OS18

Surviving A Sunset:
Being Resilient in the Face of Open-Source Disruption

Frank Menius, Atorus Research, USA

ABSTRACT

Let's face it, the world of data science and programming is changing. Open-source technologies like R and Python are gaining momentum across many industries where proprietary institutionalized software solutions once stood unchallenged. The problems and promises of larger decentralized data sources; real world data; machine learning; artificial intelligence; and faster outcome expectations are driving companies to seek out new, innovative solutions in pursuit of migrating market share. The pharmaceutical and life science industry is no different and is diving headlong into adopting open-source as an efficient tool to get therapies to market faster. So, how do you, as a statistical programmer, survive this disruption without being left behind? This paper will focus on the key skills you need to hone today to ensure your programming future without wandering off into the sunset.

DISRUPTION

Let's talk a bit about disruption. Disruption is a break or interruption in the normal course or continuation of some activity. Specifically, the disruption I am talking about is the introduction of open-source solutions into the clinical trials industry, which has been dominated by a small group of proprietary solutions. This introduction will cause change, and that change can be difficult and scary for those invested in the status quo. None are likely to be invested more than the statistical programmers that will be forced to adapt to this disruption. I am here to tell you that the disruption introduced to the industry can also be an engine of opportunity for those that embrace it.

ARE YOU READY FOR DISRUPTION

Several Years ago, traffic circles started appearing on highways all over the country. They were a way to make traffic more efficient and safer. They work, but not everyone adapted quickly, or at all. Not everyone was ready for big changes in the way they drive. Some still aren't. We have all seen it. Some people treat it like it is a stop sign. Some people forget to yield. Some people can't figure out how to get out. Then, there are just a few people who refuse to see the change at all.

So, do you think you are ready for disruption? Like with the traffic circle our industry is rapidly changing the way we have done things for years. In the clinical research industry, many programmers program their whole career with just one language. Many programmers will only ever program one part of a submission. It could be datasets, maybe it is

tables, listings, and figures (TLFs), but often programmers find themselves siloed into one topic or one specialty. Because we have followed this process for years in the clinical research industry, we have built up this infrastructure of expertise. We really only have to look internally for help when we need it.

Now let's look at some of the disruptions that come from the adoption of open-source solutions. Increasingly statistical programmers are being asked to be multi-lingual. They are expected to be able to utilize the right tool for the right problem. This may mean knowing and being able to utilize two or more programming languages efficiently. This is a big change for the industry. Ironically, most other programmers in other industries have never experienced this monoculture of programming. It is pretty unique to our industry to have only had one programming option for so long. Additionally, programmers utilizing this multi-lingual approach are being asked to be less focused on a specific topic and more adaptable. They may find themselves working on a wide range of topics including traditional datasets and TLFs, but maybe also some other things. Perhaps they are being asked to develop applications or make user interfaces for reporting. They could be developing packages for use internally or externally to their company. They could even be working on machine learning and artificial intelligence (AI), the list goes on. Additionally, expertise in open-source is likely not internal to our organizations or even our industry. Users are now faced with navigating a world where millions of people globally are producing and maintaining the tools they use. Now statistical programmers have to look outside for experts.

HOW CAN YOU SURVIVE DISRUPTION?

For several years I have been involved in the push to adopt open-source solutions within the clinical research industry. For the past year I have been blessed to have a position to actually train industry programmers on how to utilize open source for their work. Through this I have made some observations that I hope can help you be prepared for this disruption and also, take advantage of it.

Engage Early

The first thing I have noticed is that self-starters and those who engage early have a clear advantage over those who wait. When I have trained multiple groups of people from a single company, I find a difference between those who attended the first cohorts and those that attended the last. That first cohort is often much more engaged. Their questions are geared more towards how they can adopt open-source to enhance their work. While the later cohorts tend to be less motivated. Their questions focus more on the difficulty of adopting open-source in leu of their current process. I encourage you to open your mind to the idea that open-source can be an enhancement to you and not a detriment. It is likely a disruptive change is going to be coming whether you want it to or not, so why not find a way to embrace it. Early adopters gain a competitive advantage as the paradigms in the industry shift. By embracing the change, and being on the front lines instead of resisting, they find themselves in a position to influence the change and policy. These early adopters become leaders and subject matter experts.

Embrace a Diversity in Skills

The next piece of advice I have for you is to embrace a diversity in skills, especially if your organization has not yet decided how it wants to implement new solutions. There is a wide world out there, become multilingual. That may mean learning more processes in SAS, exploring R, playing with python, picking up SQL, or some other tools and languages. The more you are exposed to the easier it will be for you to adapt to any changes that come. Like with your investment portfolio, spread out your skill base so that you can adapt to any shift your company, or the industry makes.

Find Training

If you want to be multilingual and an early adopter, then you need to learn new skills. This can be scary but there are some simple steps you can take. First, if your company has training on open-source solutions, sign up for them. Become involved. If your company doesn't have this training, ask for it. Ask what your companies plans are for dealing with open-source disruption in the industry. You might find yourself in a role to influence whatever disruption does occur. If there is no training at your company that is not stopping you from learning. There are numerous resources available to you. While there are paid trainers like me who would be happy to train you, learning new skills doesn't have to cost you anything more than time and willpower. One of the best resources in the open-source world is the number of free books available online. I have included links to many of these in the references section.

Don't be Married to the Process

My next observation for you is that you should not be married to the process. It has been said that the most dangerous phrase uttered in history is "We've always done it this way". I know myself as someone who continually tries to improve processes, it is my biggest pet peeve. In clinical trials sometimes we get so bogged down in our day to day that we forget the reason we do the work. That reason is to get lifesaving therapies to patients in a safe and efficient way. Therefore, if a new idea can get those treatments to patients safely and faster, then we have a duty to embrace it. To do this in programming we need to focus on outputs and the deliverables, not the process that we use to make them. That process may change, but quality outputs don't. Programming the same output in 2 different

programming languages may take radically different processes, so focus on the end goal. If you are learning a new language don't necessarily try to replicate the steps you used in your first language. Your new language may have a more efficient or easier way to get to the results you want.

Learn to Find Help

One thing that you need to master as a new open-source programmer is learning how to find information and ask for help. I mentioned that in the open-source world expertise likely lies outside of your company and possibly outside of the clinical research industry. So how do you find it? Well, you could start by joining the community. Community Forums are groups of programmers, developers, and users who share problems, questions, tools, and solutions. Here are just some of the very active community groups that are out there where you can find answers and expertise.

- Posit Community is hosted by the developers of the R-Studio IDE and shares discussions of Data Science Operations, and how to scale them for complex environments.
- Reddit is a collection of communities (subreddits) that you can join and participate in. Here you can talk about anything from food, cars, politics, and yes R and Python Programming. Each Subreddit has its own rules and community.
- Stack Overflow is a question-and-answer forum website for computer programmers and developers and has many threads on open-source programming.
- Even GitHub has a platform for communicating with other developers and users and finding other community groups.
- For those who like to be visually walked through solutions there are thousands of videos on YouTube that can walk you through step-by-step solutions and tools.
- You can even find clubs both locally and online that bring programmers and developers together.
    - R Ladies is a worldwide organization whose mission is to promote gender diversity in the R community, and I assume you are already familiar with phuse.

This is all just a small list of community resources you can find out there but one thing that all of these have in common is that the open-source community is very active and very responsive. You will be amazed at how fast you might receive an answer to your questions. Many developers will even work to implement updates to their code based on your use cases.

Still, you may wonder where to get started. Well, you could join one or more of the community forums and ask your question directly. They also often have internal search bars and help options. If you are working on something and looking for a more immediate answer probably the easiest place to start is with a simple search engine like Google, Bing, DuckDuckGo, whatever is your preference. If you have a question, simply search for the solution. Often the first result will be one of the Reddit or Stack Overflow sites. Someone else has probably already asked your question and received an answer. Using a search engine is pretty straightforward, but there are definitely some tips that can be used to enhance your search results.

- Use quotes to get an exact match
- Search within a specific site by adding "site:<the website you want to search>" before your question
- Exclude results by putting - in front of them in the search
- Use the wildcard * if you are unsure about a term in your query and let the engine search for relevant replacements
- Combine searches with OR, AND logic
- Filter out searches by time period with AFTER:, BEFORE: or . .

Many search engines now employ an AI or large language model to assist your search. These often will summarize all the results and combine the most relevant results for you. While I found this to be helpful most of the time my suggestion with AI is going to always be "Trust but verify". Don't just take the AI answer as the truth. Try and find the AI's sources, they are typically directly below the summary. If the AI is giving you an actual code suggestion, I would be very suspicious as to if it would be entirely correct. That is not to say it isn't a good place to start it just is going to take some work on your part to verify, edit, and debug.

Finally, a rule for getting external help whether it is a community forum or AI or even a colleague, is know your companies standard operating procedures (SOPs). Don't share your sensitive data or proprietary information. If using AI check with your company to see if they have an internal firewalled AI that you can access without risk to your data. Otherwise, make an effort to ask questions without exposing your data or use dummy data.

Be Resilient

Finally, my last piece of advice is to be resilient. Change is scary. You are not going to know everything upfront. You will have to learn new skills. There are things you will not do correctly. You won't understand everything at first. You will make mistakes. This is ok. Know that you can adapt. Know that you can be successful. You are also not

alone. Many others are going through this disruption and have gone through disruptions before. They can do this and so can you. Change brings opportunity, seize it.

RECOMMENDED FREE READING RESOURCES

Hands-On Programming with R
Garrett Grolemund https://rstudio-education.github.io/hopr/

R for Data Science: Import, Tidy, Transform, Visualize, and Model Data 2nd Edition
by Hadley Wickham https://r4ds.hadley.nz/

R Programming for Data Science
by Roger D. Peng https://bookdown.org/rdpeng/rprogdatascience/

Efficient R Programming: A Practical Guide to Smarter Programming
by Colin Gillespie, Robin Lovelace https://csgillespie.github.io/efficientR/

Advanced R
by Hadley Wickham https://adv-r.hadley.nz/

Mastering Software Development in R
by Roger D. Peng, Sean Kross, Brooke Anderson https://bookdown.org/rdpeng/RProgDA/

Modern Statistics with R
From wrangling and exploring data to inference and predictive modelling
Måns Thulin http://www.modernstatisticswithr.com/

Happy Git and GitHub for the useR
by Jenny Bryan https://happygitwithr.com/

R Graphics Cookbook: Practical Recipes for Visualizing Data
by Winston Chang https://r-graphics.org/

ggplot2
by Hadley Wickham https://ggplot2-book.org/

R Packages
by Hadley Wickham https://r-pkgs.org/

Mastering Shiny
by Hadley Wickham https://mastering-shiny.org/

Big book of R [UPDATE]
by Oscar Baruffa https://www.bigbookofr.com/

Think Python: How to Think Like a Computer Scientist
by Allen B. Dawney https://greenteapress.com/wp/think-python-2e/

Automate the Boring Stuff with Python: Practical Programming for Total Beginners
by Al Sweigart https://automatetheboringstuff.com/

Python for Everybody: Exploring Data in Python 3
by Dr. Charles Russell Severance https://www.py4e.com/

Invent Your Own Computer Games with Python
by Al Sweigart https://inventwithpython.com/

The Hitchhiker's Guide to Python: Best Practices for Development
by Kenneth Reitz https://docs.python-guide.org/

Think Stats: Exploring Data Analysis
by Allen B. Downey https://greenteapress.com/wp/think-stats-2e/

Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit
by Steven Bird https://www.nltk.org/book/

And Many More…

RECOMMENDED COMMUNITY FORUMS

Posit Community https://forum.posit.co/

Reddit https://www.reddit.com/r/Rstudio
https://www.reddit.com/r/programming
https://www.reddit.com/r/datascience

Stack Overflow https://stackoverflow.com/

Github https://github.com/community

YouTube https://www.youtube.com/

R Ladies https://rladies.org/

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:
Author Name: Frank Menius
Company: Atorus Research
Work Phone: 919-412-6537
Email: frank.menius@atorusresearch.com
Website: https://www.linkedin.com/in/frank-menius/