Reintroducing mothur: 10 years later

Patrick D. Schloss[1†]

† To whom correspondence should be addressed: pschloss@umich.edu

1 Department of Microbiology and Immunology, University of Michigan, Ann Arbor, MI 48109

Observation format

Abstract

75 words

Importance

150 words

3000 words total

Few scientists set out on a nearly two decade long journey with a specific goal in mind. Often we fail to start a scientific journey because it looks too hard. Perhaps we get bogged down in all of the things that could go wrong. Perhaps we go astray from the path because we find something else that appears more interesting. Every scientist picks their own path and takes their own forks in the road. From the outside, it may appear to be a random walk. Nevertheless, these meandering journeys in are common in science.

At the risk of navel gazing, looking back on our scientific journeys can be instructive to other scientists who are overwhelmed at the prospect of looking forward at their careers [Lenski, Chemistry guy, Lyme disease mbio, (A)historical paper]. By no means is my personal journey over, but since 2002 I have been on a journey that I did not realize I was on. Now that the paper introducing the mothur software package is ten years old and has become the most cited paper published by *Applied and Environmental Microbiology*, it is worth stepping back and using the development of mothur as a story that likely has parallels to many other research stories that have taken time to develop.

I fondly recall preparing a poster for the 2002 meeting of research groups supported by the NSF-supported Microbial Observatories Program. I wanted to triumphantly show that I had sequenced more than 600 16S rRNA gene sequences from a single 0.5-g sample of Alaskan soil. This was greater sequencing depth than anyone else had achieved for a single sample. As I was preparing the poster, I walked into the office of Jo Handelsman, my postdoctoral research advisor, and laid out the outline for the poster. She asked if I could add one of those “curvy things”, a rarefaction curve, to show where I was in sampling the community. Rarefaction curves and attempts to estimate the taxonomic richness of soil had become popular because of the simple, but impactful mini-review by Jennifer Hughes and her colleagues, which introduced the field to operational taxonomic units (OTUs), rarefaction curves, and richness estimates [DOI: 10.1128/AEM.67.10.4399-4406.2001]. I do not recall whether that poster had a rarefaction curve on it, but her question primed my career.

Introducing DOTUR and friends. When Jo asked me to generate a rarefaction curve for the poster, the request was not trivial. How would I bin the sequences into OTUs? Hughes and her colleagues did it manually for datasets that had fewer than 284 sequences. Although I could possibly do that for my 600 sequences, my goal was to generate 1,000 sequences from the sample and to repeat that sampling effort for other samples. I needed something that could be automated. Furthermore, the software that Hughes used, EstimateS, required a series of tedious data formatting steps. I had found my first problem. How would I assign sequences to OTUs and use that data to estimate the richness and diversity of a sample? The second problem would be how could I compare the sequences found in one sample to another sample? The solution to the first problem, DOTUR (Distance-based OTUs and Richness), took us two years to develop [10.1128/AEM.71.3.1501-1506.2005]. DOTUR did two things: given a matrix describing the genetic distance between pairs of sequences, it would cluster those sequences into OTUs for any distance threshold to define the OTUs and then it would use the frequency of each OTU to calculate a variety of alpha diversity metrics. The solutions to the second problem would come from our work to develop software including S-LIBSHUFF [2004], SONS (Shared OTUs and Similarity) [2006], and TreeClimber [2006]. Around the same time, Catherine Lozupone and Rob Knight were developing their UniFrac tools to compare communities with a phylogenetic rather than OTU-based approach [PMID: 16332807; PMID: 17220268]. With these tools, the field of microbial ecology had a quantitative toolbox for describing and comparing microbial communities.

It is important to remember that we knew there were many problems with 16S rRNA gene sequencing. We knew there were biases from extractions and amplification [XXXX]. We knew there were chimeras [XXXX]. Getting to the distance matrix required trimming and correcting sequence errors, aligning them, and finding the most appropriate way to calculate a pairwise distance. It was also rare to have experimental replication to perform statistical tests to compare treatment groups. We frequently used a dataset comparing Scottish soils from from Alison McCaig and colleagues. This dataset consisted of two experimental groups, each replicated three times with 45 sequences in each replicate. Nevertheless, we had excuses and work arounds for these problems that served our needs. At the time, I felt that the biggest problems were how to cluster the sequences into OTUs and how to use those clusterings to test our hypotheses. Along the way we would demonstrate the utility of such tools to answer questions like where are we in the bacterial census? How many sequences would it take to see every OTU in that sample of Alaskan soil? How does the word usage of *Goodnight, Moon* compare to that of *Portrait of a Lady*? More importantly, 1,900 papers used DOTUR to facilitate their own research questions. Had we waited to solve all of the problems that plague 16S rRNA gene sequencing, we would still be waiting.

As we developed these tools, I found a unique niche in microbiology. I believe that my undergraduate and graduate training as a biological engineer prepared me to think about research questions from a systems perspective, to think quantitatively, and to understand the value of using computer programs to help solve problems. As an undergraduate engineering student, I learned the Pascal programming language and promptly forgot much of it as an engineering graduate student. As a postdoc, I learned the Perl programming language to better understand how LIBSHUFF, a tool for comparing the structure of two communities, worked since it was written in Perl [DOI: 10.1128/AEM.70.9.5485-5492.2004]. After writing my own version of LIBSHUFF and seeing the speed of the version written in C++ by my collaborator, Bret Larget, I converted my Perl version of DOTUR into C++. At the time, the conversion from Perl to C++ seemed like an academic exercise to learn a new language. My Perl version only took a minute or so to process the final collection of 1,000 sequences and the C++ version took seconds. Was that really such a big difference? In hindsight, as we now process datasets with millions of sequences, the decision to learn to C++ was critical. The ability to pick up computer languages to solve problems was enabled by my prior training. It was also a skill that was virtually unheard of in microbiology. Today researchers without the ability to program in Python or R are at a significant disadvantage.

Introducing mothur. Shortly after DOTUR was published, I received an email from Mitch Sogin asking whether DOTUR could handle more than a million sequences. Without answering his question, I asked where he found a million sequences. Little did I know that his email would represent another pivot in the development of these tools. His group would be the first to use 454 sequencing technology to generate 16S rRNA gene sequences [PMID: 16880384]. Although mothur could assign those sequences to OTUs, it was slow and required a significant amount of RAM. As I left my postdoc to start my independent career across the state from Sogin's lab at the University of Massachusetts in Amherst, my plan was to rewrite DOTUR, SONS, S-LIBSHUFF, and TreeClimber for the new world of massively parallelized sequencing. The new tool was mothur.

Milling about at a poster session at an ASM General Meeting in New Orleans, I again ran into Mitch who asked what my plans were for new tools. I told him that I wanted to make a tool like ARB (a powerful database tool and phylogenetics package), but for microbial ecology analysis. His retort was, “You and what army?” To that point, I had written every line of code and been answering many emails from people asking for help. It would be difficulty, but I needed to learn to let go and share the development process with someone else. He was right, I would need an army. That “army” ended up being Sarah Westcott who has worked on the mothur project largely from its inception. Today, mothur is over 200,000 lines of code and Sarah has touched or written nearly every line of code. Beyond writing and testing mothur’s code base, she has become a conduit for many learning the tools of microbial ecology by patiently answering questions via email and the package’s discussion forum. The community and I are lucky that Sarah has stayed with the project for more than a decade. To be honest, such dependency on a single person makes the project brittle. In hindsight, it would have been better to have developed mothur with more of an “army” or team so that there is overlap in people’s understanding of how mothur works. Although such an arrangement might work in a software engineering firm, it is not practical in an academic setting where funding is limited for developers of free, open source software packages. There are certainly projects that make this work, but they are rare.

Challenges of making open source count. Anyone can post code to GitHub with a permissive license and claim to be an open source software developer. Far more challenging is engaging the target community to make contributions to that code. Frankly, we have struggled to expand the number of people that make contributions to the mothur code base. One challenge we face is that if we looked to third parties to contribute code to mothur, they would need to know C++. Given the paucity of microbiologists with any programming skills, expecting that community to provide contributors that can write code in a syntax that prizes execution efficiency over developer efficiency was not likely. In contrast, the QIIME development team could be more distributed because their code base was primarily written in Python, which prizes developer efficiency over execution efficiency and exists as a series of wrappers to execute other developers’ code. These choices resulted in many tradeoffs that have impacted ease of installation, usability, execution speed, and flexibility. If we were offered a grant to rewrite mothur, we would likely rewrite it as an

R package that leaned heavily on the R language's C++ packages. Of course, such choices are always best in hindsight and when we started developing mothur, the ability to interface between scripting languages like R and Python and C++ code was not as well developed as it is today, For example, the modern version of the Rcpp package was first released in 2009 and its popularity was not immediate. Again, the development of mothur has been a product of the environment that it was created in. Although these decisions have largely had positive outcomes, there have been tradeoffs that caused us to sacrifice other goals.

Beyond contributing to the mothur code base, we sought out other ways to include the community as developers. The paper describing mothur included **N** co-authors, all but three (Schloss, Ryabin, and Westcott) responded to a call to provide a wiki page that described how they used an early version of mothur to analyze a data set. Our vision was that authors might use the mothur wiki to document reproducible workflows for papers using mothur but to also provide instructional materials for other seeking to adapt mothur for their uses. Again, this vision was a product of the environment. IPython notebooks (2011) and R markdown (2012) would not be developed until later. Unfortunately, once the incentive of co-authorship was removed, researchers stopped contributing their workflows to the wiki. Part of the difficulty of recruiting wiki contributors was a perception by some that the wiki was not a community resource. For example, I would frequently receive emails from people telling me that there was a typo on a specific page when the intention was that they could correct the typos without my input. We have been more successful in soliciting input and contributions from the user community through the mothur discussion forum and GitHub issue tracker. As mothur has matured, we have been dependent on the user community to use these resources to tell us what features they would like to see included in mothur. The user community also tells us where our documentation is confusing. Often we can count on people not directly affiliated with mothur to provide instruction and their own experience to other users. We are constantly trying to recruit our "army" and are happy to take any contributions we can. Whether the contributions are to the code base, discussion forum, or suggestions for new tools, these contributions have been invaluable to the growth and popularity of mothur.

Failed experiments. If we never failed, we would not be trying hard enough. Over the past decade we have tried a number of experiments to improve the usability and utility of mothur. One of our first experiments was to use mothur to generate standard vector graphic (SVG)-formatted files of heatmaps and venn diagrams depicting the overlap between microbial communities. I quickly realized that I would never put a mothur-generated figure into a manuscript I wrote. Such visuals require far too much customization to be publication-quality. Although QIIME has incorporated visualization tools through the Emperor package, the challenge of users taking default values has downsides as ordinations with black background or publishing 3-D ordinations in a 2-D medium litter the literature. Instead, we have encouraged users to use R packages to visualize mothur-generated results using the minimalR instructional materials. A second experiment we attempted was to create a graphical user interface (GUI) for running mothur. Forcing users to interact with mothur through the command line has been a significant hurdle for many. Unfortunately, the development effort required to create and maintain a GUI is significant and there is limited funding for such efforts. The newest version of QIIME (version 2) has emphasized interaction with the tools through a GUI and it remains to be seen how this experiment will go. Another downside of using a GUI is that there is a risk that reproducibility will suffer if users do not have a mechanism to document their mouse clicks. This documentation is explicit in mothur as all commands and output is recorded in a logfile. Given the heightened recent focus on reproducibility we have extended significant effort in developing instructional materials teaching users how to organize, document, and execute reproducible pipelines that allow a user to go from raw sequence data to a compiled manuscript with figures through the Riffomonas project. Finally, we collaborated with programmers through Google Summer of Code to develop commands in mothur to implement the random forest and SVM machine learning algorithms. Similar to the challenges of developing attractive visuals, fitting the algorithms' hyperparameters, testing, and deploying the resulting models requires a significant amount of customization. Furthermore, this is an active area of research where methods are still being developed and improved. Thankfully, there are numerous R and Python packages that do a better job of developing these models. Again, we have put our efforts into developing instructional materials that mothur users can use to fit such models to their data. In each of our "failed" experiments, the real problems were straying from what mothur does well and failing to grasp what we really wanted the innovation to do. In hindsight, our solution to these failure has been to provide tutorials to as a conduit between mothur and their goals.

Competition is good and healthy. From the beginning there have been online tools the Ribosomal Database Project (RDP), greengenes, and SILVA. These allowed users a straightforward method of comparing their data to those collected in a database. There are two primary downsides to these tools. First, researchers running the online tool must pay the computational expenses leading to slow process times as numerous users simultaneously attempt an analysis and when hardware becomes outdated. Eventually this limitation would result in the termination of the greengenes website. Second, these platforms provide a one-size-fits-all analysis. These tools only allow a user to analyze 16S and in some cases 18S rRNA gene sequences. If a users analyzes a different gene then the tool will not serve them. These observations resulted in two design goals we have had with mothur: bringing the analysis to a user's computer and separating a tool from the database. For example, we commonly use a sequence alignment method that was originally developed for greengenes, but use the SILVA databases. In addition, we offer the naive Bayesian classifier developed by the RDP and allow users to train it to any database they want, including customized databases. In both examples, users can align or classify non rRNA gene sequence data. As the bioinformatics tools have matured, both the RDP and SILVA offer pipelines for analyzing large datasets, albeit in one-size-fits-all black box implementations.

With the growth in popularity of 16S rRNA gene sequencing there has naturally been an expansion in the number of people developing tools to analyze these data. Months after the paper describing mothur was released, QIIME was released with similar goals. Over the past 10 years, many have attempted to compare these two programs: Pepsi vs Coke, Apple vs Windows, etc. It is never clear which software is which brand and whether it is a complement or an insult. Regardless, both programs are very popular. From my perspective, most of the differences are cosmetic. QIIME is effectively a bundle of scripts to run other developers' software. For example, with QIIME v1, it was possible to run mothur as part of QIIME. One can also run the naive Bayesian classifier through QIIME using the original code developed by the RDP. From my perspective, this caused QIIME many problems because there were numerous software dependencies that had to be installed causing frustration for users. Although the QIIME developers would create virtual machines and packaging tools to simplify installation, these fixes required sophistication by the user. In contrast, when a user runs mothur, they are running mothur. The naive Bayesian classifier code that is in mothur is a rewritten version of the original code. When we rewrite someone's software we do it with an eye to improving performance, access, and utility for non-16S rRNA gene sequence data. For example, while 454 data was popular, PyroNoise was an effective tool for denoising flowgram data. Running the original code required a linux computer and knowledge of bash and Perl scripting. When we rewrote the code for mothur, we made it accessible to people using any operating system with a simple command interface. I have found that our approach requires significant developer effort, but saves considerable user effort.

Beyond the large packages like mothur and QIIME, there has been significant growth in stand alone boutique software tools for sequence curation (e.g. PyroNoise and DADA), chimera checking (e.g. UCHIME, ChimeraSlayer, and Perseus), and clustering (e.g. USEARCH and Swarm). Where possible, we have implemented these algorithms directly into mothur. We have also used this diversity of methods to perform head-to-head comparisons. Most notable is the area of clustering algorithms where there have been a large number of algorithms developed without an obvious method to objectively compare them. Through a series of studies, we applied an objective metric, the Matthew's Correlation Coefficient (MCC), to numerous algorithms for clustering sequences into OTUs. By performing this type of analysis, we were able to objectively compare the algorithms, make recommendations to the field, and develop new algorithms that outperformed the existing algorithms. We have embraced the competition and diversity of all methods being used to analyze amplicon data as an opportunity to identify the strengths and weaknesses of the methods in an attempt to make recommendations to other researchers.

What are we proud of? * Open source packages * Platform independent * Development of open instructional materials * Data driven development of methods * Database independent - allows people outside of our field to use mothur * Reproducibility * Helped create standards - when something is this popular, you force people to use a certain suite of tools and methods

The future. I plan to continue developing mothur for as long as other researchers find it useful. One challenge of such a plan is maintaining the funding to support its development. The development of mothur was initially enabled by a grant to Mitch Sogin from the Sloan Foundation to support his VAMPS (Visualization and Analysis of Microbial Population Structures) initiative. We used that seed funding to secure an NSF grant and then a grant from NIH for tool development as part of their Human Microbiome Project. Since that project expired in 201X, we have not had funding to specifically support mothur's development. I have been fortunate to have discretionary funds generated from other projects to help support mothur. Although there is funding for new tools, there appears to be little appetite by funders to support existing tools. Emblematic of this was the NIH program, Big Data To Knowledge (BD2K), which solicited proposals through the program announcement “Extended Development, Hardening and Dissemination of Technologies in Biomedical Computing, Informatics and Big Data Science (PA-14-156)”. This opportunity appeared perfect, except that the National Institute of Allergy and Infectious Diseases (NIAID), the primary supporter of microbiome research at NIH, did not participate in the announcement. Tools like mothur have been successful and need mechanisms to continue to mature and support the needs of the research community.

As with anything in science, methods become passe. When we first developed mothur, tools like T-RFLP and DGGE were still commonly used. Today it would be hard to argue that data from those methods meaningfully advance a study relative to what one could get using 16S rRNA gene sequence data. Looking forward, many want to claim that amplicon sequencing is today's DGGE. They claim that instead researchers should move on to shotgun metagenomic sequence data. It is important to note that the two methods answer fundamentally different questions. Metagenomic sequence data tells a researcher about the functional potential and genetic diversity of a community and 16S rRNA gene sequence data describes the taxonomic composition. Both tools provide important information but they cannot easily replace each other. Although metagenomic data does provide highly resolved taxonomic information, the limit of detection is much higher than that of amplicon data. For example, we analyzed 10,000 16S rRNA sequences from each of about 500 subjects. We can think of this as representing about 1,000,000 genome equivalents (10,000 16S rRNA genes/subject x 500 subjects / 5 16S rRNA gene sequences/genome). Assuming a genome is 4 Mbp, this would represent a sequencing depth of 4 Tbp. Although such a sequencing effort is possible, albeit expensive, analyzing such a large dataset with an approach that captures the full genetic diversity of the community would be financially and technically prohibitive. Going forward, there is still a place for 16S rRNA gene sequencing. Although sequencing technologies will continue to evolve to capture longer and more high quality data, there will likely always be a need for characterizing the taxonomic diversity of microbial communities. With this in mind, there will always be a place for tools like mothur.

Of course this does not mean that such tools will remain static. We see three key areas that we will continue to help to develop. First, just as we adapted through the transitions from Sanger to 454 to MiSeq sequencing platforms, we must learn how to adapt data from PacBio, Oxford Nanopore, and other tools to our analysis pipelines. As with these earlier platforms, we must understand the error profiles so that they can be corrected. We have learned that moving forward requires that we maintain or improve sequence quality. No doubt, datasets and read lengths will improve. Second, with these improvements, we will need to continue to improve our algorithms. We have already seen that attempts to use low quality MiSeq and HiSeq data causes computational problems leading to the creation of open and closed reference clustering methods. Unfortunately, comparative analyses showed that these methods fail relative to *de novo* clustering methods. More work is needed to improve reference-based clustering methods so that larger datasets can be analyzed without sacrificing quality. Finally, there are ongoing controversies that need further exploration. These include the validity and utility of amplicon sequence variants, the validity and impact of removing low frequency sequences, and methods of identifying and removing 16S rRNA gene sequences from contaminants. In the original mothur paper, we commented that 16S rRNA gene sequencing and analysis is very much like the Red Queen in Alice in Wonderland. Although some disagreed with this analogy, we still feel it fits. The sequencing technology and rapacious appetite of researchers continues to race on. At the same time, bioinformatics tools must adapt to facilitate our research. I am confident that mothur will be up to this exciting challenge.

Acknowledgements

References

25 references

Figure 1. Caption caption caption. Footnotes footnotes footnotes.