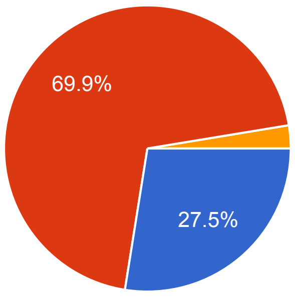
272 responses

View all responses

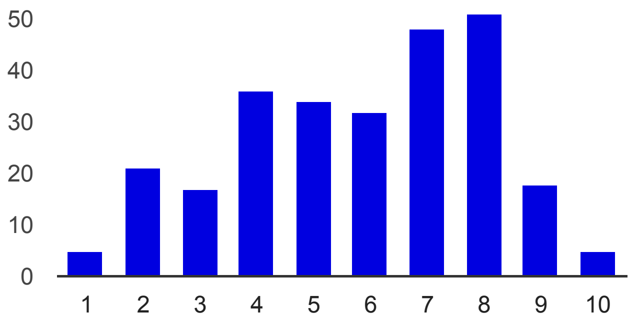
Summary

Are you located in the UK?



Yes	**74**	27.5%
No	**188**	69.9%
Is Scotland still in the UK?	**7**	2.6%

How would you rate your level of bioinformatics expertise (0 = total n00b, 10 = Heng Li) ?



1	**5**	1.9%
2	**21**	7.8%
3	**17**	6.3%
4	**36**	13.3%
5	**34**	12.6%
6	**32**	11.9%
7	**48**	17.8%
8	**51**	18.9%

9	**18**	6.7%
10	**5**	1.9%

What things most frustrate you or limit your ability to carry out bioinformatics analysis?

Multiple tools in multiple programming languages

Programming

- inconsistent use of data formats; - access to shared compute resources (in theory, I have access to lots of shared resources; in practice, queue times, disk allocations, etc are very limiting); - inability to distinguish at an early stage how much effort I need to put into software engineering for a particular program/script (i.e., I frequently over-optimize code that never gets used again, and I frequently continue to use kludgy, poorly-written code over and over again without putting in a small amount of effort to clean it up and document it, which would save me a lot of time)

- limited storage space - limits to sharing human (patient) biological data - "chr1" vs. "1" - incompatibility of tools that ought to work together - tools: lack of portability / poor packaging / bit rot

Replicates. I.e good quality input data. Poorly documented code and scripts. Resources that assume I have time to work out their idiosyncratic usage methods. Lack of knowledge. Lack of time.

Computer capacity , immature software

Bandwidth, large datasets to transfer

Software installation. Lack of bioinformatics support (1 other person in the department that does sequence analysis from start to finish).

Bioinformatics expertise is not what hampers me. It is more the frustrating things that cannot be fully automated, which slow me down: identifier mapping, writing parsers, and searching+reading the literature.

Using VMs, need more coding knowledge...

usability of bioinformatics software/tools out there

Need to learn how to use multiple programs Inconsistency in file formats

lack of competent collaborators at senior level; inadequate compute resources;

Fast and efficient data transfer; lack of suitable IT infrastructure; continually installing updates or new software.

- Tools that are difficult to install - Tools that are poorly documented - Tools where the documentation is well out of date - Tools that have implicit assumptions about versions of other tools that should be explicit - Tools that try to do too much, or have non-standard usage, that makes them hard to integrate into analysis pipelines

Bad software Bad software written in Perl Bad software which masquerades as good software Lack of well-maintained, clear-winner tools like Samtools, Khmer. The NCBI and most of the software it maintains. The many painful steps of dealing with BLAST-like results for analysing metagenomes The suckiness of a large proportion of bioinformatics web services. Dependency management

The poor documentation of programs and the number of programs available making it impossible to figure out which ones are the best to use for your analysis. There is no best practice and it is changing all the time.

My inability to script for parsing/parallelising (and the lack of time to devote to learning it properly)

Having sufficient compute and storage for large-scale exome and genome analysis.

State of bioinformatics software, formats madness

Finding tools to easily convert data formats Things supposedly in a format that are really slightly out of spec and so won't work

Lack of proper comparisons / benchmarking of methodology, and thus standardisation Poorly implemented or documented code Lack of metadata annotation in published sequence data (environmental) Lack of time

Misunderstandings with collaborators who hail from a purely biological\clinical background as opposed to a computational biology\genomics one.

Incompatible software i.e. pipelines that require multiple software, but certain versions not being compatible

Lack of a background in maths and computer science.

Poorly documented software (inlc. web tools, databases, packages, modules, ...). Parsing XML files. Converting Input/Output files from non standard formats. Abandoned software.

First of all my own understanding of how stuff works. Secondly bad documentation. Thirdly: having to install c/perl libraries/modules. Fourthly: limited computing capabilities.

lack of annotation of data

1000 different formats and having to translate between them

Lack of knowledge, dizzying array of different solutions and no way to compare them or to judge which is best for a particular application other than word of mouth, after finally selecting one discovering that collaborators prefer another or you can't install it on a particular server, ennui, lack of time to perform the actual analysis because 5 years has been spent collecting samples and sequencing them and the project should have ended 2 years ago.

documentation for some bioinformatic tools are pretty weak for non high level bioinformaticians there's quite a gap (at least I feel like there is) between a person that can use linux, command lines and do some little scripts to actually being able to do big scripts using several tools, etc.

Programming ability Knowledge of cluster and parallel computing Inability to install software on department infrastructure

Inconsistenties between formats! Biology is hard.

* Lack of clear documentation and example use cases for tools, especially those with a large set of options * Lack of good, "simple" test cases (particularly annoying when building from source; would like to check build is correct in a few minutes, i.e., not

having to run a full human genome through the tool). * Undocumented build and/or runtime dependencies; related: including large and/or duplicate dependency source trees (e.g., BOOST) to avoid such issues. * Lack of long-term stability and release/transition plans for many tools. Especially bad w.r.t. web-based services and data sources, which often disappear/move without notice. * Proliferation of new tools instead of contributing improvements to existing ones (see, for example, short read mappers). Similarly, would like to see a community-maintained, optimized (e.g., using SSE) library for key algorithms like Smith-Waterman to reduce duplicate efforts. * Bizarre software architectures, especially w.r.t. installation directory structures; please, please, please, stick with the "bin"/"share"/"lib" convention and have one main executable. * Hard-coded paths (particularly for tool locations) and similar assumptions about how our site's infrastructure is configured. * Almost all bioinformatics tools shouldn't need admin (root) privileges to install, let alone run. Don't assume install will be in /usr/local (or, even worse, /usr); should be able to build and run out of $HOME/.local (or equivalent) if I want to. * Proliferation of data formats, lack of documentation/standards for them, lack of compliance with standards (if they exist), and the insane number of parsers I have to write as a result. * Web services need to start supporting HTTPS (not just plain, un-secured HTTP). * Licensing: costs for "non-commercial/research/academic use only" software often crippling for smaller biotech startups and hospitals; custom, confusing, unclear, and/or incompatible licenses (e.g., GPL sources except for one key .c file that's marked "academic use only). Talk to a lawyer before licensing your software!

Opportunity/experience

Knowing there is a way to perform a task, but not being able to properly logic out the process in a timely manner.

Poorly documented software that is tailored for a specific organism or group of organisms. Orphan software. Software that only runs on one OS. Not enough time to spend messing with analyses and the data.

Poorly documented open-source code and manuals, very little computational support (both in the lab and the faculty in general - UofC just hired its first Chair position for Bioinformatics), large amount of software options to choose for any one task - no gold standards

-Flat text files -Poor/Awful/Incomplete/custom/docker-only software installation packages -Centralized code development -NCBI/SRA

Code that doesn't work Data that's in the wrong format

limited amount of CPUs and memory on lab server for everyone in the lab, difficulty finding metadata associated with publically available sequence data, some software is difficult to install (requires root privileges, only available for ubuntu, etc)

That there's a lot of crappy software out there that people put out without any real benchmarking.

Installation of tools not always straightforward. Mostly CLI, which is ok. Unclean data.

Often the University infrastructure is at least one major version out of date in terms of the OS. I often have to use Amazon EC2 to get around this, which costs money.

Installing programs with +++ dependancies.

outsourcing (60% discount in bioinformatics service)

Poorly designed websites; Copying data from one website to another

Lack of qualified human resources at all levels

Multitude of software/tools but not knowing which one is the "best" for my data. The uncertainty of genomic downstream analysis because of current limitations of sequencing technologies. Not having many people in the lab that can critically assess the way I analyse data.

Devoid of a linux computer inspite if being a full time bioinformatics phd candidate. The hpc cluster is awesome!

1) Lack of standards: i) formats, ii) tools to use, iii) best coding practices and above all iv) data management and sharing 2) Lack of time versus amount of data to analyse and new knowledge to absorb, all things in the way of becoming a unicorn AKA the ultimate full-stack bioinformatician

Having to wait for data to be moved around

huge amount of necessary working memory for some tools output/input format conversions dependencies of some programs (down to version level)

Lack of time to learn how to do stuff properly

Time.

Having to copy files back and forth between different cluster file systems and I/O throughout limitations.

The words "pilot study" doesn't excuse you from all aspects of good study design. Many biologists nearly complete lack of interest in learning how to analyze their own data or understanding the details of the analysis.

Poor documentation

Lack of good metadata on publicly available datasets, lack of / unclear documentation for new software leading to time wasted.

installation of different software, packages,... and getting them to work on different platforms (linux, mac, windows), both in doing this myself as well as helping students with installing software and getting it to work.

Change Legacy projects

inflexible software software with too many dependencies/complicated configuration Software that use non-standard and incompatible formats

Stupid inconsistencies between projects, e.g Encode says "chr1" and 1K genomes says "1" Lack of null datasets to use to build statistical models Insufficient replicates

My failings. Funding ineligibility. Local computing infrastructure failings. Local bureaucracy failings. PI failings (when won't collaborate or share data, or provide funding for time/hardware).

- converting between file types - lack of common 'grammar' for bioinformatics analysis - lack of easy to access computational resources - lack of solid advice on which techniques to use and when (e.g. best practices)

Lack of compute

Not enough hours in the day. I still make stupid mistakes.

Server waiting time

Software installation. Incompatible formats. Platforms competing to deliver the whole package but don't and make it difficult to export data.

Access to diverse tools in sensible environment. Benchmarking and documentation in tools. Cloud aren't yet for ad hoc and easy-entry, try-and-see tasks. Funding typically underestimated cost of solid bioinformatics.

- insufficiently described methods in papers - dbGaP - code from published papers that's not actually available - code that's not documented - code that won't install - code that doesn't work - too many formats - insufficient funding

IT departments that don't understand that they need to let people get on with things, and that some people need access to more than just Microsoft office and a few minor programs.

Lack of developers.

Lack of a good team.

Software that doesn't work Data in too many different formats Little problems with the data that require you to spend lots of time debugging Too much variability in methods and parameters that people use in their papers. All the money goes to people at large institutions.

Installation of open source packages and their dependencies is often a nightmare

Failure to provide code, and merely providing terse verbal methods section descriptions in its place, makes many papers incomprehensible.

Access to data and HIPPAA protections

- Difficult to navigate the infinite choices of software, algorithms, analyses. I want to move from 16S rRNA sequencing to metagenomics, but how do I even start if I'm doing it entirely on my own? - Difficult to find enough documentation for any tool/software for learning by myself from scratch; - People don't give enough details of chosen parameters for their analyses (e.g. in 16S rRNA gene pipelines), so it's difficult to properly compare across studies.

Lack of time to really check my code as thoroughly as I would like. Lack of standardised pipelines or prevalence of bespoke analysis.

Python modules Non-open licenses

1. Finding a good starting point. Generally I find on-line advice either too basic or too advanced. 2. Trusting that any analyses I do manage to conduct is correct. 3. Understanding what tools to use and what I need to learn before I attempt anything.

Tools that come with a million dependencies

Missing tools that I am forced to implement. The limited computer resources. The size of the human genome. Time in the day. Cluster stability.

- pipelines composed of poorly crafted software - poor documentation, which transforms what should be simple software reuse or adaptation into a hard problem -

not enough compute resources

Standard formats

Dealing with University IT policy bullshit (lack of admin privledges). My work is often halted by this if I want to test out a new package/tool on my local desktop.

No biologist can explain their experiment clearly

closed source / data not publicly available lack of support from IT-department

Few publication authors shares reproducible analysis pipeline!

• Not a lot of institutional or other support for moving beyond 101 stuff. I know what the command line is. I know how to grep. I know how to install and run something like bwa. I don't know how to write decent classes. • Dependency hell. • Software or software advances that only apply to or work in the human genome. • Being self-taught in language of choice means I have to dig to learn best practices. • Continuing own education after learning the basics.

Constraints of having to run Linux on virtual box on a windows pc. Installation/configuration of tools (no doubt compounded by my inexperience and vbox as discussed above)

Wetlab workers not being able to produce sensible sample sizes.

my skills poorly annotated microbial genomes

Too many tasks to do, more lab scientists than bioinformaticians

Published software that can only actually run on the authors laptop. Poor quality sequencing. EBI

File transfer Storage limits Controlled datasets (I appreciate privacy concerns but the turnaround for approvals can run upwards of 6 months)

Recondite documentation for other peoples' tools. Pressure of time – I always have to stop poking at my code before it's perfect to talk to people/eat/fill out surveys. Poorly commented code written by me-from-the-past (ass).

1a) Collaborators who want to do all analysis w/o prior experimental design. 1b) Collaborators who are incapable of providing experimental design. 2) Raw computing power.

undocumented tools/features/limitations.

Data homogeneisation - formating Programming skills Hardware capabilities

poor documentation hard-to-install software no sudo rights

the proxy deploying whatever I want in a server. storage cluster+quotas

Experts are so busy and the pipeline of bioinformaticians not big enough yet And my postdocs have variable issues with access to HPC

Consistent need to reprocess useful public datasets to achieve uniformity Perceived lack of institutional/mentor support/appreciation for value and necessity of implementing valid, rigorous bioinformatics methods

Computer power, size of dataset (metagenomics). Lack of knowledge re: computer languages.

-Poor familiarity with Perl a and coding in general -dependencies that crash or are outdated -uninformative error messages from open-source algorithms (ex: just "core dump" and nothing else)

Difficult to find public samples based on metadata.

Lacks of knowledge about already available tools.

Tools with very complex/limited documentation

I'm a Computer Scientist (just trying to design and develop genomic software), and I do struggle to keep on the path of learning so many genomic words when talking to my biology colleagues. Besides, there is a huge pletora of genomic apps that one cannot know whether is making the right decision or not.

Different file formats. High CPU/RAM requirements.

The complexity of the analyses relative to my limited knowledge; I'm picking up the background and terminology as I go along, but am painfully aware that a mistake through ignorance could invalidate the whole thing (particularly when there's no consensus on the best method anyway). A lot of the documentation assumes a certain level of computing/statistical knowledge that make it rather impenetrable to someone from a different background.

Not enough disk space on the cluster. Not enough free job slots on the cluster. Ridiculously complicated R-packages. Perl module and version incompatibilities. Not enough hours in the day.

Access to computing resources, clusters, cloud resources, etc.

insufficient, inaccurate software documentation time spent trying to convert among file types/rearrange data

crappy runs only on my laptop only that doesn't take into account actually installing software for real on a production system outside of your home directory. Reliance on horrible dependency hell of dynamic libraries (python, R). super user requirements of docker

As a PI, finding good people (grad students, postdocs, staff). Also, in fast-moving fields, systematic ways of assessing the latest tools.

Time

Poor documentation, especially on the effect of different parameter settings.

People re-inventing tools that already exist, rather than enhancing existing tools Long-winded arguments about which programming language is best. Poorly maitained tools lacking supporting information. Politics around data localization. Dyed-in-the-wool industry haters who assume anything commercial can't be any good.

Mathematical expertise Algorithm and model design Coding (fixing this)

- Lack of personal training in good practices - Lack of good open-sourced pipelining tools - Publishing new and better implementations of existing methodologies should be valued considerably more.

-time -successful wet lab work

Large resources failing to carefully consider the extensibility and "consumability" of the product they provide.

Software installation issues Lack of default file formats

- Need to improve coding skills but it's hard to find time - Lack of support/mentoring/oversight - lack of other bioinformatics people to talk to - working with organisms that don't have well annotated genomes - trying to find a documentation strategy that works for me (constantly forgetting to git commit)

1. Poor documentation; 2. inadequate description of a newly published bioinformatic method such that the results of a paper cannot be reproduced without back and forth emails with the author leading me to ask 'who reviewed this paper anyway?'; 3.outdated dependencies; 4. Lack of support from authors on a method they published. 5. Inadequate computational resources

I haven't never properly learnt a programming language I forget commands to use after some break

multiple and/or badly documented data formats, bad software quality, unobtainable data from published studies, incomprehensible Perl (not only...) scripts. Matherial and methods: "data was analyzed using custom Perl|Bash|awk scripts" (not available from anywhere)

Dependencies and poor documentation. My own lack of knowledge and time.

Lack of standardization in almost every aspect

1. Lack of time. 2. NGS machine errors. 3. Investigators not sharing samples, sequences and metadata, especially during MERS and Ebola outbreaks.

No root access. Poor documentation. Install dependencies in Python. Personally, knowledge of statistical modeling.

Too many meetings!

Lack of available time Lack of energy Poor internet connections everywhere that isn't my home or office

data set interoperability

statistics knowledge, R uninformative error messages

Lack of persistence of data and resources Java

Documentation unclear/incomplete, inefficient code - not multithreaded/parallel, waiting in the batch queuing system, my lack of python, R just being R. Lack of time to learn Python or other skills, people just want to know 'the answer' so have to make do with the quick and easiest solution (not necessarily best).

Time installing software

My lack of ability, obvs Lack of good students/postdocs

Told that only provide a command line interface rather than a programmatic API. This often results in unnecessary temp files wasting space and makes it difficult to write analysis as a single script that can be run reproducibly

-No standard names. For example HGNC was not standard till pretty recently. So standard it is great if they are diff. genome severs (ie. USCS, ENSEMBL), but they should talk more each together. -Standard APIs for accessing data for instance it will be a must for USCS

1 - data preparation taking more time than the actual analysis 2 - poor experimental design 3 - poor data quality (you find out after long data preparation)

Poor software engineering The whole "pet bioinformatician" thing Ancient clusters and expensive clouds Lack of understanding of basic software engineering in colleagues

So many tools...dissemination of current best practices and which tools compliment one another is a constant struggle.

Experimenters approaching the core facility after data has been generated and not using our experimental design capability (data autopsy's) Experimenters unable to afford more replicates Experimenters not recording data correctly in the first instance (dodgy excel spreadsheets)

- lack of high memory nodes on my school's cluster. I want ALL the memory. - things take too long. I want instant results. DIAMOND is a step in the right direction.

Time - being pulled in too many directions on too many projects. Underspend on data acquisition. See too many RNA-Seq experiments without replicates, or sequencing efforts unwilling to spend more money to get a complete genome vs spending uncosted time trying to extract as much as possible from a poor dataset.

- Lack of computational trainings for undergrads, and lack of appropriate lab environment i.e. computational labs designed for Bioinformatics students. - Lack of JavaScript tools (for Bioinformatics analysis)

*collaborators sending me poorly documented/named datasets *lack of database/archive curation (e.g. contaminators/mislabelling in GenBank)

Lack of Underlings

Buggy software, software which takes non-standard input and output formats, software which has a lengthy set of dependencies, software which requires you to fax/email to get a license, software with no source code so I have to treat it as a black box Also I fill up my disk quota a lot.

Data transfer speeds Poor documentation Tricky compiles with badly defined dependencies

Inconsistent naming Crap documentation Broken software Mick Watson

Lack of real standards

Programming skills, to manipulate data into formats required by software. Not much time to sit down and learn perl/Python!

Cruddy data formats, terrible software, awful documentation. All of it really!

File formats. Particularly GTF files. My own lack of formal computer science/software engineering training

Poor documentation on tools (either non-existent or the assumed knowledge level is to high). Hundreds of scripts/programs/pipelines that do the same thing, everyone has an opinion on what is best.

- Software with unclear instructions / descriptions that forces you to look at the code to understand what it's actually doing - File formats; even if standards exist they leave too much room for interpretation -> diverging implementations - Bad experimental designs or limited sample availability - Clinical data in Excel tables and with bad format

Access to data sets from other sites. Compute cluster bottlenecks in storage or compute nodes.

There's never a guide that applies to all research, so developing a pipeline separately for each different project can be time consuming. As a beginner, I find it frustrating that there's often no answers to the errors I receive from my commands. If there were a comprehensive list of errors and explanations for each program I use, that would be ideal (but in reality, unlikely to ever happen).

Storage capacity Computational capacity Interoperability of tools

-Bad documentation. -Non-bioinformatically trained collaborators ignoring guidelines for optimal/standara data formats. -Expectation to carry out analysis i)in unreasonably short times ("press a button" bioinformatics misunderstanding ii) free iii) uncredited . Resist and fight in all cases.

not enough bioinformatics support and poor infrastructure

Time to learn how to program, while doing experiments

Available GUI tools, Canned solution like docker needed to remove need for installing additional software dependencies.

disk IO (file copying)

small samples, not enough data

Too many data, too little time (and memory) Not well documented software Lack of collaboration between (some) people Usage of office (seriously guys?) Too much perl

My general lack of programming expertise

The need to learn so many different programs and lack of a computer science background. The best analysis methods change so fast it is hard to keep up.

Dependency hell when installing tools, lack of method description, hardly neutral/unbiased tool benchmarks

Lack of standardized methods by those who generate data

Badly written manuals. People's positions on interleaved and/or non-interleaved FASTQ files.

So many ideas for different analyses - only one of me.

Lack of simple, documented command line tools or similar to manipulate/parse text files on windows desktop. Loading a linux VM is a PITA. Lack of GUI tools with good feedback on function for similar tasks as above - e.g. contrast with power and interface of CellProfiler for potentially very complex image analysis. Lack of time to develop my own skills to enable training of students and create/understand a set of consistent tools. Informatics is only a small part of most of our projects (e.g. gene selection for an expression screen), so activity tends to be sporadic and very task/job centric.

I don't enjoy bioinformatics. I wanted to be a scientist and now this is apparently part of the gig. Coding is hard.

Installation of software programs. Software program updates not being compatible with each other. Overall lack of knowledge, and how confusing it can be to gain it (since I'm a beginner).

Network speed

Getting data from users in a usable or easily usable format. Ease of getting updates to software on the cluster

Confusing directions that go well above my level of understanding. As a noob who struggles with this kind of stuff, I find most new comer guides to assume the student knows way too much.

Bad metadata Getting people to hand over data Not having a time machine or clones

(1) My computer gets angry (2) I need to do pesky human things like sleep and eat. (3) Bad metadata -- can we please get consistent standards that people actually follow. I spend most of my life dealing with metadata. (4) Backups in the wet lab. Our wetlab people are awesome, but sometimes, they get busy.

Large data files (file transfer times). Large storage space requirements for intermediate files. Extremely irritating and time-consuming requirements for submitting data for publication (NCBI)

Lack of practice and how you have to start from zero if you go a while without practicing it.

Knowing what package to use that does not entail spending over £3000 on bespoke software!

file formats, papers written like press releases with few details on key steps, lack of standard workflows even for routine bioinformatics tasks

downloading a data and working and filtering the contaminated data sucks!!!! no enuf time to understand the biology and improve the coding skills better sucks !!!!

Resource limits (e.g. 3 day cap for jobs to be run on the cluster) or lack of necessary resources (e.g. not enough amounts of RAM available).

I am a biologist who taught myself to code. My most common frustration is not being as versatile a programmer as I'd like to be.

Lack of APIs for databases Closed source code in articles Unreproducible analysis in articles due to missing complete descriptions

- HPC structure - storage capacity - time

Assumed that I am bioinformatics support for everything, which includes sysadmin for installing software. Having to constantly evaluate commercial products that are wrappers for open source software. Nobody understanding/caring why software A is more appropriate for a job than software b.

At other jobs it has been IT and infrastructure, now it is metadata storage (and lack of well-defined solutions) and data integration across omics experiments.

My Advisor and hard-drive space

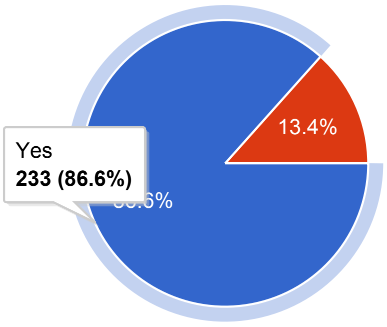
Poor documentation of tools and methods.

(US) Government network infrastructure and the arbitrary security requirements.

Each lab must effectively create a separate internal network to do any meaningful analysis.

Lack of up-to-date info when not connected to the buzz around a state-of-the-art lab.

Do you regularly use online genomic sequence resources e.g. NCBI or EBI to combine with your own data to perform integrated analysis?



Yes	**233**	86.6%
No	**36**	13.4%

If no, why not?

I have a fucking big cluster.

Haven't had the need to as yet

Never needed to

Analytical tools there are poor, irrelevant or wrong. Do use them for storing data.

I mostly used local instances of databases/genomes

Not relevant to my work

Too many tools/data available in other sites too

Because I don't have my own data - I work for one of these resources.

I'd use a reference genome and reference databases like Greengenes and Silva but haven't needed anything else. Very much a biologist who uses genomics as and when.

Difficulty getting and formating

At least I think I do, but maybe I'm not using enough. Mostly just BLAST, but I only do amplicon sequencing.

just beginning to do HiC analysis.

Not interested.

Run local instances.

not applicable for my work (to little data for the organisms i work with)

Not convienient enough, not custom enough

I don't need to use, but my colleagues do.

Not yet.

Lazy

Needs to be downloaded

low throughput

Do n ot know how.

The type of data I'm processing would not benefit from those possible comparisons

I use Qiita

I work on a non-model organism and data is scarce.

Everything I have needed so far is in house

i did previously

Never seen the benefit

Too difficult, not enough documentation.

more of a service facility. generate data, qc.

Too hard to join up disparate tools

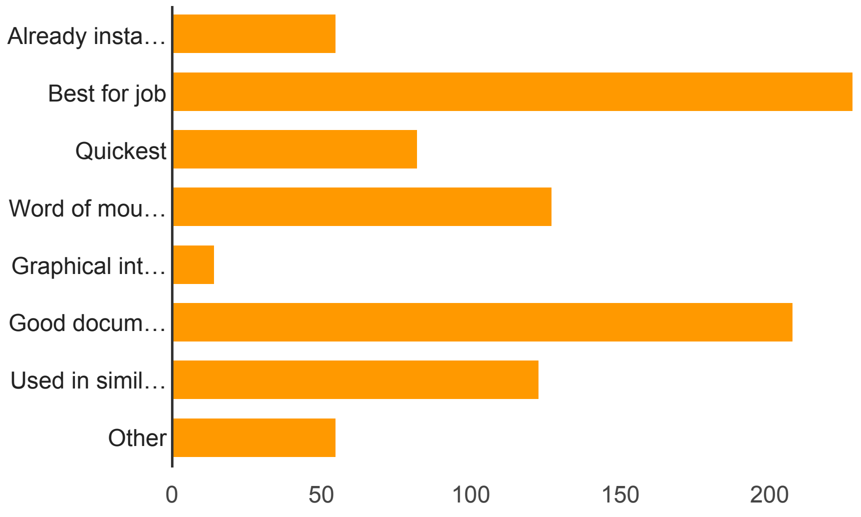
Lacking a use case

Don't integrate particularly well with mainstream Linux/HPC resource at academic institution

I work on providing this kind of data

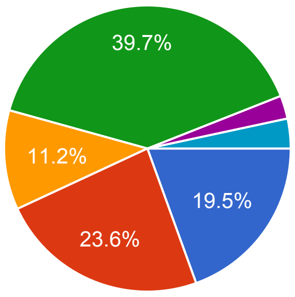
Command line is more compatible with pipelines

What is important when choosing which software to use?



Already installed on server	**55**	20.5%
Best for job	**228**	85.1%
Quickest	**82**	30.6%
Word of mouth recommendation	**127**	47.4%
Graphical interface	**14**	5.2%
Good documentation	**208**	77.6%
Used in similar analysis	**123**	45.9%
Other	**55**	20.5%

Where do you most often perform bioinformatics analysis?



Personal computer (laptop/desktop)	**52**	19.5%
Lab server	**63**	23.6%
Departmental server	**30**	11.2%
University server/cluster	**106**	39.7%
Cloud	**7**	2.6%
Other	**9**	3.4%

What problems running software if any do you most often encounter with this solution?

Updates and installs; lack of standardised system; insufficient memory

software uses too much RAM/time/cpu and limits throughput given finite compute.

setting the limits : too many resources / not enough

Installing software is usually difficult, especially if the software has many dependencies. Software that rely on GUIs instead of command line are terrible to work with on the lab server.

trouble understanding convoluted R documentation.

Compilation, dependencies,

Lack of memory or storage; bursting to cloud sounds like a good idea until we run into bandwidth and data security issues

When running on the lab server, the main problem is lack of RAM, followed by lack of computational power. However, I have access to large clusters and cloud computing facilities, so in those cases I simply run elsewhere.

None - I'm at the Broad and they have a department that supports the servers and installs all the versions of all the software (feel free to spit in jealousy)

Lack of sudo access Poorly maintained software

scaling up to really big datasets lack of parallelizability slow data transfers

Hard to install software (especially if admin privileges needed).

I struggle installing programs i.e. setting the path and making it accessible to all users.

I do not have admin access and hence rely on others to install software packages

Updates of software breaking things.

Managing a large ad hoc cluster.

Running out of memory and/or disk space

Computer too slow, Software incompatible

Resource allocation conflicts Job queuing inefficiencies Installation works on head node but not worker nodes

Once i figured out the shell, got scolded a few times for not qsub'ing jobs in $WORK, filling up $SCRATCH and other n00b mistakes I haven't had any problems - I'm working with relatively small data sets (10GB)

No standardization in command line flags and options among tools (assemblers for instance). Poor scaling. memory utilization.

Have to get some tools installed by sys admin

Software requires very specific data formats but the format is not well documented. No matter how you format the input the response is "data not formatted correctly" On what line, and what character would really help!

Remarkably few; occasionally would be nice to have more cores

Bugs in unmaintained but useful bioinformatics software.

Missing a package I want to use

Installation and configuration issues with tools that aren't quite mature.

Nobody have good tests for their software, so you have to implement your own CI server to see that everything works as expected bedtools/MACS2/STAR/etc

- hard to interpret runtime errors - software often doesn't compile on anything but Linux (or OS X if you're lucky)

Lots of software depencies that sometimes get unsupported

None yet. I imagine wen I start doing more meta genomics work (coming soon) I will have issues with space on my personal lap top.

Compiling software, especially with dependencies and linking libraries.

Dependency Hell.

I cook logic boards.

Not able to install software. Complexity of cluster use.

Unpredictable queuing times. Getting new software installed can take a while.

Software not installed. Competing for resources with coworkers.

Software not installed. Out of disk space on my quota. X11 forwarding being irritably slow.

- no root access so can't install some binary-only software - dependency hell - lack of support for the previous major version of a Linux distribution

- Tools with complex dependencies that require root access to easily install - Tools that crash or just do not work

One of the biggest problems I ruin into is institutional , we don't have admin privileges on our own competes (do on our servers though)

• Long-running jobs (for example, bwa mem with no seed) exceed the allowed time limit and are killed. • Difficulty installing with limited permissions.

We depend on third parties when running out of HD space or RAM.

We have excellent support staff who troubleshoot software packages so this tends not to be a huge problem now that we have switched over to this cluster.

Heng Li stopped answering his email.

low computing capacity running time

Something breaks somewhere and I'm dependent on the server technician being available and able to fix the problem; different people want different versions of the same software; not easy to install new software; sometimes people jam up the server with huge jobs.

None, but some better software is available but too expensive.

Dependencies/incompatibilities (on Linux) - really, didn't Windows solve this in '95 with the end of "the version of run32.dll needed is XXX.XXX, click here to install". Why is this still a problem? Documentation unsuitable for people of my own limited experience. Analysis or data aggregation is often needed very sporadically, meaning gaining genuine expertise is often not time-efficient or recall of how I got it to work last time is poor.

-New users crashing the server (but we have a stellar IT support who are quick to fix and re-train new users. -Potentially long queueing times.

Asking for new modules all the time and waiting for them to be installed is a hassle, so managing my own environment can be a problem especially when servers are dismantled, changed, upgraded, etc.

Errors while installing

Maintaining the server myself - steep learning curve to acquire the necessary sysadmin skills Compatibility issues

Package dependencies and data specific run-time errors.

Too slow, and so I am learning how to use databases and servers/clusters

dependency hell

DEPENENCIES!! Pain in the ass.

See above.

Very rarely any

- long queue times for high memory or GPU nodes - software not configured correctly - some software requires recompiling to change parameters (not ideal...), thus would have to ask support staff to do that for me. - downtime frequent

python module cannot found and permission denied , most of the services blocked

Constantly maxing out local disk capacity

Out of date libraries Occasional downtime of the cluster

User error!

Dependencies.

Generally CPU limited - but that's what the cluster is for.

Documentation problems

incompatibilities

False positives, false negatives & scalability.

1) environment incompatible with software (requires assistance from sys admins), 2) software immature (I'll correct a handful of coding errors, but more than a few obvious

bugs cause me to distrust/abandon a new tool)

Cluster staff used to maths not bioinformatics so we need to trouble shoot every new install ourselves

Lack of error documentation

Dependencies

Difficult to download new programs onto the server. Probably based on my own lack of computing ability.

Institutional bureaucracy Need for grants/cost of entry

installation, random poorly-documented bugs

Memory. Available server time. Suitable permissions to install software as needed. Needless restrictions on activity. Cost.

Poorly documented software, dependency conflicts, installation process not clear.

Competing with other projects for compute capacity can slow analyses.

Slow Installs on personal computers can sometimes be painful. Issues with dependencies, compilers, etc...

Dependencies unavailable. Only works with a particular version of Java/Python etc. Input has to be zipped. Input has to be unzipped. Output is unzipped ...

Poor documentation. Specific out-of-date dependencies. Dependencies not installing correctly. Obtuse error messages. That horrible feeling when you want to do a new bit of analysis, you find what looks like the perfect tool already designed and it turns out to be the biggest festering shite you've ever compiled on your computer

Slow connection. No root access.

lack of resources (when sharing with others)

Time-sharing, few options for extending window of use.

Firewall makes remote access more annoying; personal desktop is powerful and guaranteed to be available but not as powerful as shared servers

Novice users running custom scripts that spawn too many (tens of thousands of) jobs on one node.

Software doesn't compile on older OSes Bottleneck in installing libraries/headers since I do not have root access

Not having a cluster management system deployed yet so we can't have multiple instances of software versions installed. Can get around this by having executables called across a nfs mount point, but sometimes that isn't really a feasible option.

Installation of software when I don't have sudo permissions

Most problems involve installation of software and are mostly due to the shortcomings of the documentation.

Pre req packages ... Conflicts with other installed software

Licensing in tools/pipelies/etc

Cluster libraries tend to be old/stable, making installing some new bioinformatics tools extremely hard/impossible due to requiring very recent versions of their dependency chain.

Slow / needs more RAM

Difficulties to install some softwares with too many dependencies and no tutorial.

Data is not ubiquitous. Keeping data persistent across a laptop, a server, a SAN, and a cloud instance is most of the job. Gotta have the data in the right place and format to do the work.

'Incorrect' (for my needs) version installed (or not installed at all)

compilations and lack of sudo

Very few. More problems on the server/cluster, with versioning etc. and oversubscription.

Cluster thread limitation. Cluster maintenance is poor. Takes a while to get programs installed/updated.

having to install arcane packages which i may not be able to do due to lack of permissions.

Too much data is stored on the server. Too many users trying to be on the server at the same time.

None really, but we are lucky to have multiple sys admins and experienced bioinformaticians.

Low availability. Waiting for jobs to even start really impairs progress. And having data on some crazy custom filesystem is never a good start to interfacing with the outside world. Oh, and clusters whose sysadmins refuse to provide SSH connection to the system and instead require the use of VPNs that don't work on Linux. Wherever possible I optimize things to run on my laptop, even if it takes days or even weeks of development.

Limited memory space on nodes, on occasion

Dependency hell, failure to compile

Few problems running software as supported by a bioinformatics support service. However software that updates frequently can be an issue as there is always a delay in updating the remote install.

Long running times for certain software

Lack of dependencies, lack of permissions to install. Managing to work around this with a virtual env.

Installing software is always a big pain due to out-of-date operating system (e.g. old gcc)

Terrible error handling, poor documentation and occasionally program constructs which could be bettered by a GCSE student

Administration / Old OS versions

- Lack of control over the operating system / installed libraries installed on the cluster.
- Competition with other users for limited disk space / CPU time

The need to explore a folder structure that was not set up by me.

- running old OS version not fullfilling version requirements - installation on server without admin rights (no package manager)

Server downtime Incorrectly guessing resources requires

Program Dependency issues

Poor documentation from academic software. Need more vignettes, walkthroughs.

Not necessarily the most up-to-date versions, as administrator for server likes to benchmark new version before installation and use.

issues with university department running the server's priorities.

Compiling (libraries don't exist or are wrong version and I don't have power to change)

The pending times are too long

Updating/dependency compatibility in R

Memory and storage limitations (then I switch to university cluster).

I have to install all dependancies etc and update myself

long waiting time. Manual partitioning of input data. Obsolete or incompatible with bioinf software versions of gcc, libraries etc.

No root access, yet many tools require it

Installation

Google and Stackoverflow. Contact the author. (Response time is directly proportional to the time since the software was released).

Different Ruby or other background installation than the software needed

Incompatibility between softwares and dependencies.

Running out of disk space. Lack of sys admin support. Our sys admin installs open suse on all servers but most software I use works smoothly for Ubuntu so lose lots of time troubleshooting software on open suse.

Ad hoc software design that is often incompatible with my application

Not allowed to compile programs from source on user accounts

hangups in my connection -> changes in script saved locally but not on cluster -> running same errors again

Java is messed up Some resources are becoming slow or have large cumbersome outputs (e.g. sequence similarity searches)

Dependency management, version conflicts, issues with GCC/Clang. People not wriitng tools to be Mac compatible.

Lack of RAM (1TB)

Bioinformaticians with a biology background unable to use the SGE properly and wasting everybody's time and patience

Installation, poor documentation and lack of author response to inquiries.

Local install of packages/versions required for a program don't always work and getting administrative approval for a global solution is slow.

No root access means waiting time for software installs or trying myself without having enough Sysadmin skills

As a part-time bifo person, I think that I struggle with selecting proper defaults that fit my experiments because I don't always have a good understanding of the stats behind them. Not for lack of trying either!

Often I have no idea how exactly the software works, as the documentation, although possibly at least existing, does not nearly provide enough actual practical examples.

Difficulties to install dependencies (e.g. SciPy vX.x) or poorly documented programs

None

Some tools are clearly not designed to be parallelized. This probably due to the lack of good (read easy to use) parallelization APIs in common scripting languages.

Must completely automate installation of dependencies in order to run the software, can't just crash around until it works

Has a non-standard setup with libraries in odd places, so lots of software is tricky to compile.

Server downtime Long waits for jobs to run sometimes

Installation, though this is vcoming easier with Docker

Trying to install things that are not self contained.

Bugs and poor documentation

Connection problems to server (my wifi, campus power outage, someone *turned the server off*, etc).

Long delays in waiting for things to be installed

Undocumented or under-documented errors, which don't suggest how to fix errors. Also, usually there are zero tests, meaning that even if the software seems to install properly, I have no way to trust that it has been installed properly, or if it works.

Lack of packages available for distro

Dependencies for complicated pipelines can be difficult to install on the cluster.

Making permissions work well across members of the research group within the lager cluster that is used by many other people. Also, we don't do our own sysadmin and the sysadmin folks don't understand what we do.

It don't install right

the university server/cluster is not well managed, performance can vary widely depending on traffic

Decommissioning of HPC cluster combined with lack of support from institution for HPC infrastructure and eResearch

lack of server capacity; lack of documentation; lack of systems support; no root access to set up database servers or other infrastructure

Lacking certain software packages or modules, having trouble installing them.

Typically none, will make use of university cluster when memory/processing power required in abundance.

Not being able to quickly install/update software.

RAM and processor limitations

Development lag between my laptop and machine. I have a powerful laptop (16GB) memory which allows me to do as much as possible before running remotely.

Bad documentation

People using all resources without adjusting job priorities!

None, really.

Cluster is solid. Storage is solid. Mick Watson

java memory leaks

none - using the lab server, we manage the software ourselves

Nodes failures, queue system bugs

Hard to test long-running/big-memory jobs

Permissions

Cannot run Docker containers on the university cluster. Versions of software and parameters change quickly. Not transferable directly from one computer to another.

Lack of memory especially with java based programs.

poor programming, lack of parallelisation

can't always install software myself (no sudo) poor documentation for software - both usage and interpretation of output

Installation problems that take to much of my time to solve. Ocasional limited computational resources (many users active at the time). Connection problems outside university network (vpn failings).

It is fine for me

Installing software and its dependancies.

Installing dependencies without root access

Installing

Software not installed on the server (dependencies and linking libraries). Long queuing time. Disk capacity (yeah, on the server as well)

- compatibility - portability

unmet dependencies

running out of memory

Installing ~~poretools~~ software that relies on the latest C libraries, or with huge dependency lists which means that, as I often do not have root privileges and have to work with out-of-date OS, software simply won't install

Obsolete versions of libraries/compilers on clusters Poor or expensive cloud compute capabilities.

No free job slots on the cluster

installing poorly documented software

None really.

The mother fuckers changing to slurm and messing up a whole barrel load of lovingly maintained pipelines

Dependency limitations. Odd one off bugs that have never been documented and are difficult to create test cases to re-create. Memory.

Not enough memory, not enough disk space (although it's a decent Macbook Pro).

There is a departmental server, but access is restricted and help is limited, as there isn't a dedicated bioinformatician.

Incompatibility of software, resource limits - wall time, memory, etc.

If you train students, please say what in your view is the biggest challenge for training students in bioinformatics?

Seems like there are a bazillion little things that someone needs to pick up and so I always feel like I'm correcting people or with holding information.

They often struggle setting up their software environment

They usually start learning bioinformatics once entered in a masters or phd, too late.

My biggest issue is lack of personal knowledge in the area - can't train people without the relevant expertise yourself. I therefore rely upon others with better experience (colleagues/collaborators) and the student's own ability. This approach is fraught with risks and pitfalls, many of which we have had to dig ourselves out of already - however in the absence of simpler/easier toolchains it has been the only way we have got things done.

Finding a common vocabulary and being sure that your understanding of a concept is the same as theirs. I deal mostly with PhD students who (understandably) spend most of their time/focus on their project, and this often makes it difficult to establish a proper understanding of more general bioinformatics tools and analysis.

I am one to be trained :)

Lack of UNIX skills

Being critical of predictions.

Lack of any previous experience in Linux platform or coding.

You have to start from scratch as they have no formal training in anything combined with the fact that there is a lot to learn

Field is now too broad for a general education in "bioinformatics".

GUI oriented education. Finding the right proportion between teaching using existing programs as "black-boxes", vs "see the guts of an algorithm" but do not learn how to use it.

They claim to have trouble finding Heng Li. It's hard to find minions you can trust.

Nowadays it is rare to have any experience of the command line at all, many older bioinformaticians grew up with DOS etc which provides a useful frame of reference.

I'm a student myself, and learning bioinformatics analyses is difficult even with internet and access to many forums as many people have their own way of writing codes, preference for programs used, etc.

Real time data analysis Answers to which tool to use

Aversion to the command line.

Having them think programmatically and learn general tools like linux bash R python.

- little or no programming literacy

Command line not intuitive for some. So many ways to do things, difficult to decide what is best to teach. Broad audience.

Lack of basic computational/statistical literacy.

Mick Watson.

It is hard to get across the extent to which bioinfo. == patient data munging

Teaching them to code in R when they have been trained in Python. As much as people might prefer using Python for bioinformatics, R is essential to know.

Giving them enough confidence to feel comfortable.

Training them to actually understand what each script is doing to the data and why is it important to understand the process and not just the final output.

N/A

They hate it :-)

For students from a biological background, getting them to lose their fear of 'breaking' things on the compute servers/cluster. For students from math/stats/CS background, understanding that biology is messy and there are rarely if ever clean and clear answers.

Breadth of the field as a whole. Very difficult to keep material relevant to a particular student and teach class as a whole.

Getting them over the fear of UNIX and the command line. If they can get that they tend to do much better.

Apprehension of large data and thinking they will not get it.

NA

data is every where, don't only produce data, analyse them!

Training students to problem solve when they run into issues.

Students come from many different backgrounds, so no one training solution is viable for all. Typically results in the students needing a large amount of self-guided learning, with mixed results.

making them write parsers instead of making them understand the data and why it is the way it is

Most students do not have a solid background in math, probability, and statistics.

diversity of backgrounds, persistent "I'm not a mathematician/computer scientist" attitudes, not much institution-wide support for training where I am

Time constraints Differing backgrounds

Students want to jump ahead, and are not interesting in the basics, like How does the data look like, How is the data stored, Where is the data coming from, etc .

Getting them to accept the messiness of the real world, compared to textbook problems that actually have solutions

1) Many students dont understand why they have to leave the sanctuary of GUI based tools such as excel to move to R/UNIX/Python. They find the command line too daunting. 2) Running a class on bioinformatics using real life examples for 10+ students takes too much time and computational resource. Need either access to cloud tools ($$) or to make the example so small it defeats the object.

I don't train students - I'm a graduate student.

Slow learning curve + difficulties in establishing baseline skills in Unix and basic programming + too many tools need to be mastered.

Time

Narrowing down the audience so you're not boring some students while others struggle with "if" statements in simple programming.

Keeping up to date with material

having them learn how to google/ask for help online.

I don't.....lets train more of them!

Lack of unix skills

(I do only informal supervision of PhD students) Steep learning curve to get basic programming and command line familiarity down, which is necessary to use the computing resources available (very few GUI options here)

Getting them comfortable on the command line.

Training good software development practices.

n/a

Helping them to avoid cargo cult bioinformatics (hello GATK) and develop their own solutions to answer problems they want to solve. Bioinformatics has tended to lack the small modular tools that enable this kind of approach, although I have to say this seems to be improving.

Finding the balance between showing them exactly what to do and letting them figure some things out on their own

linux/command-line is your friend, really.

Command-line bullshittery without a doubt. http://www.pgbovine.net/command-line-bullshittery.htm

Statistics is the most important part, that computer science is not the end point.

Where to start! Students can come with so many different backgrounds.

Getting students to care about learning the details. Most just want to get from data to interpreted results without having to grasp advanced statistics or programming.

Convincing them to use version control

Installation of programs. Keeping materials and programs up to date and relevant. CLI is not an issue and is picked up relatively quickly by even complete novices.

Very limited knowledge of informatics and statistics, esp. among students primarily studying biology

The absence of basic computational and mathematic training in early undergraduate biology courses.

That most have absolutely no computing knowledge, thus you have to train that as well as the domain specific aspects.

We should teach 3 things a) principles (inc stats, which most biologists are pathetically ignorant about), b) how to use the command line c) how to use google. Given the pace of change people have to be able to teach themselves from the basics

As I describe it to students: "You are facing an incredibly unfair challenge, you are being asked to learn (at the same time), 3 NEW disciplines: Unix/Linux command line, Software tools for bioinformatics, and scripting/programming languages (Bash/R/perl/python), to do a single analysis."

lack of bioinformatics background in biology students

Have only done some informal teaching of phd students. It's hard to know where to start...

basic computer knowledge and getting programs to work. Usually, they have a good background in statistics and biology, but the programming skills are the most crucial step and the skill they are often lacking.

Teaching them to work in the terminal and that understanding what the program does is essential. Especially with troubleshooting programs, they need to understand the programs to keep moving as error messages often are not very informative.

They need I medics training and its often done peicemeal in short courses or workshops

- Focus in training goals - Highlight work

Making them skeptical about the tools that they use, and getting them to check all inputs and outputs

Reproducibility and controls.

Access to sufficient compute to carry out their studies - PhD budgets are already strained, there isn't enough to pay for compute, so they rely on (free/bad) departmental servers

They often have no training in *nix systems before coming to me - they have no programming/scripting experience from their undergraduate courses.

Statistics, statistics, statistics

Convincing them that the command line is not as scary as they think.

Lack of computing/programming skills, resulting in a very step learning curve for bio grad students. Academic programs lead by faculty that they themselves have little to no bioinformatics research experiences.

Encouraging a computational approach to thinking and managing/handling/analysing data (also sometimes convincing them that bioinformatics is science, and not just doing BLAST searches).

Hard to educate in the principle of : Think your question before you type. For bio/med background: don't be scared to make mistakes or look stupid. For comp / maths background : your question is more important that your method.

1. Teaching bioinformatics to biologists who can't code. 2. Teaching bioinformatics to coders who don't know biology.

Basic computer literacy - how to install software, troubleshooting simple errors.

Microsoft Windows

Senior academics taking on students with absolutely no previous expertise in bioinformatics just because they are keen to "have a go". Usually medics.

They're afraid to break things. They have a powerful general-purpose computer in their pocket, but the app store approach to computing makes them think they can't tell it what to do.

View that bioinformatics is easy and quick to master

command line basics

understand the biology behind running tools alone is not important churn out the needle from a haystack

- Understanding how to move for quickly profiling different potential solutions in a rapid way, and how to transition from that kind of analysis into building larger-scale analyses. - Training students to learn for themselves how to follow new developments in the field.

Depending on the background. Students with Computer Science background usually find it a challenge to understand the nuances in molecular biology Students with Biology background find it a challenge to consider command-line tools.

I have to train each student, one at a time. I also have to remember how to do everything and for some analyses I do them once every few months, so I can lose brain cells between analyses.

Teaching them that just because the tool says something doesn't mean it's biologically correct

I am a PhD student myself.

I don't train students (grad student), but I do believe there should be an emphasis on basic computer literacy in undergraduate education. We all take "intro to biology." Students should all take "intro to computing" and learn basic computer skills.

The difficulty of having both computer science and molecular biology mindsets

One can never learn enough.

Taking the time to demonstrate a balance between best practices and exploring the problem fully.

Getting them to avoid newbie errors (like using inconsistent assemblies or zero- and one-based coordinates together). No matter how many times I underline how important this is they keep doing it! Also the huge amount of time new students have to spend installing software and configuring their environment just to get started.

A complete lack of computing or biology knowledge. Some just want to know where the magic button is to give them a tree.

Training them in proper and fast coding

Getting to think like a scientist and not as a technician.

Figuring out how to balance the biology and computation... they're good at one or there other, but struggle to handle both.

biological thinking

Bioinformatics is more than one discipline: coding and analysis have different training goals A students base level of statistics knowledge is often poor Time between learning and using the knowledge is not always synced hence the trainees will often forget.

Not (yet) training students.

The need for students to learn practical data wrangling, analysis and visualization skills in R, Python, etc that often aren't taught in course work.

Lack of programming experience

Most lack a basic understanding of how a computer works....how ram is utilized, data stuctures, file types, ect

Proper understanding of underlying algorithms to know how to optimize precise and accuracy results quickly.

Basic knowledge on algorithms, coding and testing their own code.

The steep learning curve. It is often too difficult to install and test new software tools due to the number of dependencies etc that are needed with the right version numbers. This results in the use of suboptimal solutions as people use what they have already installed.

difficulties to find good candidates in comparation to some other fields. Seems to me they do not choose academia but company money instead

Resources to run real analysis

Understanding the messiness and noisiness of biological data

Training with sample data set and using tools in real life can be different and always enhancing tools library there is always something (not always the best but) new out there

Sometimes they don't know much stuff

Unix command line.

Not having an algorithmic thinking.

for biology students = algorithmic thinking for computer science students = biological thinking

Fear of command line!

Getting them to believe they can actually do it

Poor preparation for linux use and computational science in general. Conceptually unprepared for "tools of the trade".

Instilling reproducible research habits Providing adequate training in each of the different spheres bioinformatics touches (mol bio, stat, cs, evo, etc) Motivating students through tedious work (parsing, troubleshooting other people's tools, etc)

they're noobs

Educating them the fundamentals of the command line

Getting them to listen to my advice. :-)

too many people want their area of expertise to be covered rather than focusing on teaching a few useful basic skills well not sure how to teach effectively

I would send all my students on courses run by our Bioinformatic support unit - done the course myself and not looked back

Finding people that are good that will accept non-competitive pay.

Lack of fundamental skills such as numeracy and problem solving

Teaching them to always *look* at the data and think critically about their results, rather than presenting me with a p-value or ROC curve.

Finding out that some of the tools and databases I used last year don't exist anymore (again, lack of persistence).

Bioinformatics is just such a wide field - students need to be confident programmers, but also able to teach themselves to use lots of tools and understand the biological background - so lots to learn and teach. Also, with all the methods out there, I feel like not enough of a focus is placed on good research methodologies (and not just use some tool you don't understand on your data) - so maybe that's the biggest challenge?

I don't train anyone but am sometimes asked for advice by newer students. Once someone has done a few unix/python/R tutorials, how do they then apply these new skills to their data?

Making sure they understand what exactly is being done to the dataset at each step when they use a semi-automated pipeline, what the limitations, biases and assumptions are, making sure they choose parameters and tools according to their research questions, instead of just repeating commands from workshop slides. Also difficult to teach them to go beyond the basics (stacked bar plots for 16S rRNA sequencing), use different visualisations, different analyses, more statistics, etc., ask new questions to explore the data, etc.

The lack of complete tutorials that, if they were given some new data, they could follow an analysis pipeline (e.g. SNP analysis) with minimal (<30 minute) hands on time from me.

Developing an understanding of the diversity of resources available and the two-way nature of their use.

A large amount of skills and knowledge need to be learnt to be productive.

None. Our students are invariably brilliant and a pleasure to train.

Two things: Thinking that bio-bioinformatics can be learned and internalized in a 2 day or even semester long course and not realizing that it is a scientific discipline. Raw fear of computing as a tool. (Similar to math adverse students.)

Getting them to sit at the command line long enough to start doing their own troubleshooting

On the side of training Computer Engineers, the challenges are: - Getting a deep intro to genomics. - Getting to know the specific problem to solve (i.e. DNA assembly, metagenomics, etc). - Selecting the right software to use.

demystifying programming in general. good scientific computing practices (documentation, code repos, etc.). Definitely different challenges for people coming from biological vs computational backgrounds.

Beside good project and data management skills, I would say critical thinking is the biggest challenge.

Too many programs leading to confusion as to how to start Hard for them to install large frameworks Poor programming skills starting out

finding students interested in both biology and computer. Most have 1 skill but not the other.

Training on the prerequisite "boring" tools before you can conduct a meaningful biological analysis. These include: installing the needed software, command line interface, git (sometimes), statistical analyses, and including control/test data to differentiate meaningful output from garbage.

I try my best not to train anyone.

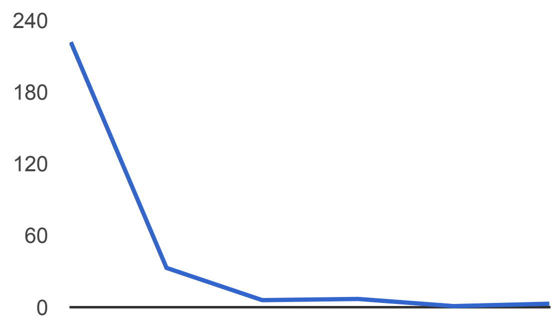
It's very interdisciplinary, so students rarely are strong in all aspects.

A lack of well documented exemplars and illustrations which take into account the time and motivation levels of undergraduates. For post grads, a lack of suitable online resources which teach logical programming and unit testing. Can someone somewhere write an intelligible guide to GIT.

Convincing them that it's worth taking the time to properly learn coding basics. It's not just a chore you can half-learn for a semester and get half-decent results on your projects; either you do it right or you get nothing.

Steep learning curve. Takes long time to dø even basic stuff.

Number of daily responses


240
180
120
60
0