Current challenges in open-source bioimage informatics

Albert Cardona & Pavel Tomancak

We discuss the advantages and challenges of the open-source strategy in biological image analysis and argue that its full impact will not be realized without better support and recognition of software engineers' contributions to the biological sciences and more support of this development model from funders and institutions.

The role of open-source development in basic research is best summarized by a quote (written in support for a grant centered on the open-source image-analysis software Fiji) from Andrew C. Oates (personal communication): "The most fundamental element is the openness; if you can't see the code of a piece of commercial software, then you cannot say what the software really does, and this is not scientific." No one should use PCR to amplify DNA without knowing what causes the band to appear on the gel. Similarly, it is not good scientific practice to press a button in a piece of software and interpret the results without understanding what the software does. Open-source software (**Box 1**) provides the necessary transparency, giving scientists the opportunity to not only fully understand the computational methods but also to adapt and improve them, building on research of others in the best scientific tradition.

We qualify this extreme introduction by recognizing that time-tested solutions can be applied to data without having access to the computer code that generated them. This typically applies to commercial platforms for biological image analysis that focus on ease of use and target biology users who require relatively routine solutions. Much modern biological research, however, is fueled by transformative advances in microscopy and demands the development of new approaches for biological image analysis. In these cases the open-source model is indispensable.

Unfortunately, several obstacles impede effective use of open-source software development principles in biological research[1,2]. Scientists often see writing computer programs as a nonscientific activity. We argue that developing competitive solutions—for example, for tracking tens of thousands of cells in a massive microscopic recording of a developing biological system or establishing the connectivity pattern of neurons in the brain—can be indispensible for answering otherwise inaccessible biological questions, while at the same time being an important advance in computer science. For such problems, innovative ideas from the computer science field are as important as proper software engineering practices to turn these ideas into fast and scalable software.

To avoid hindering scientific progress, the biological research community must engage in productive collaboration with computer scientists and programmers in the area of biological image analysis. The numerous thriving open-source projects in the area of bioinformatics testify to the value of open-source communities as natural interfaces for computer scientists and biologists to productively work together (**Box 1**). Open-source projects can have similar success in the nascent interdisciplinary research field of bioimage informatics.

The aim of bioimage informatics is to use cutting-edge computer science to achieve insights into biological problems through computational analysis of large-scale image data sets[3,4]. Quantitative measurements need to be extracted from inherently noisy image data. High-throughput screening produces quantities of images that are well beyond the ability to be analyzed by manual inspection. High-resolution, time-resolved microscopy of large biological specimens produces image data sets that approach the scale of data production in particle physics. Below we discuss how solutions to these types of problems require a transfer of knowledge from computer science into biology mediated by programmers.

From new algorithms to usable applications

The basis for progress in biological image analysis is algorithms developed by computer science researchers. In particular, the computer vision field that focuses on processing of natural images is a potentially rich source of ideas that may be applied to biological image processing[5]. However, as biological image data differ substantially from natural images (in terms of dimensionality, signal-to-noise ratios and prior knowledge of what is being imaged) the computer vision algorithm must be adapted to the specific needs of biological applications.

Computer vision is a fiercely competitive research field, and publishable algorithm advances are often meaningful yet incremental improvements of an existing

Albert Cardona is at the Institute of Neuroinformatics, University of Zurich and ETH Zürich, Zürich, Switzerland. Pavel Tomancak is at the Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany.
e-mail: tomancak@mpi-cbg.de

approach. There is therefore little incentive for researchers to put these algorithms into the open-access domain, where they can potentially be misused by competitors to show that their competing approach works better. Instead, the algorithms are compared in terms of mathematical formulation or using small, agreed-upon test data sets. Because an optimized implementation of an algorithm is often secondary to the precise mathematical formulation, the preferred tools of this community, such as Matlab (MathWorks), are built for generality. Obviously, a set of equations or a Matlab implementation that will run only on relatively small images is of little use in biology, where data set sizes can be in the terabyte range. It is therefore of paramount importance to invest in optimized and scalable implementations of the best algorithms that can be applied to biological image data.

Whose job is it to develop and support fast, user-friendly and scalable implementations of powerful algorithms adapted for biological image data? Once an efficient software solution exists, it is not uncommon that it tends to disappear along with the person who created it. A problem often related as 'the computer science PhD student moved on, and we do not know what parameters were used, neither what the magic numbers mean'.

Commercial software companies will almost certainly hide the algorithm and implementation details behind restrictive end-user licenses because the release of the source code would in most cases conflict with the revenue model. Such 'closed' software loses all transparency regarding algorithm implementation. On one hand, computer vision researchers themselves are in a prime position to come up with professional software, but they are more interested in the mathematical underpinning of the solution and often view software engineering as nonscientific activity. They certainly have no interest in supporting and maintaining the software. Many biologists, on the other hand, have serious programming expertise, but typically they will not follow the professional software engineering practices necessary for production-grade software and will be similarly unmotivated to support the software. Software engineers would be ideal for the task of creating and maintaining professional software, but the academic environment is poorly equipped to compete with the commercial sphere for engineers, and this situation is unlikely to change.

We believe an open-source software framework that fosters productive collaboration between computer-savvy biologists and bio-application–oriented computer scientists and is designed to collect and maintain useful programs that can be reused and expanded by future generations of researchers is the viable approach. But are there enough biologists and computer scientists capable of working together in this application domain to support a vibrant and productive open-source software community?

To answer this question we set up an online poll in which we asked a small sample of life scientists to evaluate their expertise in biology, programming and image analysis on the scale of 1–5 (1 being poor to 5, excellent) and select their primary scientific field (biology, computer science or other; http://fly.mpi-cbg.de/expertise/, developed by Stephan Saalfeld) (**Fig. 1**). Altogether, 309 anonymous scientists filled out the survey; 245 identified themselves as biologists and 57 as computer scientists. The results showed that programming and image-analysis skills correlate (**Fig. 1a**), with computer scientists understandably believing in their interdisciplinary skills more than biologists. Most biologists judged their image-analysis skills as average. Because few claimed to be good programmers, their perception must be that they can do image analysis even without any programming skills. Amusingly, some computer scientists judged their own programming skills to be below average. As expected, expertise in biology and programming was strongly anticorrelated (**Fig. 1b**), and the representatives of both disciplines clearly separated into their respective areas of expertise.

BOX 1 OPEN-SOURCE SOFTWARE

A piece of software is referred to as 'open-source' if its source code is publicly accessible and distributed with an open-source license that finetunes what can and cannot be done with the software. Among the most popular open-source licenses is the GNU General Public License, which "allows free distribution under the condition that further developments and applications are put under the same license"[11] (also called 'copyleft'), ensuring that the derived source code remains open and free. In contrast, another popular open-source license, the Berkeley Software Distribution, allows the derived software to become proprietary and closed, facilitating commercial exploitation. The open-source movement is a branch of the Free Software Movement initiated in 1983 by Richard M. Stallman to give freedom to computer users by replacing the proprietary software with free software[12]. The term 'open source' was adopted in 1998 by Open Source Initiative to make this development model more attractive to the corporate world by avoiding the term 'free'.

Most of the programs that power the internet are being developed as open-source software. Some well-known examples are Linux (a unix-like operating system), the Apache web server, the MySQL database, the Perl and Python programming languages and the Open Office suite. Dedicated nonprofit organizations, such as the Apache Software Foundation, have been established to nurture the open-source development communities. Many successful companies find value in the ability to customize software for their needs and fund open-source software. For instance Google runs a highly successful open-source mentoring program Google Summer of Code that funds students to work with seasoned open-source developers on small software projects, bringing new talent to the open-source platforms.

In biology, BioPerl[13] and Bioconductor[14] represent prime examples of large open-source software communities with profound impact on sequence and transcriptomics data analysis. In biological image analysis, ImageJ[7] has been the dominant platform, allowing easy extension owing to convenient plug-in architecture. Advanced software libraries such as the Insight Toolkit (ITK)[15] have been collecting algorithmic solutions in bioimage analysis, but because of their inherent complexity they have been relatively inaccessible to practicing biologists. Recent quantum leaps in microscopy technology inspired establishment of new and rapidly growing bioimage open-source communities that build on ImageJ (Fiji[10], ImageJ2 (ref. 9) and Cell Profiler[7]), ITK (Vaa3d[8] and BioimageXD[16]) or start anew (Icy[17]).

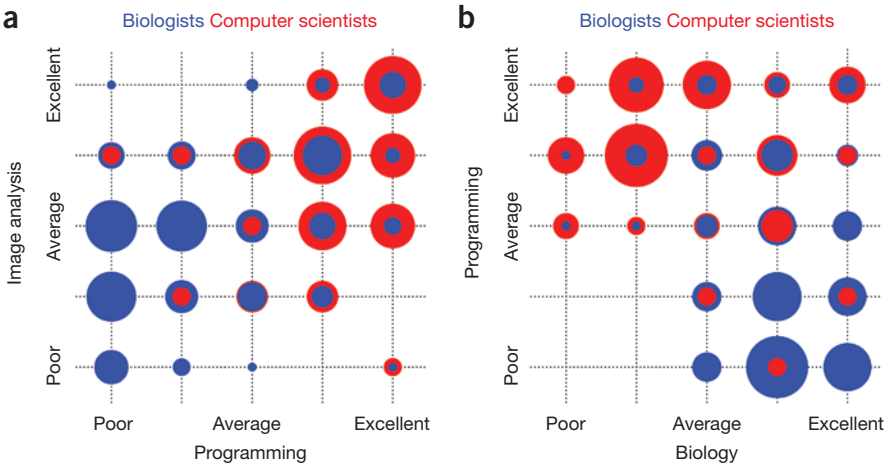

Figure 1 | Informal online self-evaluation of scientists' expertise relevant for bioimage informatics. (**a**,**b**) Results for image analysis versus programming (**a**) and biology versus programming (**b**). Size of plotted circles is proportional to the percentage of responders selecting a given combination of categories, and data are color-coded by the reported primary expertise.

What is relevant for the discussion here, many biologists have some programming skills and some even trusted themselves to be excellent programmers. Similarly, some computer scientists apparently have fairly good knowledge of biology and most judged themselves to be above average programmers. Few individuals had both excellence in biology and programming, and apparently no scientists viewed themselves as weak in both disciplines, which is encouraging. There are certainly biases in this unscientific survey, but it suggests that biologists with strong computer skills and computer scientists with keen interest in biological image analysis exist and that at least some of them have in common the passion for writing computer code. Nevertheless, as programming skills vary even among computer scientists, interdisciplinary collaboration in bioimage informatics should be supplemented by expertise in professional software engineering so that the resulting tools are accessible, scalable and can be maintained in the long run.

A necessary prerequisite for biologists, computer scientists and programmers to invest their time in implementing and distributing new algorithm solutions through open-source platforms to the scientific community is to recognize the value of this contribution to basic research. The generally dismissive position toward software engineering in life sciences is untenable because biological investigations often require new computational approaches to be developed. Moreover, bioimage software projects focused on solving specific biological questions result in new software tools whose impact reaches beyond the question that motivated their development. The measuring stick of success in scientific circles is publication, and therefore software must be published to promote the scientific career of the researchers doing the software engineering work.

Some well known journals such as *Nature Methods*, *Bioinformatics*, *Genome Biology*, *Neuroimage* and *PLoS Computational Biology* systematically publish reports of new software. But more typically software is embedded in methods sections of biology papers, and the programmers are listed as middle authors. This does little to boost their scientific track record. One possibility discussed in the field is to establish a new journal dedicated to bioimage informatics. In the current environment of massive proliferation of scientific periodicals, the proposal to establish a new journal is typically greeted with substantial skepticism. The computer science researchers rightly point out that their field has numerous, well-established publishing venues in the form of annual conferences with peer-reviewed paper submission and traditional printed or online journals. Unfortunately these platforms are too technical for biologists who are typically not even aware of their existence.

We believe that there is a space for an online journal that would focus on practical implementations of bioimage informatics algorithms. Such a journal should provide a detailed but accessible explanation of the algorithmic principles and document the use and parameters of an open-source software implementation of the algorithm. The journal would provide a much needed outlet for the activities of life scientists developing image-analysis tools and serve as an accessible venue for biologists to find new solutions that can be built upon through a mandatory open-source model.

Alternatively, if there is insufficient support in the community for a new journal, an image-analysis portal could be established to collect existing software and to identify the best solution based on user rating after publication. This approach using social networking would require the acceptance of popularity criteria such as downloads or user ratings as being comparable to journal publication.

Although an argument can be made that software engineering is not a science, we believe that the biological research community should adopt a pragmatic position and explicitly incorporate software engineering into the basic research enterprise by generating publishing venues and attractive career paths for programmers in the life sciences. The success and widespread acceptance of sequence bioinformatics software projects that are routinely published, and support long-term career development of the researchers involved, should serve as a precedent for the bioimage-analysis field.

Collaborative software development

Productive collaboration between computer-savvy biologists and bio-application–oriented computer scientists requires mechanisms for stimulating this process. The open-source community provides valuable lessons on how to work collaboratively and share ideas. The prime example is the collaborative coding spree usually referred to as a 'hackathon'.

In the bioimage informatics community, hackathons are widely used by the Fiji (http://fiji.sc/) community, which has held six hackathons hosted by major scientific institutions since its inception in 2007. The Fiji hackathons bring together 10–20 programmers and programming-savvy biologists from the Fiji community and other related projects to work together on loosely defined problems associated with image analysis using Fiji. The participants spend about two weeks sequestered in a room working collaboratively on the Fiji code. Participants usually aim to develop code for

their own biological research project, yet benefit enormously from getting to know each other, working together and combining their expertise and code libraries. The end result is a dramatic expansion and improvement of the capabilities of the platform (**Fig. 2** and **Supplementary Video 1**).

Thus far, Fiji hackathons have been funded mostly from research grants of the participating laboratories with substantial *ad hoc* injection of support in the form of travel, accommodations and equipment from the hosting institutions. In return the hosting institution receives access to active scientific software developers from around the world and can, in most cases, convince them to give tutorials and one-on-one consultations. The purpose of the hackathon, however, is not to develop solutions for the host institutions but to work collaboratively on common research problems with the overall improvement of the platform being the inevitable positive side effect.

Hackathons are an excellent forum to build bridges between various open-source software projects. The last hackathons in Madison, Wisconsin, USA, and in Dresden, Germany, were a joint venture of the Fiji and ImageJ2 projects and brought together representatives of several other bioimage-analysis software projects. One tangible result of the collaborative coding spree was the commitment of several projects to adopt Fiji's ImgLib library for image data representation, which will substantially enhance their future interoperability (http://www.scijava.org/).

As every platform has its strengths and weaknesses, and even the young bioimage informatics field is fragmented into many independent projects, it is important at this early stage to discuss the mechanisms for cooperation, exchange of algorithms and data. Even if the different projects use fundamentally incompatible programming languages (Java versus C++) it is possible to discuss approaches and data-exchange protocols, the so-called wrapping of functions from one platform into another, and common strategies for representation of image data. We strongly believe that this communication fosters healthy competitiveness among the platforms for better usability, smarter algorithms and faster implementations.

Support for open-source projects

Despite the collaborative advantages of open-source software development, a major obstacle is the lack of a sustainable funding model for open-source projects. From our personal experience, traditional academic funding agencies show strong reluctance to provide support for open-source software development. Part of the problem may be in the common perception that open-source software is developed by enthusiasts in their spare time and as such comes essentially for free. Although this may have been the case at the beginning of the open-source movement, the vast majority of current contributors to the major open-source platforms are compensated for their efforts (for example, Linux kernel developers).

As the projects that start as small 'garage' start-ups driven by the passion of their founders mature, they gather a following, and with that come increasing demands on maintenance, development and communication with a growing user community that cannot be sustained without dedicated funding. The sooner the funding agencies realize this, the better because some of the promising open-source endeavors simply die without funding from academic sources. A notable exception to this was US National Institutes of Health (NIH) Image, now called ImageJ[6].

Notably, funding agencies are not opposed to paying tens of thousands of dollars to buy commercial licenses for software packages that are rigid, opaque regarding their inner workings and can only be run in batch mode with great difficulty and at great cost (one license per computer node in the cluster). We argue that research funds serve scientific research better by paying for development and customization of open-source software that benefits not only the funded research group but also all other groups with similar image-processing needs. Even with publicly funded open-source projects there is certainly space for commercial solutions, but it should be the companies that adapt their business models to use the fruits of academic research and not the other way around.

Open-source projects tend to be very long-term, whereas research funding is usually provided for a fixed period of time. Although traditional research grants are needed to get open-source projects off the ground, in the long run these projects rely on institutional support to survive in the academic environment. This model works very well because many bioimage

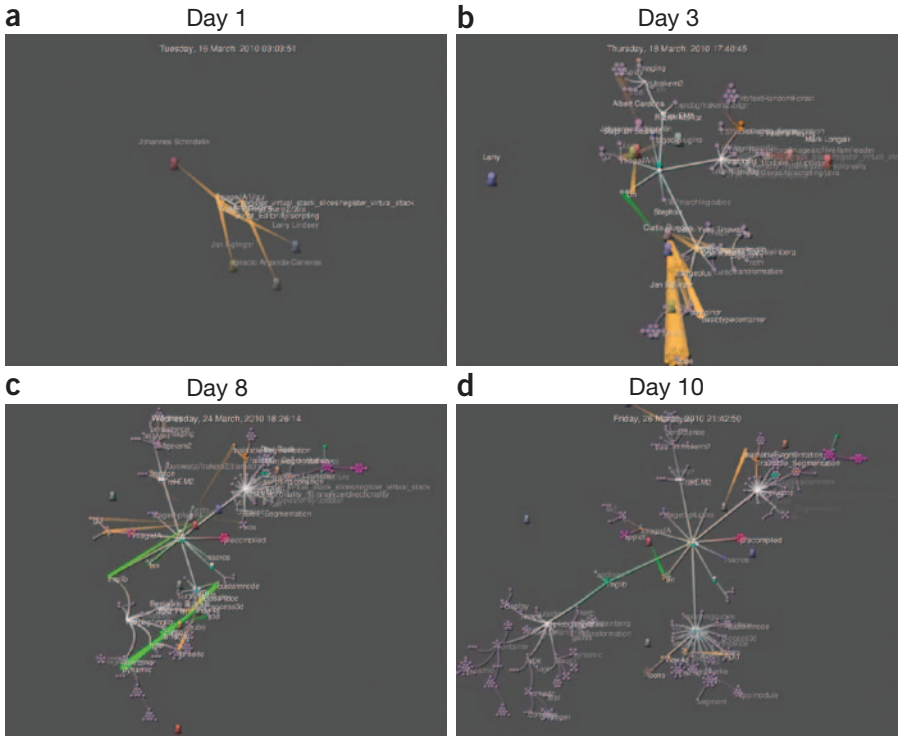

Figure 2 | Visualization of the 'hackathon effect'. (**a**) Situation at the beginning of the Fiji Hackathon at the European Molecular Biology Laboratory (16–26 March 2010). (**b**,**c**) Developer activity during the hackathon. (**d**) Overview of the code generated during the 10-day coding spree. Shown are screenshots from a video generated by the 'gource' tool. Modifications to files of the Fiji project are depicted as rays from the symbols for developers (pawns) to the files represented by a tree of colored balls.

informatics open-source platforms are firmly anchored at prestigious scientific institutions (CellProfiler[7] at Broad Institute, Vaa3D[8] at Howard Hughes Medical Institute Janelia Farm, ImageJ2 (ref. 9) at the Laboratory for Optical and Computational Instrumentation at University of Wisconsin at Madison, Fiji[10] at Max Planck Institute of Molecular Cell Biology and Genetics and others), which support entire research groups dedicated to the platform development, maintain computational infrastructure and promote the platform by associating their institution with it. The institutes benefit through internal access to experts that can help them solve image-analysis challenges and by attracting new talent that gravitates toward highly visible and successful software projects. We see a long-term partnership with established academic institutions, providing a career track for the crucial core developers at the heart of each project, augmented by grant support, as two major ingredients for the long-term success of open-source projects in the academic environment. Similar models have already been applied at Howard Hughes Medical Institute Janelia Farm, which created a Scientific Computing Unit, and at the Max Planck Institute of Molecular Cell Biology and Genetics, which established an Image Processing Facility.

Conclusions

An open-source strategy for analysis of biological image data is a sensible approach taken by many leading scientists in the field. It is an ongoing process to convince professional experts in computer science to take up biological image data challenges and advance their research agenda in this application domain. Because the size and complexity of biological image data sets is on the rise, an important ingredient for progress is modern software engineering practices that result in user-friendly, high-performance open-source software. Regardless of who does the open-source implementations—be it the biologists, computer vision experts or professional programmers—it is of the utmost importance to recognize their contribution to the scientific inquiry by providing publishing venues, grant support and career development options in the scientific context.

Note: Supplementary information is available at http://www.nature.com/doifinder/10.1038/nmeth.2082.

ACKNOWLEDGMENTS

We thank S. Saalfeld for programming the online survey and plotting the results shown in **Figure 1**, J. Schindelin for generating the visualization of Fiji development progress shown in **Figure 2**, J. Schindelin, M. Longair, C. Rueden and D.J. White for insightful discussions about the merits of the open-source strategy, and A.C. Oates for critical reading of the manuscript. P.T. was supported by Human Frontier Science Program Young Investigator grant RGY0083 and the European Research Council Community's Seventh Framework Programme (FP7/2007-2013) grant agreement 260746.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial interests.

1. Ince, D.C., Hatton, L. & Graham-Cumming, J. *Nature* **482**, 485–488 (2012).
2. Morin, A. *et al. Science* **336**, 159–160 (2012).
3. Peng, H. *Bioinformatics* **24**, 1827–1836 (2008).
4. Swedlow, J.R. & Eliceiri, K.W. *Trends Cell Biol.* **19**, 656–660 (2009).
5. Danuser, G. *Cell* **147**, 973–978 (2011).
6. Schneider, C.A., Rasband, W.S. & Eliceiri, K.W. *Nat. Methods* **9**, 671–675 (2012).
7. Carpenter, A.E. *et al. Genome Biol.* **7**, R100 (2006).
8. Peng, H., Ruan, Z., Long, F., Simpson, J.H. & Myers, E.W. *Nat. Biotechnol.* **28**, 348–353 (2010).
9. Eliceiri, K.W. *et al. Nat. Methods* **9**, 697–710 (2012).
10. Schindelin, J. *et al. Nat. Methods* **9**, 676–682 (2012).
11. Holtgrewe, U. *Time Soc.* **13**, 129–146 (2004).
12. Stallman, R. in *Open sources. Voices from the open source Revolution* (eds., DiBona, C., Ockman, S. & Stone, M.) 53–71 (O'Reilly & Associates, 1999).
13. Stajich, J.E. *et al. Genome Res.* **12**, 1611–1618 (2002).
14. Gentleman, R.C.V. *et al. Genome Biol.* **5**, R80 (2004).
15. Ibanez, L., Schroeder, W., Ng, L. & Cates, J. *The ITK Software Guide* (Kitware Inc., 2003).
16. Kankaanpää, P. *et al. Nat. Methods* **9**, 683–689 (2012).
17. de Chaumont, F. *et al. Nat. Methods* **9**, 690–696 (2012).