# A meta-analysis of bioinformatics software benchmarks reveals that publication-bias unduly influences software accuracy

Paul P. Gardner[1,2,3]*, James M. Paterson[1,2], Fatemeh Ashari-Ghomi[1,2], Sinan U. Umu[1,2], Stephanie McGimpsey[4,5], Aleksandra Pawlik[6]

**Abstract**

Computational biology has provided widely used and powerful software tools for testing and making inferences about biological data. In the face of increasing volumes of data, heuristic methods that trade software speed for mathematical completeness must be employed. We are interested in whether trade-offs between speed and accuracy are reasonable. Also, what factors are indicative of accurate software?

In this work we mine published benchmarks of computational biology software, we collect data on the relative accuracy and speed of different software and then test to see what factors influence accuracy e.g. speed, author reputation, journal impact or recency.

We found that author reputation, journal impact, the number of citations, software speed and age are not reliable predictors of software accuracy. This implies that useful bioinformatics software is not only the domain of famous researchers, and that any researchers are capable of producing good software. In addition, we found that there exists an excessive number of slow and inaccurate software tools across multiple sub-disciplines of bioinformatics. Meanwhile, there are very few tools of middling accuracy and speed. We hypothesise that a strong publication bias is unduly influencing the publication and development of bioinformatic software tools. In other words, at present software that is not highly ranked on speed and not highly ranked on accuracy is difficult to publish due to editorial and reviewer practices. This leaves an unfortunate gap in the literature upon which future software refinements could be constructed.

[1] *School of Biological Sciences, University of Canterbury, Christchurch, New Zealand.*
[2] *Biomolecular Interaction Centre, University of Canterbury, Christchurch, New Zealand.*
[3] *Bio-Protection Research Centre, University of Canterbury, Christchurch, New Zealand.*
[4] *McConomy School of Dance, Derry, Ireland.*
[5] *Research for Good, Limavady, Ireland.*
[6] *New Zealand eScience Infrastructure, 49 Symonds St, Auckland, New Zealand.*
***Corresponding author**: paul.gardner@canterbury.ac.nz

## Background

Computational biology software is widely used and has produced some of the most cited publications in the scientific corpus [1, 2, 3]. This software includes implementations of methods for sequence alignment and homology inference [4, 5, 6, 7], phylogenetic analysis [8, 9, 10, 11, 12], statistical analysis of survival patterns in biomedicine [13, 14], biomolecular structure analysis [15, 16, 17, 18, 19], visualization and data collection [20, 21]. However, the popularity of a software tool does not necessarily imply that it is either accurate or computationally efficient, instead usability, ease of installation, operating system and other factors may play a greater role.

Progress in the biological sciences is increasingly limited by the ability to analyse increasing volumes of data, therefore the dependence of biologists on software is also increasing [22]. There is an increasing use of technological solutions for automating biological data generation (e.g. next-generation sequencing, mass-spectroscopy, cell-tracking and species tracking), therefore the biological sciences have become increasingly dependent upon computational software for processing large quantities of data [22]. As a consequence, the computational efficiency of data processing and analysis software is of great importance to decrease the energy and time costs of research [23]. Furthermore, even small error rates can have a major impact on the number of false inferences as datasets become larger [24].

The gold-standard for determining accuracy is for independent researchers to conduct benchmarks, which can serve a useful role in reducing the over-optimistic reporting of software accuracy [25, 26] and the self-assessment trap [27]. Benchmark studies typically use a number of positive and negative control datasets, predictions can then be partitioned into true or false groups and a variety of metrics can be used to evaluate the performance of different predictions [28, 29]. Some benchmarks now use live, or frequently updated, results to indicate the latest developments in software performance [30, 31, 32]. The aim of these benchmarks is to independently identify tools

that make acceptable compromises in terms of scoring schemes and the resulting potential for false predictions, true predictions and speed, and are therefore suited for wide adoption by the community.

For common computational biology tasks, a proliferation of software-based solutions often exists [33, 34, 35, 36, 37]. While this may generally be a good problem to have, and points to a diversity of options from which practical solutions can be selected, many possible options creates a dilemma for users. In the absence of any recent gold-standard benchmarks, how should scientific software be selected? In the following work, we presume that "biological accuracy" is the most desirable feature of software. Biological accuracy is the degree to which predictions or measurements reflect the truths of biological systems, this is usually determined by comparing results of software to the results of established reference data. These are commonly expert-derived curated datasets. In some fields biological accuracy is very difficult to ascertain, for example, in phylogenetics it is nearly impossible to know the ancestral relationships between organisms. In situations like this, researchers can use a mix of simulated or high-confidence datasets.

A number of possible predictors of software quality are used by the community of computational biology software users. Some accessible, quantifiable and frequently used proxies for identifying high quality software include: **1. Recency:** recently published software tools may have built upon the results of past work or be an update to an existing software. Therefore, these could be more accurate and faster. **2. Wide adoption:** a software tool may be widely used because it is fast and accurate, or because it is well-supported and user-friendly. In fact, "large user base", "word-of-mouth", "wide-adoption", "personal recommendation," and "recommendation from a close colleague," are frequent responses to surveys of "how do scientists select software?" [38, 39, 40]. **3. Journal impact:** high profile journals are run by editors and reviewers who carefully select and curate the best manuscripts. Therefore, high impact journals may be more likely to select manuscripts describing good software [41]. **4. Author/Group reputation:** the key to any project is the skills of the people involved, including maintaining a high collective intelligence [39, 42, 43]. As a consequence, an argument could be made that well respected and high-profile authors will produce better software [44, 45]. **5. Speed:** software is frequently said to trade accuracy for speed. For example, heuristic software such as the popular homology search tool, BLAST, compromise the mathematical guarantee of optimal solutions for more speed [4, 7]. Some researchers may naively interpret this fact as slower software is likely to be more accurate. But speed may also be influenced by the programming language [46], and/or level of hardware optimisation [47, 48]; In general the implementation is likely to have more of an impact (e.g. brute-force approaches versus rapid and sensitive pre-filtering [49, 50]).

Other factors that influence whether a software tool is selected include: whether the documentation is good, user-friendly, word-of-mouth and "used in a similar analysis" [40]. This sort of information is not as readily quantifiable as the above measures. However, citation metrics may be a useful proxy. The word-of-mouth factor may also explain the reason why some software continues to be used, in spite of poor relative performance [51].

In the following study, we explore factors that may be indicative of software accuracy. This, in our opinion, should be one of the prime reasons for selecting a software tool. We have mined the large and freely accessible PubMed database [52] for benchmarks of computational biology software, and manually extracted accuracy and speed rankings for **243** software packages. For each software tool, we have collected measures that may be predictive of accuracy, and may be subjectively employed by the researcher community as a proxy for software quality. These include relative speed, relative age, the productivity and impact of the corresponding authors, journal impact and the number of citations.

## Results

We have collected relative accuracy and speed ranks for **243** distinct software tools. This software has been developed for solving a broad cross-section computational biology tasks. These include software for homology search [53], genome sequence analysis (e.g. read mapping or sequence assembly) [54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67], multiple sequence alignment [68, 69, 70, 71, 72], cell tracking [73], transcriptome analysis [74, 75, 76, 77], RNA interaction prediction [78], protein interactions [79], protein structure prediction [80, 81], epistasis [82], metagenomic analysis [83, 84], repetitive sequence prediction [85], proteomics [86, 87] and phylogenetics [88, 89, 90, 91, 92]. Each software tool was benchmarked in at least one of **43** publications that satisfy the Boulesteix criteria [93]. In brief, the Boulesteix criteria are: 1. the main focus of the article is a benchmark. 2. the authors are reasonably neutral. 3. the test data and evaluation criteria are sensible.

For each of the publications describing these methods, we have (when possible) identified the 2014 journal impact factor (JIF), published by Thomson Reuters [94] and the H5-index published by Google Scholar Metrics. We have collected the H-indices and M-indices [44] for the corresponding authors for each method, and the number of times the publication(s) associated with a method has been cited using Google Scholar (data collected over a 1 month period in early 2016).

We have computed the Spearman's correlation coefficient for each pairwise combination of the mean normalised accuracy and speed ranks, the year published, mean relative age (compared to software in the same benchmarks), journal IF and H5 metrics, the total number of citations, the relative number of citations (compared to software in the same benchmarks) and the maximum H and M indices for the corresponding authors. The results are presented in Figure 1A. We found significant associations between most of the citation-based metrics (journal H5, JIF, citations, relative citations, H-index

and M-index). There is also a strong association between the year of publication, the relative age and many of the citation-based metrics.

We found that author reputation metrics, journal impacts and the age of methods were **not** significantly correlated with either method accuracy or speed (see Figure 1). The strongest association was between accuracy and journal impact factor (Spearman's rho = 0.1, P-value = 0.16). A linear model of these parameters and accuracy also failed to identify a correlation between these (accuracy: R2 = -0.03, P-value = 0.94; speed: R2 = 0.04, P-value = 0.66).

$$\begin{aligned} accuracy = & c_0 + c_1 \times speed + c_2 \times JIF + c_3 \times H5 + \\ & c_4 \times citations + c_5 \times Hindex + \\ & c_6 \times Mindex + c_7 \times relativeAge + \\ & c_8 \times relativeCitations \end{aligned}$$

$$\begin{aligned} speed = & c_0 + c_1 \times accuracy + c_2 \times JIF + c_3 \times H5 + \\ & c_4 \times citations + c_5 \times Hindex + \\ & c_6 \times Mindex + c_7 \times relativeAge + \\ & c_8 \times relativeCitations \end{aligned}$$

To further validate this result, we compute a correlation between speed and accuracy for each benchmark and used weighted sum Z-tests [95]. This also failed to identify a significant relationship (sum Z=-0.1, P-value=0.5).

In order to gain a deeper understanding of the distribution of available bioinformatic software tools on a speed versus accuracy landscape, we ran a Monte Carlo permutation test. The ranks extracted from each benchmark were randomly permuted, generating 10,000 randomized speed and accuracy ranks. In the cells of a $10 \times 10$ grid spanning the normalised speed and accuracy ranks we computed a Z-score for the observed number of methods in a cell, compared to the expected distributions generated by the randomized ranks. The results of which are shown in Figure 2. We identified 21 bins where there was a significant excess or dearth of methods. For example, there was an excess of "slow and inaccurate" software (Z=1.6, P-value=0.05) and "slow and accurate" software (Z=1.9, P-value=0.03). We find that the amount of software classed as "fast and accurate" and "fast and inaccurate" are at approximately the expected proportions based upon the permutation test. The number of significant results is in excess and not due to multiple testing, as the probability of finding 21 of 100 tests significant by chance is low (P-value = $2 \times 10^{-8}$, exact binomial test) [96].

There is a large reduction in the number of software tools that are classed as intermediate in terms of both speed and accuracy based upon our permutation test (Figure 2). The cells corresponding to the four central underrepresented deciles are highlighted (Z = -1.7, -2.1, -2.1 and -2.4, P-values = 0.04, 0.02, 0.02 and 0.008, respectively, reading from top to bottom, left to right). We also tested the relative age of the software tools in significantly over- or under-represented regions of the speed vs accuracy plot. We found that the "slow and inaccurate" methods were generally published earlier than other methods (W = 31, P=0.007, one-tailed Wilcoxon test) (Figure S8).

## Discussion

We have gathered data on the relative speed and accuracies of **243** bioinformatic methods from **43** benchmarks that were published between 2005 and 2016. We show that there is an under-representation of software that has both intermediate levels of accuracy and speed. There may be a number of factors that drive this phenomena. One likely explanation is that bioinformatic software tools suffer from a form of publication bias [97]. Our community of developers, reviewers and editors may be unwilling to publish software that is neither the fastest nor the most accurate (we have anecdotal evidence to support this, N=1). If correct, this is unfortunate. Some problems that lack fast and accurate solutions force researchers to make unnecessary compromises in terms of accuracy and time. If our hypothesis that this underrepresentation is due to publication bias is valid then why is there an enrichment of slow and inaccurate software (P-value=0.05, empirical distributions from permutation tests)? How could these methods be published? We have found that slow and inaccurate software is generally published earlier than alternative methods (P=0.007, one-tailed Wilcoxon test), therefore comparisons were not required to publish these tools and the accuracies may have been over-optimistically reported [98].

We found that no commonly used proxy for study quality is correlated with software accuracy. Neither, author reputation, number of citations, journal impact, relative age or speed appear to be associated with accurate software. Linear mixed models of these values also fail to identify predictors of software accuracy, as do methods for combining P-values. A great deal of criticism has been leveled specifically at journal impact factors [99, 100, 101]. Our finding show that this measure is not reflective of software quality. But neither is the H5 index which is thought to be a more robust measures of journal impact [102]. The poor relationship between accuracy and both author reputation and the number of citations is particularly troubling. Both are related to "word of mouth" and "previously used in a similar analysis" which a recent survey of researchers suggested is a major influence on the selection of software tools [40]. This implies that the recorded high citation rates for bioinformatic software [1, 2, 3] is more a reflection of user-friendliness and the Matthew Effect [101, 103]. The Matthew Effect is a biblical reference that can be paraphrased as "the rich get richer and the poor get poorer". In this context, highly cited software is more likely to be used and cited again, irrespective of relative performance.

The lack of any relationship between software speed and accuracy is surprising. The slower software tools are found to be overrepresented at both high and low levels of accuracy (Figure 2). Likewise accurate software was found in both high and low speed ranges. A simple gedankenexperiment may be

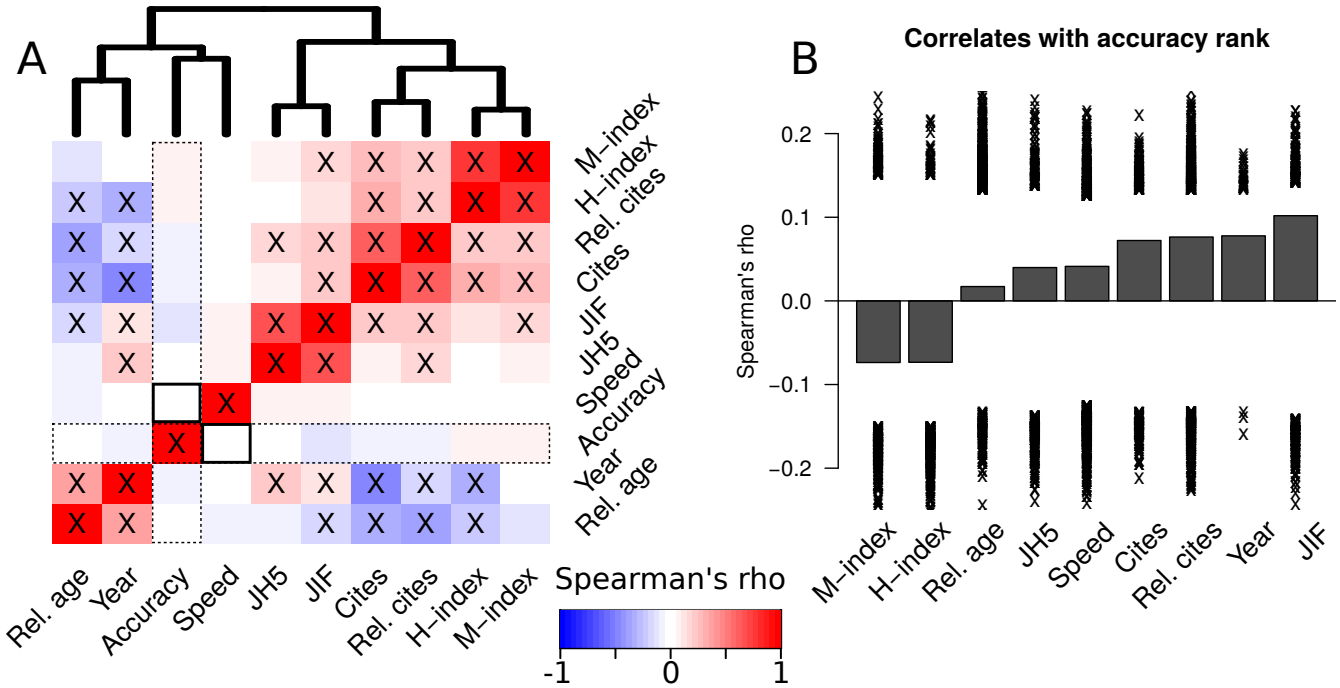


**Figure 1. A.** A heatmap indicating the relationships between proposed predictors of software quality. Spearman's rho is used to infer correlations between metrics such as the H and M indices of corresponding authors, number of citations, journal impact factors and H5 indices, the year and relative age of software and the mean relative rankings of software speed and accuracy. Red colours indicate a high positive correlation, blue colours indicate a high negative correlation. Correlations with a P-value less than 0.05 are indicated with a 'X'. The dashed rectangular area is illustrated in more detail in **B**, the bold square is shown in more detail in **Figure 2**.The dendrogram was computed using the default 'heatmap.2 function in R, which computes a Euclidean distance matrix and a complete-linkage hierarchical clustering. **B.** A barplot illustrating the correlation as measured by Spearman's rho between and normalised accuracy ranks and potential factors that may be predictive of accuracy. In order to give an appreciation of the difference between the observed effect-sizes and significant effect sizes, we generated 10,000 permuted accuracy ranks for each benchmark and recorded Spearman's rho for the significant correlations (P¡0.05). These values are marked with "x"s in the barplot.

sufficient to prove that slow software is less thoroughly tested than fast software. This is because typical software development is an iterative process, where methods are refined by successive rounds of coding, testing and evaluation [104]. We can assume that similar time-spans are spent on most projects, for example, the span of a MSc or PhD degree. As a consequence slow software undergoes fewer development cycles than fast methods, and is therefore tested less, often resulting in less accurate and slower software. The fact that fast and inaccurate software is relatively rare does support this argument. The lack of any further apparent relationship between speed and accuracy implies there are stronger influences on accuracy rankings than speed alone.

## Conclusions

Scientific progress is a grinding process made by testing hypotheses using appropriate positive and negative controls, in combination with independent experiments and replication [105, 106]. We think that software and analysis tools should be no exception to these principles. The continual increases in the complexity of software tools creates an increased likelihood that software bugs are introduced [107]. Scientific software should therefore be thoroughly tested by developers, however, the results of developer and author-derived tests should be treated with caution due to a range of potential conscious and unconscious biases [27]. Our study shows that no commonly-used impact-based metrics are related to software accuracy, therefore the only reasonable way to select software tools is through software benchmarks.

We have shown that slow and inaccurate software is typically published early in the development of a field (see Figure S8). Subsequent software tools may be high accuracy or high speed. However, as further software tools become available, those that do not rank highly in terms of accuracy or speed are under-represented in the literature. Our conjecture is that this is due to a publication bias in computational biology software literature, with publishers, editors and reviewers implicitly requiring software to be notable in at least one of these regards in order to be published. This hole in the literature leaves

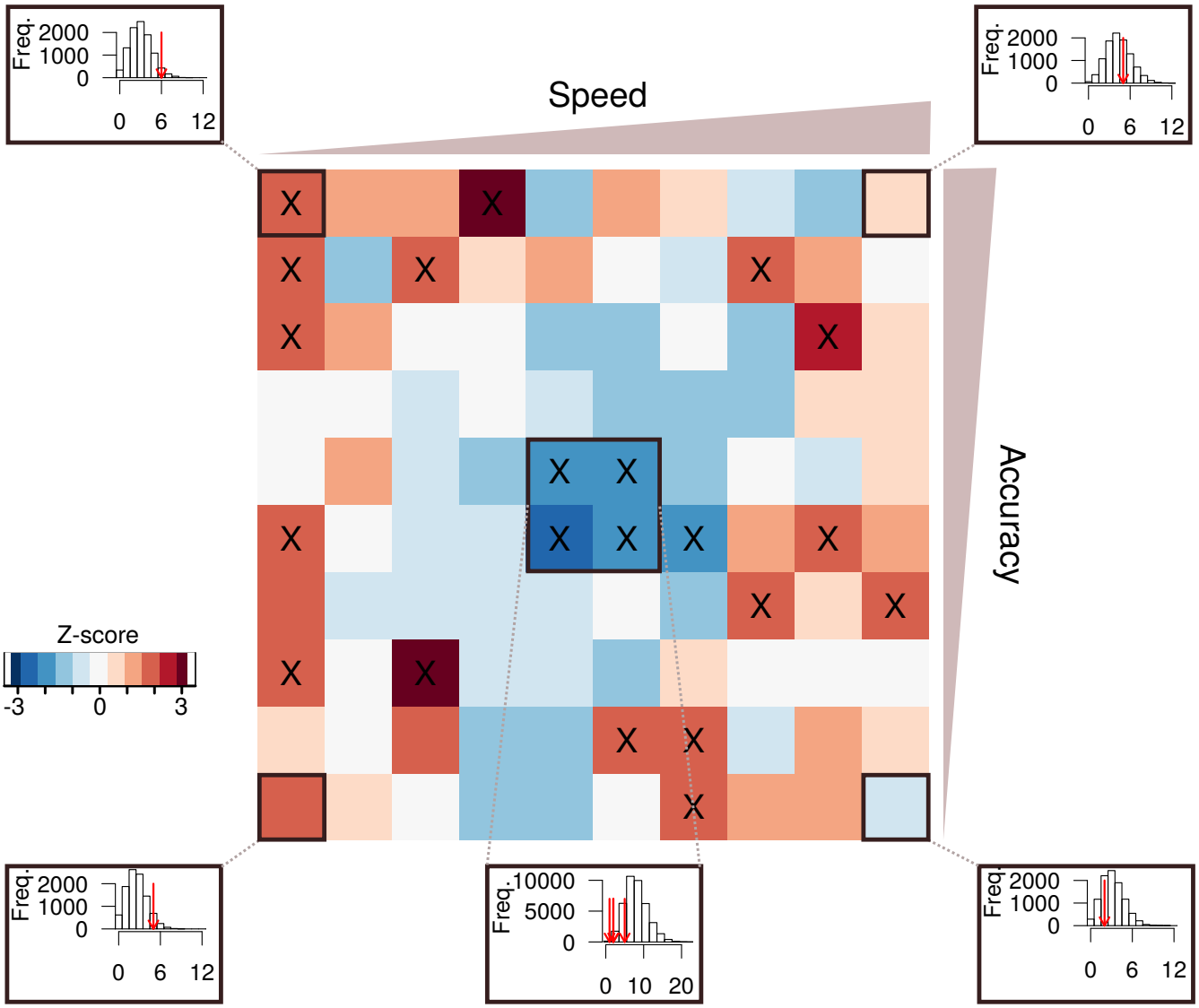


**Figure 2.** A heatmap indicating the relative paucity or abundance of software in the range of possible accuracy and speed rankings. Red colours indicate an abundance of software tools in an accuracy and speed category, while blue colours indicate scarcity of software in an accuracy and speed category. The abundance is quantified using a Z-score computation for each bin, this is derived from 10,000 random permutations of the speed and accuracy ranks from each benchmark. Mean normalised ranks of accuracy and speed have been binned into 100 classes (a $10 \times 10$ grid) that range from comparatively slow/inaccurate to comparatively fast/accurate. Z-scores with a P-value less than 0.05 are indicated with a 'X'.

an unfortunate gap upon which further software refinements cannot be made.

The software that could be used in building research workflows can make use of software tools that are of medium accuracy and speed. These may prove to be useful rapid pre-filters for more accurate, yet computationally-demanding approaches. These could also be used when datasets are too large to be processed by more accurate software tools, i.e. when accurate tools are slow or memory hungry [108, 109]. These tools may also serve as an indication that some approaches may not be worth pursuing further.

We propose that the full spectrum of software tool accuracies and speeds serve a useful purpose to the research community. Like negative results, if honestly reported, illustrate to the research community that certain approaches are not practical research avenues [105, 110, 111]. The current practises of publishers, editors, reviewers and authors of software tools appear to have deprived our community of tools for building effective and productive workflows.

A potential avenue for further exploration is to compare the starting point for software development projects as certain approaches may produce more rapid gains than others. For example, starting with biologically plausible and fast methods will theoretically allow more rapid gains in accuracy through iterative method refinement than starting with slow and mathematically complete approaches. At the very least these may be used as rapid data filters for reducing the size and complexity of problems, prior to employing more rigorous methods.

We have shown that accurate software is not necessarily the most recently released method or the product of high profile lab groups or selected by high impact journals. Software that is widely used or is either slow or fast is also not necessarily the most accurate. Therefore, accurate software may be the product of features we have not been able to capture. Possibly, hard work, good ideas and sound method testing, as well as technical ability, experience and education levels in software development [104].

Finally, we think that the field of computational biology could benefit from embracing an increased number of independent software comparison studies [93], in parallel with the popular challenge-based benchmarks such as CASP, Assemblathon, Alignathon, DREAM and RGASP [112]. We are hopeful that this, along with the relaxation of the publication bias we have described, will reduce the over-optimistic and misleading reporting of method accuracy [25, 26, 27].

## Methods

In order to evaluate predictors of computational biology software accuracy, we mined the published literature, extracted data from articles, connected these with bibliometric databases, and tested for correlates with accuracy. We outline these steps in further detail below.

**Criteria for inclusion:** We are interested in using computational biology benchmarks that satisfy Anne-Laure Boulesteix's (ALB) three criteria for a "neutral comparison study" [93]. Firstly, the main focus of the article is the comparison and **not** the introduction of a new method. Secondly, the authors should be reasonably neutral, which means that the authors should not generally have been involved in the development of the methods included in the benchmark. Thirdly, the test data and evaluation criteria should be sensible. This means that the test data should be independent of data that methods have been trained upon, and that the evaluation measures appropriately quantify correct and incorrect predictions.

Literature mining: We identified an initial list of 10 benchmark articles that satisfy the ALB-criteria. These were identified based upon previous knowledge of published articles and were supplemented with several literature searches (e.g. '"benchmark" AND "cputime"' was used to query both GoogleScholar and Pubmed [52, 113]). We used these articles to seed a machine-learning approach for identifying further candidate articles and to identify new search terms to include.

For our machine-learning-based literature screening, we computed a score ($s(a)$) for each article that tells us the likelihood that it is a benchmark. In brief, our approaches uses 3 stages:

1. Remove high frequency words from the title and abstract of candidate articles (e.g. 'the', 'and', 'of', 'to', 'a', ...)
2. Compute a log-odds score for the remaining words
3. Use a sum of log-odds scores to give a total score for candidate articles

For stage 1, we identified a list of high frequency (e.g. f(word) $>$ 1/10,000) words by pooling the content of two control texts [114, 115].

For stage 2, in order to compute a log-odds score for bioinformatic words, we computed the frequency of words that were not removed by our high frequency filter in two different groups of articles: bioinformatics-background and bioinformatics-benchmark articles. The text from bioinformatics-background articles were drawn from the bioinformatics literature, but these were not necessarily associated with benchmark studies. For background text we used Pubmed ([52, 113] to select 8,908 articles that contained the word "bioinformatics" in the title or abstract and were published between 2013 and 2015. We computed frequencies for each word by combining text from titles and abstracts for the background and training articles. A log-odds score is computed for each word using the following formula: $lo(w) = \log_2 \frac{\text{ffl} f_{tr}(word)+\delta}{f_{bg}(word)+\delta}$, where is a prior probability ($\delta = 10^{-5}$, by default), $f_{bg}(word)$ and $f_{tr}(word)$ are the frequencies of a *word* in the background and training datasets respectively. Word frequencies are computed by counting the number of times a word appears in the pool of titles and abstracts, the counts are normalised by the total number of words in each set.

Thirdly, we also collected a group of candidate benchmark articles by mining Pubmed for articles that are likely to be benchmarks of bioinformatic software, these may match the

terms: "((bioinformatics) AND (algorithms OR programs OR software)) AND (accuracy OR assessment OR benchmark OR comparison OR performance) AND (speed OR time)". Further terms used in this search were progressively added as relevant enriched terms were identified in later iterations. The final query is given in **supplementary materials**.

A score is computed for each candidate article by summing the log-odds scores for the words in title and abstract, i.e. $s(a) = \sum_i^N lo(w_i)$. The high scoring candidate articles are then manually evaluated against the ALB-criteria. Accuracy and speed ranks are extracted from the articles that meet the criteria, and these are also added to the set of training articles. The evaluated candidate articles that do not meet the ALB-criteria are incorporated into the set of background articles. This process is iterated a number of times and has resulted in the identification of **43** benchmark articles, that contain **102** different benchmarks, together these rank **243** distinct software packages.

There is a potential for bias to have been introduced into this dataset. Some possible forms of bias include converging on a niche group of benchmark studies due to the literature mining technique that we have used. A further possibility is that benchmark studies themselves are biased, either including very high performing or very low performing software tools. To address each of these concerns we have attempted to be as comprehensive as possible in terms of benchmark inclusion, as well as include comprehensive benchmarks. By which we mean studies that include all available software tools that address a biological problem.

**Data extraction:** for each article that met the ALB-criteria and contained data on both the accuracy and speed from their tests we extracted ranks for each method. Many articles contained multiple benchmarks, in these cases we selected a range of these, the provenance of which is stored with the accuracy metric and raw speed and accuracy rank data for each method. In line with rank-based statistics, the cases where methods were tied are resolved by using a midpoint rank (e.g. if method 3 and 4 are tied, the rank 3.5 is used) [116]. Each rank extraction was independently verified by at least one other co-author to ensure both the provenance of the data could be established and that the ranks were correct. The ranks for each benchmark were then normalised to lie between 0 and 1 using the formula $\frac{r-1}{n-1}$ where 'r' is a method's rank and 'n' is the number of methods in the benchmark. For methods that were benchmarked multiple times with multiple metrics (e.g. BWA is evaluated in 6 different articles [58, 59, 61, 62, 63, 65]) a mean normalised rank is used to summarise the performance.

For each method we identified the corresponding publications in GoogleScholar, the total number of citations was recorded, the corresponding authors were also identified and if these had public GoogleScholar profiles we extracted their H-index and calculated a M-index ($\frac{H-index}{y}$) where '$y$' is the number of years since their first publication. For the journals that each method is published in we extracted the "journal impact factor" (JIF) and the H5-index from Thompson-Reuters and GoogleScholar Metrics databases respectively. The year of publication was also recorded for each method. A "relative age" and "relative citations" was also computed for each method. For each benchmark, software was ranked by year of first publication (or number of citations), ranks were assigned and then normalised as described above. Methods ranked in multiple evaluations were then assigned a mean value for "relative age" and "relative citations".

**Statistical analysis:** For each method we have up to 10 statistics (1. corresponding author's H-index, 2. corresponding author's M-index, 3. journal H5 index, 4. journal impact factor, 5. normalised accuracy rank, 6. normalised speed rank, 7. number of citations, 8. relative age, 9. relative number of citations, 10. year first published). These have been evaluated in a pairwise fashion to produce Figure 1 A&B, the R code for these is given in the supplement.

For each benchmark of three or more methods, we extracted the published accuracy and speed ranks. In order to identify if there is an enrichment of certain accuracy and speed pairings we constructed a permutation test. The individual accuracy and speed ranks were reassigned to methods in a random fashion and each new accuracy and speed rank pairing was recorded. For each benchmark this procedure was repeated 10,000 times. These permuted rankings were normalised and compared to the real rankings to produce the 'x' points in Figure 1B and the heatmap and histograms in Figure 2. The heatmap in Figure 2 is based upon Z-scores ($Z = \frac{x-\bar{x}}{\sigma}$). For each cell in a $10 \times 10$ grid a Z-score is computed to illustrate the abundance or lack of methods in a cell relative to the permuted data.

## Data availability

The raw datasets are available here:
`https://docs.google.com/spreadsheets/d/14xIY2PHNvxmV9MQLpbzSfFkuy1RlzDHbBOCZLJKcGu8/edit?usp=sharing`
Additional documentation, code, figures and raw data is available here:
`https://github.com/UCanCompBio/speed-vs-accuracy-meta-analysis`

## Acknowledgements

The authors acknowledge the valued contribution of invaluable discussions with Anne-Laure Boulesteix, Shinichi Nakagawa, Suetonia Palmer and Jason Tylianakis. Michael A. Black, Murray Cox, Raquel Norel, Alexandros Stamatakis, Jens Stoye,Tandy Warnow, provided valuable feedback on drafts of the manuscript.

## References

[1] Carolina Perez-Iratxeta, Miguel A Andrade-Navarro, and Jonathan D Wren. Evolving research trends in bioinformatics. *Brief. Bioinform.*, 8(2):88–95, March 2007.

[2] Richard Van Noorden, Brendan Maher, and Regina Nuzzo. The top 100 papers. *Nature*, 514(7524):550–553, 30 October 2014.

[3] Jonathan D Wren. Bioinformatics programs are 31-fold over-represented among the highest impact scientific papers of the past two decades. *Bioinformatics*, 5 May 2016.

[4] S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–410, October 1990.

[5] J D Thompson, D G Higgins, and T J Gibson. CLUSTAL w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, 22(22):4673–4680, November 1994.

[6] J D Thompson, T J Gibson, F Plewniak, F Jeanmougin, and D G Higgins. The CLUSTAL_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Res.*, 25(24):4876–4882, 15 December 1997.

[7] S F Altschul, T L Madden, A A Schäffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, 25(17):3389–3402, 1 September 1997.

[8] Joseph Felsenstein. Confidence limits on phylogenies: An approach using the bootstrap. *Evolution*, 39(4):783–791, 1 July 1985.

[9] N Saitou and M Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4(4):406–425, July 1987.

[10] D Posada and K A Crandall. MODELTEST: testing the model of DNA substitution. *Bioinformatics*, 14(9):817–818, 1998.

[11] Fredrik Ronquist and John P Huelsenbeck. MrBayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, 19(12):1572–1574, 12 August 2003.

[12] Koichiro Tamura, Joel Dudley, Masatoshi Nei, and Sudhir Kumar. MEGA4: Molecular evolutionary genetics analysis (MEGA) software version 4.0. *Mol. Biol. Evol.*, 24(8):1596–1599, August 2007.

[13] E L Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *J. Am. Stat. Assoc.*, 53(282):457–481, 1 June 1958.

[14] David R Cox. Regression models and life-tables. *J. R. Stat. Soc. Series B Stat. Methodol.*, pages 187–220, 1972.

[15] G M Sheldrick. Phase annealing in SHELX-90: direct methods for larger structures. *Acta Crystallogr. A*, 46(6):467–473, 1 June 1990.

[16] George M Sheldrick. A short history of SHELX. *Acta Crystallogr. A*, 64(Pt 1):112–122, January 2008.

[17] T A Jones, J Y Zou, S W Cowan, and M Kjeldgaard. Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallogr. A*, 47 ( Pt 2):110–119, 1 March 1991.

[18] R A Laskowski, M W MacArthur, D S Moss, and J M Thornton. PROCHECK: a program to check the stereochemical quality of protein structures. *J. Appl. Crystallogr.*, 26(2):283–291, 1 April 1993.

[19] Zbyszek Otwinowski and Wladek Minor. [20] processing of x-ray diffraction data collected in oscillation mode. In *Methods in Enzymology*, volume Volume 276, pages 307–326. Academic Press, 1997.

[20] P J Kraulis. MOLSCRIPT: a program to produce both detailed and schematic plots of protein structures. *J. Appl. Crystallogr.*, 24(5):946–950, 1 October 1991.

[21] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, T N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic Acids Res.*, 28(1):235–242, 1 January 2000.

[22] Vivien Marx. Biology: The big challenges of big data. *Nature*, 498(7453):255–260, 13 June 2013.

[23] Joel Gombiner. Carbon footprinting the internet. *Consilience-The Journal of Sustainable Development*, 5(1), 2011.

[24] John D Storey and Robert Tibshirani. Statistical significance for genomewide studies. *Proc. Natl. Acad. Sci. U. S. A.*, 100(16):9440–9445, 5 August 2003.

[25] Anne-Laure Boulesteix. Over-optimism in bioinformatics research. *Bioinformatics*, 26(3):437–439, 1 February 2010.

[26] Monika Jelizarow, Vincent Guillemot, Arthur Tenenhaus, Korbinian Strimmer, and Anne-Laure Boulesteix. Over-optimism in bioinformatics: an illustration. *Bioinformatics*, 26(16):1990–1998, 15 August 2010.

[27] Raquel Norel, John Jeremy Rice, and Gustavo Stolovitzky. The self-assessment trap: can we all be better than average? *Mol. Syst. Biol.*, 7(1):537, 1 January 2011.

[28] J P Egan. *Signal Detection Theory and ROC-analysis*. Series in Cognition and Perception. Academic Press, New York, 1975.

[29] T Hall, S Beecham, D Bowes, D Gray, and S Counsell. A systematic literature review on fault prediction performance in software engineering. *IEEE Trans. Software Eng.*, 38(6):1276–1304, November 2012.

[30] J M Bujnicki, A Elofsson, D Fischer, and L Rychlewski. LiveBench-1: continuous benchmarking of protein structure prediction servers. *Protein Sci.*, 10(2):352–361, February 2001.

[31] Tomasz Puton, Lukasz P Kozlowski, Kristian M Rother, and Janusz M Bujnicki. CompaRNA: a server for continuous benchmarking of automated methods for RNA

secondary structure prediction. *Nucleic Acids Res.*, 42(8):5403–5406, April 2014.

[32] Michael Barton. nucleotides · genome assembler benchmarking. http://nucleotid.es/. Accessed: 2015-12-18.

[33] Joseph Felsenstein. Phylogeny programs. *Internet address: http://evolution. gs. washington. edu/phylip/software. html*, 1995.

[34] Stephen Altschul, Barry Demchak, Richard Durbin, Robert Gentleman, Martin Krzywinski, Heng Li, Anton Nekrutenko, James Robinson, Wayne Rasband, James Taylor, and Cole Trapnell. The anatomy of successful computational biology software. *Nat. Biotechnol.*, 31(10):894–897, October 2013.

[35] Vincent J Henry, Anita E Bandrowski, Anne-Sophie Pepin, Bruno J Gonzalez, and Arnaud Desfeux. OMICtools: an informative directory for multi-omic data analysis. *Database*, 2014, 14 July 2014.

[36] Wikipedia contributors. List of sequence alignment software. https://en.wikipedia.org/w/index.php?title=List_of_sequence_alignment_software&oldid=693586242, 3 December 2015. Accessed: 2015-12-18.

[37] Wikipedia contributors. List of RNA structure prediction software. https://en.wikipedia.org/w/index.php?title=List_of_RNA_structure_prediction_software&oldid=693718881, 4 December 2015. Accessed: 2015-12-18.

[38] Jo Erskine Hannay, Carolyn MacLeod, Janice Singer, Hans Petter Langtangen, Dietmar Pfahl, and Greg Wilson. How do scientists develop and use scientific software? In *Proceedings of the 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering*, SECSE '09, pages 1–8, Washington, DC, USA, 2009. IEEE Computer Society.

[39] Lucas N Joppa, Greg McInerny, Richard Harper, Lara Salido, Kenji Takeda, Kenton O'Hara, David Gavaghan, and Stephen Emmott. Troubling trends in scientific software use. *Science*, 340(6134):814–815, 17 May 2013.

[40] Nicholas Loman and Thomas Connor. Bioinformatics infrastructure and training survey, 2015.

[41] E Garfield. Citation indexes for science; a new dimension in documentation through association of ideas. *Science*, 122(3159):108–111, 15 July 1955.

[42] Anita Williams Woolley, Christopher F Chabris, Alex Pentland, Nada Hashmi, and Thomas W Malone. Evidence for a collective intelligence factor in the performance of human groups. *Science*, 330(6004):686–688, 29 October 2010.

[43] Kendra S Cheruvelil, Patricia A Soranno, Kathleen C Weathers, Paul C Hanson, Simon J Goring, Christopher T Filstrup, and Emily K Read. Creating and maintaining high-performing collaborative research teams: the importance of diversity and interpersonal skills. *Front. Ecol. Environ.*, 12(1):31–38, 1 February 2014.

[44] J E Hirsch. An index to quantify an individual's scientific research output. *Proc. Natl. Acad. Sci. U. S. A.*, 102(46):16569–16572, 15 November 2005.

[45] Lutz Bornmann, Rüdiger Mutz, and Hans-Dieter Daniel. Are there better indices for evaluation purposes than the h index? a comparison of nine different variants of the h index using data from biomedicine. *J. Am. Soc. Inf. Sci.*, 59(5):830–837, 1 March 2008.

[46] Mathieu Fourment and Michael R Gillings. A comparison of common programming languages used in bioinformatics. *BMC Bioinformatics*, 9:82, 5 February 2008.

[47] Michael Farrar. Striped Smith–Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics*, 23(2):156–161, 15 January 2007.

[48] Lorenzo Dematté and Davide Prandi. GPU computing for systems biology. *Brief. Bioinform.*, 11(3):323–333, May 2010.

[49] J Schaeffer. The history heuristic and alpha-beta search enhancements in practice. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(11):1203–1212, November 1989.

[50] Christos H Papadimitriou. Computational complexity. In *Encyclopedia of Computer Science*, pages 260–265. John Wiley and Sons Ltd., Chichester, UK.

[51] Lina Wadi, Mona Meyer, Joel Weiser, Lincoln D Stein, and Juri Reimand. Impact of knowledge accumulation on pathway enrichment analysis. Technical report, 19 April 2016.

[52] Eric W Sayers, Tanya Barrett, Dennis A Benson, Evan Bolton, Stephen H Bryant, Kathi Canese, Vyacheslav Chetvernin, Deanna M Church, Michael Dicuccio, Scott Federhen, Michael Feolo, Lewis Y Geer, Wolfgang Helmberg, Yuri Kapustin, David Landsman, David J Lipman, Zhiyong Lu, Thomas L Madden, Tom Madej, Donna R Maglott, Aron Marchler-Bauer, Vadim Miller, Ilene Mizrachi, James Ostell, Anna Panchenko, Kim D Pruitt, Gregory D Schuler, Edwin Sequeira, Stephen T Sherry, Martin Shumway, Karl Sirotkin, Douglas Slotta, Alexandre Souvorov, Grigory Starchenko, Tatiana A Tatusova, Lukas Wagner, Yanli Wang, W John Wilbur, Eugene Yaschenko, and Jian Ye. Database resources of the national center for biotechnology information. *Nucleic Acids Res.*, 38(Database issue):D5–16, January 2010.

[53] Eva K Freyhult, Jonathan P Bollback, and Paul P Gardner. Exploring genomic dark matter: a critical assessment of the performance of homology search methods on noncoding RNA. *Genome Res.*, 17(1):117–125, 6 January 2007.

[54] Sebastian Jünemann, Karola Prior, Andreas Albersmeier, Stefan Albaum, Jörn Kalinowski, Alexander Goesmann,

Jens Stoye, and Dag Harmsen. GABenchToB: a genome assembly benchmark tuned on bacteria and benchtop sequencers. *PLoS One*, 9(9):e107014, 8 September 2014.

[55] Hong Tran, Jacob Porter, Ming-An Sun, Hehuang Xie, and Liqing Zhang. Objective and comprehensive evaluation of bisulfite short read mapping tools. *Adv. Bioinformatics*, 2014:472045, 15 April 2014.

[56] Wenyu Zhang, Jiajia Chen, Yang Yang, Yifei Tang, Jing Shang, and Bairong Shen. A practical comparison of de novo genome assembly software tools for next-generation sequencing technologies. *PLoS One*, 6(3):e17915, 14 March 2011.

[57] Mostafa M Abbas, Qutaibah M Malluhi, and Ponnuraman Balakrishnan. Assessment of de novo assemblers for draft genomes: a case study with fungal genomes. *BMC Genomics*, 15 Suppl 9:S10, 8 December 2014.

[58] Suying Bao, Rui Jiang, Wingkeung Kwan, Binbin Wang, Xu Ma, and You-Qiang Song. Evaluation of next-generation sequencing software in mapping and assembly. *J. Hum. Genet.*, 56(6):406–414, June 2011.

[59] Ségolène Caboche, Christophe Audebert, Yves Lemoine, and David Hot. Comparison of mapping algorithms used in high-throughput sequencing: application to ion torrent data. *BMC Genomics*, 15:264, 5 April 2014.

[60] Dimitrios Kleftogiannis, Panos Kalnis, and Vladimir B Bajic. Comparing memory-efficient genome assemblers on stand-alone and cloud infrastructures. *PLoS One*, 8(9):e75505, 27 September 2013.

[61] Ayat Hatem, Doruk Bozdağ, Amanda E Toland, and Ümit V Çatalyürek. Benchmarking short sequence mapping tools. *BMC Bioinformatics*, 14:184, 7 June 2013.

[62] Sophie Schbath, Véronique Martin, Matthias Zytnicki, Julien Fayolle, Valentin Loux, and Jean-François Gibrat. Mapping reads on a genomic sequence: an algorithmic overview and a practical comparative analysis. *J. Comput. Biol.*, 19(6):796–813, June 2012.

[63] Matthew Ruffalo, Thomas LaFramboise, and Mehmet Koyutürk. Comparative analysis of algorithms for next-generation sequencing read alignment. *Bioinformatics*, 27(20):2790–2796, 15 October 2011.

[64] Xiao Yang, Sriram P Chockalingam, and Srinivas Aluru. A survey of error-correction methods for next-generation sequencing. *Brief. Bioinform.*, 14(1):56–66, January 2013.

[65] Manuel Holtgrewe, Anne-Katrin Emde, David Weese, and Knut Reinert. A novel and well-defined benchmarking method for second generation read mapping. *BMC Bioinformatics*, 12:210, 26 May 2011.

[66] Owen J L Rackham, Petros Dellaportas, Enrico Petretto, and Leonardo Bottolo. WGBSSuite: simulating whole-genome bisulphite sequencing data and benchmarking differential DNA methylation analysis tools. *Bioinformatics*, 31(14):2371–2373, 15 July 2015.

[67] Howard W Huang, NISC Comparative Sequencing Program, James C Mullikin, and Nancy F Hansen. Evaluation of variant detection software for pooled next-generation sequence data. *BMC Bioinformatics*, 16:235, 29 July 2015.

[68] Julie D Thompson, Benjamin Linard, Odile Lecompte, and Olivier Poch. A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. *PLoS One*, 6(3):e18093, 31 March 2011.

[69] Paulo A S Nuin, Zhouzhi Wang, and Elisabeth R M Tillier. The accuracy of several multiple sequence alignment programs for proteins. *BMC Bioinformatics*, 7:471, 24 October 2006.

[70] Fabiano Sviatopolk-Mirsky Pais, Patrícia de Cássia Ruy, Guilherme Oliveira, and Roney Santos Coimbra. Assessing the efficiency of multiple sequence alignment programs. *Algorithms Mol. Biol.*, 9(1):4, 6 March 2014.

[71] Muhammad Tariq Pervez, Masroor Ellahi Babar, Asif Nadeem, Muhammad Aslam, Ali Raza Awan, Naeem Aslam, Tanveer Hussain, Nasir Naveed, Salman Qadri, Usman Waheed, and Muhammad Shoaib. Evaluating the accuracy and efficiency of multiple sequence alignment methods. *Evol. Bioinform. Online*, 10:205–217, 7 December 2014.

[72] Kevin Liu, C Randal Linder, and Tandy Warnow. Multiple sequence alignment: a major challenge to large-scale phylogenetics. *PLoS Curr.*, 2:RRN1198, 19 November 2010.

[73] Martin Maška, Vladimír Ulman, David Svoboda, Pavel Matula, Petr Matula, Cristina Ederra, Ainhoa Urbiola, Tomás España, Subramanian Venkatesan, Deepak M W Balak, Pavel Karas, Tereza Bolcková, Markéta Streitová, Craig Carthel, Stefano Coraluppi, Nathalie Harder, Karl Rohr, Klas E G Magnusson, Joakim Jaldén, Helen M Blau, Oleh Dzyubachyk, Pavel Křížek, Guy M Hagen, David Pastor-Escuredo, Daniel Jimenez-Carretero, Maria J Ledesma-Carbayo, Arrate Muñoz Barrutia, Erik Meijering, Michal Kozubek, and Carlos Ortiz-de Solorzano. A benchmark for comparison of cell tracking algorithms. *Bioinformatics*, 30(11):1609–1617, 1 June 2014.

[74] Yue Li, Zhuo Zhang, Feng Liu, Wanwipa Vongsangnak, Qing Jing, and Bairong Shen. Performance comparison and evaluation of software tools for microRNA deep-sequencing data analysis. *Nucleic Acids Res.*, 40(10):4298–4305, May 2012.

[75] Bingxin Lu, Zhenbing Zeng, and Tieliu Shi. Comparative study of de novo assembly and genome-guided assembly strategies for transcriptome reconstruction based on RNA-Seq. *Sci. China Life Sci.*, 56(2):143–155, February 2013.

[76] Ruolin Liu, Ann E Loraine, and Julie A Dickerson. Comparisons of computational methods for differential alternative splicing detection using RNA-seq in plant systems. *BMC Bioinformatics*, 15:364, 16 December 2014.

[77] Shailesh Kumar, Angie Duy Vo, Fujun Qin, and Hui Li. Comparative assessment of methods for the fusion transcripts detection from RNA-Seq data. *Sci. Rep.*, 6:21597, 10 February 2016.

[78] Adrien Pain, Alban Ott, Hamza Amine, Tatiana Rochat, Philippe Bouloc, and Daniel Gautheret. An assessment of bacterial small RNA target prediction programs. *RNA Biol.*, 12(5):509–513, 2015.

[79] Domonkos Tikk, Philippe Thomas, Peter Palaga, Jörg Hakenberg, and Ulf Leser. A comprehensive benchmark of kernel methods to extract protein-protein interactions from literature. *PLoS Comput. Biol.*, 6:e1000837, 1 July 2010.

[80] Rachel Kolodny, Patrice Koehl, and Michael Levitt. Comprehensive evaluation of protein structure alignment methods: scoring by geometric measures. *J. Mol. Biol.*, 346(4):1173–1188, 4 March 2005.

[81] Björn Wallner and Arne Elofsson. All are not equal: a benchmark of different homology modeling programs. *Protein Sci.*, 14(5):1315–1327, May 2005.

[82] Junliang Shang, Junying Zhang, Yan Sun, Dan Liu, Daojun Ye, and Yaling Yin. Performance analysis of novel methods for detecting epistasis. *BMC Bioinformatics*, 12:475, 15 December 2011.

[83] Stinus Lindgreen, Karen L Adair, and Paul P Gardner. An evaluation of the accuracy and speed of metagenome analysis tools. *Sci. Rep.*, 6:19233, 18 January 2016.

[84] Adam L Bazinet and Michael P Cummings. A comparative evaluation of sequence classification programs. *BMC Bioinformatics*, 13:92, 10 May 2012.

[85] Surya Saha, Susan Bridges, Zenaida V Magbanua, and Daniel G Peterson. Empirical comparison of ab initio repeat finding programs. *Nucleic Acids Res.*, 36(7):2284–2294, April 2008.

[86] Eva Lange, Ralf Tautenhahn, Steffen Neumann, and Clemens Gröpl. Critical assessment of alignment procedures for LC-MS proteomics and metabolomics measurements. *BMC Bioinformatics*, 9:375, 15 September 2008.

[87] Chao Yang, Zengyou He, and Weichuan Yu. Comparison of public peak detection algorithms for MALDI mass spectrometry data analysis. *BMC Bioinformatics*, 10:4, 6 January 2009.

[88] Kevin Liu, C Randal Linder, and Tandy Warnow. RAxML and FastTree: comparing two methods for large-scale maximum likelihood phylogeny estimation. *PLoS One*, 6(11):e27731, 21 November 2011.

[89] Jimmy Yang and Tandy Warnow. Fast and accurate methods for phylogenomic analyses. *BMC Bioinformatics*, 12 Suppl 9:S4, 5 October 2011.

[90] Maribeth Oscamou, Daniel McDonald, Von Bing Yap, Gavin A Huttley, Manuel E Lladser, and Rob Knight. Comparison of methods for estimating the nucleotide substitution matrix. *BMC Bioinformatics*, 9:511, 1 December 2008.

[91] Md Shamsuzzoha Bayzid and Tandy Warnow. Naive binning improves phylogenomic analyses. *Bioinformatics*, 29(18):2277–2284, 15 September 2013.

[92] Kevin Liu, Serita Nelesen, Sindhu Raghavan, C Randal Linder, and Tandy Warnow. Barking up the wrong treelength: the impact of gap penalty on alignment and tree accuracy. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 6(1):7–21, January 2009.

[93] Anne-Laure Boulesteix, Sabine Lauer, and Manuel J A Eugster. A plea for neutral comparison studies in computational sciences. *PLoS One*, 8(4):e61562, 24 April 2013.

[94] Eugene Garfield. The history and meaning of the journal impact factor. *JAMA*, 295(1):90–93, 4 January 2006.

[95] Dmitri V Zaykin. Optimally weighted z-test is a powerful method for combining probabilities in meta-analysis. *J. Evol. Biol.*, 24(8):1836–1841, 2011.

[96] Matthew D Moran. Arguments for rejecting the sequential bonferroni in ecological studies. *Oikos*, 100(2):403–405, 2003.

[97] Anne-Laure Boulesteix, Veronika Stierle, and Alexander Hapfelmeier. Publication bias in methodological computational research. *Cancer Inform.*, 14(Suppl 5):11–19, 15 October 2015.

[98] Anne-Laure Boulesteix. Ten simple rules for reducing overoptimistic reporting in methodological computational research. *PLoS Comput. Biol.*, 11(4):e1004191, April 2015.

[99] The Plos Medicine Editors. The impact factor game. *PLoS Med.*, 3(6):e291, 6 June 2006.

[100] Mike Rossner, Heather Van Epps, and Emma Hill. Show me the data. *J. Cell Biol.*, 179(6):1091–1092, 17 December 2007.

[101] Vincent Larivière and Yves Gingras. The impact factor's matthew effect: A natural experiment in bibliometrics. *J. Am. Soc. Inf. Sci.*, 61(2):424–427, 1 February 2010.

[102] A W Harzing and R van der Wal. Comparing the google scholar h-index with the ISI journal impact factor. *Research in Int. Management Products*, 2008.

[103] Robert K Merton and Others. The matthew effect in science. *Science*, 159(3810):56–63, 1968.

[104] G V Wilson. Where's the real bottleneck in scientific computing? *Am. Sci.*, 2006.

[105] John P A Ioannidis. Why most published research findings are false. *PLoS Med.*, 2(8):e124, August 2005.

[106] Ramal Moonesinghe, Muin J Khoury, and A Cecile J. Most published research findings are False—But a little replication goes a long way. *PLoS Med.*, 4(2):e28, 27 February 2007.

[107] Diego Darriba, Tomas Flouri, and Alexandros Stamatakis. The state of software in evolutionary biology. 1 January 2015.

[108] T M Lowe and S R Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.*, 25(5):955–964, March 1997.

[109] Z Weinberg and W L Ruzzo. Sequence-based heuristics for faster annotation of non-coding RNA families. *Bioinformatics*, 22(1):35–39, January 2006.

[110] C Workman and A Krogh. No evidence that mRNAs have lower folding free energies than random sequences with the same dinucleotide distribution. *Nucleic Acids Res.*, 27(24):4816–4822, December 1999.

[111] E Rivas and S R Eddy. Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. *Bioinformatics*, 16(7):583–605, July 2000.

[112] Paul C Boutros, Adam A Margolin, Joshua M Stuart, Andrea Califano, and Gustavo Stolovitzky. Toward better benchmarking: challenge-based methods assessment in cancer genomics. *Genome Biol.*, 15(9):462, 17 September 2014.

[113] J McEntyre and D Lipman. PubMed: bridging the information gap. *CMAJ*, 164(9):1317–1319, 1 May 2001.

[114] Lewis Carroll. *Alice's adventures in Wonderland*. Macmillan and Co., London, 1865.

[115] J R R Tolkien. *The Hobbit, Or, There and Back Again*. George Allen & Unwin, UK, 1937.

[116] H B Mann and D R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Stat.*, 18(1):50–60, 1947.