# Deep Learning does not Replace Bayesian Modeling

## Comparing research use via citation counting

Breck Baldwin

8/20/2021

# 1 Introduction

Stan, a Bayesian modeling language, was released in 2012 to considerable fanfare. A shiny new inference algorithm, HMC with NUTS ([1]), promised and delivered fits that could not be fit before. And then deep learning happened. The release of TensorFlow ([2]) opened the doors of deep learning to all and by it 2016 began getting significant traction. One result of this was people often assumed that deep learning's successes usurped Bayesian modeling's domain. This is not in our collective imagination–Bayesians like to 'believe things' after all–NSF reviews came back dismissing Stan funding because all the interesting work was assumed to be happening with deep learning. Recently in the UK, open skepticism was expressed about impact claims for Bayesian software in response to a research grant. Mind you, deep learning and Bayesian modeling are conceptual cousins but in the end are very different from each other. They are better thought of as complementary than as antagonistic. Yet Bayesians found themselves in deep learning's shadow somehow.

It is 2021 and this document does some very simple analysis around use of Bayesian and deep learning packages as evidenced in the research literature to get a perspective on what actually happened and is happening. The comparison aims to approach the following goals with very simple research citation metrics:

1. How does Bayesian modeling software stack up against deep learning without appeal to feature comparisons, performance arguments on suspect data sets or achieved closeness to Platonic ideals? Just go out and count how many citations the respective approaches have in the research literature. Counting and categorization, that's it.
2. Asses the impact of Bayesian modeling software using deep learning metrics as a yard stick–how big a fraction of a huge thing are we?
3. Contextualize the roles each approach has by looking at subject distributions. The technologies have very different use cases so one would expect variation.

Citation counting is a crude metric but it has the advantage of simplicity. In compiling these metrics I came away with very different opinion than I started with so I thought it worth sharing. My prior assumptions were that Bayesian modeling was very niche and scurrying around doing very useful and important science but niche none the less. This analysis led me to revise that opinion considerably.

For the purposes of this document we take Bayesian software to be Stan ([3]) and PyMC3 ([4]) with ecosystem components included that are likely to be cited. That includes the simplified syntax interfaces to Stan, brms ([5]), RstanArm ([6]), and interface packages to Stan, RStan ([7]) and PyStan ([8])[1]. The analysis applies only to packages that are in current development so the venerated, and very high impact, packages like BUGS ([9]) and JAGS ([10]) are not considered although including them would double our counts. There is also the best named Bayesian package of all time, "emcee: The MCMC Hammer," ([11]) which enjoys tremendous use in the astrophysics community but is too specialized to be considered a general Bayesian package so it is not included. Also, the ecosystems are actually much larger but they are unlikely to be cited in the research literature and we have to stop some place. For deep learning we take TensorFlow, its interface Keras ([12]) and PyTorch ([13]) as roughly equivalent entities for the comparison. Theano ([14]) is no longer in development. It should be noted that both PyTorch and TensorFlow have implemented HMC with NUTS for Bayesian inference but those are recent developments that have not made much of an impact yet.

The key resource behind this document is Elsevier's https://scopus.com (https://scopus.com) research search engine that provides a tightly curated[2] search index that includes sources outside of the academic behemoth's own journals. It also provides a solid API (Application Programmer Interface) and a classification of journals into subject areas. The actual form of Scopus queries is discussed below which should allow the questioning reader to verify and update the counts for the various packages discussed.

# 2 High level prevalence of Deep Learning vs Bayesian Modeling

In the process of grant writing I create metrics to help justify projects, lately in partnership with PyMC and ArviZ through the scientific fiscal sponsor NumFOCUS of which Stan is a member as well. Since Bayesian modeling seems always in the shadow of deep learning I started tracking deep learning software packages as well for comparison. Below we see the relative citations of the top deep learning packages TensorFlow, PyTorch and the support package Keras to the Bayesian packages PyMC3, Stan with support/derivative packages RStan, RStanArm, PyStan and brms.

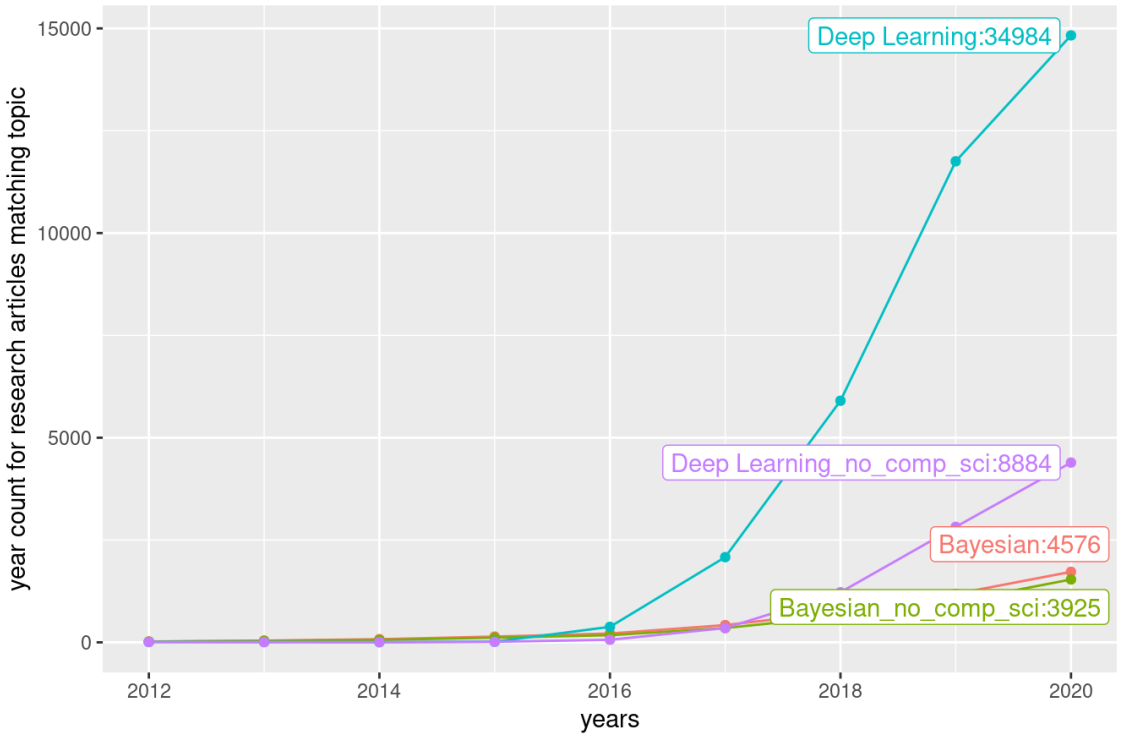


Fig 1: Relative counts with and without computer science journals

Reading from top of Fig 1, assuming raw research citations counts matter, Bayesian packages are overwhelmed by deep learning packages with a count of nearly 35,000 citations to a Bayesian count of 4,600–nearly a 10x difference. One could excuse a bit of Bayesian despondency in the face of the wave depicted above–this ratio supports my expectations. But graph also shows relative counts for research citations outside of computer science which drops the count of deep learning from 35k to 8.8k–a 75% drop. That drop IS a surprise. I didn't have a prior on deep learning's distribution across subject categories but I didn't expect it to be so skewed. The Bayesian ecosystem shows a total of 4.5k of which 14% are computer science–that ratio does not surprise me.

# 3 Subject Category Breakdowns

So what is going on here? The next question is what does the the distribution look like for non-computer science subject areas? Below is a table of subject counts for packages starting in 2006 when PyMC was first released to 2021 with relative percentages listed:

| | Bayesian | Deep Learning | totals | Stan | PyMC | PyStan | RStanArm | RStan | brms | PyTorch | Keras | TensorFlow | detail totals |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Agricultural and Biological Sciences | 1275/61% | 820/39% | 2095 | 821/29% | 94/3% | 3/0% | 114/4% | 365/13% | 440/15% | 102/4% | 397/14% | 534/19% | 2870 |
| Arts and Humanities | 385/23% | 1305/77% | 1690 | 209/10% | 17/1% | 1/0% | 28/1% | 78/4% | 204/10% | 389/19% | 365/18% | 729/36% | 2020 |

| | Bayesian | Deep Learning | totals | Stan | PyMC | PyStan | RStanArm | RStan | brms | PyTorch | Keras | TensorFlow | detail totals |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Biochemistry, Genetics and Molecular Biology | 752/25% | 2232/75% | 2984 | 451/12% | 133/4% | 12/0% | 63/2% | 120/3% | 213/6% | 407/11% | 951/25% | 1424/38% | 3774 |
| Business, Management and Accounting | 92/13% | 598/87% | 690 | 59/7% | 11/1% | 0/0% | 7/1% | 20/2% | 21/3% | 107/13% | 216/26% | 397/47% | 838 |
| Chemical Engineering | 51/5% | 933/95% | 984 | 19/2% | 27/2% | 0/0% | 1/0% | 5/0% | 3/0% | 184/15% | 383/31% | 602/49% | 1224 |
| Chemistry | 125/8% | 1370/92% | 1495 | 55/3% | 46/2% | 1/0% | 10/1% | 12/1% | 21/1% | 295/16% | 551/30% | 866/47% | 1857 |
| Computer Science | 727/2% | 29365/98% | 30092 | 463/1% | 233/1% | 15/0% | 22/0% | 76/0% | 100/0% | 7420/21% | 9058/25% | 18178/51% | 35565 |
| Decision Sciences | 314/10% | 2702/90% | 3016 | 260/7% | 21/1% | 1/0% | 20/1% | 77/2% | 28/1% | 504/14% | 988/27% | 1766/48% | 3665 |
| Dentistry | 0/0% | 19/100% | 19 | 0/0% | 0/0% | 0/0% | 1/4% | 0/0% | 1/4% | 1/4% | 11/44% | 11/44% | 25 |
| Earth and Planetary Sciences | 481/23% | 1587/77% | 2068 | 159/6% | 300/12% | 22/1% | 11/0% | 41/2% | 19/1% | 289/11% | 690/27% | 1003/40% | 2534 |
| Economics, Econometrics and Finance | 97/45% | 118/55% | 215 | 70/27% | 7/3% | 0/0% | 4/2% | 23/9% | 18/7% | 14/5% | 60/23% | 68/26% | 264 |
| Energy | 54/4% | 1330/96% | 1384 | 26/2% | 20/1% | 1/0% | 1/0% | 6/0% | 9/1% | 124/7% | 606/36% | 883/53% | 1676 |
| Engineering | 394/3% | 14323/97% | 14717 | 198/1% | 175/1% | 5/0% | 10/0% | 43/0% | 35/0% | 2918/17% | 5051/29% | 9090/52% | 17525 |
| Environmental Science | 790/45% | 949/55% | 1739 | 501/22% | 77/3% | 1/0% | 69/3% | 226/10% | 249/11% | 103/4% | 480/21% | 586/26% | 2292 |
| Health Professions | 94/14% | 593/86% | 687 | 62/7% | 5/1% | 1/0% | 6/1% | 21/2% | 42/5% | 143/17% | 234/28% | 331/39% | 845 |
| Immunology and Microbiology | 230/62% | 141/38% | 371 | 141/30% | 28/6% | 3/1% | 11/2% | 46/10% | 73/15% | 21/4% | 75/16% | 74/16% | 472 |
| Materials Science | 84/3% | 3069/97% | 3153 | 34/1% | 47/1% | 1/0% | 0/0% | 5/0% | 5/0% | 673/18% | 1127/30% | 1905/50% | 3797 |
| Mathematics | 753/9% | 7878/91% | 8631 | 607/6% | 114/1% | 6/0% | 31/0% | 156/2% | 47/0% | 1928/19% | 2532/25% | 4812/47% | 10233 |
| Medicine | 887/22% | 3062/78% | 3949 | 572/12% | 88/2% | 6/0% | 71/1% | 154/3% | 286/6% | 548/11% | 1393/28% | 1793/37% | 4911 |
| Multidisciplinary | 330/36% | 594/64% | 924 | 181/16% | 41/4% | 3/0% | 27/2% | 76/7% | 111/10% | 93/8% | 251/22% | 378/33% | 1161 |
| Neuroscience | 563/38% | 910/62% | 1473 | 321/17% | 70/4% | 5/0% | 35/2% | 85/5% | 239/13% | 219/12% | 326/18% | 538/29% | 1838 |
| Nursing | 25/69% | 11/31% | 36 | 16/33% | 0/0% | 0/0% | 3/6% | 1/2% | 15/31% | 1/2% | 5/10% | 8/16% | 49 |
| Pharmacology, Toxicology and Pharmaceutics | 74/33% | 148/67% | 222 | 60/21% | 5/2% | 0/0% | 4/1% | 16/6% | 8/3% | 26/9% | 67/24% | 97/34% | 283 |
| Physics and Astronomy | 494/9% | 4812/91% | 5306 | 134/2% | 324/5% | 31/0% | 8/0% | 23/0% | 31/0% | 906/14% | 1928/30% | 3062/47% | 6447 |
| Psychology | 798/87% | 119/13% | 917 | 461/37% | 32/3% | 1/0% | 62/5% | 159/13% | 397/32% | 18/1% | 51/4% | 75/6% | 1256 |
| Social Sciences | 695/23% | 2330/77% | 3025 | 437/12% | 44/1% | 3/0% | 48/1% | 145/4% | 295/8% | 474/13% | 806/22% | 1474/40% | 3726 |
| Veterinary | 35/81% | 8/19% | 43 | 21/37% | 0/0% | 0/0% | 5/9% | 12/21% | 7/12% | 0/0% | 6/11% | 6/11% | 57 |

Table 1 shows the basic subject split between Bayesian modeling and deep learning with total for the two categories, following are the individual package counts with a 'detail totals' as well for the individual packages. Note that the totals do not match because an article can match more than one package and as result gets contributes to the count of multiple packages. On average an article's journal is classified into two subject categories.

Staring with the subject that got us here we see that computer science has 727 Bayesian publications vs 29,365 deep learning publications for 98% of the share of 30,092 total. The left most two columns are the same queries used for the graph in Fig 1 with the total in column 3. Columns 4-12 are individual packages with percentage of the total in the 13th column. Sticking with computer science, one sees that TensorFlow has 18,178 references with 51% of the count–note that the total counts are higher because an article can matched by more that one query, each package gets a query, so the article count is 3 if Keras, PyMC and TensorFlow queries match the references and it has a computer science subject classification.

I leave the tea leaf interpretation to others but I'll make the observation that Bayesians are playing a pretty big game in a lot of areas of science and as someone interested in getting funding to Bayesian software this is a pretty strong argument that we are highly relevant.

# 4 Details of the queries

Below are the queries used to identify the packages. Direct citation linking does not work well with these packages since many of them had no journal publication to cite in a referring article. As a result a string search was used to identify the citations in the reference section which was determined by Scopus.com's document parser. I would estimate a 5% false positive rate based based on informal examination of return sets.

For an example of how the queries were developed, the stand alone query 'rstan' had the addition of rejecting articles that mentioned 'mit' because there was a common reference to 'http://wwwmath.mit.edu/~rstan/ec/ (http://wwwmath.mit.edu/~rstan/ec/)' that existed prior to 2012 when Stan was released. The name 'Stan' presented obvious difficulties, see https://statmodeling.stat.columbia.edu /2019/04/29/we-shouldntve-called-it-stan-i-shouldve-listened-to-bob-and-hadley/ (https://statmodeling.stat.columbia.edu/2019/04 /29/we-shouldntve-called-it-stan-i-shouldve-listened-to-bob-and-hadley/). The query shown achieved desired count drop off at 2012 with reasonable precision and each of the 'OR' terms contributed to coverage significantly. How the word 'mc-stan.org' was tokenized is unknown but there were no results before 2013 for the solo query.

| | |
|---|---|
| Stan | REF((gelman AND hoffman AND stan) OR mc-stan.org) |
| brms | REF(brms AND burkner) |
| PyStan | REF(pystan) |
| RStanArm | REF(rstanarm) |
| RStan | REF(rstan AND NOT mit) |
| PyMC | REF(PyMC3 OR (PyMC* AND fonnesbeck)) |
| TensorFlow | REF(tensorflow) |
| PyTorch | REF(pytorch) |
| Keras | REF(Keras) |
| Bayesian | REF(((gelman AND hoffman AND stan) OR mc-stan.org) OR (brms AND burkner) OR (pystan) OR (rstanarm) OR (rstan AND NOT mit) OR (PyMC3 OR (PyMC* AND fonnesbeck))) |
| Deep Learning | REF((tensorflow) OR (pytorch) OR (Keras)) |

PyMC presented some challenges in that the string PyMC was referenced in early publications but also is ambiguous with chemical terms, so like Stan, author names were included to restrict search as was the case with 'brms.' These queries were initially developed to search entire documents rather than the current restriction to the references section of papers. I would expect the false positive rate to be even lower.

The above queries can be run with the advance search option at https://scopus.com (https://scopus.com) and are generated from the

API queries actually used where '+' was replaced with ' ' above. An account is required which is included for many education institutions.

# 5 Conclusions

As often happens conducting a closer look reveals subtleties. Deep learning dominates the computer science part of science and obviously it dominates in the commercial sector which this article does not address. However the remaining parts of science show a more balanced distribution across the approaches. This makes sense since the approaches are very different and serve different goals.

As a Bayesian advocate I'd like to leave the funding agencies with the observation that Bayesian software carries roughly one third of the research load by research citation count (3925/(3925+8884)) without the backing of Fortune 500 companies like Facebook and Google. Just recently I learned of a funding denial because the funders didn't believe that Bayesian software could have the impact that was being claimed–that opinion could well be self fulfilling in the future but as of now it is not justified.

# 6 Access to source/scripts and acknowledgements

Supporting technology for this document includes the R language ([15]), the 'tidyverse' group of packages ([16]) that give data and visualization tools and this Rmarkdown document ([17]) rendered with 'bookdown' ([18]) developed using the Rstudio ([19]) IDE (Integrated Development Environment). Supporting package include 'kable-extra' ([20]), a table formatter, the 'scales' ([21]) package for converting to percentages, 'ggrepel' ([22]) for labeling line graphs with dynamically shifting labels, 'jsonlite' ([23]) for JSON data parsing, 'httr' ([24]) for GET requests, 'stringr' ([25]) for regular expression matching and 'redux' ([26]) for access to the Redis server ([27]) which I highly recomend using as a caching layer for this sort of project.

I'd like to thank Andrew Gelman and the Laplace lab at Columbia University for support.

Oriol Abril Pla brought up emcee as a popular Bayesian system worth consideration which was a big oversight on my part as well as providing insightful commentary.

I intend this to be an evolving document with periodic updates at it's github repo at https://github.com/breckbaldwin /ScientificSoftwareImpactMetrics (https://github.com/breckbaldwin/ScientificSoftwareImpactMetrics) where it lives as `DeepLearningAndBayesianSoftware.Rmd` , is rendered as `DeepLearningAndBayesianSoftware.html` and is viewable as https://breckbaldwin.github.io/ScientificSoftwareImpactMetrics/DeepLearningAndBayesianSoftware.html (https://breckbaldwin.github.io/ScientificSoftwareImpactMetrics/DeepLearningAndBayesianSoftware.html). Note that github pages changes urls around in odd ways for the rendered view of hte html page. All source code is viewable in the .Rmd document.

I can be reached at breckbaldwin@gmail.com (mailto:breckbaldwin@gmail.com) or via the github issues on the repository.

# References

1. Hoffman MD, Gelman A. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *J Mach Learn Res*. 2014;15(1):1593-1623.
2. Abadi M, Barham P, Chen J, et al. Tensorflow: A system for large-scale machine learning. In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*.; 2016:265-283.
3. Stan Development Team. Stan. Published online 2012. http://mc-stan.org (http://mc-stan.org)
4. Salvatier J, Wiecki TV, Fonnesbeck C. Probabilistic programming in python using PyMC3. *PeerJ Computer Science*. 2016;2:e55.
5. Bürkner P-C. brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software*. 2017;80(1):1-28. doi:10.18637/jss.v080.i01 (https://doi.org/10.18637/jss.v080.i01)
6. Goodrich B, Gabry J, Ali I, Brilleman S. Rstanarm: Bayesian applied regression modeling via Stan. Published online 2020. https://mc-stan.org/rstanarm (https://mc-stan.org/rstanarm)
7. Stan Development Team. RStan: The R interface to Stan. Published online 2020. http://mc-stan.org/ (http://mc-stan.org/)
8. Stan Development Team. PyStan: The Python interface to Stan. Published online 2020. http://mc-stan.org/ (http://mc-stan.org/)
9. Lunn D, Jackson C, Best N, Thomas A, Spiegelhalter D. The BUGS book. *A Practical Introduction to Bayesian Analysis, Chapman Hall, London*. Published online 2013.
10. Plummer M, others. JAGS: A program for analysis of bayesian graphical models using gibbs sampling. In: *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*. Vol 124. Vienna, Austria.; 2003:1-10.

11. Foreman-Mackey D, Hogg DW, Lang D, Goodman J. Emcee: The MCMC hammer. *Publications of the Astronomical Society of the Pacific*. 2013;125(925):306.
12. Chollet F, others. Keras. Published online 2015.
13. Paszke A, Gross S, Massa F, et al. PyTorch: An imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, dAlché-Buc F, Fox E, Garnett R, eds. *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc.; 2019:8024-8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf)
14. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. Published online 2020. http://arxiv.org/pdf/1605.02688.pdf (http://arxiv.org/pdf/1605.02688.pdf)
15. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing; 2018. https://www.R-project.org/ (https://www.R-project.org/)
16. Wickham H, Averick M, Bryan J, et al. Welcome to the tidyverse. *Journal of Open Source Software*. 2019;4(43):1686. doi:10.21105/joss.01686 (https://doi.org/10.21105/joss.01686)
17. Xie Y, Dervieux C, Riederer E. *R Markdown Cookbook*. Chapman; Hall/CRC; 2020. https://bookdown.org/yihui/rmarkdown-cookbook (https://bookdown.org/yihui/rmarkdown-cookbook)
18. Xie Y. *Bookdown: Authoring Books and Technical Documents with R Markdown*. Chapman; Hall/CRC; 2016. https://bookdown.org/yihui/bookdown (https://bookdown.org/yihui/bookdown)
19. RStudio Team. *RStudio: Integrated Development Environment for r*. RStudio, PBC.; 2020. http://www.rstudio.com/ (http://www.rstudio.com/)
20. Zhu H. *kableExtra: Construct Complex Table with 'Kable' and Pipe Syntax*.; 2021. https://CRAN.R-project.org/package=kableExtra (https://CRAN.R-project.org/package=kableExtra)
21. Wickham H, Seidel D. *Scales: Scale Functions for Visualization*.; 2020. https://CRAN.R-project.org/package=scales (https://CRAN.R-project.org/package=scales)
22. Slowikowski K. *Ggrepel: Automatically Position Non-Overlapping Text Labels with 'Ggplot2'*.; 2021. https://CRAN.R-project.org/package=ggrepel (https://CRAN.R-project.org/package=ggrepel)
23. Ooms J. The jsonlite package: A practical and consistent mapping between JSON data and r objects. *arXiv:14032805 [statCO]*. Published online 2014. https://arxiv.org/abs/1403.2805 (https://arxiv.org/abs/1403.2805)
24. Wickham H. *Httr: Tools for Working with URLs and HTTP*.; 2020. https://CRAN.R-project.org/package=httr (https://CRAN.R-project.org/package=httr)
25. Wickham H. *Stringr: Simple, Consistent Wrappers for Common String Operations*.; 2019. https://CRAN.R-project.org/package=stringr (https://CRAN.R-project.org/package=stringr)
26. FitzJohn R. *Redux: R Bindings to 'Hiredis'*.; 2018. https://CRAN.R-project.org/package=redux (https://CRAN.R-project.org/package=redux)
27. Redis Development Team. Redis. Published online 2020. http://redis.io (http://redis.io)

---

1. There are also lighter weight interfaces as well interface to other languages that I ignore due to low expected research mentions, see https://https://mc-stan.org/users/interfaces/ (https://https://mc-stan.org/users/interfaces/) for a listing.↩
2. https://scolar.google.com (https://scolar.google.com) genrally yields double the counts for similar queries but no subject classification or API to code against. The increased counts are likely due to inclusion of non-peer reviewed sites like https://arxiv.org (https://arxiv.org)↩