July). Incentives for replication in economics. *National Bureau of Economic Research*. https://doi.org/10.3386/w23576

Gentleman, R., & Lang, D. T. (2007, March). Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics*, *16*(1), 0 1–23. https://doi.org/10.1198/106186007X178663

GitHub. (2024). GitHub Pages. *GitHub*. https://pages.github.com/

Glover, M. J., Jones, E., Masconi, K. L., Sweeting, M. J., Thompson, S. G., Powell, J. T., Ulug, P., & Bown, M. J. (2018, May). Discrete event simulation for decision modeling in health care: Lessons from abdominal aortic aneurysm screening. *Medical Decision Making*, *38*(4), 439–451. https://doi.org/10.1177/0272989X17753380

Goldacre, B., Morley, J., & Hamilton, N. (2022, April). Better, broader, safer: Using health data for research and analysis. *Department of Health and Social Care*. https://www.gov.uk/government/publications/better-broader-safer-using-health-data-for-research-and-analysis/better-broader-safer-using-health-data-for-research-and-analysis

Gomes, D. G. E., Pottier, P., Crystal-Ornelas, R., Hudgins, E. J., Foroughirad, V., Sánchez-Reyes, L. L., Turba, R., Martinez, P. A., Moreau, D., Bertram, M. G., Smout, C. A., & Gaynor, K. M. (2022, November). Why don't we share data and code? Perceived barriers and benefits to public archiving practices. *Proceedings of the Royal Society B: Biological Sciences*, *289*(1987), 20221113. https://doi.org/10.1098/rspb.2022.1113

Grimm, V., Augusiak, J., Focks, A., Frank, B. M., Gabsi, F., Johnston, A. S. A., Liu, C., Martin, B. T., Meli, M., Radchuk, V., Thorbek, P., & Railsback, S. F. (2014, May). Towards better modelling and decision support: Documenting model development, testing, and analysis using TRACE. *Ecological Modelling*, *280*, 129–139. https://doi.org/10.1016/j.ecolmodel.2014.01.018

Grimm, V., Railsback, S. F., Vincenot, C. E., Berger, U., Gallagher, C., DeAngelis, D. L., Edmonds, B., Ge, J., Giske, J., Groeneveld, J., Johnston, A. S. A., Milles, A., Jacob nabe-Nielsen, J. G. P., Radchuk, V., Rohwäder, M.-S., Stillman, R. A., Thiele, J. C., Ayllón, D., & Ayllón, D. (2020). The ODD protocol for describing agent-based and other simulation models: A second update to improve clarity, replication, and structural realism. *Journal of Artificial Societies and Social Simulation*, *23*(2), 7. https://doi.org/10.18564/jasss.4259

Hardwicke, T. E., & Vazire, S. (2024, July). Transparency is now the default at Psychological Science. *Psychological Science*, *35*(7), 708–711. https://doi.org/10.1177/09567976231221573

Harper, A., Mustafee, N., & Yearworth, M. (2021, February). Facets of trust in simulation studies. *European Journal of Operational Research*, *289*(1), 197–213. https://doi.org/10.1016/j.ejor.2020.06.043

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Del Río, J. F., Wiebe, M.Peterson, P. … Oliphant, T. E. (2020, September). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Heather, A., Monks, T., & Harper, A. (2025, January). Johnson et al. 2021 computational reproducibility assessment. version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14622070

Heather, A., Monks, T., & Harper, A. (2025a, January). Computational reproducibility assessments: Summary. Version 1.0.2. *Zenodo*. https://doi.org/10.5281/zenodo.14717263

Heather, A., Monks, T., & Harper, A. (2025b, January). Shoaib and Ramamohan 2022 computational reproducibility assessment. Version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14622306

Heather, A., Monks, T., & Harper, A. (2025, January). Anagnostou et al. 2022 computational reproducibility assessment. version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14622364

Heather, A., Monks, T., & Harper, A. (2025, January). Hernandez et al. 2015 computational reproducibility assessment. Version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.13971045

Heather, A., Monks, T., & Harper, A. (2025, January). Huang et al. 2019 computational reproducibility assessment. Version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14621728

Heather, A., Monks, T., & Harper, A. (2025, January). Kim et al. 2021 computational reproducibility assessment. Version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14621982

Heather, A., Monks, T., & Harper, A. (2025c, January). Wood, et al. 2021 computational reproducibility assessment. Version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14622158

Heather, A., Monks, T. & Harper, A., Lim, (2025, January). 2020 computational reproducibility assessment. version 1.0.0. *Zenodo*. https://doi.org/10.5281/zenodo.14621885

Heather, A., Monks, T., Harper, A., Mustafee, N., & Mayne, A. (2024, June). Protocol for assessing the computational reproducibility of discrete-event simulation models on STARS. *Zenodo*. https://doi.org/10.5281/zenodo.12179846

Henderson, A. S., Hickson, R. I., Furlong, M., McBryde, E. S., & Meehan, M. T. (2024, March). Reproducibility of COVID-era infectious disease models. *Epidemics*, *46*, 100743. https://doi.org/10.1016/j.epidem.2024.100743

Hernandez, I., Ramirez-Marquez, J. E., Starr, D., McKay, R., Guthartz, S., Motherwell, M., & Barcellona, J. (2015, May). Optimal staffing strategies for points of dispensing. *Computers & Industrial Engineering*, *83*, 172–183. https://doi.org/10.1016/j.cie.2015.02.015

Hrynaszkiewicz, I., Harney, J., & Cadwallader, L. (2021, April). A survey of code sharing practice and policy in computational biology. *OSF*. https://doi.org/10.31219/osf.io/f73a6

Huang, S., Maingard, J., Kok, H. K., Barras, C. D., Thijs, V., Chandra, R. V., Brooks, D. M., & Asadi, H. (2019, June). Optimizing resources for endovascular clot retrieval for acute ischemic stroke, a discrete event simulation. *Frontiers in Neurology*, *10*. https://doi.org/10.3389/fneur.2019.00653

Hunter, J. D. (2007, May). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Institute of Electrical and Electronics Engineers. (2024). About content in IEEE Xplore. *IEEE Explore*. https://ieeexplore.ieee.org/Xplorehelp/overview-of-ieee-xplore/about-content

Janssen, M. A., Pritchard, C., & Lee, A. (2020, December). On code sharing and model documentation of published individual and agent-based models. *Environmental Modelling & Software*, *134*, 104873. https://doi.org/10.1016/j.envsoft.2020.104873

Johnson, K. M., Sadatsafavi, M., Adibi, A., Lynd, L., Harrison, M., Tavakoli, H., Sin, D. D., & Bryan, S. (2021, March). Cost effectiveness of case detection strategies for the early detection of COPD. *Applied Health Economics and Health Policy*, *190*(2), 203–215. https://doi.org/10.1007/s40258-020-00616-2

Kim, L. G., Sweeting, M. J., Armer, M., Jacomelli, J., Nasim, A., & Harrison, S. C. (2021, June). Modelling the impact of changes to abdominal aortic aneurysm screening and treatment services in England during the COVID-19 pandemic. *PLOS ONE*, *16*(6), e0253327. https://doi.org/10.1371/journal.pone.0253327

Konkol, M., Kray, C., & Pfeiffer, M. (2019, February). Computational reproducibility in geoscientific papers: Insights from a series of studies with geoscientists and a reproduction study. *International Journal of Geographical Information Science*, *33*(2), 408–429. https://doi.org/10.1080/13658816.2018.1508687

Krafczyk, M. S., Shi, A., Bhaskar, A., Marinov, D., & Stodden, V. (2021, March). Learning from reproducing computational results: Introducing three principles and the reproduction package. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *379*(2197), 20200069. https://doi.org/10.1098/rsta.2020.0069

Laurinavichyute, A., Yadav, H., & Vasishth, S. (2022, August). Share the code, not just the data: A case study of the reproducibility of articles published in the Journal of Memory and Language under the open data policy. *Journal of Memory and Language*, *125*, 104332. https://doi.org/10.1016/j.jml.2022.104332

Lim, C. Y., Bohn, M. K., Lippi, G., Ferrari, M., Loh, T. P., Yuen, K.-Y., Adeli, K., & Horvath, A. R. (2020, December). Staff rostering, split team arrangement, social distancing (physical distancing) and use of personal protective equipment to minimize risk of workplace transmission during the COVID-19 pandemic: A simulation study. *Clinical Biochemistry*, *86*, 15–22. https://doi.org/10.1016/j.clinbiochem.2020.09.003

Loder, E., Macdonald, H., Bloom, T., & Abbasi, K. (2024, March). Mandatory data and code sharing for research published by the BMJ. *BMJ*, *384*, q324. https://doi.org/10.1136/bmj.q324

Management Science. (2019, June). Code and data disclosure policy. *Management science*. https://pubsonline.informs.org/page/mnsc/code-and-data-disclosure-policy

McCullough, B. D., McGeary, K. A., & Harrison, T. D. (2006). Lessons from the JMCB archive. *Journal of Money, Credit, and Banking*, *38*(4), 1093–1107. https://www.jstor.org/stable/3838995

McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.). *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

McManus, E., Turner, D., Gray, E., Khawar, H., Okoli, T., & Sach, T. (2019, September). Barriers and facilitators to model replication within health economics. *Value in Health*, *22*(9), 1018–1025. https://doi.org/10.1016/j.jval.2019.04.1928

Merkel, D. (2014, March). Docker: Lightweight Linux containers for consistent development and deployment. *Linux Journal*, *20140*(239), 2: 2.

Monks, T., Currie, C. S. M., Onggo, B. S., Robinson, S., Kunc, M., & Taylor, S. J. E. (2019, January). Strengthening the reporting of empirical simulation studies: Introducing the STRESS guidelines. *Journal of Simulation*, *13*(1), 55–67. https://doi.org/10.1080/17477778.2018.1442155

Monks, T., & Harper, A. (2025). Computer model and code sharing practices in healthcare discrete-event simulation: A systematic scoping review. *Journal of Simulation*, *19*(1), 108–123. https://doi.org/10.1080/17477778.2023.2260772

Monks, T., Harper, A., & Mustafee, N. (2024). Towards sharing tools and artefacts for reusable simulations in healthcare. *Journal of Simulation*, 1–20. https://doi.org/10.1080/17477778.2024.2347882

Morin, A., Urban, J., & Sliz, P. (2012, July). A quick guide to software licensing for the scientist-programmer. *PLOS Computational Biology*, *8*(7), e1002598. https://doi.org/10.1371/journal.pcbi.1002598

Nature Computational Science. (2023, November). Code sharing in the spotlight. *Nature Computational Science*, *3*(11), 907–907. https://doi.org/10.1038/s43588-023-00566-4

Nature Human Behaviour. (2024, January). Promoting reproduction and replication at scale. *Nature Human Behaviour*, *8*(1), 1–1. https://doi.org/10.1038/s41562-024-01818-7

Nature Machine Intelligence. (2022, October). Revisiting code reusability. *Nature Machine Intelligence*, *4*(10), 801–801. https://doi.org/10.1038/s42256-022-00554-9

NHS England RAP Community of Practice. (2024, November). Reproducible analytical pipelines (RAP). *Rap community of practice*. https://nhsdigital.github.io/rap-community-of-practice/

NISO Reproducibility Badging and Definitions Working Group. (2021, January). Reproducibility badging and definitions. *National Information Standards Organization (NISO)*. https://doi.org/10.3789/niso-rp-31-2021

Nwanosike, E. M. (2023, September). Direct oral anticoagulants (DOACs) use in patients with renal insufficiency and obesity. *University of Huddersfield*. https://pure.hud.ac.uk/ws/portalfiles/portal/83727839/Final_thesis_Nwanosike.pdf

Obels, P., Lakens, D., Coles, N. A., Gottfried, J., & Green, S. A. (2020, June). Analysis of open data and computational reproducibility in registered reports in psychology. *Advances in Methods and Practices in Psychological Science*, *3*(2), 229–237. https://doi.org/10.1177/2515245920918872

The pandas development team. (2024, September). Pandas-dev/pandas:pandas. *Zenodo*. https://doi.org/10.5281/zenodo.3509134

Philip, A. M., Prasannavenkatesan, S., & Mustafee, N. (2023, June). Simulation modelling of hospital outpatient department: A review of the literature and bibliometric analysis. *Simulation*, *99*(6), 573–597. https://doi.org/10.1177/00375497221139282

Plotly Technologies Inc. (2015). Collaborative data science. *Plotly Technologies Inc*. https://plot.ly

Python Core Team. (2024). Python. *Python Software Foundation*. https://www.python.org/

R Core Team. (2024). R: A language and environment for statistical computing. *R Foundation for Statistical Computing*. https://www.R-project.org/

Reproducible Analytical Pipelines. (2022, June). Reproducible analytical pipelines (RAP) strategy. *Government analysis function*. https://analysisfunction.civilservice.gov.uk/policy-store/reproducible-analytical-pipelines-strategy/

Robinson, A., Asaduzzaman, M., Jena, R., & Naemi, R. (2023). Simulation as a tool to model potential workflow enhancements in radiotherapy treatment pathways – a systematic review. *Journal of Applied Clinical Medical Physics*, *24*(10), e14132. https://doi.org/10.1002/acm2.14132

Roy, S., Venkatesan, S. P., & Goh, M. (2021, October). Healthcare services: A systematic review of patient-centric logistics issues using simulation. *The Journal of the Operational Research Society*, *72*(10), 2342–2364. https://doi.org/10.1080/01605682.2020.1790306

Sadatsafavi, M., Ghanbarian, S., Adibi, A., Kate Johnson, J. M. F., Flanagan, W., Bryan, S., & Sin, D. (2019, February). Development and validation of the evaluation platform in COPD (EPIC): A population-based outcomes model of COPD for Canada. *Medical Decision Making*, *39*(2), 152–167. https://doi.org/10.1177/0272989X18824098

Salleh, S., Thokala, P., Brennan, A., Hughes, R., & Booth, A. (2017, September). Simulation modelling in healthcare: An umbrella review of systematic literature reviews. *Pharmacoeconomics*, *35*(9), 937–949. https://doi.org/10.1007/s40273-017-0523-3

Salmon, A., Rachuba, S., Briscoe, S., & Pitt, M. (2018, December). A structured literature review of simulation modelling applied to emergency departments: Current patterns and emerging trends. *Operations Research for Health Care*, *19*, 1–13. https://doi.org/10.1016/j.orhc.2018.01.001

Samuel, S., & Mietchen, D. (2024, January). Computational reproducibility of Jupyter notebooks from biomedical publications. *GigaScience*, *13*, giad113. https://doi.org/10.1093/gigascience/giad113

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013, October). Ten simple rules for reproducible computational research. *PLOS Computational Biology*, *9*(10), e1003285. https://doi.org/10.1371/journal.pcbi.1003285

Schwander, B., Nuijten, M., Evers, S., & Hiligsmann, M. (2021). Replication of published health economic obesity models: Assessment of facilitators, hurdles and reproduction success. *Pharmacoeconomics*, *39*(4), 433–446. https://doi.org/10.1007/s40273-021-01008-7

Shoaib, M., & Ramamohan, V. (2022, March). Simulation modeling and analysis of primary health center operations. *Simulation*, *98*(3), 183–208. https://doi.org/10.1177/00375497211030931

Stagge, J. H., Rosenberg, D. E., Abdallah, A. M., Akbar, H., Attallah, N. A., & James, R. (2019, February). Assessing data availability and research reproducibility in hydrology and water resources. *Scientific Data*, *6*(1), 190030. https://doi.org/10.1038/sdata.2019.30

Stockemer, D., Koehler, S., & Lentz, T. (2018, October). Data access, transparency, and replication: New insights from the political behavior literature. *PS, Political Science & Politics*, *51*(4), 799–803. https://doi.org/10.1017/S1049096518000926

Stodden, V., Krafczyk, M. S., & Bhaskar, A. (2018, June). Enabling the verification of computational results: An empirical evaluation of computational reproducibility. In *Proceedings of the First International Workshop on Practical Reproducible Evaluation of Computer Systems, P-RECS'18* (pp. pages 1–5). Association for Computing Machinery. https://doi.org/10.1145/3214239.3214242

Stodden, V., Seiler, J., & Ma, Z. (2018, March). An empirical analysis of journal policy effectiveness for computational reproducibility. *Proceedings of the National Academy of Sciences*, *115*(11), 2584–2589. https://doi.org/10.1073/pnas.1708290115

Team SimPy. (2024). Simpy: Discrete event simulation for Python. *Team Simpy*. https://simpy.readthedocs.io/

Thompson, S. G., Bown, M. J., Glover, M. J., Jones, E., Masconi, K. L., Michaels, J. A., Powell, J. T., Ulug, P., & Sweeting, M. J. (2018, August). Screening women aged 65 years or over for abdominal aortic aneurysm: A modelling study and health economic evaluation. *Health Technology Assessment*, *22*(43), 1–142. https://doi.org/10.3310/hta22430

Trisovic, A., & Lau, M. K. (2022, February). Thomas Pasquier, and Mercè Crosas. A large-scale study on research code quality and execution. *Scientific Data*, *9* (1), 60. https://doi.org/10.1038/s41597-022-01143-6

Ucar, I., Smeets, B., & Simmer, A. A. (2019, July). Discrete-event simulation for R. *Journal of Statistical Software*, *90*(2), 1–30. https://doi.org/10.18637/jss.v090.i02

Ushey, K., & Wickham, H. (2024). Renv: Project environments. *Renv*. https://rstudio.github.io/renv/

Van Den Eynden, V., Knight, G., Vlad, A., Radler, B., Tenopir, C., Leon, D., Manista, F., Whitworth, J., & Corti, L. (2016). Survey of Wellcome researchers and their attitudes to open research. *Wellcome Trust*. https://doi.org/10.6084/m9.figshare.4055448.v1

Van Der Ham, R. (2018, July). Salabim: Discrete event simulation and animation in Python. *Journal of Open Source Software*, *3*(27), 767. https://doi.org/10.21105/joss.00767

Van Gulick, A., & Curtin, L. (2024, October). Resources for selecting and using generalist repositories to support NIH data sharing. *Zenodo*. https://doi.org/10.5281/ZENODO.13987727

Wilsdorf, P., Wolpers, A., Hilton, J., Haack, F., & Uhrmacher, A. (2023, February). Automatic reuse, adaptation, and execution of simulation experiments via provenance patterns. *ACM Transactions on Modeling and Computer Simulation*, *33*(1–2), 1–27. https://doi.org/10.1145/3564928

Winter, S., Timperley, C. S., Hermann, B., Cito, J., Bell, J., Hilton, M., & Beyer, D. (2022, November). A retrospective study of one decade of artifact evaluations. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2022* (pp. 145–156). Association for Computing Machinery. https://doi.org/10.1145/3540250.3549172L

Wood, B. D. K., Müller, R., Brown, A. N., & Azer, S. A. (2018, December). Push button replication: Is impact evaluation evidence for international development verifiable? *PLOS ONE*, *13*(12), e0209416. https://doi.org/10.1371/journal.pone.0209416

Wood, R. M., McWilliams, C. J., Thomas, M. J., Bourdeaux, C. P., & Vasilakis, C. (2020, September). COVID-19 scenario modelling for the mitigation of capacity-dependent deaths in intensive care. *Health Care Management Science*, *23*(3), 315–324. https://doi.org/10.1007/s10729-020-09511-7

Wood, R. M., Pratt, A. C., Kenward, C., McWilliams, C. J., Booton, R. D., Thomas, M. J., Bourdeaux, C. P., & Vasilakis, C. (2021, May). The value of triage during periods of intense COVID-19 demand: Simulation modeling study. *Medical Decision Making*, *41*(4), 393–407. https://doi.org/10.1177/0272989X21994035

Zhang, X., Lhachimi, S. K., & Rogowski, W. H. (2020, April). Reporting quality of discrete event Simulations in healthcare-results from a generic reporting checklist. *Value in Health*, *230*(4), 506–514. https://doi.org/10.1016/j.jval.2020.01.005

Appendices

Appendix A. Frameworks and guidelines used in evaluation

A. 1. *STRESS-DES*

As presented in Monks et al. (2019) (Monks et al., 2019)

- **1.1 Purpose—**Purpose of the model—Explain the background and objectives for the model.
- **1.2 Outputs—**Model outputs—Define all quantitative performance measures that are reported, using equations where necessary. Specify how and when they are calculated during the model run along with how any measures of error such as confidence intervals are calculated.
- **1.3 Aims—**Experimentation aims—If the model has been used for experimentation, state the objectives that it was used to investigate.
  - (A) Scenario-based analysis—Provide a name and description for each scenario, providing a rationale for the choice of scenarios and ensure that item 2.3 (below) is completed.
  - (B) Design of experiments—Provide details of the overall design of the experiments with reference to performance measures and their parameters (provide further details in data below).
  - (C) Simulation Optimisation—(if appropriate) Provide full details of what is to be optimised, the parameters that were included and the algorithm(s) that was be used. Where possible provide a citation of the algorithm(s).
- **2.1 Diagram—**Base model overview diagram—Describe the base model using appropriate diagrams and description. This could include one or more process flow, activity cycle, or equivalent diagrams sufficient to describe the model to readers. Avoid complicated diagrams in the main text. The goal is to describe the breadth and depth of the model with respect to the system being studied.
- **2.2 Base logic—**Base model logic—Give details of the base model logic. Give additional model logic details sufficient to communicate to the reader how the model works.
- **2.3 Scenarios—**Scenario logic—Give details of the logical difference between the base case model and scenarios (if any). This could be incorporated as text or where differences are substantial could be incorporated in the same manner as 2.2.
- **2.4 Algorithms—**Algorithms—Provide further detail on any algorithms in the model that (for example) mimic complex or manual processes in the real world (i.e., scheduling of arrivals/appointments/operations/maintenance, operation of a conveyor system, machine breakdowns, etc.). Sufficient detail should be included (or referred to in other published work) for the algorithms to be reproducible. Pseudo-code may be used to describe an algorithm.
- **2.5.1 Entities—**Components—entities—Give details of all entities within the simulation including a description of their role in the model and a description of all their attributes.
- **2.5.2 Activities—**Components—activities—Describe the activities that entities engage in within the model. Provide details of entity routing into and out of the activity.
- **2.5.3 Resources—**Components—resources—List all the resources included within the model and which activities make use of them.
- **2.5.4 Queues—**Components—queues—Give details of the assumed queuing discipline used in the model (e.g., First in First Out, Last in First Out, prioritisation, etc.). Where one or more queues have a different discipline from the rest, provide a list of queues, indicating the queuing discipline used for each. If reneging, balking or jockeying occur, etc., provide details of the rules. Detail any delays or capacity constraints on the queues.
- **2.5.5 Entry/exit—**Components—entry/exit points—Give details of the model boundaries, i.e., all arrival and exit points of entities. Detail the arrival mechanism (e.g., "thinning" to mimic a non-homogenous Poisson process or balking).
- **3.1 Data sources—**Data sources—List and detail all data sources. Sources may include: interviews with stakeholders, samples of routinely collected data, prospectively collected samples for the purpose of the simulation study, public domain data published in either academic or organisational literature. Provide, where possible, the link and digital object identifier (DOI) to the data or reference to published literature. All data source descriptions should include details of the sample size, sample date ranges, and use within the study.
- **3.2 Pre-processing—**Pre-processing—Provide details of any data manipulation that has taken place before its use in the simulation, e.g., interpolation to account for missing data or the removal of outliers.
- **3.3 Inputs—**Input parameters—List all input variables in the model. Provide a description of their use and include parameter values. For stochastic inputs provide details of any continuous, discrete, or empirical distributions used along with all associated parameters. Give details of all time-dependent parameters and correlation. Clearly state:
  - Base case data.
  - Data use in experimentation, where different from the base case.
  - Where optimisation or design of experiments has been used, state the range of values that parameters can take.
  - Where theoretical distributions are used, state how these were selected and prioritised above other candidate distributions.
- **3.4 Assumptions—**Assumptions—Where data or knowledge of the real system is unavailable what assumptions are included in the model? This might include parameter values, distributions, or routing logic within the model.
- **4.1 Initialisation—**Initialisation—Report if the system modelled is terminating or non-terminating. State if a warm-up period has been used, its length and the analysis method used to select it. For terminating systems state the stopping condition. State what if any initial model conditions have been included, e.g., pre-loaded queues and activities. Report whether initialisation of these variables is deterministic or stochastic.
- **4.2 Run length—**Run length—Detail the run length of the simulation model and time units.

- **4.3 Estimation—**Estimation approach—State the method used to account for the stochasticity: For example, two common methods are multiple replications or batch means. Where multiple replications have been used, state the number of replications and for batch means, indicate the batch length and whether the batch means procedure is standard, spaced, or overlapping. For both procedures provide a justification for the methods used and the number of replications/size of batches.
- **5.1 Language—**Software or programming language—State the operating system and version and build number. State the name, version, and build number of commercial or open source discrete-event simulation (DES) software that the model is implemented in. State the name and version of general-purpose programming languages used (e.g., Python 3.5). Where frameworks and libraries have been used provide all details including version numbers.
- **5.2 Random—**Random sampling—State the algorithm used to generate random samples in the software/programming language used, e.g., Mersenne Twister. If common random numbers are used, state how seeds (or random number streams) are distributed among sampling processes.
- **5.3 Execution—**Model execution—State the event processing mechanism used, e.g., three phase, event, activity, process interaction. Note that in some commercial software the event processing mechanism may not be published. In these cases, authors should adhere to item 5.1 software recommendations. State all priority rules included if entities/activities compete for resources. If the model is parallel, distributed and/or use grid or cloud computing, etc., state and preferably reference the technology used. For parallel and distributed simulations the time management algorithms used. If the High Level Architecture (HLA) is used, then state the version of the standard, which run-time infrastructure (and version), and any supporting documents (Federation Object Models (FOMs), etc.).
- **5.4 System**—System specification State the model run time and specification of hardware used. This is particularly important for large-scale models that require substantial computing power. For parallel, distributed and/or use grid or cloud computing, etc. state the details of all systems used in the implementation (processors, network, etc.).
- **6.1 Sharing**—Computer model sharing statement—Describe how someone could obtain the model described in the paper, the simulation software and any other associated software (or hardware) needed to reproduce the results. Provide, where possible, the link and DOIs to these.

A. 2. Generic reporting checklist

As outlined in Zhang et al., (2020).

Model conceptualisation

- **1 Decision problem—**Is the focused health-related decision problem clarified? . . .the decision problem under investigation was defined. DES studies included different types of decision problems, e.g., those listed in previously developed taxonomies.
- **2 Setting/condition—**Is the modeled healthcare setting/health condition clarified? . . .the physical context/scope (e.g., a certain healthcare unit or a broader system) or disease spectrum simulated was described.
- **3 Structure**—Is the model structure described? . . .the model's conceptual structure was described in the form of either graphical or text presentation.
- **4 Time horizon**—Is the time horizon given? . . .the time period covered by the simulation was reported.
- **5 Scenarios**—Are all simulated strategies/scenarios specified? . . .the comparators under test were described in terms of their components, corresponding variations, etc.
- **6 Population**—Is the target population described? . . .the entities simulated and their main attributes were characterised.

Parameterisation and uncertainty assessment

- **7 Data sources**—Are data sources informing parameter estimations provided? . . .the sources of all data used to inform model inputs were reported.
- **8 Inputs**—Are the parameters used to populate model frameworks specified? . . .all relevant parameters fed into model frameworks were disclosed.
- **9 Uncertainty**—Are model uncertainties discussed? . . .the uncertainty surrounding parameter estimations and adopted statistical methods (e.g., 95% confidence intervals or possibility distributions) were reported.
- **10 Sensitivity**—Are sensitivity analyses performed and reported? . . .the robustness of model outputs to input uncertainties was examined, for example via deterministic (based on parameters' plausible ranges) or probabilistic (based on a priori-defined probability distributions) sensitivity analyses, or both.

Validation

- **11 Face validity**—Is face validity evaluated and reported? . . .it was reported that the model was subjected to the examination on how well model designs correspond to the reality and intuitions. It was assumed that this type of validation should be conducted by external evaluators with no stake in the study.
- **12 Cross validity**—Is cross validation performed and reported? . . .comparison across similar modelling studies which deal with the same decision problem was undertaken.
- **13 External validity**—Is external validation performed and reported? . . .the modeler(s) examined how well the model's results match the empirical data of an actual event modeled.

- **14 Predictive validity**—Is predictive validation performed or attempted? . . .the modeler(s) examined the consistency of a model's predictions of a future event and the actual outcomes in the future. If this was not undertaken, it was assessed whether the reasons were discussed.

Generalisability and stakeholder involvement

- **15 Generalisability**—Is the model generalizability issue discussed? . . .the modeler(s) discussed the potential of the resulting model for being applicable to other settings/populations (single/multiple application).
- **16 Stakeholders**—Are decision makers or other stakeholders involved in modelling? . . .the modeler(s) reported in which part throughout the modelling process decision makers and other stakeholders (e.g., subject experts) were engaged.
- **17 Funding**—Is the source of funding stated? . . .the sponsorship of the study was indicated.
- **18 Limitations**—Are model limitations discussed? . . .limitations of the assessed model, especially limitations of interest to decision makers, were discussed.

A. 3. STARS framework

As described in Monks et al., (2024).

Essential components

- **Open licence**—Free and open-source software (FOSS) licence (e.g., MIT, GNU Public Licence (GPL)).
- **Dependency management**—Specify software libraries, version numbers, and sources (e.g., dependency management tools like virtualenv, conda, poetry).
- **FOSS model**—Coded in FOSS language (e.g., R, Julia, Python).
- **Minimum documentation**—Minimal instructions (e.g., in README) that overview (a) what model does, (b) how to install and run model to obtain results, and (c) how to vary parameters to run new experiments.
- **ORCID**—ORCID for each study author.
- **Citation information**—Instructions on how to cite the research artefact (e.g., CITATION.cff file).
- **Remote code repository**—Code available in a remote code repository (e.g., GitHub, GitLab, BitBucket).
- **Open science archive**—Code stored in an open science archive with FORCE11 compliant citation and guaranteed persistance of digital artefacts (e.g., Figshare, Zenodo, the Open Science Framework (OSF), and the Computational Modelling in the Social and Ecological Sciences Network (CoMSES Net)).

Optional components

- **Enhanced documentation**—Open and high-quality documentation on how the model is implemented and works (e.g., via notebooks and markdown files, brought together using software like Quarto and Jupyter Book). Suggested content includes:
  - Plain English summary of project and model.
  - Clarifying licence.
  - Citation instructions.
  - Contribution instructions.
  - Model installation instructions.
  - Structured code walk through of model.
  - Documentation of modelling cycle using *TRAnsparent and Comprehensive model Evaluation* (TRACE).
  - Annotated simulation reporting guidelines.
  - Clear description of model validation including its intended purpose.
- **Documentation hosting**—Host documentation (e.g., with GitHub pages, GitLab pages, BitBucket Cloud, Quarto Pub).
- **Online coding environment**—Provide an online environment where users can run and change code (e.g., BinderHub, Google Colaboratory, Deepnote).
- **Model interface**—Provide web application interface to the model so it is accessible to less technical simulation users.
- **Web app hosting**—Host web app online (e.g., Streamlit Community Cloud, ShinyApps hosting).

A. 4. Journal badges

Journal badges were identified from several sources:

- Association for Computing Machinery (ACM) (NISO Reproducibility Badging and Definitions Working Group, 2021).
- National Information Standards Organisation (NISO) (NISO Reproducibility Badging and Definitions Working Group, 2021).
- Center for Open Science (COS) (Blohowiak et al., 2023).
- Institute of Electrical and Electronics Engineers (IEEE) (Institute of Electrical and Electronics Engineers IEEE, 2024).
- *Psychological Science* journal (Association for Psychological Science APS, 2024; Hardwicke & Vazire, 2024).

The criteria for each badge are described in Table A1, with the badges grouped into three themes, as defined by NISO (NISO Reproducibility Badging and Definitions Working Group, 2021).

Table A1. Criteria for journal badges.

Badge	Criteria
"Open objects" badges	
ACM "Artefacts Available" (Association for Computing Machinery ACM, 2020)	● Artefacts are archived in a repository that is: (a) public (b) guarantees persistence (c) gives a unique identifier.
NISO "Open Research Objects (ORO)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	Criteria for ACM badge plus: ● Open licence.
NISO "Open Research Objects - All (ORO-A)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	Criteria for NISO "ORO" badge plus: ● Complete (all relevant artefacts available).
COS "Open Code" (Blohowiak et al., 2023)	Criteria for NISO "ORO" badge plus: ● Documents (a) how code is used (b) how it relates to article (c) software, systems, packages and versions.
IEEE "Code Available" (Institute of Electrical and Electronics Engineers IEEE, 2024)	● Complete (all relevant artefacts available).
"Object review" badges	
ACM "Artefacts Evaluated - Functional" (Association for Computing Machinery ACM, 2020)	● Documents (a) inventory of artefacts (b) sufficient description for artefacts to be exercised. ● Artefacts relevant to the paper. ● Complete (all relevant artefacts available). ● Scripts can be successfully executed.
ACM "Artefacts Evaluated - Reusable" (Association for Computing Machinery ACM, 2020)	Criteria for ACM "Functional" badge plus: ● Artefacts are carefully documented and well-structured to the extent that reuse and repurposing is facilitated, adhering to norms and standards.
IEEE "Code Reviewed" (Institute of Electrical and Electronics Engineers IEEE, 2024)	● Complete (all relevant artefacts available). ● Scripts can be successfully executed.
"Reproduced" badges	
ACM "Results Reproduced" (Association for Computing Machinery ACM, 2020)	● Reproduced results (assuming (a) acceptably similar (b) reasonable time frame (c) only minor troubleshooting).
NISO "Results Reproduced (ROR-R)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	Same as ACM "Results Reproduced".
IEEE "Code Reproducible" (Winter et al., 2022)	Same as ACM "Results Reproduced".
Psychological Science "Computational Reproducibility" (Association for Psychological Science APS, 2024; Hardwicke & Vazire, 2024)	Criteria for ACM "Results Reproduced" plus: ● README file with step-by-step instructions to run analysis. ● Dependencies (e.g., package versions) stated. ● Clear how output of analysis corresponds to article.

Abbreviations: ACM, Association for Computing Machinery; COS, Center for Open Science; IEEE, Institute of Electrical and Electronics Engineers; NISO, National Information Standards Organisation.

Appendix B. Full journal badge evaluation

Table B1. Evaluation of studies against badge criteria.

Badge	Studies that met criteria
"Open objects" badges	
ACM "Artefacts Available" (Association for Computing Machinery (ACM), 2020)	1/8 (12.5%)
NISO "Open Research Objects (ORO)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	1/8 (12.5%)
NISO "Open Research Objects—All (ORO-A)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	0/8 (0.0%)
COS "Open Code" (Blohowiak et al., 2023)	1/8 (12.5%)
IEEE "Code Available" (Institute of Electrical and Electronics Engineers (IEEE), 2024)	1/8 (12.5%)
"Object review" badges	
ACM "Artefacts Evaluated—Functional" (Association for Computing Machinery (ACM), 2020)	0/8 (0.0%)
ACM "Artefacts Evaluated—Reusable" (Association for Computing Machinery (ACM), 2020)	0/8 (0.0%)
IEEE "Code Reviewed" (Institute of Electrical and Electronics Engineers (IEEE), 2024)	1/8 (12.5%)
"Reproduced" badges	
ACM "Results Reproduced" (Association for Computing Machinery (ACM), 2020)	1/8 (12.5%)
NISO "Results Reproduced (ROR-R)" (NISO Reproducibility Badging and Definitions Working Group, 2021)	1/8 (12.5%)
IEEE "Code Reproducible" (Winter et al., 2022)	1/8 (12.5%)
Psychological Science "Computational Reproducibility" (Association for Psychological Science (APS), 2024; Hardwicke & Vazire, 2024)	0/8 (0.0%)

Abbreviations: ACM, Association for Computing Machinery; COS, Center for Open Science; IEEE, Institute of Electrical and Electronics Engineers; NISO, National Information Standards Organisation.

Appendix C. Code repositories

The following table provides links to the code repositories containing each of the computational reproducibility assessments and evaluations. There are also links to the associated Quarto websites and archives, ensuring easy access to all relevant resources.

This second table provides links to the code repositories from the original articles.

Table C1. Links for reproduction and evaluation.

Description	URL
Compilation of results from the eight studies	
Repository	https://github.com/pythonhealthdatascience/stars_wp1_summary
Archive	(Heather et al., 2025a)
Website	https://pythonhealthdatascience.github.io/stars_wp1_summary/
Reproduction: Hernandez et al. (2015)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-hernandez-2015
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-hernandez-2015/
Reproduction: Huang et al. (2019)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-huang-2019
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-huang-2019/
Reproduction: Lim et al. (2020)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-lim-2020
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-lim-2020/
Reproduction: Kim et al., (2021)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-kim-2021/
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-kim-2021/
Reproduction: Johnson et al. (2021)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-johnson-2021
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-johnson-2021/
Reproduction: R. M. Wood et al. (2021)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-wood-2021
Archive	(Heather, Monks & Harper, 2025c)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-wood-2021/
Reproduction: Shoaib & Ramamohan (2022)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-shoaib-2022
Archive	(Heather, Monks & Harper, 2025b)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-shoaib-2022/
Reproduction: Anagnostou et al. (2022)	
Repository	https://github.com/pythonhealthdatascience/stars-reproduce-anagnostou-2022
Archive	(Heather, Monks, & Harper, 2025)
Website	https://pythonhealthdatascience.github.io/stars-reproduce-anagnostou-2022/

Table C2. Links to original study repositories.

Description	URL
Original study: Hernandez et al. (2015)	
Repository	https://github.com/ivihernandez/staff-allocation
Original study: Huang et al. (2019)	
Repository	https://github.com/shiweih/desECR
Original study: Lim et al. (2020)	
Repository	https://github.com/chaose5/COVID-roster-simulation
Original study: Kim et al. (2021)	
Repository	https://github.com/mikesweeting/AAA_DES_model
Original study: Johnson et al. (2021)	
Repository	https://github.com/KateJohnson/epicR/tree/closed_cohort
Original study: R. M. Wood et al. (2021)	
Repository	https://github.com/nhs-bnssg-analytics/triage-modelling
Original study: Shoaib & Ramamohan (2022)	
Repository	https://github.com/shoaibiocl/PHC-
Original study: Anagnostou et al. (2022)	
Repository	https://gitlab.com/anabrunel/charm
Archive	(Anagnostou, 2022)