

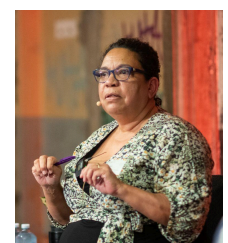
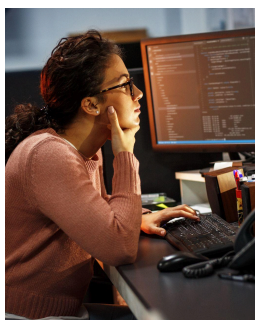
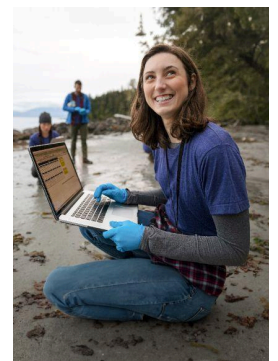
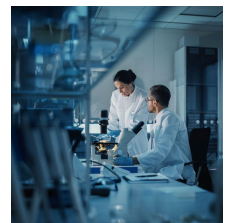


Australian Research Data Commons

Unearthing Research Software

By Eva Maxfield Brown and Nic Weber,
for the *Australian Research Data Commons*

February 2024



Text & infographics only.

DOI: 10.5281/zenodo.10530616

Suggested citation:

ARDC Ltd. (2024). *Unearthing Research Software*.

Viewed online at:

<http://doi.org/10.5281/zenodo.10530616>

Published:

29 February 2024

Cover images clockwise from top right:

Gorodenkoff - 402341966 / [adobestock.com](https://www.adobestock.com)

Hero Images - 316159695 / [adobestock.com](https://www.adobestock.com)

Marc Grimwade / ARDC

Gorodenkoff - 236237514 / [adobestock.com](https://www.adobestock.com)

BalanceFormCreative - 188531911 / [adobestock.com](https://www.adobestock.com)

thefirstglory - 391783296 / [adobestock.com](https://www.adobestock.com)

Mario Purisic - jG1z5o7NCq4 / unsplash.com

Acknowledgement of Country

We acknowledge the traditional custodians throughout Australia and their continuing connection to, and deep knowledge of, the land and waters. We pay our respects to Elders both past and present.

Contents

| | |
|--|-----------|
| List of Tables | 3 |
| List of Figures | 4 |
| Executive Summary | 5 |
| 1. Introduction | 7 |
| 2. Methods | 8 |
| 3. Data | 9 |
| 4. Model | 10 |
| 5. Results | 11 |
| 5.1. Example 1: Software Predicted but Not Verified | 11 |
| 5.2. Example 2: Software Predicted and Verified | 12 |
| 6. Findings and Discussion | 13 |
| 6.1. Overall Software Production | 13 |
| 6.2. ARDC Thematic Groups | 15 |
| 6.3. Fields of Research | 18 |
| 6.3.1. Awards Given and Software Produced | 19 |
| 6.4. Research Evaluation Committees | 21 |
| 6.5. Organisations | 23 |
| 6.5.1. Organisation Groups | 24 |
| 6.6. Funding | 26 |
| 7. Implications for Policy | 27 |
| 8. Appendix 1 - Examples of Model Prediction | 29 |
| 8.1. 5 Highest Software Predicted Probability Scores for ARC Grants - 2010-19 | 29 |
| 8.2. 5 Lowest Software Predicted Probability Scores - 2010-19 | 31 |
| 9. Appendix 2 - Template For A Research Software Development Plan (SDP) | 33 |
| Works Cited | 35 |

List of Tables

- Table 1.** tfidf-logit model evaluation scores against NSF and ARC award data
- Table 2.** ARDC Thematics Research Group, number of awards received, number of awards predicted to produce software, and percentage of awards predicted to produce software.
- Table 3.** Top ten most software producing Fields of Research for Fields of Research with equal to or greater than the median number of awards across all Fields of Research (two digit groupings).
- Table 4.** Top ten most software producing Research Evaluation Committees (RECs) for RECs with equal to or greater than the median number of awards across all RECs (two digit groupings).
- Table 5.** Top ten most awarded organisations and their predicted produced software.
- Table 6.** Software production by organisation grouping (Group of Eight, ATN, and Other).

List of Figures

- Figure 1.** Overall software production between 2010-2019
- Figure 2.** Software production over time as a percentage of awards each year
- Figure 3.** Overall software production between 2010-2019 grouped by ARDC Thematic Research Areas
- Figure 4.** Software production over time as a percentage of awards each year grouped by ARDC Thematic Research Areas
- Figure 5.** Top ten most software producing Fields of Research (FoR) for FoR with equal to or greater than the median number of awards across all FoR (two digit groupings)
- Figure 6.** Correlation between number of awards and the percent of awards which produce software grouped by Fields of Research (two-digit groupings)
- Figure 7.** Top ten most software producing Research Evaluation Committees (REC) for RECs with equal to or greater than the median number of awards across all RECs
- Figure 8.** Top ten most awarded organisations with percent of awards predicted to produce software
- Figure 9.** Software production over time as a percentage of awards each year grouped by Organisation type (Group of Eight, ATN, and Other)
- Figure 10.** Distribution of funding award funding amounts grouped by model prediction

Executive Summary

The Australian Research Data Commons' (ARDC) 'National Agenda for Research Software' succinctly describes three challenges in seeing, shaping, and sustaining research software that are important for a robust Australian research enterprise (ARDC, 2022).

This report primarily addresses challenges to surface the scale of production of research software by providing a novel quantification of how much software is produced, by whom, and with what ARC research funding. Using machine learning techniques we predicted the amount of research software that was produced by ARC grants from 2010-2019. We used verified ARC-funded software and grant metadata to evaluate our existing predictive model, and achieved a .71 (F1) accuracy score - which outperforms previous software prediction tasks in the science of science (Brown et al, 2023).

Our major findings include the following:

- We predict that ~47% of 2010-2019 ARC grants produced software infrastructure, tools, or code of some kind.
- There is a linear increase of software produced by ARC grants - as the volume of grants increases so too does the amount of software produced. This finding holds true across fields, organisation types, and over time.
- Despite disciplinary and methodological differences we estimate that 25% of all Humanities, Arts, and Social Sciences research grants produce software (HASS & Indigenous Research according to the ARDC's 'Thematic Areas')
- There is no significant relationship between the amount of funding awarded, and the probability that the research will produce software.

In short, we predict there is a substantially large amount of software produced by ARC funding, that this production is distributed across diverse research areas, and that software production is increasing. We believe the work described in this report also holds important implications for both the 'Use' and the 'Sustainability' challenged identified by ARDC's 'National Agenda for Research Software'.

We recommend 4 areas of potential future research and policy development:

1. **Require software production descriptions (SPD) in grant applications** - such documentation can be lightweight, easy to complete, and produce important guarantees to funders that the work is not invisible labour nor redundant with previous efforts. In Appendix 1 we provide a template for SPDs.
2. **Require awardees to report software as a grant outcome** - Recognizing that important basic and applied research contributions are made through software development can both surface more potentially valuable ARC funded research products to be reused, but can also provide a knowledge base for evaluating software production descriptions (SPDs).
3. **Conduct a national research software census** - This would activate a network of researchers and support staff to both identify and document research software, and help ARC better understand where future research software funding should be directed for sustainability.
4. **Identify workforce needs in research software engineering and support** - We show that previous estimates of the increase in labour supporting research software is out of line with the amount of software that is likely produced. We believe there is an important future consideration for how ARC funded research software is being supported and maintained by both research software engineers (RSEs) and students.

1. Introduction

Software is pervasive in contemporary research – it underpins nearly all data collection and analysis activities in engineering and the sciences, and plays an increasingly critical role in the creation, dissemination, and interpretation of scholarship in the arts and humanities.

But despite the importance of software to research it remains an often overlooked and underfunded resource. The ARDC's 'National Agenda for Research Software' describes the motivations behind formally recognizing software as a first-class research output, and in doing so poses three potential challenges:

- **Seeing Software:** Coordinated action is needed to boost software's visibility as an output of research - one that requires substantial intellectual effort as input and careful management and maintenance of its output.
- **Shaping Software:** Extended training and support for researchers will improve the quality and reusability of what gets produced in research settings.
- **Sustaining Software:** Identifying and then building pathways for sustaining critical software infrastructure will be vital to match policy with practice.

This report quantifies the scale of Australia's annual software research production using past award metadata and discoverable Australian Research Council (ARC)-funded research software. Our findings provide important external validation of previous qualitative and survey-based research focused on the undersupply of research software support, the difficulty of discovering and reusing research software, and the necessary sociotechnical infrastructures to meaningfully sustain research software.

2. Methods

Estimating the amount of software produced by ARC funded research required, generally, three research tasks: curating evaluation data, evaluating an existing predictive model, and analysing results of applying the model to the ARC awards database:

- **Data (Section 3):** We developed a new corpus that includes verified ARC-funded software repositories that are linked to grant metadata.
- **Model (Section 4):** We used an existing Term-Frequency Inverse-Document Frequency (TFIDF) Logistic Regression Model (described in Brown et. al 2023) to predict software in the evaluation corpus. We achieved an F1¹ score of 0.71 - which exceeds a benchmark produced in previous work.
- **Application (Section 5):** We applied the prediction model to a dataset of ARC grants awarded from 2010-2019, and performed exploratory data analysis to suggest future research directions.

¹ The F1 is a useful metric for evaluating NLP tasks (such as this) because it provides a harmonised measure of precision and recall. In this project the prediction is binary (Software_Produced vs Software_Not_Produced) - and precision metrics tend to overestimate the accuracy of a binary prediction while recall metrics tend to underestimate accuracy.

3. Data

We used the GitHub API to search for keywords in a software repository's README.md file². This search yielded approximately 1200 potential ARC-funded research repositories. We then used a regular expression to find ARC Grant Identifiers in the primary README.md file for each repository. This yielded 106 potential matches. Next, using the ARC award database we linked each software repository to an ARC award by the grant identifier. After removing duplicate awards we were left with 56 repositories. We then manually inspected and annotated each of the 56 repositories to verify that they contain research software. The resulting dataset is an **ARC evaluation corpus** to be used in evaluation testing.

² The keyword terms we searched for are enumerated [here](#)

4. Model

In previous work we trained prediction models on a large corpus of manually verified National Science Foundation (NSF) funded software repositories (n=917) (Brown, et al, 2023). The best performing model (tfidf-logit) was able to achieve an F1 score of .67³. We measured the performance of this model against the **ARC evaluation corpus**. The predictive task we performed took the text of a grant abstract as an input. The model we used, a logistic regression model trained with TF-IDF word embeddings (tfidf-logit), provides an output that is a binary classification - either Software_Produced or Software_Not_Produced. In table 1 (below) we evaluate the performance of the tfidf-logit model against both the NSF and ARC evaluation corpora. The similarity in F1 scores indicates that the model continues to be applicable in precision and recall, and should perform well at predicting whether an ARC grant produced software based on the features of its grant metadata.

Table 1. tfidf-logit model evaluation scores against NSF and ARC award data

| Dataset | Model | Precision | Recall | F1 |
|---------|-------------|-----------|--------|---------------|
| NSF | tfidf-logit | 0.674 | 0.673 | 0.6736 |
| ARC | tfidf-logit | 0.815 | 0.696 | 0.719 |

³ The F1 is a useful metric for evaluating NLP tasks (such as this) because it provides a harmonised measure of precision and recall. In this project the prediction is binary (Software_Produced vs Software_Not_Produced) - and precision metrics tend to overestimate the accuracy of a binary prediction while recall metrics tend to underestimate accuracy.

5. Results

We applied our predictive model (tfidf-logit) to the text of abstracts for 13,784 grants funded by ARC from 2010-2019. Our model predicts that 47% (n=6501) of ARC grants during this period produced software. In the following sections we further analyse these predictions, but first offer some insight as to the benefits and limitations to the machine learning approach we are using in this report⁴.

5.1. Example 1: Software Predicted but Not Verified

Below is an example of an ARC Future Fellowships grant. The proposal explicitly mentions software, and we should expect the model to predict software production (and in fact we would worry if it did not):

Grant: FT120100943

Title: Algorithmics for visual analytics of massive complex networks.

Abstract: The project will provide new scalable algorithms for visual analytics of massive complex networks. These fast algorithms will enable security analysts to detect abnormal behaviours such as money laundering, biologists to understand protein-protein interaction networks, and support software engineers' new ways of understanding large software systems.

Software Predicted Probability: 0.8291702665289034

Software Not Predicted Probability: 0.17082973347109665

The probability of our estimate provides some “certainty” – allowing us to estimate how well the prediction task might perform on obvious cases (such as this). We should interpret this as the model being 80% certain that the grant produced software (and 20% chance of being wrong). However, even this seemingly obvious example is hard to validate. When we searched for software produced under this grant and author we found none - there is no website, repository, or mention of software in any published papers related to the author during the period of performance (2012-2018 – we extended our search by two years to account for any post-award publication). This case could suggest a fault in our model, but it might just as easily demonstrate its utility – even using manual methods of inspection it can be incredibly hard to establish a verifiable link between research software production and an award.

⁴ In Appendix 2 we show examples of abstracts for ARC grant awards that were predicted to create software and those that were not.

5.2. Example 2: Software Predicted and Verified

The second example we offer is a Discovery Project award to the mathematician Dr. Craig Hodgson at the University of Melbourne.

Grant: DP190102363

Title: Classical and quantum invariants of low-dimensional manifolds.

Abstract: This project aims to advance our understanding of knots and 3-dimensional spaces, which arise naturally in fields as diverse as physics, computer graphics, chemistry and biology. Recent ideas from quantum field theory link physics to topology in dimensions 3 and 4, leading to powerful invariants of knots and 3-dimensional manifolds that include the Jones polynomial and the 3D-index. This project aims to resolve key questions relating these quantum invariants to classical topology and geometry. The project will have a major impact in low-dimensional topology, and lead to deep and unexpected connections between mathematics and mathematical physics.

Software Predicted Probability: 0.8812043988611511

Software Not Predicted Probability: 0.1187956011388489

Nowhere in this abstract or in the outcomes does it appear that the grant would have produced, used, or improved research software. However, the novelty of the proposed work builds upon research software produced using ARC funding where Dr. Hodgson and colleagues produced the C++ library Snap (Coulson et al, 2000). In fact, Dr Hodgson continues to use that software over the 24 years since it was produced (e.g., mentioned in Goodman et al 2008; Heard et al 2010), and the software continues to be cited by researchers. The continued reuse of this software is despite the fact that it has no suggested citation information or stable identifier⁵, and it is hosted on SourceForge where it continues to be downloaded around the world.⁶ The software, related to the current grant, has been reused by a high-impact library in mathematics in 2020 and even a children's geometry game in 2021⁷. In short, we believe this demonstrates that predicting software with an embedding space (as our approach here takes) can help surface what are tangential links between the language used in a grant abstract and the software produced in a research agenda supported over decades by ARC.

⁵ See the citations over time at [Google Scholar](#) as well as the [SNAP software website's landing page](#)

⁶ See the [SourceForge timeline view](#)

⁷ [Tardis](#) library and [Torus](#) game from Jeff Weeks

6. Findings and Discussion

6.1. Overall Software Production

We predict that approximately 47% of all ARC funded grants between 2010-2019 produced research software.

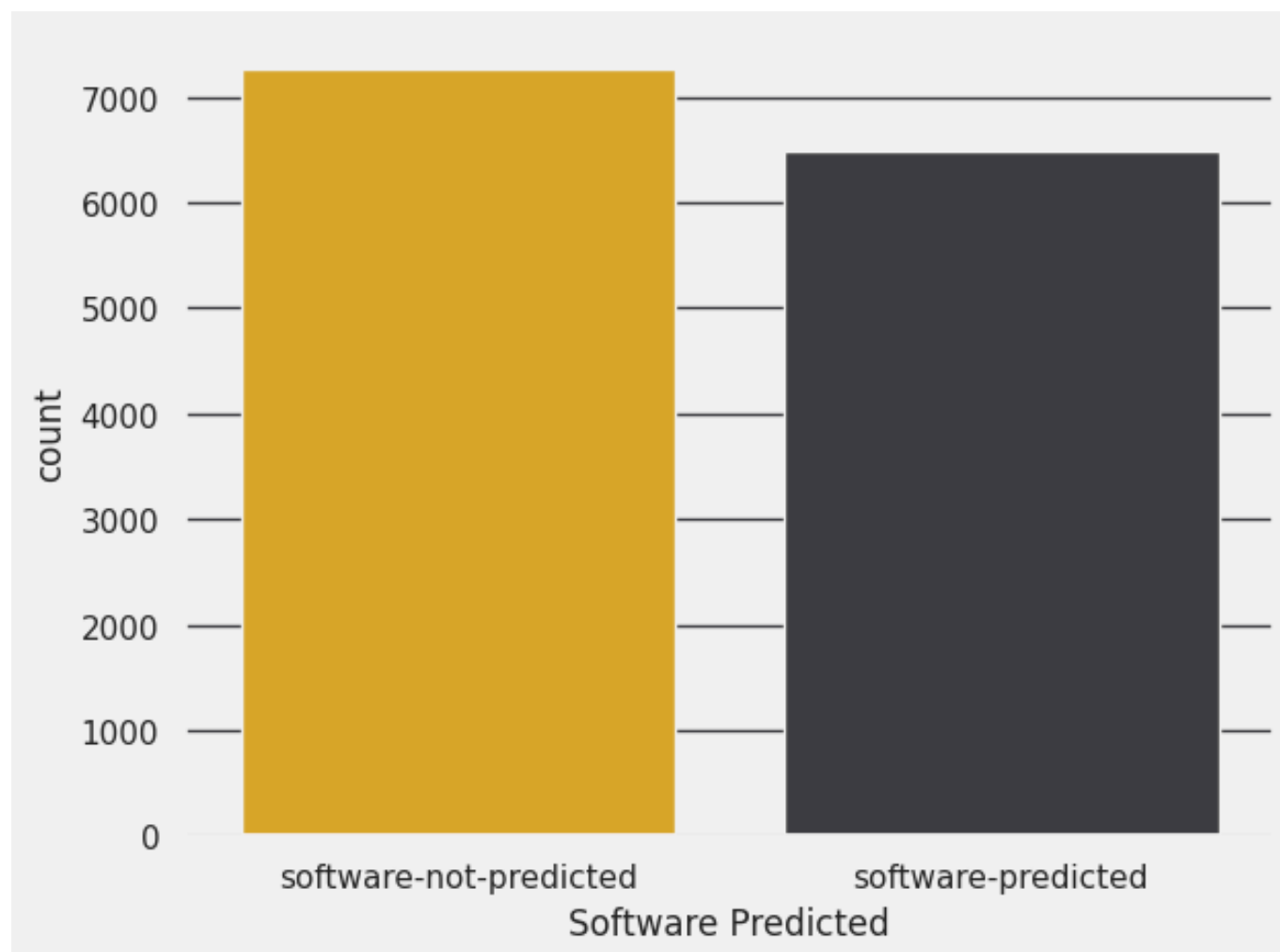


Figure 1. Overall software production between 2010-2019

Grouping predictions by the year in which the award was funded, our model indicates that predicted software production has increased over time.

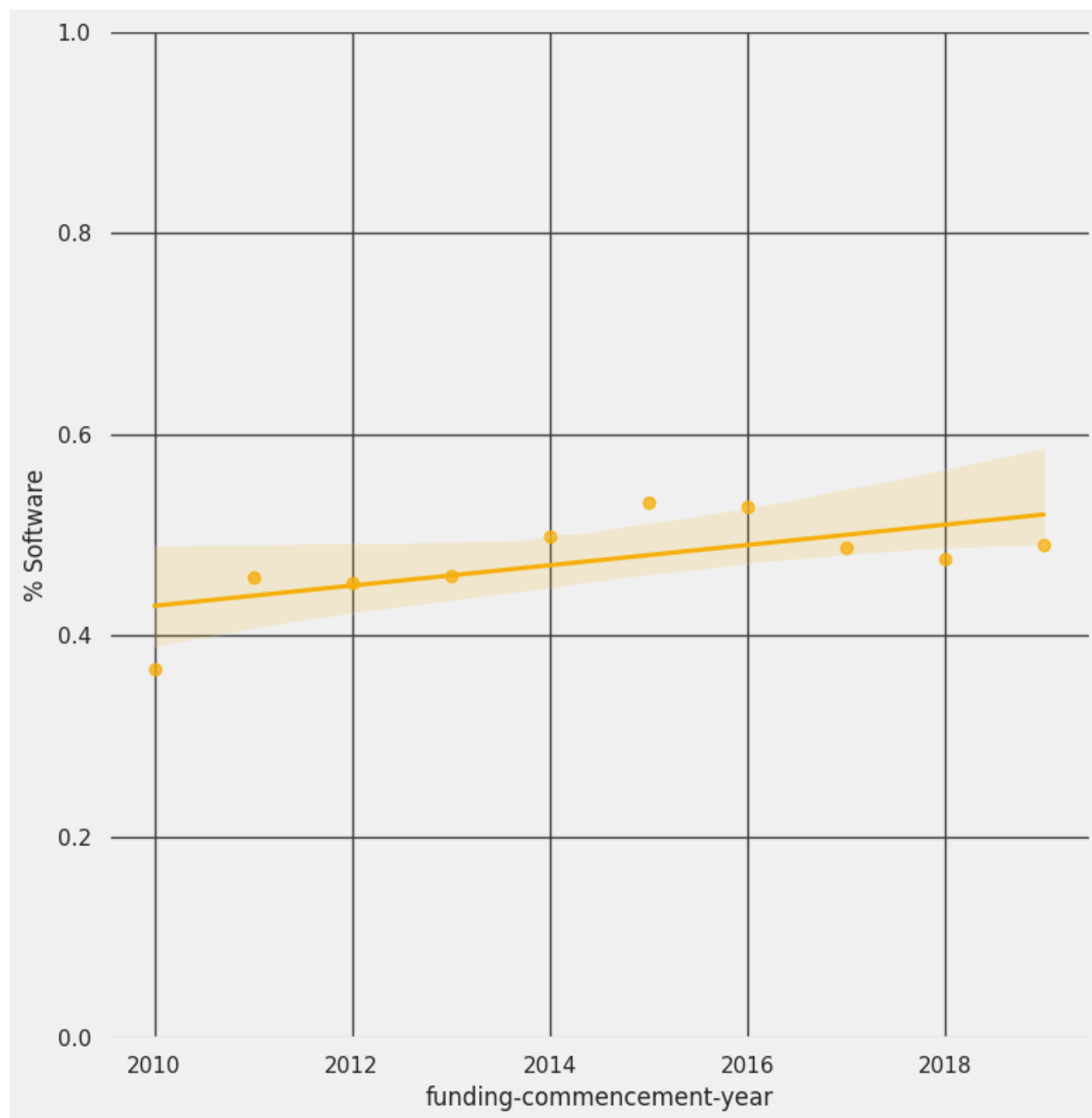


Figure 2. Predicted software production over time as a percentage of awards each year

6.2. ARDC Thematic Groups

Using the Australian Research Data Commons' thematic groupings⁸ we estimate that 49% (n=1830) of awards made to researcher's in the Planet (Earth and Environment) category produced software. Greater than 1/3 or 38% of People (Health and Medical) research awards produced software. We estimate that, surprisingly, 25% (n=988) of all HASS and Indigenous research awards produced software - a group that was largely underrepresented in previous work that described research software discoverability (Stevens, 2022). For all research falling outside these three groupings we bin into an "Other Areas" category.

Table 2. ARDC Thematics Research Group, number of awards received, number of awards predicted to produce software, and percentage of awards predicted to produce software.

| Research Group | Awards # | Software # | Software % |
|------------------------------|----------|------------|------------|
| Other Areas | 4585 | 3092 | 67.437 |
| Planet (Earth & Environment) | 3766 | 1830 | 48.593 |
| People (Medical & Health) | 1529 | 587 | 38.391 |
| HASS & Indigenous | 3890 | 988 | 25.984 |

⁸ The groupings are Planet, People, and HASS & Indigenous Research as described in <https://ardc.edu.au/case-study/thematic-rdc/>

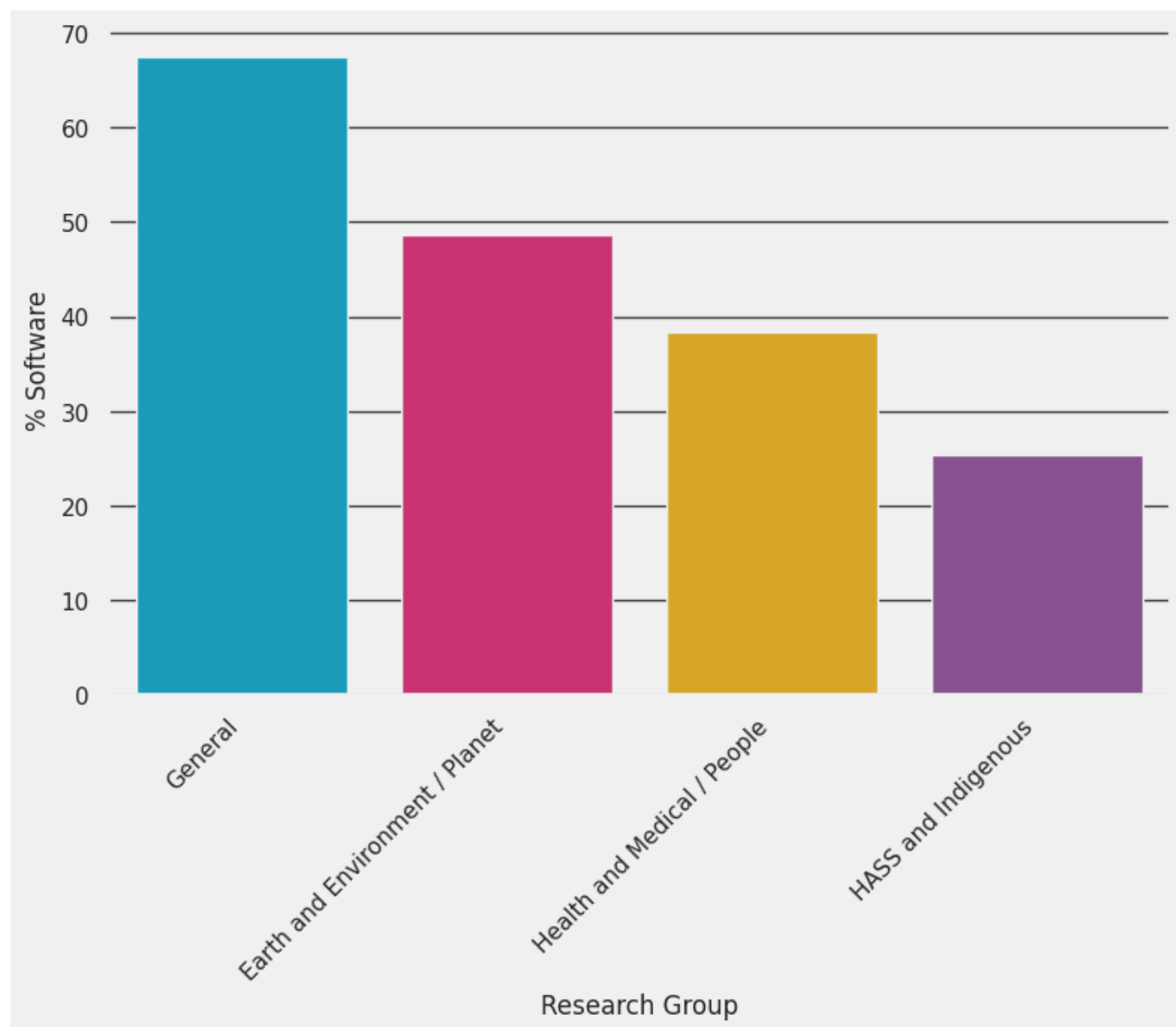


Figure 3. Overall software production between 2010-2019 grouped by ARDC Thematic Research Areas

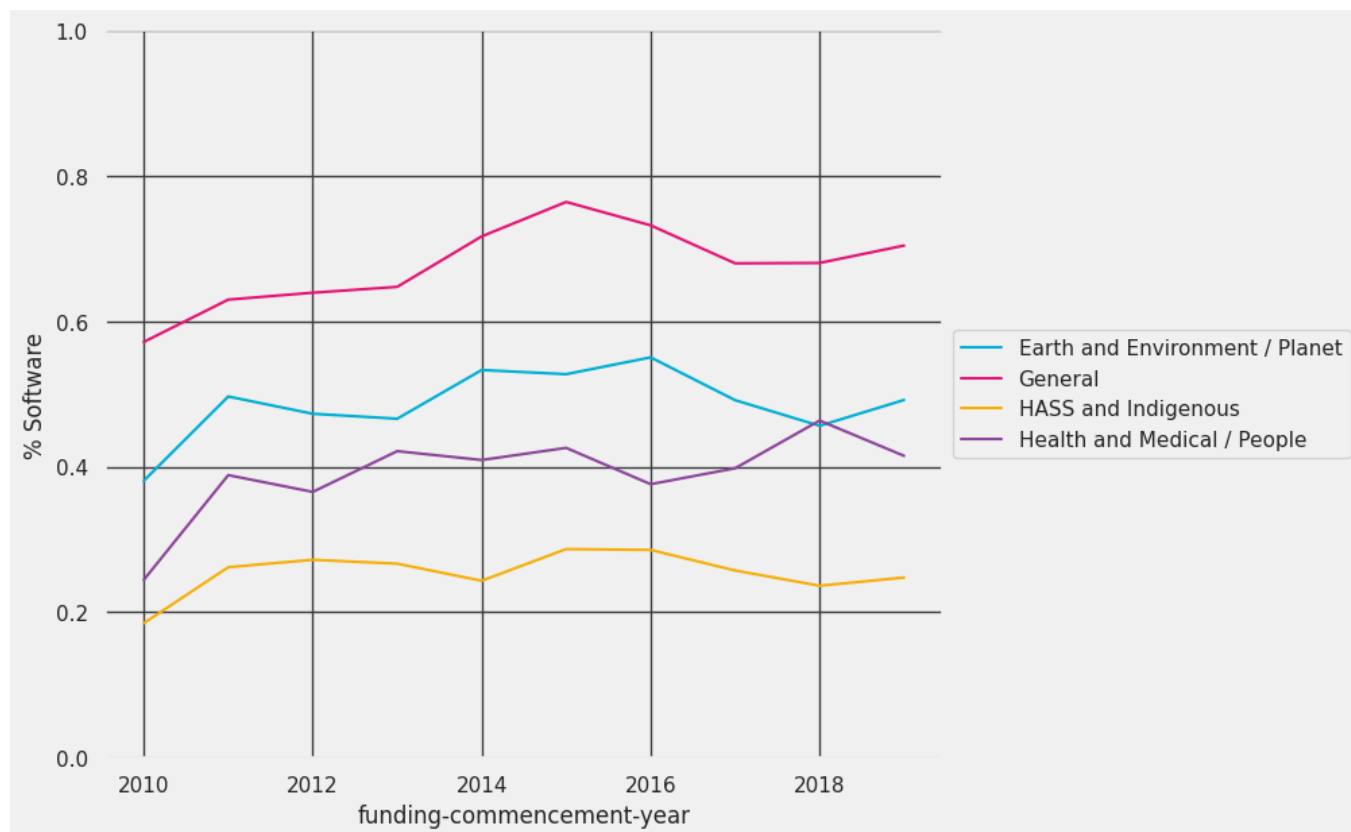


Figure 4. Software production over time as a percentage of awards each year grouped by ARDC Thematic Research Areas

A more detailed view over time shows that over the course of the entire 2010 - 2019 period, this pattern is generally followed: “General” awards by percentage produced the most software, then Earth and Environment / Planet, then Health and Medical / People, followed by HASS and Indigenous. In 2018, Health and Medical / People narrowly overtook Earth and Environmental / Planet in software production, and appears to be increasing the software produced by year at a greater rate than other categories. Notably, this detailed view also demonstrates that over the past decade each Thematic Group had a higher software production rate in 2019 than in 2010 - In other words, **all Thematic Groups are increasing in software production.**

6.3. Fields of Research

Using the 2008 ARC designated two-digit Field of Research (FoR) codes, we explored which fields produced the highest proportion of software relative to the number of grants received. We limited our analysis to fields that received greater than or equal to the median number of awards made in a given year. Across the decade of 2010-2019, we predicted that FoR fields produced software at a rate greater than 50% - meaning half of all awards made to these FoRs likely produced some form of software tool or code. In our analysis, we predicted that 'Mathematical Sciences' is the top software producer (75% of all awards made) and this mirrors our findings in analysing National Science Foundation (NSF) software production by directorate (where Mathematical and Physical Sciences were estimated to produce the most software by grant). However, we hold this finding with a bit of caution - our definition of software used to train the predictive model was inclusive of words related to algorithms and general scripting that commonly appear in mathematics grant abstracts. Unfortunately, mathematics is a particularly hard field to verify software production - as early as 1979 mathematicians have been advocating for greater transparency in software development and use (Crowder et al, 1979).

Table 3. Top ten most software producing Fields of Research for Fields of Research with equal to or greater than the median number of awards across all Fields of Research (two digit groupings).

| Field of Research | Awards # | Software # | Software % |
|------------------------------------|----------|------------|------------|
| Mathematical Sciences | 669 | 502 | 75.037369 |
| Technology | 507 | 370 | 72.978304 |
| Chemical Sciences | 840 | 581 | 69.166667 |
| Information and Computing Sciences | 768 | 483 | 62.890625 |
| Engineering | 2182 | 1294 | 59.303391 |
| Physical Sciences | 859 | 503 | 58.556461 |
| Biological Sciences | 1930 | 1036 | 53.678756 |
| Earth Sciences | 635 | 283 | 44.566929 |
| Medical and Health Sciences | 685 | 269 | 39.270073 |
| Psychology and Cognitive Sciences | 686 | 239 | 34.839650 |

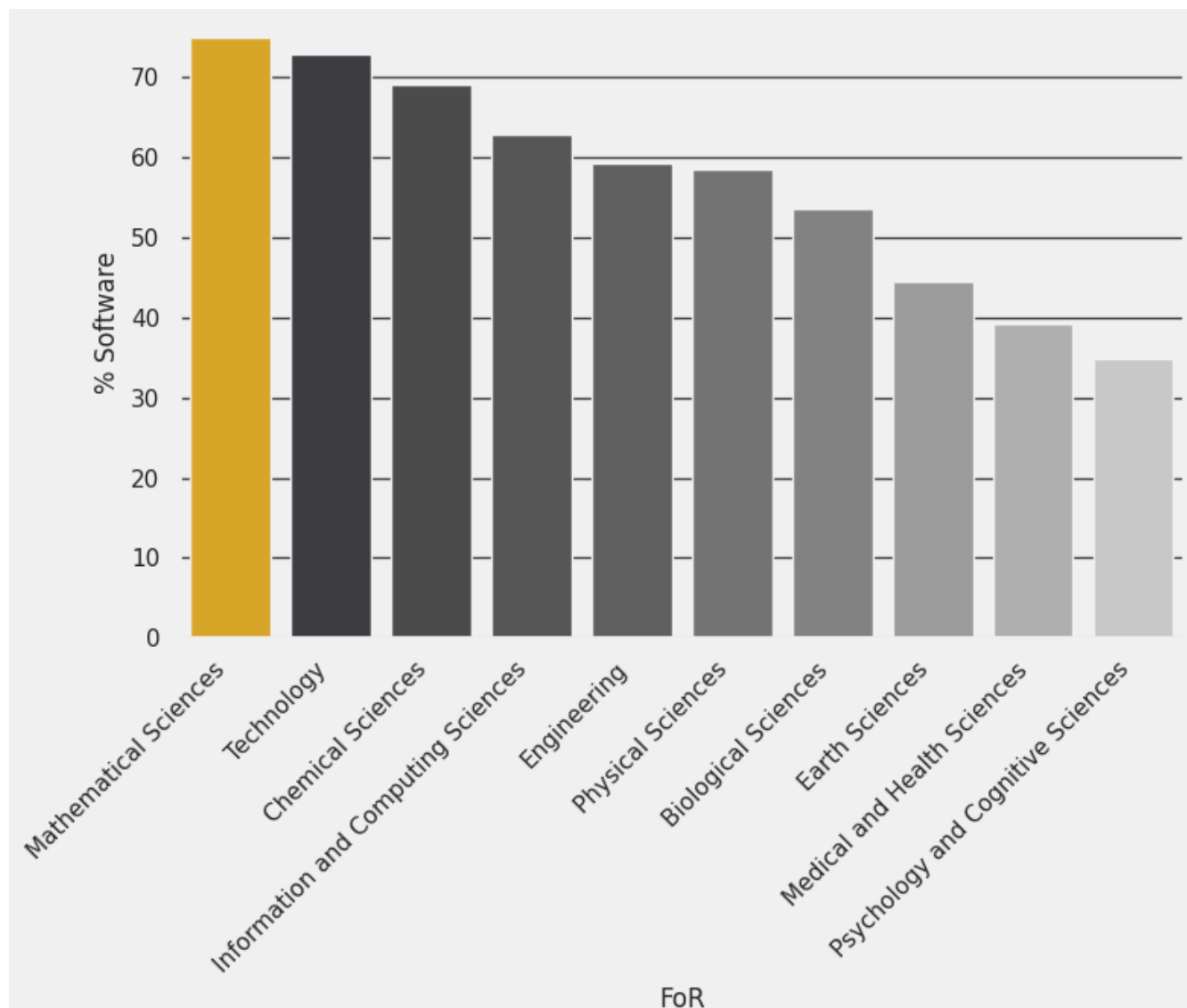


Figure 5. Top ten most software producing Fields of Research (FoR) for FoR with equal to or greater than the median number of awards across all FoR (two digit groupings)

6.3.1. Awards Given and Software Produced

A relationship between the amount of ARC grants that are awarded and the software that is produced by a field (FoR) is plotted below. We show a positive correlation between award volume and software produced – regardless of the field – we should note that the more grants that fields receive the higher the percentage of software produced. We caution against seeing this as a linear relationship that will hold if the independent variable (volume of awards) is manipulated – there is no reason to believe that simply increasing the amount of awards made to a field or organisation will cause more software to be produced. Instead, we believe there are important field and organisational differences that result in software being produced at different rates given the amount of rates that are produced (which is true of many other relationships between field and grant productivity as measured by publications, data, etc).

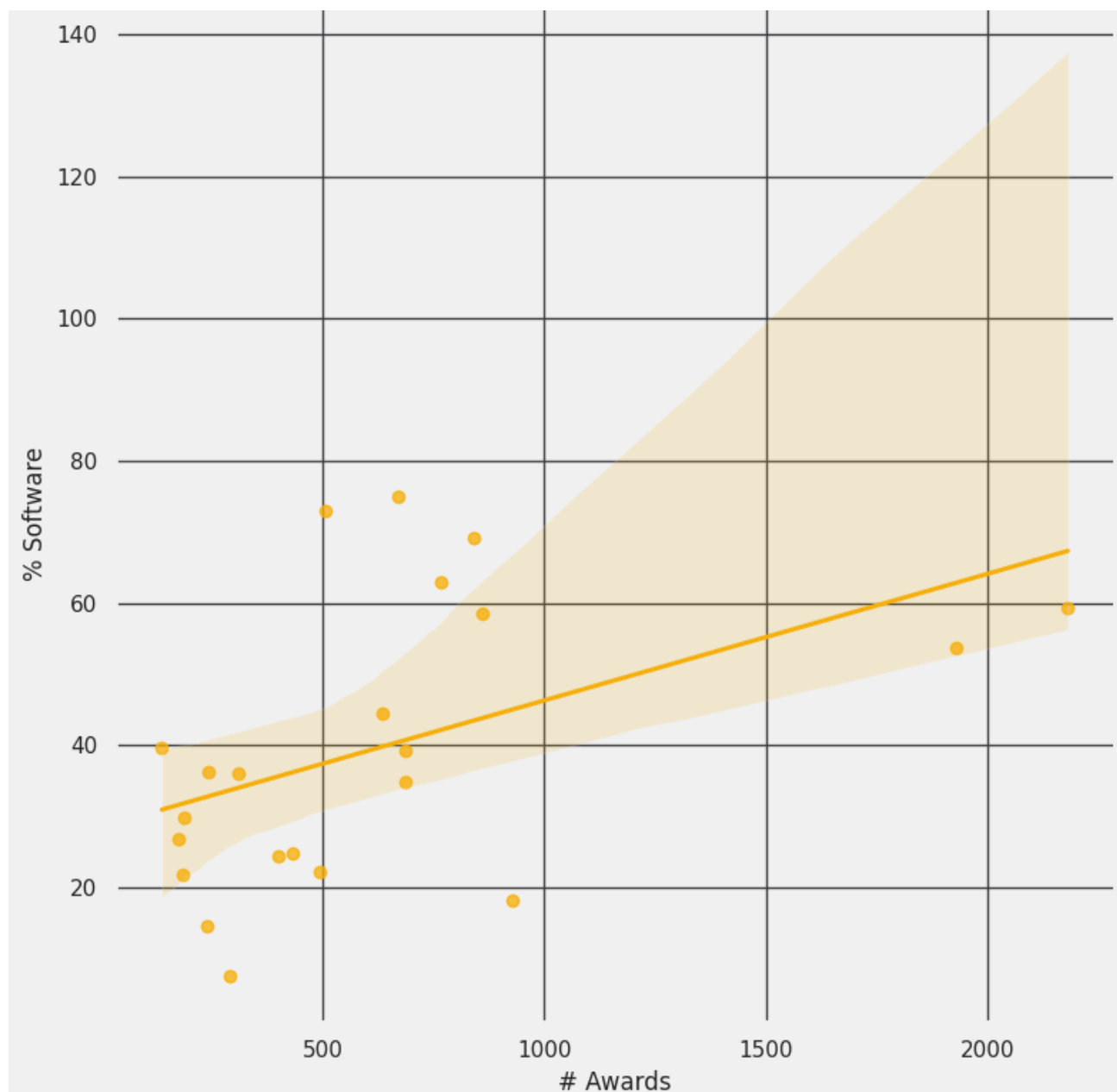


Figure 6. Correlation between number of awards and the percent of awards which produce software grouped by Fields of Research (two-digit groupings)

To further explore this relationship we revisited the total amount of software produced by year over the past decade of ARC funded research (figure 2). We showed a positive relationship between the two variables - over time more software is produced each year. We believe this relationship is rightfully linear - it should be expected that, overall, the more awards that are made the more software will be produced. Externalities that may affect this relationship negatively might include reduced funding (which would vary in impact to different fields) the consolidation of software by a field (e.g. a common package or library is released and renders new software production less important), and commercialization of software which hamper innovation (even temporarily).

6.4. Research Evaluation Committees

While FoR codes provide a fine-grained look into which fields software is being produced within, grouping by Research Evaluation Committee (REC) gives us a middle ground between FoR codes and the ARDC Thematic Groupings.

Table 4. Top ten most software producing Research Evaluation Committees (RECs) for RECs with equal to or greater than the median number of awards across all RECs (two digit groupings).

| REC | Awards # | Software # | Software % |
|--|----------|------------|------------|
| Technology | 507 | 370 | 72.978304 |
| Mathematical, Information and Computing Sciences | 1437 | 985 | 68.545581 |
| Physical, Chemical and Earth Sciences | 2334 | 1367 | 58.568980 |
| Engineering and Environmental Sciences | 2580 | 1391 | 53.914729 |
| Biological and Biotechnological Sciences | 2114 | 1076 | 50.898770 |
| Medical and Health Sciences | 1371 | 508 | 37.053246 |
| Economics and Commerce | 552 | 200 | 36.231884 |
| Humanities and Creative Arts | 1660 | 409 | 24.638554 |
| Education and Human Society | 1215 | 191 | 15.720165 |

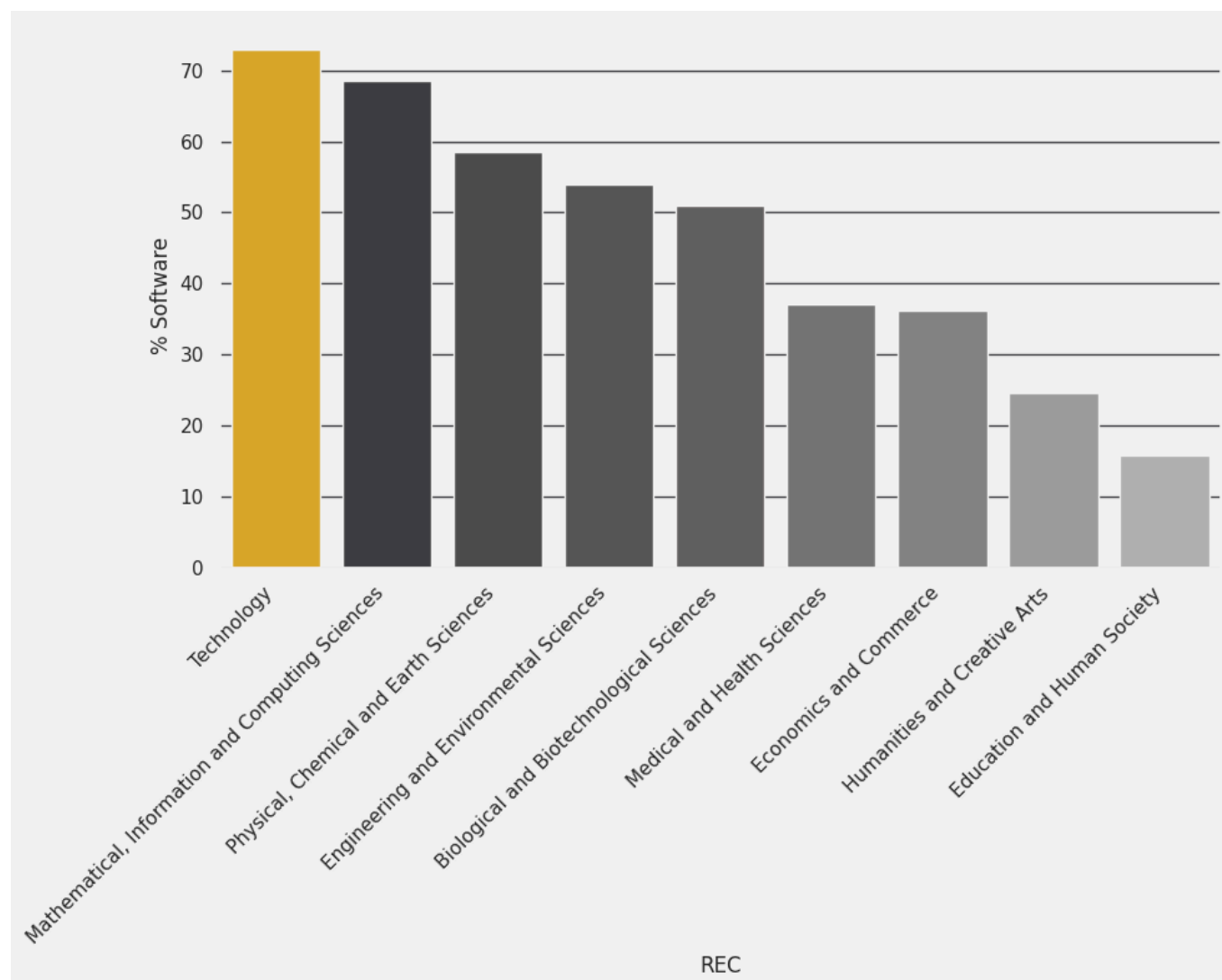


Figure 7. Top ten most software producing Research Evaluation Committees (REC) for RECs with equal to or greater than the median number of awards across all RECs

The awards reviewed by the “Technology” REC tended to produce the highest proportion of software relative to the number of awards given. Further, the overall distribution of software production by REC follows a trend of what we might expect with many STEM fields, and is reflected in the newly revised FoR codes (previous section).

6.5. Organisations

Next, we explored the production of research software by research organisations in Australia. In table 5 below we show the Top 10 universities by the number of ARC grants received. We predicted software created by those awards and therefore can report the percentage of research software production by organisation. The average percent of software production is 48% with a standard deviation of 3.1%. We interpret the relatively low deviation from the mean as being evidence that there is not a substantial benefit or penalty to software production based on organisation (at least by our selective sample of top-10 award winning organisations).

Table 5. Top ten most awarded organisations and their predicted produced software.

| Organisation | Awards # | Software # | Software % |
|-------------------------------------|----------|------------|------------|
| The University of New South Wales | 1479 | 749 | 50.642326 |
| The University of Queensland | 1457 | 740 | 50.789293 |
| The University of Melbourne | 1413 | 666 | 47.133758 |
| Monash University | 1249 | 637 | 51.000801 |
| The Australian National University | 1188 | 579 | 48.737374 |
| The University of Sydney | 1187 | 591 | 49.789385 |
| The University of Western Australia | 647 | 296 | 45.749614 |
| The University of Adelaide | 583 | 293 | 50.257290 |
| Macquarie University | 412 | 182 | 44.174757 |
| Queensland University of Technology | 368 | 156 | 42.391304 |

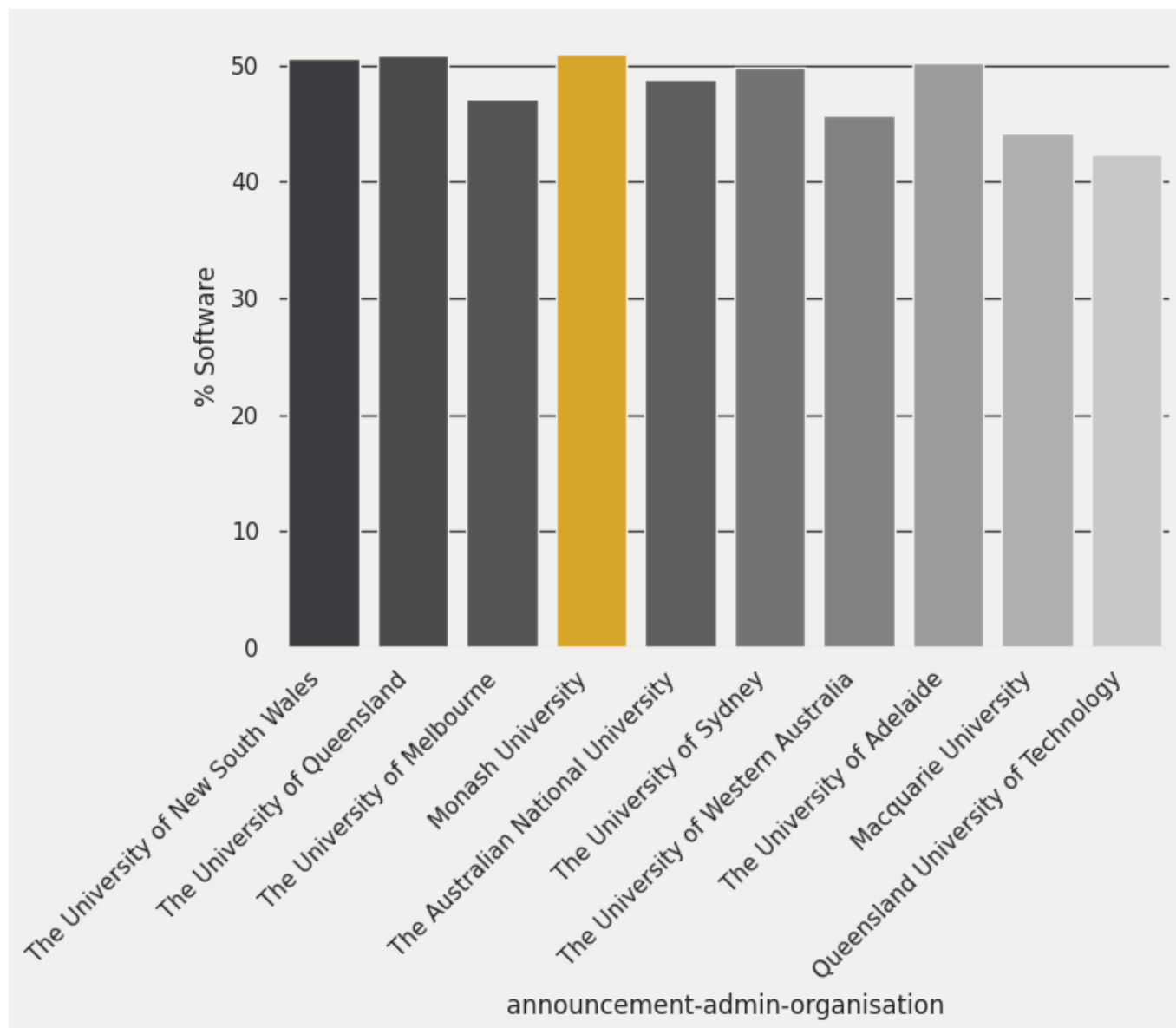


Figure 8. Top ten most awarded organisations with percent of awards predicted to produce software

6.5.1. Organisation Groups

To further explore software production by organisation we analysed the distribution of three university groupings: The “Group of Eight”, the Australian Technology Network (ATN), and all other universities. Given these large distributions we do not see a significant difference in the percentage of research grants producing research software. We believe that it is more instructive to view the production of software over time – each group of universities is also following a linear trend of increasing the amount of software produced over time (regardless of their prestige).

Table 6. Software production by organisation grouping (Group of Eight, ATN, and Other).

| Organisation Grouping | # Awards | # Software | % Software |
|-----------------------|----------|------------|------------|
| Group of Eight | 9203 | 4551 | 49.451266 |
| Other | 3285 | 1350 | 41.095890 |
| ATN | 1282 | 596 | 46.489860 |

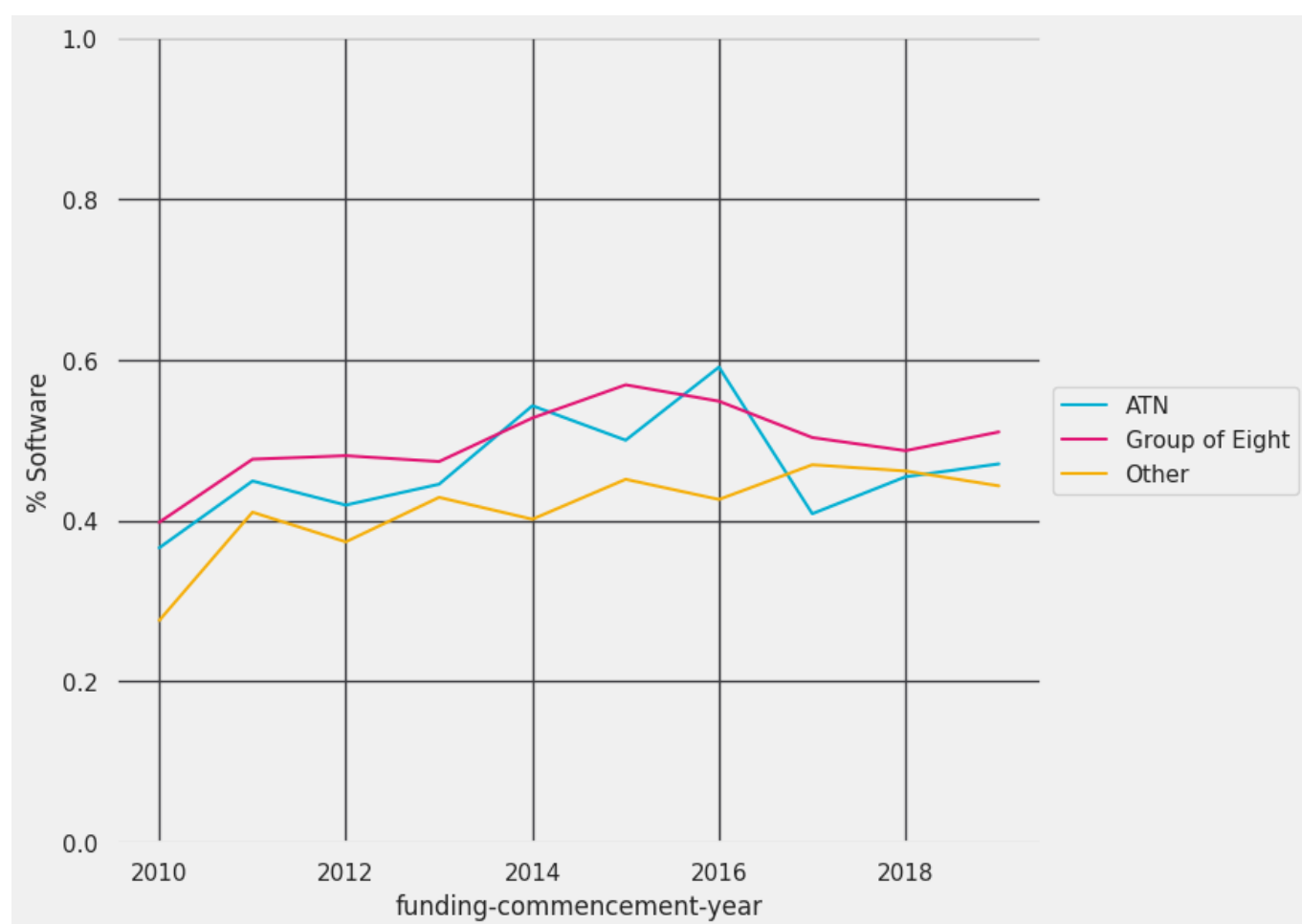


Figure 9. Software production over time as a percentage of awards each year grouped by Organisation type (Group of Eight, ATN, and Other)

6.6. Funding

We explored any relationship that might exist between the amount of an ARC research grant and the probability that the grant produced software. We found no significant relationship between funding amount and rate of software produced.

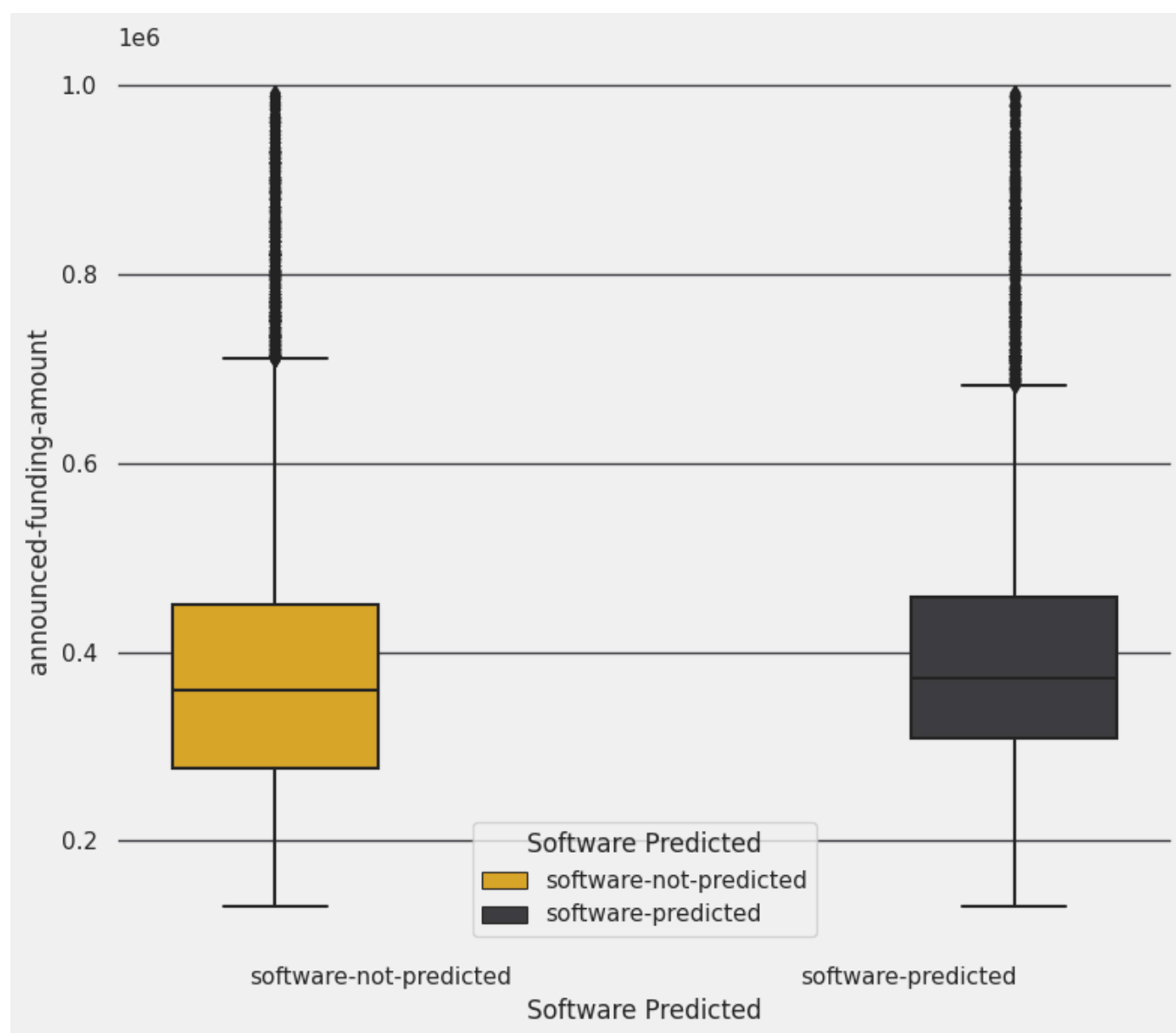


Figure 10. Distribution of funding award funding amounts grouped by model prediction

7. Implications for Policy

Our analysis provides a quantitative estimate of ARC funded research software production. We found that across organisations, fields, and time – software is increasingly produced by ARC-funded researchers. We found no significant difference between the amount of money received and the probability that a researcher will produce software, and no evidence that a researcher at a more ‘prestigious’ university will produce software. In short, software production is distributed across fields and organisations relatively evenly, and is increasing over time.

We believe our findings add important context to the undersupply of research software support, the difficulty of discovering and reusing research software, and the necessary sociotechnical infrastructures to meaningfully sustain research software. Next we attempted to situate our findings amongst previous ARDC reports in order to recommend concrete next-steps for realising a national agenda for research software in Australia.

- A 2021 survey by Barker and Buchhorn estimated there were 6,000 people supporting research software in Australia. This figure represented a fourfold increase over estimates of research software support from 2019. There were, in estimate, 4,000 funded positions including 2,500 full-time equivalents supporting software. The survey also notes that an additional 2,000 individuals were in research software support roles which would represent a tenfold growth since 2019.
- Stevens (2022) surveyed Australian researchers about how they discover, evaluate, and reuse research software. Participants reported difficulty in efficiently locating relevant software tools and the ability to meaningfully access software that was referenced (or mentioned) in a research publication. When evaluating research software for potential reuse most software producers (coders) used difficulty of access, price, and quality of documentation as key decision making criteria.
- Ram and Howison (2023) convened focus groups and produced empirical use cases around the need for infrastructural support of research software use and production. To overcome some structural issues in sustaining research software they proposed adopting software bill of materials (SBOM) documentation to detail dependencies, a software infrascopes to illuminate adoption trends, web analytics for analysing the impact of software served from websites, and linking software with data catalogues. These solutions responded to consistent calls for transparency around development and usage while also targeting reproducibility, assessment of community health, and discovery through enhanced connections between outputs and applications.

Our work reinforces the need for better documentation, discoverability, and management of research software described in both reports. We build upon these findings in making four recommendations for future research and policy development for sustainable software in Australia's National Agenda:

Require software production descriptions (SPD) in grant applications - such documentation can be lightweight, easy to complete, and produce important guarantees to funders about the delivery of promised software. SPDs might also help ensure that research software work is not invisible labour nor redundant with previous efforts. In Appendix 1 we provide a template for SPDs to conceptualise what this type of policy intervention might look like and require of grant applicants. We view SPDs as being a continuously updated document that could also incorporate a software registry, software bill of materials, and other software management plans.

Require awardees to report software as a grant outcome - Recognizing that important basic and applied research contributions are made through software development can both surface potentially valuable ARC funded research products to be reused, but can also provide a knowledge base for evaluating software production descriptions (SPDs). We also note that research software outcomes are commonly required by other funding agencies such as the NIH and NSF in the USA, and the UKRI (see 'what to report PDF') in the UK.

Conduct a national research software census - This would activate a network of researchers and support staff to both identify and document research software, and help ARC better understand where future research software funding should be directed for sustainability. Building upon the proposal by Ram and Howison (2023), this might also enable linking more ARC-funded research outcomes together such as datasets and publications, and in turn give a more holistic view of the research impact of grant funding.

Identify workforce needs in research software engineering and support - Over the past decade (2010-2019) we show a positive correlation between software produced and grant making. Barker and Buchhorn's estimate that there has been a tenfold growth in unfunded research software support since 2019 appears troubling. The ARC has not increased their funding levels tenfold between 2019-2021, and from our analysis we have no reason to believe that software production has increased tenfold during this period. The significant increase in individuals supporting and producing software may speak to increasing complexity, challenges in ongoing software maintenance (e.g. technical debt), or simply a change in labour needs of research organisations. Future work might try to better understand both research software support - in roles such as Research Software Engineer (RSE) - and student employment under grant-funded research. Accounting for where software labour is coming from and what it is going towards would provide a clearer picture of the differences we see between a sharp increase in the number of people working on research software and the relatively steady increase in the amount of software being produced.

8. Appendix 1 - Examples of Model Prediction

8.1. 5 Highest Software Predicted Probability Scores for ARC Grants - 2010-19

Grant: DP190102363

Title: Classical and quantum invariants of low-dimensional manifolds.

Abstract: This project aims to advance our understanding of knots and 3-dimensional spaces, which arise naturally in fields as diverse as physics, computer graphics, chemistry and biology. Recent ideas from quantum field theory link physics to topology in dimensions 3 and 4, leading to powerful invariants of knots and 3-dimensional manifolds that include the Jones polynomial and the 3D-index. This project aims to resolve key questions relating these quantum invariants to classical topology and geometry. The project will have a major impact in low-dimensional topology, and lead to deep and unexpected connections between mathematics and mathematical physics.

Software Predicted Probability: 0.881

Software Not Predicted Probability: 0.119

Grant: DP130103694

Title: Triangulations in dimensions 3 and 4: discrete and geometric structures.

Abstract: Recently there have been spectacular advances in understanding 3-dimensional spaces and the interaction between ideas in mathematical physics (quantum invariants) and such spaces. This project aims at practical methods for finding geometric structures and advancing our understanding of the information that physics is providing about these spaces.

Software Predicted Probability: 0.868

Software Not Predicted Probability: 0.132

Grant: DP160104502

Title: Invariants, geometric and discrete structures on manifolds.

Abstract: This project aims to develop practical methods for finding geometric and discrete structures on manifolds in both low and high dimensions and advancing our understanding of the information that physics is providing about these spaces. Recently there have been spectacular advances in understanding 3-D spaces and the interaction between ideas in mathematical physics (quantum invariants, string theory) and such spaces.

In this project, the first aim is to construct structures with good geometric properties on 3- and 4-manifolds, using triangulations. The second aim is to study combinatorial decompositions of n -manifolds, using our new technique of multisections and also searching for polyhedral metrics of non-positive curvature. The third aim is to connect quantum invariants and geometric structures, again using triangulations.

Software Predicted Probability: 0.834

Software Not Predicted Probability: 0.166

Grant: FT120100943

Title: Algorithmics for visual analytics of massive complex networks.

Abstract: The project will provide new scalable algorithms for visual analytics of massive complex networks. These fast algorithms will enable security analysts to detect abnormal behaviours such as money laundering, biologists to understand protein-protein interaction networks, and support software engineers new ways of understanding large software systems.

Software Predicted Probability: 0.829

Software Not Predicted Probability: 0.171

Grant: DP150100294

Title: Advanced three-dimensional Computer Vision Algorithms for 'Find and Grasp' Future Robots.

Abstract: This project addresses crucial limitations of existing vision systems for the robot grasping of irregular objects in messy living environments. This project aims to undertake fundamental research into novel three-dimensional vision algorithms, exploiting multiple modalities (two-dimensional+three-dimensional+video) for scene labelling, object classification, scene segmentation and grasp synthesis to enable future robots to operate in unstructured environments with highly occluded and cluttered objects. It is expected to significantly advance research and to have broad applications, including home robotics to improve the quality of life of elders and people with special needs. These algorithms may also be used in security (explosive manipulation) and agriculture (field crop harvesting).

Software Predicted Probability: 0.827

Software Not Predicted Probability: 0.173

8.2. 5 Lowest Software Predicted Probability Scores - 2010-19

Grant: DE180100682

Title: Do teacher research experiences affect student outcomes in science?

Abstract: This project aims to determine whether authentic scientific research projects undertaken by teachers translate into improved science-related outcomes for students. Measures to redress declining levels of understanding and interest in science among Australian secondary students over more than two decades have had little effect. Using a cross-national multiple case study design, the project plans to generate new knowledge about improving teachers' scientific literacy and classroom practice that translates to improved student understanding of, and engagement with, science. This will afford significant social and economic benefits through targeted research-informed education strategies that meet the needs of Australia's future STEM workforce.

Software Predicted Probability: 0.187

Software Not Predicted Probability: 0.813

Grant: DP1096184

Title: Environmental stress indicators in coral skeletons.

Abstract: Coral reefs are critical for Australia's tourism and fisheries industries, cultural heritage and international conservation responsibilities. The proposed research will test and document two newly identified stress indicators in corals, one of which will allow stress to be documented by visual inspection on living reef flats. Both new techniques will allow documentation of historical records of stress events, thus improving understanding of reef dynamics through intervals of climate change, and importantly, they also may help detect 'early warning signs' of poor health in living reef corals. Thus, the research will inform both palaeoclimate studies and current reef management strategies.

Software Predicted Probability: 0.202

Software Not Predicted Probability: 0.798

Grant: DP1094932

Title: A mechanistic understanding of coral reef recovery.

Abstract: This project will provide the scientific basis to inform management policies to promote and maintain healthy coral reefs, both in Australia and overseas, which are suffering through climate change impacts. This work, which contributes directly to National Research Priority An Environmentally Sustainable Australia, will provide environmental benefits through understanding how degraded reefs can recover. The Great Barrier Reef alone is worth more than \$6 billion in tourism and fisheries revenue, and understanding how to maintain healthy coral reefs will contribute to the long-term sustainable growth of these industries. It will also help ensure continued use and provision of reef goods and services to coastal communities in tropical Australia.

Software Predicted Probability: 0.206

Software Not Predicted Probability: 0.794

Grant: FS110200034

Title: A Changing Climate on the Great Barrier Reef: Present and Future Implications.

Abstract: The Great Barrier Reef is fundamental to the economy of Australia. This national and international icon needs to be preserved in the face of a changing world to ensure on-going sustainability of our marine resources. Ocean acidification, warming water temperatures, increased freshwater disrupt the sensitive symbiotic association of corals the major structure building organisms of reefs. Understanding how these environmental stressors result in the decrease in coral health is fundamental to prevent loss of our coral reefs and an important step towards preserving them for future generations.

Software Predicted Probability: 0.206

Software Not Predicted Probability: 0.794

Grant: DE130100572

Title: Understanding the ecological and economic implications of reef fish larval dispersal.

Abstract: Until we understand larval dispersal, the movement of reef fish during their juvenile stage, we cannot sustainably manage coral reef ecosystems. This project will use sophisticated mathematical tools to understand how larval dispersal influences the ecology and management of the Great Barrier Reef and a fishery in Papua New Guinea.

Software Predicted Probability: 0.207

Software Not Predicted Probability: 0.793

9. Appendix 2 - Template for a Research Software Development Plan (SDP)

The Australian Research Council (ARC) recognizes the growing importance of software in research across all disciplines. In line with this, the ARC encourages the development of research software as part of grant applications. However, to ensure responsible and effective use of public funds, proposals involving significant software development must provide a clear and compelling justification for this approach.

Justification Requirements: Applicants seeking ARC funding for research software development should provide a comprehensive justification demonstrating the following:

1. Need:
 - a. Clearly define the research problem or challenge that the software will address. Explain why existing software solutions are inadequate or unavailable.
 - b. Demonstrate the potential impact of the proposed research software on the field. Explain how it will advance knowledge, improve research methods, or address critical research questions.
 - c. Outline the intended users of the software. Specify the research community or other stakeholders who will benefit from its development.
2. Technical Feasibility:
 - a. Provide a detailed technical description of the proposed software. This should include the software architecture, development methodology, planned functionalities, and desired technical specifications.
 - b. Explain the technical expertise available to the research team. Demonstrate the team's capacity to develop and maintain the proposed software.
 - c. Describe the development timeline and resources required. Provide a realistic and well-defined plan for software development, including milestones and deliverables.
3. Cost-Benefit Analysis:
 - a. Compare the costs of developing new software versus acquiring existing solutions. Consider factors such as development and maintenance costs, licensing fees, customization requirements, and potential cost savings through increased efficiency or improved research capabilities.
 - b. Quantify the expected benefits of the software. Estimate the value it will deliver in terms of increased research productivity, improved data analysis, or enhanced research collaboration.

- c. Demonstrate the long-term sustainability of the software. Explain how the research team will ensure ongoing maintenance, support, and updates after the grant funding period.
4. Open Access and Dissemination:
 - a. Commit to making the research software openly accessible to the research community. This may involve sharing the source code through public repositories, publishing documentation, and providing user support.
 - b. Describe the plan for disseminating the software and associated research results. Explain how you will promote the software's adoption and encourage its use by other researchers.
5. Ethical Considerations:
 - a. Address any potential ethical concerns associated with the software. This may include issues related to data privacy, intellectual property, or responsible use of technology.
 - b. Outline the measures you will take to mitigate these risks and ensure ethical development and deployment of the software.

Submission Guidelines

Justifications for research software development should be clearly identified and separate from the main research proposal. Applicants should ensure their justification adheres to the following:

- **Clarity and conciseness:** Present the information in a clear and concise manner, avoiding unnecessary technical jargon.
- **Adequate evidence:** Back up claims with data, references, and supporting evidence.
- **Alignment with ARC priorities:** Explain how the software development aligns with the ARC's strategic priorities and funding goals.

Potential Evaluation Criteria

- ARC reviewers could evaluate justifications for research software development based on the following criteria:
 - Significance of the research problem addressed by the software.
 - Technical feasibility and expertise of the research team.
 - Cost-effectiveness of developing the software compared to existing solutions.
 - Potential impact and benefits of the software for the research community.
 - Commitment to open access and dissemination of the software.
 - Consideration of ethical implications and mitigation strategies.




Works Cited

- Australian Research Data Commons. (2022). A National Agenda for Research Software (1.0). Zenodo. <https://doi.org/10.5281/zenodo.6378082>
- Barker, M., & Buchhorn, M. (2022). Research Software Capability in Australia. Zenodo. <https://doi.org/10.5281/zenodo.6335998>
- Brown, E. M., Schwartz, L., Huang, R. L., & Weber, N. (2023). Soft-Search: Two Datasets to Study the Identification and Production of Research Software. *arXiv preprint arXiv:2302.14177*.
- Coulson, D. Goodman., A Hodgson., C and, Neumann. W. (2000) "Computing arithmetic invariants of 3-manifolds." *Experiment. Math.* 9 (1) 127 - 152.
- Goodman, O., Heard, D., & Hodgson, C. (2008). Commensurators of cusped hyperbolic manifolds. *Experimental Mathematics*, 17(3), 283-306.
- Heard, D., Hodgson, C., Martelli, B., & Petronio, C. (2010). Hyperbolic graphs of small complexity. *Experimental Mathematics*, 19(2), 211-236.
- Ram, K. & Howison, J. (2003). Research Software Visibility Infrastructure Priorities Report. <https://doi.org/10.5281/zenodo.8404846>
- Stevens, F. (2022). Understanding How Researchers Find Research Software for Research Practice. <https://doi.org/10.5281/zenodo.7340033>






Australian Research Data Commons

CONTACT

-  ardc.edu.au
-  +61 3 9902 0585
-  contact@ardc.edu.au

FOLLOW

-  [@ardc_au](https://twitter.com/ardc_au)
-  [australian-research-data-commons](https://www.linkedin.com/company/australian-research-data-commons)
-  [subscribe to our newsletter](#)



Australian Government



ARDC is enabled by NCRIS