18th Annual Bioinformatics Open Source Conference
BOSC 2017
Prague, Czech Republic
July 22-23, 2017
http://open-bio.org/wiki/BOSC_2017

Image: "Data Plumbing" by Gregg Helt

Welcome to BOSC 2017! The Bioinformatics Open Source Conference has been held annually since 2000 in conjunction with the Intelligent Systems for Molecular Biology (ISMB) Conference. BOSC is organized by the Open Bioinformatics Foundation (OBF), a non-profit group that promotes the practice and philosophy of Open Source software development and Open Science within the biological research community.

Sponsors

We are grateful to our new sponsors: eLife (cutting-edge technology for cutting-edge research), The Hyve (open source solutions for bioinformatics), Mozilla Science Lab (a community of researchers, developers, and librarians making research open and accessible), Repositive Limited (connecting the genomics community with the data they need), and Seven Bridges (the Biomedical Data Analysis Platform), as well as GigaScience (an open access, open data, open peer-review journal) as a returning sponsor.

BOSC 2017 Organizing Committee

Nomi Harris (Chair), Heather Wiencko (Co-Chair)

Brad Chapman, Peter Cock, Chris Fields, Bastian Greshake, Karsten Hokamp, Hilmar Lapp, Monica Munoz-Torres

Program Committee (Those marked with * also reviewed Late-Breaking Lightning Talk abstracts):

Kai Blin*, Christian Brueffer*, Scott Cain*, Jacqueline Campbell, Brad Chapman*, John Chilton, Peter Cock*, Karen Cranston, Anamaria Crisan, Michael Crusoe, Philip Ewels, Christopher Fields*, Konrad Förstner*, George Githinji, Bastian Greshake*, Björn Grüning, Nomi Harris*, Michael Heuer*, Karsten Hokamp*, Hans-Rudolf Hotz*, Shaun Jackman, Hilmar Lapp, Jessica Maia, Scott Markel*, Hervé Ménager, Mónica Muñoz-Torres, Frank Nothaft, Konstantin Okonechnikov, Kieran O'Neill*, Lorena Pantano*, Michael Reich, Surya Saha*, Gianluca Della Vedova*, Heather Wiencko*, Jason Williams

BOSC is a community effort—we thank all who made it possible, including the organizing committee, the program committee, the session chairs, our sponsors, and the ISMB SIG chair, Steven Leard.

If you are interested in reviewing abstracts for BOSC 2018, please email bosc@open-bio.org.

Program

BOSC includes two full days of talks, posters, and Birds of a Feather interest groups (BOFs). Session topics this year include Workflows, Data Science & Visualization, Developer Tools and Libraries for Open Science and Reproducibility, Open Data, and Community Building and Citizen Science. Like last year, there is also a session of late-breaking lightning talks. The longer talks this year are mostly 15 minutes (plus 3 minutes for questions); lightning talks are 5 minutes, with a short time allocated for questions at the end of the session.

This year's keynote speakers will be Madeleine Ball and Nick Loman. Our panel topic this year is "Open Data--Standards, Opportunities and Challenges", with panelists Madeleine Ball, Carole Goble, Nick Loman, and Andrew Su, and moderator Mónica Muñoz-Torres.

Posters

Check https://www.iscb.org/cms_addon/conferences/ismbeccb2017/posterlist.php?cat=B for your poster number. ISMB/ECCB requests the following presentation schedule:

- Session A: Odd Numbered Posters – 6:00 pm - 7:00 pm, Saturday, July 22
- Session A: Even Numbered Posters – 6:00 pm - 7:00 pm, Sunday, July 23

Poster setup/takedown times:

- Setup: Saturday, July 22 between 7:30 am - 10:00 am (Congress Foyer, Level 2)
- Posters should be removed at 7:00 pm, Sunday, July 23.

Pay-your-own-way Dinners

We invite you to join BOSC organizers and attendees at optional pay-your-own-way dinners both nights of BOSC at 7:30pm. Both restaurants are within walking distance of the conference center. You must RSVP to attend, and both dinners usually fill up.

The Saturday dinner will be at Podolská Kotva: http://www.podolskakotva.cz/
There's space for 30 people, and you can RSVP at: http://doodle.com/poll/74kp3m6si5pne3mt

The Sunday dinner will be at Podolka Podolí: http://podoli.restauracepodolka.cz/
There's space for 20 people, and you can RSVP at: http://doodle.com/poll/zw2ti3z6dsgdy2sb

Special thanks to volunteer Petr Danecek for planning both dinners!

If you want to join us for dinner, you must RSVP by Saturday at lunchtime.

Dinner attendees are responsible for paying for their own food and drinks (cash preferred).

OpenBio Codefest

The annual pre-BOSC Codefest is an opportunity for anyone interested in open science, biology and programming to meet, discuss and work collaboratively. Codefest is FREE and open to all (even if you're not attending BOSC). Codefest 2017 will take place July 20-21 at Brmlab, Prague (a non-profit, community-run hackerspace). See https://www.open-bio.org/wiki/Codefest_2017 for details and to sign up.

OBF Membership

Professionals, scientists, students, and others active in open science or open source software or in the life sciences are invited to join BOSC's parent organization, the Open Bioinformatics Foundation (the OBF). The OBF grew out of the volunteer projects BioPerl, BioJava and Biopython and was formally incorporated in 2001 in order to handle modest requirements of hardware ownership, domain name management and funding for conferences and workshops. In 2005, we enacted bylaws for the first time, and along with it created a formal membership.

In 2012, after overwhelming approval in a membership vote, we changed from being independently incorporated to joining Software In The Public Interest, Inc., a fiscal sponsorship organization that aligns well with our own values and culture. We continue to maintain our own membership so that our community has a role in shaping our direction and vision. You can find information on how to join OBF on the OBF wiki at http://www.open-bio.org/wiki/Membership. There is no membership fee.

If you are interested in meeting and talking to some of the OBF Directors and members, please join us at one of the BOSC dinners (see above) or the “Welcome to BOSC” Birds of a Feather (BoF) session on the first day of the conference during lunch.

BOSC 2017 Schedule at a Glance

Day 1 (Saturday, July 22)	
Time	Session
8:30-9:30	(ISMB announcements & ISMB keynote)
10:00-10:10	BOSC announcements
10:10-10:45	Introduction to the Open Bioinformatics Foundation; OBF in the Google Summer of Code; Codefest 2017 Report
10:45-12:20	Session: Workflows
12:20-14:00	Lunch, Posters, Birds of a Feather discussions (BoFs)
14:00-15:00	Session: Developer tools and libraries for open science and reproducibility
15:00-16:00	Session: Data Science & Visualization
16:30-17:30	Keynote (Madeleine Ball)
17:30-18:30	Birds of a Feather discussions (BoFs)
18:00-19:00	Posters (odd-numbered posters present)

Day 2 (Sunday, July 23)	
Time	Session
8:30-9:30	(ISMB announcements & ISMB keynote)
10:00-10:05	BOSC announcements
10:05-11:30	Session: Community Building and Citizen Science
11:30-12:20	Session: Late-Breaking Lightning Talks
12:30-14:00	Lunch, Posters, Birds of a Feather discussions (BoFs)
14:00-15:00	Session: Open Data
15:00-16:00	Panel: Open Data--Standards, Opportunities and Challenges
16:30-17:30	Keynote (Nick Loman) and closing remarks
17:30-18:30	Birds of a Feather discussions (BoFs)
18:00-19:00	Posters (even-numbered posters present)

Complete schedule of talks

(Check https://www.open-bio.org/wiki/BOSC_2017_Schedule for updates)

Day 1 (Saturday, July 22, 2017)

Session	Title	Speaker (or Session Chair)	Start Time	End Time
BOSC opening	Introduction and welcome	Nomi Harris (Chair, BOSC 2017)	10:00	10:10
BOSC opening	The Open Bioinformatics Foundation	Hilmar Lapp	10:10	10:20
BOSC opening	OBF in the Google Summer of Code. Wrapping up 2016 and presenting the 2017 projects	Kai Blin	10:20	10:30
BOSC opening	Codefest 2017 summary	Brad Chapman	10:30	10:45
	Session: Workflows	Chair: Brad Chapman	10:45	12:20
Workflows	Rabix Executor: an open-source executor supporting recomputability and interoperability of workflow descriptions	Janko Simonovic	10:45	10:56
Workflows	Rabix Composer: an open-source integrated development environment for the Common Workflow Language	Ivan Batic	10:56	11:05
Workflows	CWL-svg: an open-source workflow visualization library for the Common Workflow Language	Maja Nedeljkovic	11:05	11:07
Workflows	CWL-ts: an open-source TypeScript library for building developer tools for the Common Workflow Language	Maja Nedeljkovic	11:07	11:11
Workflows	(Q&A for previous 4 talks)		11:11	11:15
Workflows	The GA4GH Tool Registry Service (TRS) and Dockstore - Year One	Denis Yuen	11:15	11:22
Workflows	The GA4GH Task Execution System (TES) and Funnel Server	Brian O'Connor	11:22	11:28

Workflows	The GA4GH Workflow Execution Schema (WES)	Peter Amstutz	11:28	11:34
Workflows	The GA4GH/DREAM Infrastructure Challenges	Brian D. O'Connor	11:34	11:41
Workflows	(Q&A for previous 4 talks)		11:41	11:45
Workflows	Workflows interoperability with Nextflow and Common WL	Kevin Sayers	11:45	11:50
Workflows	CWL Viewer: The Common Workflow Language Viewer	Stian Soiland-Reyes	11:50	11:55
Workflows	Screw: tools for building reproducible single-cell epigenomics workflows	Kieran O'Neill	11:55	12:00
Workflows	(Q&A for previous 3 lightning talks)		12:00	12:03
Workflows	BioThings Explorer: Utilizing JSON-LD for Linking Biological APIs to Facilitate Knowledge Discovery	Jiwen Xin	12:03	12:08
Workflows	Discovery and visualisation of homologous genes and gene families using Galaxy	Anil S. Thanki	12:08	12:13
Workflows	YAMP : Yet Another Metagenomic Pipeline	Alessia Visconti	12:13	12:18
Workflows	(Q&A for previous 3 lightning talks)		12:18	12:20
Birds of a Feather discussions (BoFs)	Feel free to organize a BoF!		12:20	13:40
Lunch			12:30	14:00
	Session: Developer tools and libraries for open science and reproducibility	Chair: Hilmar Lapp	14:00	15:00
Developer tools / reproducibility	MultiQC: Visualising results from common bioinformatics tools	Phil Ewels	14:00	14:18
Developer tools / reproducibility	NGL – a molecular graphics library for the web	Alexander S Rose	14:18	14:23

Developer tools / reproducibility	GRAPHSPACE: Stimulating interdisciplinary collaborations in network biology	Aditya Bharadwaj	14:23	14:28
Developer tools / reproducibility	Efficient detection of well-hopping duplicate reads on Illumina patterned flowcells	Timothy Booth	14:28	14:33
Developer tools / reproducibility	(Q&A for first 3 lightning talks)		14:33	14:36
Developer tools / reproducibility	An ensemble approach for gene set testing analysis with reporting capabilities	Monther Alhamdoosh	14:36	14:41
Developer tools / reproducibility	OpenMS 2.0: a flexible open-source software platform for mass spectrometry data analysis	Timo Sachsenberg	14:41	14:46
Developer tools / reproducibility	Interoperable, collaborative multi-platform variant calling with bcbio	Brad Chapman	14:46	14:51
Developer tools / reproducibility	Gene Set Variation Analysis in cBioPortal	Kees van Bochove	14:51	14:56
Developer tools / reproducibility	(Q&A for last 4 lightning talks)		14:56	15:00
	Session: Data Science & Visualization	Chair: Karsten Hokamp	15:00	16:00
Data Science & Visualization	The backbone of research reproducibility: sustainable and flexible tool deployment	Björn Grüning	15:00	15:18
Data Science & Visualization	Reproducible bioinformatics software with GNU Guix	Pjotr Prins	15:18	15:23
Data Science & Visualization	Reproducible and user-controlled software management in HPC with GNU Guix	Ricardo Wurmus	15:23	15:28
Data Science & Visualization	A Ubiquitous Approach to Reproducible Bioinformatics across Computational Platforms	John Chilton	15:28	15:33
Data Science & Visualization	(Q&A for first 4 lightning talks)		15:33	15:36
Data Science & Visualization	Revitalizing a classic bioinformatics tool using modern technologies: the case of the	Keiichiro Ono	15:36	15:41

	Cytoscape Project			
Data Science & Visualization	The SPOT ontology toolkit : semantics as a service	Olga Vrousgou	15:41	15:46
Data Science & Visualization	Biopython Project Update 2017	Sourav Singh	15:46	15:51
Data Science & Visualization	(Q&A for last 3 lightning talks)		15:51	15:54
Coffee Break			16:00	16:30
Keynote	Open Sourcing Ourselves	Madeleine Ball	16:30	17:30
Birds of a Feather (BoFs)	Feel free to organize one!		17:30	18:30
Posters (odd-numbered posters present)			18:00	19:00
Dinner	Optional pay-your-own-way dinner (you must RSVP in advance--space limited)	Podolská Kotva	19:30	

Day 2 (Sunday, July 23, 2017)

Session	Title	Speaker (or Session Chair)	Start Time	End Time
Day 2	BOSC Announcements	Heather Wiencko (Co-Chair, BOSC 2017)	10:00	10:05
	Session: Community Building and Citizen Science	Chair: Peter Cock	10:05	11:30
Community	BeerDeCoded: exploring the beer metagenome	Jonathan Sobel	10:05	10:23
Community	Supporting curation communities & collecting technical dividends	Monica Munoz-Torres	10:23	10:41
Community	Journal of Open Source Software (JOSS)	Pjotr Prins	10:41	10:59
Community	Building an open, collaborative, online infrastructure for bioinformatics training	Bérénice Batut	10:59	11:17

Community	Software and social strategies for community sourced biological networks and ontologies	Dexter Pratt	11:17	11:22
Community	Distance-based, online bioinformatics training in Africa: the H3ABioNet experience	Nicola Mulder	11:22	11:27
Community	(Q&A for previous 2 lightning talks)		11:27	11:30
	Session: Late-Breaking Lightning Talks	Chair: Nomi Harris	11:30	12:30
LBLTs	Recent object formation in the core of Galaxy	Martin Cech	11:30	11:35
LBLTs	Reproducibility of computational workflows is automated using continuous analysis	Brett Beaulieu-Jones	11:35	11:40
LBLTs	Full-stack genomics pipelining with GATK4 + WDL + Cromwell	Kate Voss	11:40	11:45
LBLTs	ToolDog - generating tool descriptors from the ELIXIR tool registry	Kenzo-Hugo Hillion	11:45	11:50
LBLTs	BioThings SDK: a toolkit for building high-performance data APIs in biology	Chunlei Wu	11:50	11:55
LBLTs	(Q&A for first 5 LBLTs)		11:55	12:00
LBLTs	Integrating cloud storage providers for genomic analyses	Ted Liefeld	12:00	12:05
LBLTs	Fighting Superbugs with Open Source Software	Kai Blin	12:10	12:15
LBLTs	Users, Communication, and a Light Application-Level API: A Request for Comments	Seth Carbon	12:15	12:20
LBLTs	(Q&A for last 4 LBLTs)		12:20	12:25
BoFs	Feel free to organize one!		12:30	13:40
Lunch			12:30	14:00
	Session: Open Data	Chair: Bastian Greshake	14:00	15:00

Open Data	RADAR-CNS - Research Infrastructure for processing wearable data to improve health	Nivethika Mahasivam	14:00	14:18
Open Data	Using Wikidata as an open, community-maintained database of biomedical knowledge	Andrew Su	14:18	14:36
Open Data	Emerging public databases of clinical genetic test results: Implications for large scale deployment of precision medicine	Stephen Lincoln	14:36	14:41
Open Data	Discovering datasets with DATS in DataMed	Philippe Rocca-Serra	14:41	14:46
Open Data	Bioschemas for life science data	Carole Goble	14:46	14:51
Open Data	Introducing the Brassica Information Portal: Towards integrating genotypic and phenotypic Brassica crop data	Annemarie Eckes	14:51	14:56
Open Data	(Q&A for previous 4 lightning talks)		14:56	15:00
Panel	**Panel: Open Data--Standards, Opportunities and Challenges**	***Moderator***: Monica Munoz-Torres ***Panelists***: Madeleine Ball, Carole Goble, Nick Loman, Andrew Su	15:00	16:00
Coffee Break			16:00	16:30
Keynote	Open data meets ubiquitous sequencing: challenges and opportunities	Nick Loman	16:30	17:30
BoFs	Feel free to organize one!		17:30	18:30
Posters (even-numbered posters present)			18:00	19:00
Dinner	Optional pay-your-own-way dinner (must RSVP in advance--space limited)	Podolka Podolí: http://podoli.restauracepodolka.cz/	19:30	

Any last-minute schedule updates will be posted at
http://www.open-bio.org/wiki/BOSC_2017_Schedule

Keynote Speakers

Madeleine Ball

Madeleine Ball is Executive Director of Open Humans Foundation and co-founder of Open Humans, a nonprofit project enabling individuals to engage studies and share data.

Open Humans is an open source online platform and community, created through funding support from the Robert Wood Johnson Foundation and the John S. and James L. Knight Foundation. Open Humans enables members to connect data from a variety of sources, including genome, microbiome, activity tracking, and GPS data – and then invites members to share their data with projects and work with research studies. By using an individual-centered approach, Open Humans enables new research opportunities, including: data sharing by individuals, cohort sharing across studies, anonymous engagement with studies, and citizen-led projects.

Dr. Ball is also supported by a Shuttleworth Foundation Fellowship, supporting her continued vision for new approaches to openness in human health research and data sharing. Previously, she was Director of Research at the Harvard Personal Genome Project.

Dr. Ball's keynote talk topic is "Open Sourcing Ourselves."

Nick Loman

Nick Loman is known as a vocal proponent of open genomic data in healthcare. A Professor of Microbial Genomics and Bioinformatics at the University of Birmingham, Dr. Loman explores the use of cutting-edge genomics and metagenomics approaches to human pathogens. He promotes the use of open data to facilitate the surveillance and treatment of infectious disease.

Dr. Loman helped establish real-time genomic surveillance of Ebola in Guinea and Zika in Brazil (via the ZiBRA project, which states that "Data will be subject to open release as it is generated"). In another recent project, real-time genomic data was used to analyze a small outbreak of Salmonella enteritidis in the UK. Through this sharing of genomic datasets, researchers were able to confirm that the cases were linked to a larger, national-scale outbreak. Dr. Loman is one of the authors of Poretools, and he regularly shares cutting-edge Nanopore data and protocols for using it. In collaboration with Lex Nederbragt, Dr. Loman is developing an open-source repository of sequencing and bioinformatics benchmarking datasets called Seqbench.

Dr. Loman's talk title is "Open data meets ubiquitous sequencing: challenges and opportunities."

Panel: Open Data--Standards, Opportunities and Challenges

Moderator **Mónica Muñoz-Torres** (@monimunozto) is the biocuration lead for Berkeley Bioinformatics Open-Source Projects (BBOP) at Lawrence Berkeley National Laboratory. She leads the Community Curation group within the global initiative to sequence and annotate the genomes of 5,000 arthropods (i5K Initiative) and is the chair of the International Society for Biocuration (ISB).

Madeleine Ball is the Executive Director and co-founder of Open Humans, an organization dedicated to enabling individuals to access their data and share it with research studies. She is also PI of the Open Humans Public Data Sharing study. In recognition of her vision for opening human health data, Madeleine was recently awarded a Shuttleworth Foundation Fellowship.

Carole Goble is a full professor in the School of Computer Science at the University of Manchester, UK, where she leads an eScience group of Research Software Engineers and researchers. She is known for her work on semantic technologies, metadata, ontologies, workflow management systems, and Virtual Research Environments. She leads the development of a bunch of pioneering software including Apache Taverna Workflow Manager, myExperiment workflow sharing platform and the FAIRDOM-SEEK asset management platform for Systems Biology. In 2010 she was elected a Fellow of the Royal Academy of Engineering for her contributions to e-Science. Goble was appointed Commander of the Order of the British Empire (CBE) in 2014 for services to science.

Nick Loman (@pathogenomenick) is known as a vocal proponent of open genomic data in healthcare. A Professor of Microbial Genomics and Bioinformatics at the University of Birmingham, Dr. Loman explores the use of cutting-edge genomics and metagenomics approaches to human pathogens. He promotes the use of open data to facilitate the surveillance and treatment of infectious disease. Dr. Loman helped establish real-time genomic surveillance of Ebola in Guinea and Zika in Brazil (via the ZiBRA project, which states that "Data will be subject to open release as it is generated").

Andrew Su (@andrewsu) is Professor at the Scripps Research Institute in the Department of Integrative, Structural and Computational Biology. His research focuses on building and applying bioinformatics infrastructure for biomedical discovery, with a particular emphasis on leveraging crowdsourcing for genetics and genomics. His projects include the Gene Wiki, BioGPS, MyGene.Info, and Mark2Cure, each of which engages “the crowd” to help organize biomedical knowledge.

Talk and Poster Abstracts

Title	OBF in the Google Summer of Code. Wrapping up 2016 and presenting the 2017 projects
Author	*Kai Blin*
Affiliation	Technical University of Denmark, Novo Nordisk Foundation Center for Biosustainability
Contact	kblin@biosustain.dtu.dk
URL	https://obf.github.io/GSoC
License	CC-BY

Google's Summer of Code program [1] is focused on introducing students to open source software development. Students are paired up with mentors from participating organisations and earn a stipend while spending their summer semester break getting an exposure to real-world software development practices. In the past years, the Open Bioinformatics Foundation has participated in the Google Summer of Code six times.

In 2016, the Open Bioinformatics Foundation has acted as an umbrella organisation for seven projects from the open source bioinformatics community, and seven students successfully finished the program [2]. In 2017, OBF is an umbrella for six open source bioinformatics projects. At the time of writing, the students for 2017 have not been selected yet, but at the time of BOSC, the projects will be well under way.

This talk will present an overview of the projects hosted under the OBF umbrella in last year's round of Google Summer of Code, as well as present the projects in the current round.

Rabix Executor: an open-source executor supporting the recomputability and interoperability of the Common Workflow Language

Janko Simonovic[1], Gaurav Kaushik[1], Sinisa Ivkovic[1], Luka Stojanovic[1], Nebojsa Tijanic[1], Adrian Sharma[1], Brandi Davis-Dusenbery[1]

[1]Seven Bridges, Cambridge, MA, USA. Email: janko.simonovic@sevenbridges.com

Project Website: http://rabix.io
Source Code: https://github.com/rabix/bunny
License: Apache 2.0 license

Biomedical data has become increasingly easy to generate in large quantities and the methods used to analyze it have proliferated rapidly. Reproducible methods are required to learn from large volumes of data and to reliably disseminate findings and methods within the scientific community. Workflow specifications and execution engines provide a framework with which to perform a sequence of analyses and help address issues with reproducibility and portability across environments. One such specification is the Common Workflow Language (CWL), an emerging standard which provides a robust and flexible framework for describing data analysis tools and workflows. CWL requires executors or workflow engines to interpret the specification and turn tools and workflows into computational jobs, as well as provide additional tooling such as error logging, file organization, and job scheduling optimizations to allow for easy computing on large volumes of data. To this end, we have developed the Rabix Executor, an open-source workflow engine that improves reproducibility through reusability and interoperability of workflow descriptions. We define five major components of the Rabix Executor -- frontend, bindings, engine, queue, and backend -- each of which is abstracted from the other to maintain a modular design so that components can be used as needed. Developers are able to design custom frontends (e.g. a custom graphical user interface or command line interface), build bindings for the engine to parse a specific set of workflow languages, employ a specific queuing or scheduling protocol of their choice, and submit computational jobs to different backends (e.g. Amazon Web Services, a high-performance computing cluster, a local machine).

For workflow decomposition, the Rabix Executor employs an abstract software model which was defined to be a superset of all known workflow languages; this enables the use of different workflow languages and versions. To demonstrate this, we are capable of running all drafts and versions of Common Workflow Language and CWL v1.0 workflows composed of tools in previous versions of CWL. To our knowledge, Rabix Executor is the only CWL implementation which maintains full backwards compatibility of tools and workflows, removing the need for refactoring existing bioinformatics applications. In scaling benchmarks, the Rabix Executor is capable of running tens of thousands of concurrent jobs from a batch of whole genome sequencing workflows. The modular and abstracted design of the Rabix Executor is intended to allow for reproducible bioinformatics analysis on various infrastructures and continual support of a growing library of bioinformatics tools and workflows.

Rabix Composer: an open-source integrated development environment for the Common Workflow Language

Ivan Batic[1], Maja Nedeljkovic[1], Boban Pajic[1], Nebojsa Tijanic[1], Luka Stojanovic[1], Marijan Lekic[1], Adrian Sharma[1], Gaurav Kaushik[1], Brandi Davis-Dusenbery[1]

[1]Seven Bridges, Cambridge, MA, USA. Email: ivan.batic@sevenbridges.com

Project Website: http://rabix.io
Source Code: https://github.com/rabix/cottontail-frontend
License: Apache 2.0 license

The Common Workflow Language (CWL) is an emerging standard for describing data analysis workflows which allows for portability between compute environments, improved reproducibility, and the ability to add custom extensions to fit institutions' needs. The robust and flexible framework provided by CWL has led to its adoption by the National Cancer Institute (NCI) Cancer Cloud Pilots, the NCI Genomic Data Commons, and academic and industrial organizations worldwide. Over time, the CWL community has worked hard to improve the CWL syntax to make it easy to read, easy to parse, and comprehensive in the scope of workflow parameters and behaviors it captures. A trade-off of this approach, however, is that complex bioinformatics workflows may consist of dozens of tools and hundreds of parameters which can be time-consuming to describe manually in CWL; a whole genome sequencing workflow may be hundreds of lines of CWL code alone.

To support the CWL community, we've created the Rabix Composer, a stand alone integrated development environment which provides rich visual and text-based editors for CWL. Our vision for the Rabix Composer is to enable rapid workflow composition and testing, provide version control and the ability to add documentation, share tools easily with online platforms and developers, and allow integration with online services such as GitHub. The Rabix Composer was designed by integrating more than 500 pieces of feedback from Seven Bridges users regarding our previous software development kit for CWL.

The Rabix Composer is part of the Rabix project (http://rabix.io), an open-source effort to provide tooling for the bioinformatics developer community. The Rabix project includes the Rabix Executor, a workflow engine that executes CWL descriptions and their associated Docker containers locally on a laptop, HPC, or on multiple cloud environments such as the Seven Bridges Platform. Using the Rabix Executor in combination with the Composer enables developers to create, run and debug bioinformatics applications locally before scaling. Together, these technologies enhance data analysis reproducibility, simplify software sharing/publication, and reduce the friction when designing a new workflow or replicating a scientific finding.

CWL-svg: an open-source workflow visualization library for the Common Workflow Language

Maja Nedeljkovic[1], Boban Pajic[1], Ivan Batic[1], Adrian Sharma[1], Gaurav Kaushik[1], Brandi Davis-Dusenbery[1]

[1]Seven Bridges, Cambridge, MA, USA. Email: maja.nedeljkovic@sevenbridges.com

Project Website: http://rabix.io
Source Code:https://github.com/rabix/cwl-svg
License: Apache 2.0 license

As the Common Workflow Language (CWL) becomes more widely adopted among the bioinformatics community, the volume and complexity of publicly available CWL has increased. The flexibility and portability of CWL encourages developers to tackle difficult pipelines and edge-cases, enabling them to describe intricate processes, which can be executed in multiple environments. However, complex workflows can be challenging for users to interpret. As CWL syntax has matured, the syntactic shortcuts added in recent versions to make the language easier to write can simultaneously make it more difficult to interpret. As a result of these combined aspects of CWL, some workflows, such as BCBio (bcbio-nextgen), can have hundreds of lines of code. Understandably, these workflows can be difficult to understand and debug without external visualization tools. As part of the Rabix Composer, an integrated development environment for CWL, we developed an open-source, workflow visualization library called CWL-svg. The CWL-svg library takes a CWL description of a workflow and creates a scalable vector graphics (SVG) image to provide a visual representation for more intuitive user interactivity. CWL-svg can be used either as a standalone library, which renders SVG files from CWL, or as part of a larger interface. Our goal with CWL-svg was to create a visualization library which would most clearly represent the relevant parts of the CWL description to the user. We incorporated user feedback gleaned from our previous iteration of our CWL Workflow Editor to create a more intuitive and informative user interface. Implementation of the CWL-svg library allows users to select nodes within the workflow (e.g. a tool or file) to highlight immediate connections, rearrange nodes, and use fluid zoom resizing to make workflow details visible at all resolutions. The library also implements an auto-align algorithm which untangles complicated workflows in a visually pleasing arrangement. These design details, combined with meticulous optimizations and attention to efficiency, make CWL-svg ideal for handling complex bioinformatics workflows.

CWL-ts: an open-source TypeScript library for building developer tools for the Common Workflow Language

Maja Nedeljkovic[1], Ivan Batic[1], Luka Stojanovic[1], Adrian Sharma[1], Gaurav Kaushik[1], Brandi Davis-Dusenbery[1]

[1]Seven Bridges, Cambridge, MA, USA. Email: maja.nedeljkovic@sevenbridges.com

Project Website: http://rabix.io
Source Code: https://github.com/rabix/cwl-ts
License: Apache 2.0 license

The Common Workflow Language (CWL) is an emerging standard for workflow description, and its adoption is rapidly growing. However, few tools for working with CWL exist, and those that do are command line based, which can be difficult to approach for new users. In order to grow the CWL community, facilitate on-boarding of new users, and entice institutions to transition to the new standard, CWL requires better developer tools. Yet, building these GUI-rich developer tools takes both time and resources. Fortunately, domain-specific developer tools tend to have common reusable pieces that can help save time during the development process. For example, different user interfaces could be built using the same underlying domain-related logic. With this idea in mind, we developed a data model library called CWL-ts. CWL-ts is a utility library for creating developer tools and user interfaces that work with CWL. CWL-ts has three main functions: abstracting the version of CWL (currently supporting v1.0 and a Seven Bridges flavor of draft-2), providing an API for manipulating the CWL document, and validating the document. The library is written in TypeScript but can be used as a plain JavaScript or Node.js library. For this reason, CWL-ts can be used in a variety of different applications and platforms. Likewise, TypeScript provides the added benefit of types to JavaScript, which is especially useful in the context of CWL. CWL-ts is already being used as the domain backbone of the Rabix Composer, an integrated development environment for the Common Workflow Language. Our goal is to grow the utility of CWL-ts as CWL develops further, bolstering it with test coverage and full conformance to the CWL specification, and offering it to the community as a starting point for creating GUI-rich developer tools.

The GA4GH Tool Registry Service (TRS) and Dockstore - Year One

Denis Yuen[1], Brian O'Connor[2], Andrew Duncan[3] , Vincent Chung[3], Xiang Kun Liu[3], Janice Patricia[3], Han Yuan Cao[3], Gary Luu[3], Vincent Ferretti[3], Lincoln Stein[3]

[1] Ontario Institute for Cancer Research, MaRS Centre, Toronto, Ontario. Email: denis.yuen@oicr.on.ca

[2] UC Santa Cruz Genomics Institute, University of California, Santa Cruz, CA, USA

[3] Ontario Institute for Cancer Research, MaRS Centre, Toronto, Ontario.

Project Website: https://dockstore.org
Source Code: https://github.com/ga4gh/dockstore/
License: Apache License 2.0 https://www.apache.org/licenses/LICENSE-2.0.html

Workflows written for the PCAWG (Pan-Cancer Analysis of Whole Genomes) study created a challenge for the cloud projects team at OICR and our collaborators due to the highly heterogenous nature of our fourteen computing environments (cloud and HPC, commercial and academic, geographically distributed). We met the challenge by distributing our workflows in Docker containers described by a proprietary descriptor. As we wrapped up, we realized that this approach could be of use to others so we adopted CWL (Common Workflow Language) descriptors and split out the Dockstore project as its own open-source website and associated utilities. This project reached a 1.0 milestone in September 2016.

Tools registered in Dockstore are encouraged to include open-source build instructions for the Docker image, pulling in open-source binaries and/or source code into the image, being built on a publicly visible third-party service and accompanied by a programmatic descriptor of the tool including metadata. In practice, this has meant Dockerfiles and Common Workflow Language (CWL) or Workflow Description Language (WDL) files checked into GitHub in public repositories, built on Quay.io, and indexed on Dockstore.org.

We have also donated a Tool Registry Service (TRS) (https://goo.gl/zfhR4q) API to the Global Alliance for Genomics and Health (GA4GH) in the hope that like-minded groups can implement it in order to facilitate the sharing of Dockerised tools in the sciences. Currently, Dockstore indexes workflows for PCAWG, the workflows for the GA4GH-DREAM challenge, the UCSC Genomics Institute, and more while welcoming contributions from all bioinformaticians.

In this talk we also present our lessons learned creating CWL+Docker images, open-source and commercial platforms for running tools and workflows on Dockstore, and interoperability challenges when converting tools between tool registries. We also highlight new Dockstore features such as test parameters, pluggable file provisioning, private tool support, and workflow visualization.

The GA4GH Task Execution System (TES) and Funnel Server

Alexander Buchanan[1], Adam Struck[1], Kyle Ellrott[1]
[1] Oregon Health and Science University

Project Website: https://github.com/ga4gh/task-execution-schemas
Source Code: https://github.com/ohsu-comp-bio/funnel
License: MIT License

The standardization of workflows and tools definitions and containerization of software has provided a way to deploy reproducible computing in a truly portable fashion. The standardization effort is hampered by the large number of infrastructural backends available. These infrastructural backends range from queuing systems, such as SLURM, GridEngine or Condor, to cloud-based solutions backed by Amazon, Microsoft Azure, Google Compute or OpenStack. When building a workflow engine, developers are burdened with writing a different backend for each of the infrastructures they want to support. This means that we now have portable code but not portable deployment systems.

The Global Alliance for Genomics and Health (GA4GH) Data Working Group is an international consortium that seeks to provide common APIs that enable genomic computing. They already have defined APIs for querying genomic read and variant data. Expanding on that effort, the Container and Workflows group is working to design a number of APIs to enable interoperable exchange of reproducible workflows and tools. The Task Execution Schema (TES) is designed to be an API that will allow users and developers a common method for issuing batch jobs to compute environments. In principle, this API is very similar to the AWS Batch API or the Google Genomics Pipeline API, but the Task Execution Schema has been designed to be platform agnostic, so that the user can deploy the same tasks in multiple environments.

TES is designed to be simple and robust. Users create a JSON message that describes the required input files to be staged, the container images, the commands to be run as well as the result files and directories that need to be copied out at the end of the run. The user then submits the task request via HTTP POST. The status of submitted tasks, including logging, can be tracked via the HTTP based API. Thus client libraries are extremely light weight. We have validated that the TES API can be used to backend several existing workflow engines. These include the Seven Bridges CWL engine, Bunny, and the Broad Institute's WDL engine, Cromwell, and Galaxy.

As a specification, any vendor has the opportunity to provide a TES compliant service. The first platform to provide the TES API is called Funnel. Funnel is an open source project from Oregon Health and Science University, built in collaboration with Intel, and is designed to execute tasks across multiple backends, including Condor, OpenStack and the Google Compute Environment. With the TES API users can quickly move between environments and service providers. We expect that the number of supported backends and servers that provide the TES API will grow very quickly.

The GA4GH Workflow Execution Schema (WES)

Peter Amstutz[1], Brian O'Connor[2] , Christopher Ketchum[2], Jeff Gentry[3], Kyle Ellrott[4]

[1] Curoverse, 212 Elm St, Somerville, MA, USA. peter.amstutz@curoverse.com

[2] UC Santa Cruz Genomics Institute, University of California, Santa Cruz, CA, USA

[3] Broad Institute, 415 Main Street Cambridge, MA, USA

[4] Oregon Health and Science University, 3181 S.W. Sam Jackson Park Rd. Portland, OR, USA

Project Website: http://ga4gh.org/#/cwf-team
Source Code: https://github.com/ga4gh/workflow-execution-schemas
License: Apache 2.0

The Workflow Execution Schema (WES) is a lightweight HTTP REST API defining how an application can submit requests to workflow execution systems in a standardized way. It is being developed by the Containers and Workflows task team of the Data Working Group (DWG) of the Global Alliance for Genomics and Health (GA4GH), which is dedicated to defining and promoting technical standards to improve portability and interoperability of data analysis workflows in Life Sciences.

A researcher wishing to perform analysis on a data set collected by another institution often faces both technical and regulatory hurdles that make it difficult to simply obtain a copy of the data to analyze locally. Federated computing has the promise to reduce these barriers to science by making it possible to bring the analysis to the data.

Workflow execution engines (such as Arvados, Toil, Rabix, Cromwell, Consonance, etc) will support this API to facilitate running federated analysis on multiple providers. A client using WES submits a description of the workflow to execute along with a set of input parameters. Workflows may be written using the Common Workflow Language (CWL), Workflow Definition Language (WDL), or other workflow languages supported by the target platform. Input parameters to the workflow execution are described using a JSON object.

The client may then monitor execution of the workflow to get status, errors, and logs. When a workflow completes successfully, output parameters to the workflow execution are also described using a JSON object. References to input and output files are described using URIs.

We present an update on development of WES and outline the power of standardized workflow execution through projects like the GA4GH/Dream Infrastructure Challenge. We believe adoption of WES will enable researchers to more easily perform analysis on restricted data, provide a common target for front-end user interfaces that submit workflows, and support interoperability among workflows executing on different providers or written in different languages through the use of a common API for invocation.

The GA4GH/DREAM Infrastructure Challenges

Brian O'Connor[1*], Peter Amstutz[2], Geraldine Van der Auwera[3], Brad Chapman[4], James Eddy[5], Kyle Ellrott[6], Jeff Gentry[3], Justin Guinney[5], Christopher Ketchum[1], Umberto Ravaioli[7], Jeremiah Savage[8], J. Seth Strattan[9], Joseph Shands[1], Denis Yuen[10]

[1] UC Santa Cruz Genomics Institute, University of California, Santa Cruz, CA, USA

[2] Curoverse, 212 Elm St, Somerville, MA 02144, USA

[3] Broad Institute, 415 Main Street Cambridge, MA 02142, USA

[4] Harvard University, Cambridge, MA 02138, USA

[5] Sage Bionetworks, 1100 Fairview Ave. N. Mailstop M1-C108, Seattle WA 98109, USA

[6] OHSU, 3181 S.W. Sam Jackson Park Rd. Portland, Oregon 97239-3098, USA

[7] University of Illinois at Urbana-Champaign, 1308 West Green Street, Urbana, IL 61801, USA

[8] University Chicago, 5801 S Ellis Ave, Chicago, IL 60637, USA

[9] Stanford University, 450 Serra Mall, Stanford, CA 94305, USA

[10] Ontario Institute for Cancer Research, MaRS Centre, Toronto, Ontario, Canada

* briandoconnor@ucsc.edu

Project Website: http://synapse.org/GA4GHToolExecutionChallenge
Source Code: https://github.com/ga4gh/dockstore/ and various repositories for each workflow/tool used in the challenge
License: Apache License 2.0 https://www.apache.org/licenses/LICENSE-2.0.html

Genomic datasets continue to grow in size and complexity and this hampers the ability to move data to compute environments for analysis. As a result, there is an increasing need to distribute algorithms across multiple clouds and regions and to perform analysis “in place” with compute environments that are either close by, or integrated with, storage solutions. The value, and necessity, of a distributed compute model, where algorithms are containerized, sent to remote clouds, and data transfer is minimized, has inspired the work of several groups to tackle the need for standards in this field. Three highly active and related containerized tool/workflow groups are promoting standards and technologies in this space: the GA4GH Containers and Workflows Task Team, the NCI Containers and Workflows Interest Group, and the NIH Commons Framework Working Group on Workflow Sharing and Docker Registry. The GA4GH group has spent the last year creating standards for tool and workflow sharing (the Tool Registry Service API standard), task execution on clouds (the Task Execution Service API standard), and workflow execution on clouds (the Workflow Execution Service API standard) along with high-quality implementations for each (including Dockstore.org, Funnel, and Toil respectively).

To get concrete on the on the utility of these standards and implementations, the GA4GH group and Sage Bionetworks have organized a series of “infrastructure challenges”. The Phase 1 challenge (http://synapse.org/GA4GHToolExecutionChallenge) showed that a Docker-based tool can be shared through the GA4GH Tool Registry Service (TRS) and executed in a wide variety of systems, producing the same result. 35 groups completed this challenge successfully and these represented a wide range of execution platforms including Open Source systems like Cromwell and Toil to commercial offerings like Curoverse, DNAnexus, and Seven Bridges Genomics. Here we present the next phase of the challenge, Phase 2, which looks to use realistic workflows (from groups such as the Broad, Genomic Data Commons, bcbio, and ENCODE) and to leverage the DREAM Challenge infrastructure for organization and community outreach. Participants that are successful in the challenge will show that complex, real-world genomics workflows can be run successfully in different systems and produce the same results. The challenges described here, and future challenges beyond Phase 2, provide a foundation for future large scale projects, such as ICGCMed, that will depend on reliably and reproducibly analysis running across multiple clouds in a highly decentralized and distributed fashion.

Workflows interoperability with Nextflow and Common WL.

Kevin Sayer[1], Paolo Di Tommaso[1], Evan Floden[1,2], Maria Chatzou[1], Cedric Notredame[1]

[1] Centre for Genomic Regulation (CRG), Dr. Aiguader 88, 08003 Barcelona, Spain.

[2] Universitat Pompeu Fabra (UPF), 08003 Barcelona, Spain.

Email: kevin.sayers@crg.eu.

Project Website: http://www.nextflow.io

Source Code: https://github.com/nextflow-io/nextflow

License: GPLv3

Reproducibility has become one of biology's most pressing issues. This impasse has been fuelled by the combined reliance on increasingly complex data analysis methods and the exponential growth of biological datasets. When considering the installation, deployment and maintenance of bioinformatic pipelines, an even more challenging picture emerges due to the lack of community standards. Moreover, the effect of limited standards on reproducibility is amplified by the very diverse range of computational platforms and configurations on which these applications are expected to be applied (workstations, clusters, HPC, clouds, etc.).

Nextflow[1] is a pipeline orchestration tool that has been designed to address exactly these issues. It provides a domain specific language (DSL) which streamlines the writing of complex distributed computational pipelines in a portable and replicable manner. It allows the seamless parallelization and deployment of any existing application with minimal development and maintenance overhead, irrespective of the original programming language. The built-in support for software containers guarantees numerical stability and enables truly replicable computational workflows across multiple execution platforms.

This talk will introduce the Nextflow support the Common Workflow Language[2]. CWL is a community driven specification for describing analysis workflows and tools in portable manner and that is being used by a number of institutions. The presentation will discuss how Nextflow can be used as the execution engine for workflows defined by using the CWL specification, the benefits and disadvantages of this approach, the current limitations and open challenges of this project.

As a proof of concept, we will show how community based CWL pipelines can be deployed by using the Nextflow execution runtime.

References

1. Nextflow enables reproducible computational workflows (Nature Biotech, April 2017, doi:10.1038/nbt.3820)
2. Common WL, https://dx.doi.org/10.6084/m9.figshare.3115156.v2

CWL Viewer: The Common Workflow Language Viewer

Mark Robinson[1], Stian-Soiland-Reyes[1], Michael Crusoe[2], Carole Goble[1]

[1] The University of Manchester, UK. Email: soiland-reyes@manchester.ac.uk

[2] Common Workflow Language project.

Project Website: https://view.commonwl.org/
Source Code: https://github.com/common-workflow-language/cwlviewer

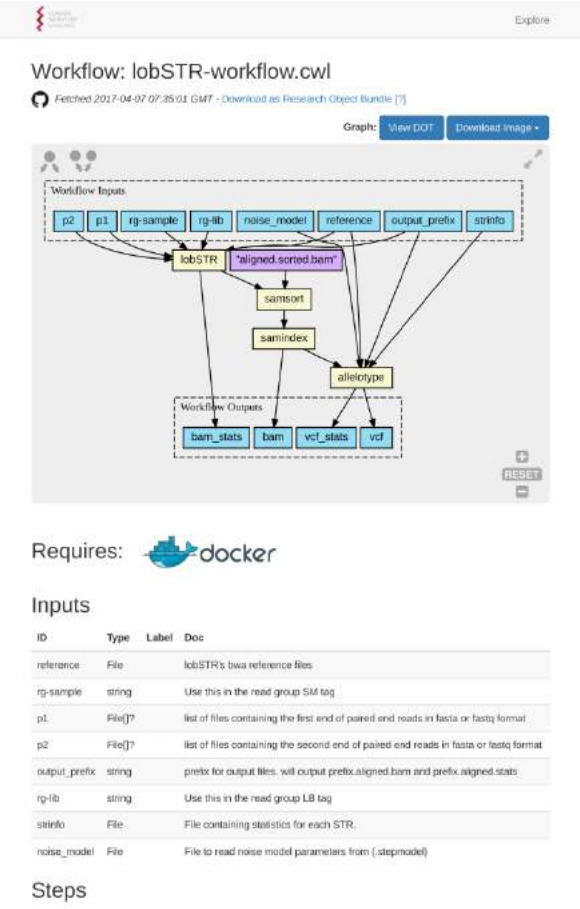
Abstract

The Common Workflow Language (CWL) project emerged from the BOSC 2014 Codefest as a grassroots, multi-vendor working group to tackle the portability of data analysis workflows. It's specification for describing workflows and command line tools aims to make them portable and scalable across a variety of computing platforms.

At its heart CWL is a set of structured text files (YAML) with various extensibility points to the format. However, the CWL syntax and multi-file collections are not conducive to workflow browsing, exchange and understanding: for this we need a visualization suite.

CWL Viewer is a richly featured CWL visualization suite that graphically presents and lists the details of CWL workflows with their inputs, outputs and steps. It also packages the CWL files into a downloadable Research Object Bundle including attribution, versioning and dependency metadata in the manifest, allowing it to be easily shared. The tool operates over any workflow held in a GitHub repository. Other features include: path visualization from parents and children nodes; nested workflows support; workflow graph download in a range of image formats; a gallery of previously submitted workflows; and support for private git repositories and public GitHub including live updates over versioned workflows.

The CWL Viewer is the de facto CWL visualization suite and has been enthusiastically received by the CWL community.



Workflow: lobSTR-workflow.cwl

Fetched 2017-04-07 07:35:01 GMT - Download as Research Object Bundle [?]

Requires:

Inputs

ID	Type	Label	Doc
reference	File		lobSTR's bwa reference files
rg-sample	string		Use this in the read group SM tag
p1	File[]?		list of files containing the first end of paired end reads in fasta or fastq format
p2	File[]?		list of files containing the second end of paired end reads in fasta or fastq format
output_prefix	string		prefix for output files. will output prefix.aligned.bam and prefix.aligned.stats
rg-lib	string		Use this in the read group LB tag
strinfo	File		File containing statistics for each STR.
noise_model	File		File to read noise model parameters from (.stepmodel)

Steps

Screw: tools for building reproducible single-cell epigenomics workflows

Kieran O'Neill[1,2], B Decato[3], A Goncearenco[4], A Khandekar[4], B Busby[4], and A Karsan[1,2]

[1] Pathology Department, University of British Columbia, Vancouver, Canada
[2] Michael Smith Genome Sciences Centre, BC Cancer Agency, Vancouver, Canada
[3] Molecular & Computational Biology Department, University of Southern California, Los Angeles, California, USA
[4] National Center for Biotechnology Information, National Institutes of Health, Bethesda, Maryland, USA

DNA methylation is a heritable epigenetic mark that shows a strong correlation with transcriptional activity. The gold standard for detecting DNA methylation is whole genome bisulfite sequencing (WGBS). Recently, WGBS has been performed successfully on single cells (SC-WGBS) [1]. The resulting data represents a fundamental shift in the capacity to measure and interpret DNA methylation, especially in rare cell types and contexts where subtle cell-to-cell heterogeneity is crucial, such as in stem cells or cancer. However, SC-WGBS comes with unique technical challenges which require new analysis techniques to address. Furthermore, although some software tools have been published, and several existing studies have tended to use similar methods, no standardized pipeline for the analysis of SC-WGBS yet exists.

Simultaneously, there has been a drive within bioinformatics towards improved reproducibility. Textual descriptions of bioinformatic analyses are deeply inadequate, and often require "forensic bioinformatics" to reproduce [2]. Recreating the exact results of a study requires not only the exact code, but also the exact software (down to version, compilation options, etc). Common Workflow Language (CWL) provides a framework for specifying complete workflows, while Docker allows for bundling of the exact software and auxiliary data used in an analysis within a container that can be executed anywhere. Together, these have the potential, via repositories such as Dockstore [3], to enable completely reproducible bioinformatics research.

Here we present Screw (Single Cell Reproducible Epigenomics Workfow). Screw is a collection of standard tools and workflows for analysing SC-WGBS data, implemented in CWL, with an accompanying Docker image. Screw provides the parts for constructing fully-reproducible SC-WGBS analyses. Tools provided include quality control visualization, clustering and visualisation of cells by pairwise dissimilarity measures, construction of recapitulated-bulk methylomes from single cells of the same lineage, generation of bigWig methylation tracks for downstream visualization, and wrappers around published tools such as DeepCpG [4] and LOLA [5]. Screw has the added benefit that CWL's compatibility with interactive GUI-based workflow tools such as Galaxy can lower the barriers to use for less-technical wet lab biologist users.

CWL sources for Screw are available under the MIT license at https://github.com/Epigenomics-Screw/Screw. Tools and workflows are available from Dockstore under Epigenomics-Screw namespace, for example https://dockstore.org/workflows/Epigenomics-Screw/Screw/screw-preprocess

1. Schwartzman, Tanay (2015) Nature Reviews Genetics 16:716–26.

2. Gentleman (2005) Statistical applications in genetics and molecular biology. doi: 10.2202/1544-6115.1034

3. O'Connor et al. (2017) F1000Research 6:52.

4. Angermueller, Lee, Reik, Stegle (2016) bioRxiv 055715.

5. Sheffield, Bock (2015) Bioinformatics 32:587–589.

BioThings Explorer: Utilizing JSON-LD for Linking Biological APIs to Facilitate Knowledge Discovery

Jiwen Xin[1], Cyrus Afrasiabi[1], Sebastien Lelong[1] , Ginger Tsueng[1], Chunlei Wu[1]
[1]The Scripps Research Institute, 10550 N Torrey Pines Rd, La Jolla, CA 92037 *kevinxin@scripps.edu*, *cwu@scripss.edu*

Project Website: http://biothings.io/explorer
Source Code: https://github.com/biothings/biothings_explorer_web
License: Apache License 2.0

RESTful APIs have been widely used to distribute biological data. And many popular biological APIs, such as MyGene.info, MyVariant.info, Drugbank, Reactome, Wikipathways and Ensembl, adopt JSON as their primary data format. These disparate resources feature diverse types of biological entities, e.g. variants, genes, proteins, pathways, drugs, symptoms, and diseases. The integration of these API resources would greatly facilitate scientific domains such as translational medicine, where multiple types of biological entities are involved, and often from different resources.

To fulfill the task of integrating API resources, we have designed a workflow pattern using a semantic web technologies. This workflow pattern uses JSON-LD, a W3C standard for representing Linked Data. In our proposal, each API specifies a JSON-LD context file, which provides Universal Resource Identifier (URI) mapping for each input/output types. Besides, an API registry system is created, where API metadata info, such as query syntax, input/output types is collected, allowing API calls to be generated automatically. By utilizing this workflow, we are able to link different API resources through the input/output types which they share in common. For example, MyGene.info adopts Entrez Gene ID as its input type, which is also one of the output types for MyVariant.info. Thus, data in these two APIs could be linked together through Entrez Gene ID.

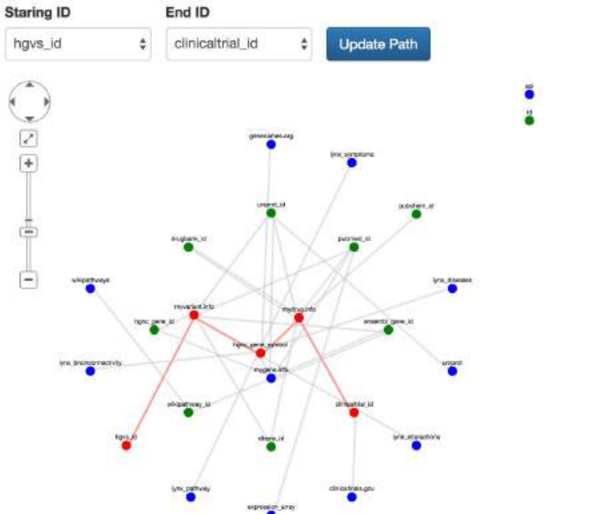


Fig. 1. List all biological terms and APIs currently available at BioThings Explorer. The red line indicates the connecting path between two selected biological terms.

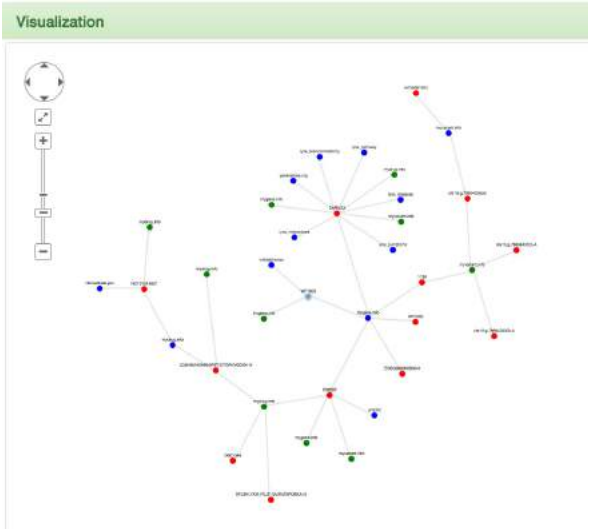


Fig. 2. Demonstration of utilizing BioThings Explorer web interface to crawl for the potential connections between biological entities in multiple domains, e.g. genes, proteins, drugs, variants, diseases through various API resources.

Following this workflow, we have developed a Python package as well as a web visualization interface named 'BioThings Explorer' using Cytoscape.js (Fig. 1, Fig. 2). These tools empower users to explore the relationship between different biological entities through the vast amount of biological data provided by various API resources in a visually organized manner. For example, users could easily explore all biological pathways in which a rare Mendelian disease candidate gene is involved, and then find all genes as well as chemical compounds which could regulate these biological pathways (IPython Notebook Demo: https://goo.gl/sx34T2), thus providing potential treatment options.

Discovery and visualisation of homologous genes and gene families using Galaxy

Anil S. Thanki, Nicola Soranzo, Wilfried Haerty, Robert P. Davey
Earlham Institute, Norwich Research Park, Norwich NR4 7UH, UK

Anil Thanki: Anil.Thanki@earlham.ac.uk

Source Code: https://github.com/TGAC/earlham-galaxytools

License: MIT License

The phylogenetic information inferred from the study of homologous genes helps us to understand the evolution of gene families and plays a vital role in finding ancestral gene duplication events as well as identifying regions that are under positive selection within species.

The Ensembl GeneTrees pipeline generates gene trees based on coding sequences and provides details about exon conservation, and is used in the Ensembl Compara project to discover homologous gene families. Since expertise is required to configure and run the pipeline via the command-line, we created GeneSeqToFamily, an open-source Galaxy workflow based on Ensembl GeneTrees. GeneSeqToFamily helps users to run potentially large-scale gene family analyses without requiring the command-line while still allowing tool parameters, configurations, and the tools themselves to be modified.

At present, we are using this workflow on a set of–vertebrate genomes (human, dog, chicken, kangaroo, macropod, opossum, mouse, platypus, and tasmanian devil), with some analyses comprising more than 13000 gene families. Gene families discovered with GeneSeqToFamily can be visualised using the Aequatus.js interactive tool, integrated within Galaxy as a visualisation plugin.

Handling this large number of input datasets became problematic for both Galaxy itself and certain tools such as T-Coffee which adversely affected memory allocation and file IO processes. We have made modifications to both the T-Coffee wrapper scripts and low-level changes to the Galaxy framework as a whole to help address these issues.

We are also working on integrating protein domain information from SMART (a Simple Modular Architecture Research Tool) to complement discovered gene families, as well as the incorporation of PantherDB into the workflow for validation of families.

Title	YAMP : Yet Another Metagenomic Pipeline
Author	*Alessia Visconti*, Tiphaine C Martin, Mario Falchi
Affiliation	Department of Twins Research & Genetic Epidemiology, King's College London
Contact	alessia.visconti@kcl.ac.uk
URL	https://github.com/alesssia/YAMP
License	GNU GPL 3

Thanks to the increased cost-effectiveness of high-throughput technologies, the number of studies focusing on microorganisms (bacteria, archaea, microbial eukaryotes, fungi, and viruses) and on their connections with human health and diseases has surged, and, consequently, a plethora of approaches and software has been made available for their study, making it difficult to select the best methods and tools.

Here we present *Yet Another Metagenomic Pipeline* (YAMP) that, starting from the raw sequencing data and having a strong focus on quality control (QC), allows, within hours, the data processing up to the functional annotation. Specifically, the QC (performed by means of several tools from the BBmap suite [1]), allows de-duplication, trimming, and decontamination of metagenomics (and metatranscriptomics) sequences, and each of these steps is accompanied by the visualisation of the data quality. The QC is followed by multiple steps aiming at characterising the taxonomic and functional diversity of the microbial community. Namely, taxonomic binning and profiling is performed by means of MetaPhlAn2 [2], which uses clade-specific markers to both detect the organisms present in a microbiome sample and to estimate their relative abundance. The functional capabilities of the microbiome community are currently assessed by the HUMAnN2 pipeline [3] which first stratifies the community in known and unclassified organisms using the MetaPhlAn2 results and the ChocoPhlAn pan-genome database, and then combines these results with those obtained through an organism-agnostic search on the UniRef proteomic database. The next YAMP release, currently under development, will also support MOCAT2 [4] and an optimised version of the HUMAnN2 pipeline. QIIME [5] is used to evaluate multiple diversity measures.

YAMP is constructed on Nextflow [6], a framework based on the dataflow programming model, which allows writing workflows that are highly parallel, easily portable (including on distributed systems), and very flexible and customisable, characteristics which have been inherited by YAMP. Users can decide the flow of their analyses, for instance limiting them to the QC or using already QC-ed sequences. New modules can be added easily and the existing ones can be customised – even though we have already provided default parameters deriving from our own experience. While YAMP is developed to be routinely used in clinical research, the expert bioinformaticians will appreciate its flexibility and modularisation. YAMP is accompanied by a Docker container [7], that saves the users from the hassle of installing the required software, increasing, at the same time, the reproducibility of the YAMP results.

References

1. https://sourceforge.net/projects/bbmap
2. Truong, D.T., et al. *Metaphlan2 for enhanced metagenomic taxonomic profiling.* Nature methods 12(10), 902–903 (2015)
3. https://bitbucket.org/biobakery/humann2
4. Kultima, J.R., et al. *MOCAT2: A metagenomic assembly, annotation and profiling framework.* Bioinformatics 32(16), 2520–2523 (2016).
5. Caporaso, J.G., et al. *QIIME allows analysis of high-throughput community sequencing data.* Nature Methods 7(5), 335–336 (2010)
6. Di Tommaso, P., et al. *Nextflow enables reproducible computational workflows.* Nature Biotechnology 35, 316–319 (2017)
7. https://www.docker.com

MultiQC: Visualising results from common bioinformatics tools

Philip Ewels[1], Max Käller[2]

[1] Department of Biochemistry and Biophysics, Science for Life Laboratory, Stockholm University, Stockholm 106 91, Sweden. Email: phil.ewels@scilifelab.se

[2] Science for Life Laboratory, School of Biotechnology, Division of Gene Technology, Royal Institute of Technology, Stockholm, Sweden.

Project Website: http://multiqc.info/
Source Code: https://github.com/ewels/MultiQC
License: GNU GPLv3

A typical bioinformatics analysis pipeline run can generate hundreds , if not thousands of files. To properly assess the results of the pipeline log files from each tool for each sample should be checked. This can be an impossible task, leading to cherry picking of logs and problems progressing through to later stages of analysis.

MultiQC is an open-source Python package that parses files from nearly 40 different bioinformatics tools. It creates a stand-alone HTML report visualising key metrics, allowing fast and accurate evaluation of analysis results. It is now in its second year, with an increasingly large user community.

At its core, MultiQC works by searching supplied directories for log files that it recognises. It parses these and produces a report with interactive plots and tables describing key metrics. Parsed data is saved to machine readable files for simple downstream processing. Much of the behaviour of MultiQC can be customised through configuration files, plus the code is structured in such a way that it is easy to write plugins to extend the core functionality.

Here, I will describe how to get the best out of MultiQC, including how to customise reports and overcome common problems. I will describe how plugins can be used to pull sample metadata from LIMS software and how results can be pushed to a database for long term trend visualisation.

NGL – a molecular graphics library for the web

Alexander S. Rose[1,2], Stephen K. Burley[1,2,3]

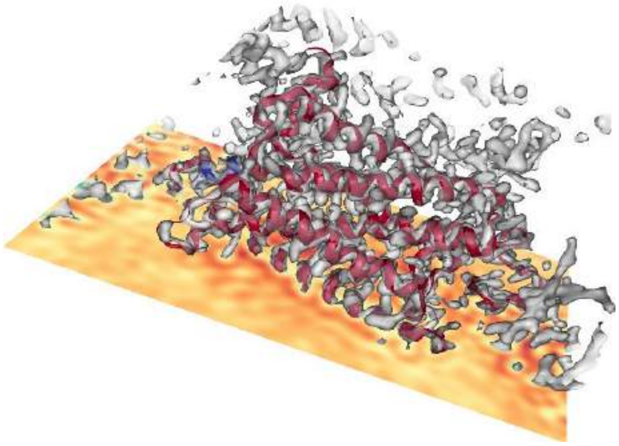
[1]RCSB Protein Data Bank [2]San Diego Supercomputer Center, UC San Diego [3]Rutgers, The State University of New Jersey

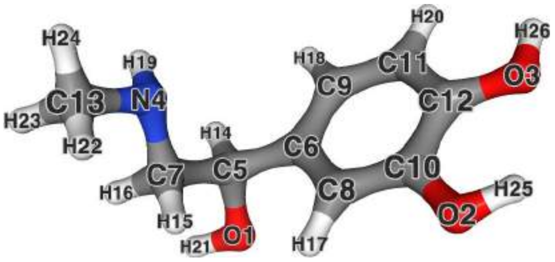
Source Code & Project Website: https://github.com/arose/ngl/
License: MIT license

Interactive visualization and annotation of large macromolecular complexes on the web is becoming a challenging problem as experimental techniques advance at an unprecedented rate. Integrative/Hybrid approaches are increasing being used to determine 3D structures of biological macromolecules by combining information from X-ray crystallography, NMR spectroscopy, and cryo-electron microscopy with data from diverse complementary experimental and computational methods. The wealth, size and complexity of available and future structures make scalable visualization and annotation solutions more important than ever. The web can provide easy access to the resulting visualizations for all interested parties, including colleagues, collaborators, reviewers and educators. Herein, we utilize the web-based NGL library to provide 3D visualization of experimental and computational data integrated with general molecular graphics.

The NGL library has a versatile API to control every aspect from data loading, processing and rendering. A distinguishing feature of NGL is its scalability to system with a million atoms and more. Further, the library supports many file formats for small molecules, macromolecular structures, molecular dynamics trajectories, maps for crystallographic, microscopy and general purpose volumetric data. Annotations can be loaded from text, json, msgpack or xml files. A wide array of customizable representations is available. Molecular data can be displayed as balls, sticks, cartoons, surfaces and labels or with specialized representations such as hyperballs and ropes. Volumetric data can be rendered as isosurfaces, point clouds or volume slices. Additional file parsers and data representations can be added through a plugin system.

The NGL library allows developers to create custom visualization solutions for specialized or novel 3D data derived from bioinformatics calculations and biophysical/biochemical experiments. The resulting interactive visualizations enable spatial understanding and exploratory analyses. Furthermore, these web-based tools simplify data exchange and foster collaborative analysis.





GRAPHSPACE: Stimulating interdisciplinary collaborations in network biology

Aditya Bharadwaj[1], Divit P. Singh[1], Anna Ritz[2], Allison N. Tegge[3], Christopher L. Poirel[4], Pavel Kraikivski[5], Neil Adames[6], Kurt Luther[1], Shiv D. Kale[7], Jean Peccoud[6], John J. Tyson[5], and T. M. Murali[1]

[1]Dept. of Computer Science, Virginia Tech, email: adb@vt.edu, [2]Biology Dept., Reed College, [3]Dept. of Statistics, Virginia Tech, [4]RedOwl Analytics, [5]Dept. of Biological Sciences, Virginia Tech, [6]Dept. of Chemical and Biological Engineering, Colorado State University, [7]Biocomplexity Institute, Virginia Tech

Project Website: `http://graphspace.org`
Source Code: `https://github.com/Murali-group/GraphSpace`
License: GNU General Public License v3

Computational analysis of molecular interaction networks has become pervasive in systems biology. Despite the existence of several software systems and interfaces to analyze and view networks, interdisciplinary research teams in network biology face several challenges in sharing, exploring, and interpreting computed networks in their collaborations.

GRAPHSPACE is a web-based system that provides a rich set of user-friendly features designed to stimulate and enhance network-based collaboration:

- Users can upload richly-annotated networks, irrespective of the algorithms or software used to generate them. GRAPHSPACE networks follow the JSON format supported by Cytoscape.js [1]. Users of Cytoscape [3] can export their networks and upload them directly into GRAPHSPACE.
- Users can create private groups, invite other users to join groups, and share networks with groups.
- A user may search for networks with a specific property or that contain a specific node or collection of nodes.
- A powerful layout editor allows users to efficiently modify node positions, edit node and edge styles, save new layouts, and share them with other users.
- Researchers may make networks public and provide a persistent URL in a publication, enabling other researchers to explore these networks.
- A comprehensive RESTFul API streamlines programmatic access to GRAPHSPACE features.
- A Python module called `graphspace_python` allows a user to rapidly construct a graph, set visual styles of nodes and edges, and then upload the graph, all within tens of lines of code. It is very easy to integrate this script into a user's software pipeline.

Currently, GraphSpace supports more than 100 users who have stored more than 21,000 graphs (most of them private) containing a total of over 1.4 million nodes and 3.8 million edges. Conceptually, GRAPHSPACE serves as a bridge between visualization and analysis of individual networks supported by systems such as Cytoscape [3] and the network indexing capabilities of NDex [2]. We anticipate that GRAPHSPACE will find wide use in network biology projects and will assist in accelerating all aspects of collaborations among computational biologists and experimentalists, including preliminary investigations, manuscript development, and dissemination of research.

References

[1] M. Franz, C. T. Lopes, G. Huck, Y. Dong, O. Sumer, and G. D. Bader. Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics*, 32:309–311, Sep 2015.

[2] D. Pratt, J. Chen, D. Welker, R. Rivas, R. Pillich, V. Rynkov, K. Ono, C. Miello, L. Hicks, S. Szalma, A. Stojmirovic, R. Dobrin, M. Braxenthaler, J. Kuentzer, B. Demchak, and T. Ideker. NDEx, the Network Data Exchange. *Cell Syst*, 1(4):302–305, Oct 2015.

[3] M. E. Smoot, K. Ono, J. Ruscheinski, P. L. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432, Feb 2011.

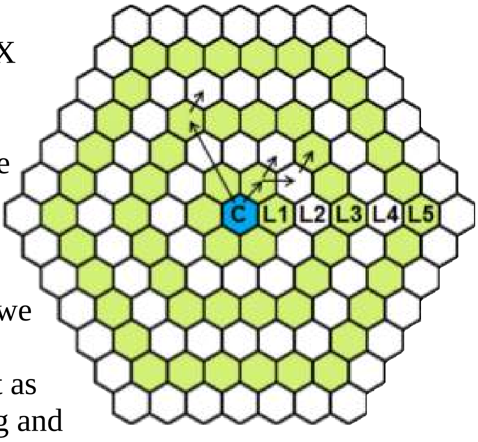
Efficient detection of well-hopping duplicate reads on Illumina patterned flowcells

Timothy Booth[1], Judith Risse[2], Antony Miles[1], Richard Talbot[1], Timothee Cezard[1],
Donald Dunbar[1], Karim Gharbi[1]

[1] Edinburgh Genomics, University of Edinburgh, Edinburgh, United Kingdom. E-mail: tim.booth@ed.ac.uk
[2] Bioinformatics, Wageningen University and Research, Wageningen, The Netherlands. E-mail: judith.risse@wur.nl

Source Code: https://github.com/EdinburghGenomics/well_duplicates

Introduction



Duplication of fragments on the Illumina HiSeq4000 and HiSeqX sequencing machines can occur when DNA leaks into adjacent empty wells on the patterned flow cell tile (the figure is a schematic representation of a small part of a flow cell tile and the arrows represent the spread of well-hopping fragments). This results in a pattern of localised repeats similar to the related issue of optical duplicate reads seen in earlier MiSeq and HiSeq flowcell types. Different names have been proposed for this but we refer to these duplicates as “well duplicates”[1]. Detecting and minimising technical duplicates in DNA sequencing is important as these constitute wasted sequencing capacity due to under-loading and may also lead to erroneous results where the number of reads needs to be quantified, such as in RNA-seq. Well duplicates until now could only be detected after mapping to a reference genome.

Materials & Methods

Our Python duplicate scanner incorporates both the detection algorithm and a new library to efficiently read the binary BCL files produced off the sequencer. The approach is to sample a representative number of centre wells from each tile (marked C in the diagram) and to scan the vicinity of each for duplicate sequences, allowing for a Levenshtein edit distance of 2 by default. To validate the results we analysed a phiX dataset with different loading concentrations on each lane and compared our results to those of the de facto standard tool, Picard MarkDuplicates [2]. Our raw results are fundamentally different in the sense that Picard ‘knows’ all duplicates and can then report the optical fraction, while our tool reports the duplicate fraction over all sampled wells. Therefore we apply a scaling factor between 0.5 and 1 based on estimated cluster size to obtain a value directly comparable with Picard.

Discussion

Our tool is suitable for early-stage QC of new Illumina runs since it does not depend on demultiplexing or mapping of the reads to a reference, runs in minutes, and can be run as soon as the required number of cycles are complete. Our approach demonstrably compares well with Picard without requiring a mapping step, and can produce Picard-equivalent metrics after scaling. The raw values from our scanner are themselves informative and consistent and are being used successfully by Edinburgh Genomics to monitor sequencing run quality.

References

1. https://github.com/samtools/hts-specs/issues/121
2. https://broadinstitute.github.io/picard

An ensemble approach for gene set testing analysis with reporting capabilities

Monther Alhamdoosh[1], Milica Ng[1], Matthew E. Ritchie[2,3]

[1]CSL Limited, Bio21 Institute, 30 Flemington Road, Parkville, Victoria 3010, Australia. Email: monther.alhamdoosh@csl.com.au

[2]Molecular Medicine Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3052, Australia.

[3]School of Mathematics and Statistics, The University of Melbourne, Parkville, Victoria 3010, Australia.

Project Website http://bioconductor.org/packages/EGSEA/
Source Code: https://github.com/Bioconductor-mirror/EGSEA/
License: GPL-2

Gene set enrichment (GSE) analysis allows researchers to efficiently extract biological insight from long lists of differentially expressed genes by interrogating them at a systems level. In recent years, there has been a proliferation of GSE analysis methods and hence it has become increasingly difficult for researchers to select an optimal GSE tool based on their particular data set. Moreover, the majority of GSE analysis methods do not allow researchers to simultaneously compare gene set level results between multiple experimental conditions. The ensemble of genes set enrichment analyses (EGSEA) is a method developed for RNA-sequencing data that combines results from twelve algorithms and calculates collective gene set scores to improve the biological relevance of the highest ranked gene sets. EGSEA's gene set database contains around 25,000 gene sets from sixteen collections. It has multiple visualization capabilities that allow researchers to view gene sets at various levels of granularity. EGSEA has been tested on simulated data and on a number of human and mouse data sets and, based on biologists' feedback, consistently outperforms the individual tools that have been combined. Our evaluation demonstrates the superiority of the ensemble approach for GSE analysis, and its utility to effectively and efficiently extrapolate biological functions and potential involvement in disease processes from lists of differentially regulated genes.

References

Alhamdoosh, M., Ng, M., Wilson, N.J., Sheridan, J.M., Huynh, H., Wilson, M.J., & Ritchie, M.E. (2017). Combining multiple tools outperforms individual methods in gene set enrichment analyses. *Bioinformatics*, 33 (3).

OpenMS 2.0: a flexible open-source software platform for mass spectrometry data analysis

Timo Sachsenberg[1,2,*], Julianus Pfeuffer[3], Oliver Kohlbacher[1,2,4], Reinert Knut[3] and the OpenMS developers

[1] Applied Bioinformatics, Dept for Computer Science, University of Tuebingen, Sand 14, 72076 Tuebingen, Germany.

[3] Center for Bioinformatics, University of Tuebingen, Sand 14, 72076 Tuebingen, Germany.

[3] Algorithmic Bioinformatics, Institute for Bioinformatics, FU Berlin, Takustrasse 9, 14195 Berlin, Germany.

[4] Biomolecular Interactions, Max Planck Institute for Developmental Biology, Spemannstr 35, 72076 Tuebingen, Germany.

[*] E-mail: sachsenb@informatik.uni-tuebingen.de

Project Website: http://www.openms.org
Source Code: https://github.com/OpenMS
License: BSD 3-Clause

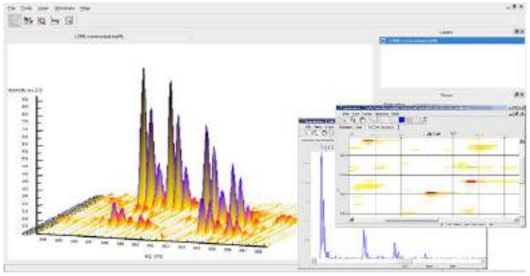


Figure 1: Inspecting raw mass spectra with the OpenMS visualization tool TOPPView.

Abstract

High-throughput mass spectrometry has become a versatile technique to tackle a large range of questions in the life sciences. Being able to quantify diverse classes of biomolecules opens the way for improved disease diagnostics, elucidation of molecular structure, and functional and phenotypic studies. OpenMS is an open-source software package spanning the whole range from algorithms, to libraries, to scripting, to tools, and powerful workflows to transform mass spectrometric data to biological knowledge. OpenMS provides a framework consisting of core data structures, key algorithms, and I/O functions for C++ developers that intend to implement novel algorithms. These data structures and algorithms are also exposed as Python modules for rapid software prototyping. They also form the basis for implementing a more than 185 distinct tools providing specific functionality to process mass spectrometric data. A common interface and use of open, standardized data formats enable chaining these into complex, custom-tailored analysis workflows. The project is supported by a lively community of developers worldwide.

OpenMS greatly benefits from a range of open-source projects by wrapping their functionality. To package these third-party tools into a coherent solution, we deploy OpenMS on all major operating systems. We provide self-contained, stand-alone installers for command line execution on cluster environments, as well as plugins for integration into workflow systems like KNIME. KNIME is an open-source, industry supported integration platform and workflow. In addition to the OpenMS plugin, hundreds of third-party plugins provide analysis tools for advanced statistics, integration of scripting languages, machine learning, database connectors or automated internet queries. This enables users to perform mass spectrometry data processing as well as downstream statistical data analysis in a single workflow. KNIME allows storing the full analysis workflow including all parameters and scripts. Parts of the workflow can be reused in other projects, and complete analysis tasks become reproducible.

Apart from development the development team also provide free consulting and regular training events to users and developers alike.

Title	Interoperable, collaborative multi-platform variant calling with bcbio
Authors	*Brad Chapman*, Rory Kirchner, Lorena Pantano, Shannan Ho Sui, Oliver Hofmann
Affiliations	Harvard Chan School, Bioinformatics Core (http://bioinformatics.sph.harvard.edu/), The University of Melbourne Centre for Cancer Research (https://umccr.github.io/)
Contact	bchapman@hsph.harvard.edu
Availability	https://github.com/chapmanb/bcbio-nextgen
Documentation	https://bcbio-nextgen.readthedocs.org/en/latest/contents/cwl.html
License	MIT

bcbio (https://github.com/chapmanb/bcbio-nextgen) is an open, community effort to develop validated and scalable variant calling, RNA-seq and small RNA analyses. Last year at BOSC, we discussed our work to port bcbio's internal workflow representation to use the community developed Common Workflow Language (CWL: http://www.commonwl.org/). This transition removed barriers that prevented bcbio interoperability.

The practical benefit of changing to standardized workflow definitions is that bcbio works on multiple heterogeneous platforms. Using CWL, bcbio runs with Curoverse's Arvados (https://arvados.org/), UCSC's Toil (http://toil.ucsc-cgl.org/) and Seven Bridges' rabix bunny (http://rabix.io/). In addition, conversion to the Workflow Description Language (WDL: https://software.broadinstitute.org/wdl/) provides in-progress support for Broad's Cromwell (https://github.com/broadinstitute/cromwell) and DNAnexus's APIs (https://github.com/dnanexus-rnd/dxWDL). There is also ongoing work with other communities actively developing CWL integration, including Nextflow (https://github.com/nextflow-io/cwl2nxf) and Galaxy (https://github.com/common-workflow-language/galaxy).

Widespread bcbio interoperability allows running in many computational environments without the overhead of maintaining bcbio specific integrations. Users can run locally or on high performance computing clusters with schedulers like SLURM, SGE and PBSPro. In addition, CWL enabled runners work across the three major cloud providers: Amazon Web Services, Google Compute Engine and Microsoft Azure. Commercial platforms like Curoverse, Seven Bridges and DNAnexus enable clinical labs to run in controlled environments. The key component of this diverse support is collaboration through the CWL standard. This demonstrates the importance of community standard development, especially in research environments where it is typically difficult to fund maintenance of large scale infastructure development.

The talk will discuss the practicalities of adjusting bcbio to use CWL and WDL. We balance infrastructure work for the transition to CWL with continued improvement of workflows and community support. Testing and documentation of bcbio is more complex since we validate workflows, like germline and somatic variant calling, in many environments. This requires coordination between groups with different focus as platforms, analyses and standards develop. High level collaboration is increasingly important as we do more complex science, and we'll describe the role of the open bioinformatics community in enabling it.

Gene Set Variation Analysis in cBioPortal

Pieter Lukasse[1], Fedde Schaeffer[1], Oleguer Plantalech Casals[1], Sander Tan[1], Sjoerd van Hagen[1]

[1] The Hyve, Arthur van Schendelstraat 650, 3511 MJ Utrecht, The Netherlands. Email: office@thehyve.nl

[1] The Hyve, Cambridge Innovation Center, 1 Broadway, 14th Floor, Cambridge, MA 02142, USA

Project Website: http://www.cbioportal.org
Source Code: https://github.com/cbioportal
License: GNU Affero General Public License v3

Abstract

cBioPortal is an open source application for interactive analysis and visualization of large scale cancer genomics datasets, originally developed by Memorial Sloan Kettering Cancer Center, New York, and since 2015 by a larger community including The Hyve, The Netherlands. Here we present the recent development we (The Hyve) did for supporting Gene Set Variation Analysis (GSVA) in cBioPortal, across the different views. We show how the new GSVA data is displayed in the oncoprint, with support for hierarchical clustering, and how it can be used in a variety of other plots and analyses.

This functionality is new in cBioPortal and could be useful to any of the users of this popular open source cancer genomics platform. It enables a new dimension of exploratory analysis in cBioPortal, allowing researchers to search and find patterns at the level of molecular processes where multiple genes that are known to work in concert, instead of exploring the data at the level of individual genes.

In our presentation we aim to share the details of this important update to the cBioPortal platform, also highlighting the way we worked with a large Pharma customer and other cBioPortal developers to implement this new feature.

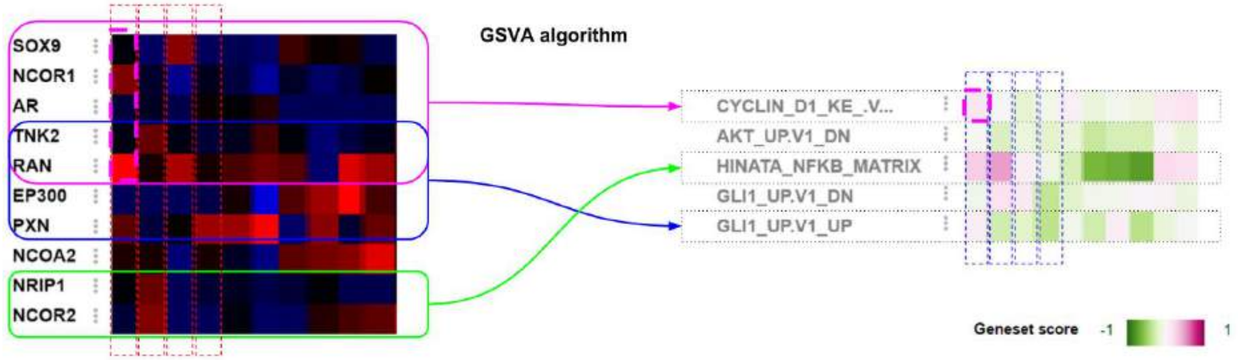


Figure 1: overview of GSVA algorithm. The goal is to highlight each gene set / sample combinations where the **genes in the given gene set** are coordinately up- or down-regulated.

The backbone of research reproducibility: sustainable and flexible tool deployment

Bjoern Gruening[1], John Chilton[2] Johannes Köster[3], Ryan K Dale[4], Yasset Perez-Riverol[5]
[1]University of Freiburg, Department of Computer Science, Georges-Köhler-Allee 106, 79110 Freiburg, Germany
[2]Department of Biochemistry and Molecular Biology, Penn State University, University Park, PA 16801, US
[3]Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, The Netherlands
[4]Laboratory of Cellular and Developmental Biology, National Institute of Diabetes and Digestive and Kidney Diseases, National Institutes of Health, Bethesda, MD 20892
[5]EMBL Outstation, European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridge, U.K.

Project: bioconda.github.io & biocontainers.pro
Code: github.com/bioconda/bioconda-recipes
Contact: bjoern.gruening@gmail.com
License: The MIT License (MIT)

A massive amount of diverse data is generated in biomedical research. To manage it and extract useful information, bioinformatic tools and software must be developed. The development of a tool should always follow a similar process. First, to solve a scientific question or a need source code is developed that can be distributed as is. To help deployment and ease the usage, the code is packaged in various package formats. The tool using the code is then deployed and used. Ideally, documentation, training and support are also provided to guide users, illustrate the solution, and advertise it.

This process, from development to support, is the golden path to develop a good tool. But issues with deployment and sustainability of the tool are found for many bioinformatic tools. What bioinformatician has not dealt with the situation of missing tool dependencies, or an older version of a tool that could not be installed due to various reasons? Deployment and sustainability of tools are therefore a major threat for productivity and reproducibility in science.

For deployment, we need a package manager that is OS- and programming language-agnostic, as bioinformatics tools are developed in many available languages and can be intended to be used on every major operating systems, including enterprise ones. Moreover all available packages should be permanently cached to be always reachable to enable reproducibility in the long term.

Here we describe a community effort to create a flexible, scalable and sustainable system to fix the tool deployment problem once and for all. Bioconda is a platform for distribution of bioinformatics software using Conda, an open source package manager developed by ContinuumIO which is, independent of any programming language and OS. Installation of Conda packages are fast and robust. No root privileges are required and multiple versions of every software can be installed and managed in parallel. Supported by an extensive documentation, writing a Conda package is very simple, easing the contribution. Thanks to its big and fast-growing community, more than 2,000 bioinformatic packages have been developed in the 18 month it has existed. These packages are long-term stored in a public repository (Cargo Port, the distribution center of the Galaxy Project), resolving the sustainability issue. Moreover, a technique called layer donning has been recently introduced to build containers automatically and very efficiently for all Conda package. These are automatically contributed to the BioContainers repository.

Development of conda packages through the BioConda community eases the packaging and the deployment of any bioinformatic tool. The interface with Cargo Port enables sustainability by mirroring all sources. Building efficient Linux containers automatically ensures an even higher layer of abstraction and isolation of the base system. Thanks to these collaborative projects, their community and their collaborations, tools can be easily packaged, deployed and will be always available to help biomedical research.

Reproducible bioinformatics software with GNU Guix

Pjotr Prins[*], Ben Woodcroft[†], Ricardo Wurmus[‡]

April 16, 2017

Website: https://www.gnu.org/software/guix/packages/
Repository: https://git.savannah.gnu.org/cgit/guix.git/
License: GPL3

Anyone who has been bitten by dependencies and would like a fully reproducible software stack should take note. Through GNU Guix we lost the fear of combining computer languages and binary deployment because all dependencies, including command line invocations of tools, are guaranteed to work.

In this talk I will share the great experience we have of packaging, deploying, publishing and distributing software via GNU Guix of a complex web service with hundreds of dependencies that has multiple servers under http://genenetwork.org/. With GeneNetwork we are creating an environment that people can do genetics on their laptop through a front-end API, e.g., for R and Python, or the browser, using content addressable storage, such as Arvados Keep, and reproducible software deployment with GNU Guix, for analysis through reproducible pipelines, such as PBS and CWL. We are even using GNU Guix to deploy pipelines on the ORNL Beacon supercomputer.

HPC computing environment and especially super computing has its bag of challenges when it comes to software deployment. As scientists we often do not get root access which means that we either depend on what software is available or we build software in a dedicated directory using tools such as Brew, Conda or even from source. Unfortunately these solutions depend on already installed tools from an underlying distribution, often proprietary or dated compilers, and, for example, modules. Any binary that gets produced, therefore, tends to be totally unique, both in the generated binary and its set of dependencies. This is bad. Bad for trouble shooting and bad for pursuing reproducible science.

With GNU Guix we have packaged more than 300 R packages, 400 Python packages, 500 Perl packages and 150 Ruby packages, including some 200 specific bioinformatics packages with some rather difficult to package tools, such as Sambamba. Thanks to the GNU Guix community it is the largest ongoing bioinformatics packaging attempt next to Debian BioMed and Bioconda.

I will discuss the work on GNU Guix 'channels', reproducible build-systems and non-root installations and moving forward on putting Guix in containers, using work flow engines, so that jobs can run on distributed systems, such as Arvados. In this talk I will explain how GNU Guix differs from distributions, such as Debian BioMed, how it can happily be deployed on any existing distribution, why it does not actually need containers, and how it can be part of Bioconda.

[*] University Medical Center Utrecht, The Netherlands, Email: pjotr.public34@thebird.nl
[†] University of Queensland, Australia
[‡] Max-Delbrück-Centrum für Molekulare Medizin (MDC), Germany

Reproducible and user-controlled software management in HPC with GNU Guix

Ricardo Wurmus[*], Altuna Akalin[†]

18th Bioinformatics Open Source Conference (BOSC) 2017, Prague, CR

Website: https://gnu.org/software/guix
Repository: https://git.savannah.gnu.org/cgit/guix.git
License: GNU General Public License version 3 (or later)

Reproducibility is the corner stone of science, and there is growing awareness among researchers of the problem of reproducibility in the field of bioinformatics research[1]. Computational research fields such as bioinformatics crucially depend on the reproducibility of software packages and the computational pipelines that are made up of them. Unfortunately, traditional software deployment methods generally do not take reproducibility into account.

On high-performance computing (HPC) systems two conflicting approaches to managing software collide: system administrators manage these large systems in a highly conservative manner, whereas the researchers using these systems may require up-to-date tool chains as well as libraries and scientific software in countless variations. Users often fall back to *ad-hoc* software deployment to satisfy their immediate requirements. As a result, HPC system users often have no guarantee that they will be able to reproduce results at a later point in time, even on the same system, and they have little hope of being able to reproduce the same software environment elsewhere.

We present GNU Guix and the functional package management paradigm and show how it can improve reproducibility and sharing among researchers[2]. Functional package management differs from other software management methodologies in that reproducibility is a primary goal. With GNU Guix users can freely customize their own independent software profiles, recreate workflow-specific application environments, and publish a package set to enable others to reproduce a particular workflow, without having to abandon package management or sharing. Profiles can be rolled back or upgraded at will by the user, independent from system administrator-managed packages.

We will introduce functional package management with GNU Guix, demonstrate some of the benefits it enables for research, such as reproducible software deployment, workflow-specific profiles, and user-managed environments, and share our experiences with using GNU Guix for bioinformatics research at the Max Delbrck Center. We will also compare the properties and guarantees of functional package management with the properties of other application deployment tools such as Docker or Conda.

References

[1] Peng, R.: Reproducible Research in Computational Science. Science, 02 Dec 2011: Vol. 334, Issue 6060, pp. 1226-1227. DOI: 10.1126/science.1213847

[2] Courts, L., Wurmus, R.: Reproducible and User-Controlled Software Environments in HPC with Guix. In: Hunold, Sascha, et al., eds. Euro-Par 2015: Parallel Processing Workshops: Euro-Par 2015 International Workshops, Vienna, Austria, August 24-25, 2015, Revised Selected Papers. Vol. 9523. Springer, 2015.

[*] GNU Guix co-maintainer, System administrator, Berlin Institute of Medical Systems Biology, Max Delbrueck Center for Molecular Medicine, Berlin, Germany. Email: ricardo.wurmus@mdc-berlin.de

[†] Principal investigator, Berlin Institute of Medical Systems Biology, Max Delbrueck Center for Molecular Medicine, Berlin, Germany.

A Ubiquitous Approach to Reproducible Bioinformatics across Computational Platforms

John Chilton[1], Marius van den Beek[2], Björn Grüning[3], Johannes Köster[4], and the Galaxy Team
[1]Department of Biochemistry and Molecular Biology, Penn State University, University Park, PA 16801, US
[2]Institut Curie, 26 rue d'Ulm, F-75248 Paris, France
[3]University of Freiburg, Department of Computer Science, Georges-Köhler-Allee 106, 79110 Freiburg, Germany
[4]Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, The Netherlands
jmchilton@gmail.com

Project: http://galaxyproject.org/ Code: http://github.com/galaxyproject/galaxy

Reproducible data analysis requires reproducible software installation. There are many approaches to reproducible software "installation" – DebianMed, Docker, EasyInstall, homebrew-science, software modules, and others.. Many work well in cloud and container-enabled environments – where the researcher has full control of a virtual machine or container host and may choose whatever software installation mechanism makes sense. However, these same approaches are less appropriate at high performance computing (HPC) centers where large centralized resources mean such freedom is unavailable. On the other hand, the HPC-centric approaches do not provide options such as ready-to-run software containers ideal for the cloud. Furthermore, some approaches are built to work with command-line scripting while others are built for specific computational platforms or deployment technologies such as the Galaxy Tool Shed or Dockstore. Here we outline an approach that covers all of these scenarios with a great deal of flexibility – allowing for the execution of the same binaries regardless of which technologies are selected.

Bioconda is a set of recipes for the popular binary package manager Conda containing thousands of bioinformatics and general purpose scientific software recipes. The BioContainer project builds environments using these these packages for technologies such as Docker and rkt - with over 2,000 containers already available. We will highlight how these projects have provided powerful new features to Snakemake and Galaxy and how they can be used to improve Common Workflow Language (CWL) tools.

We will demonstrate how to annotate Snakemake, Galaxy, and CWL tools and workflows to utilize Bioconda packages - allowing the same packages and (for compiled languages) binaries across different platforms. We will show that leveraging Bioconda increased the reproducibility of Snakemake workflows, vastly improved the development and deployment experience in Galaxy over its custom built Tool Shed package manager, and can provide a path forward for reproducibility with CWL in environments that are not Docker enabled.

For Galaxy and CWL, we will show that these same annotations can allow the platform to directly find or build BioContainers for tool execution. We will argue these containers are going to be superior to custom built ones in most cases - they are very small, automatically built (no development, annotation, registration, etc. are required by the artifact creator), and allow for the same packages and binaries to be used inside and outside of a container.

Revitalizing a classic bioinformatics tool using modern technologies: the case of the Cytoscape Project

Keiichiro Ono[1], Eric Sage[1], Barry Demchak[1]
1 University of California, San Diego. La Jolla, CA USA. Email: {kono, edsage, bdemchak}@ucsd.edu

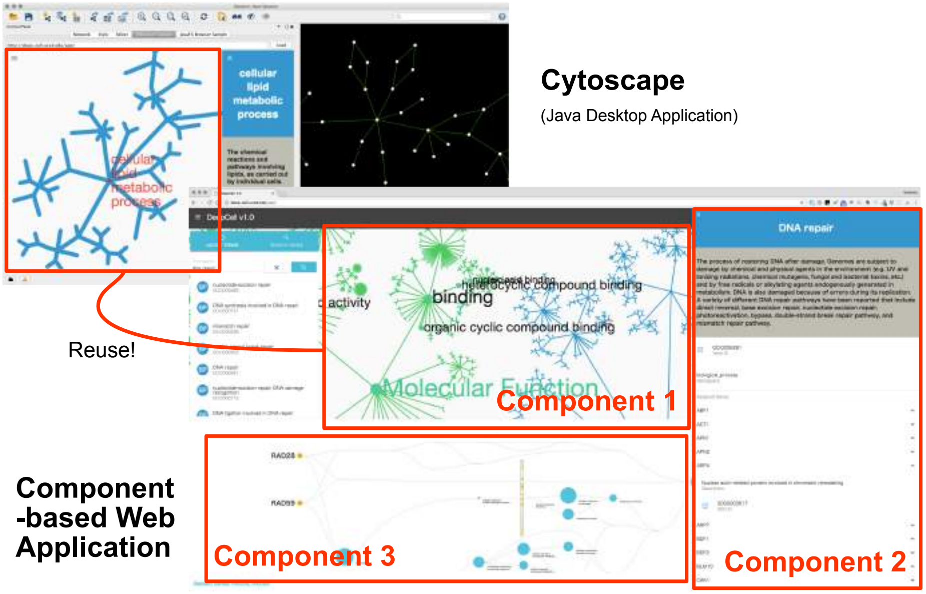
Project Website: http://www.cytoscape.org/
Source Code: https://github.com/cytoscape/
License: LGPL/MIT License

Abstract

To date, widely-used bioinformatics tools are often based on long established programming systems that are unable to effectively leverage or interoperate with new tools developed on modern programming systems running on modern web platforms. Abandoning and rewriting existing tools represents both risky and large investments of both calendar and monetary resources. In this presentation, we will discuss new approaches that enable small bioinformatics software developer teams to economically construct bridges between existing applications and modern web-based tools. We will highlight construction and use of portable and reusable UI components and computational services.

For our case study, we use Cytoscape, the de-facto standard network visualization platform in biology. Its first version was released in 2002, and it is still actively developed by the Cytoscape Consortium. It is a desktop Java-based plugin architecture, and third party developers have released over 300 Java-based apps – these apps represent a large and valuable ecosystem that cannot be abandoned even if Cytoscape itself were to be redeveloped (e.g., as a microservice architecture or single-page web application). Additionally, because Java is a relatively closed programming system, there is no easy way for Cytoscape core or apps to integrate emerging web technology that implements a complex user interface and or data visualization modules. This hurdle is often insurmountable for open source software (OSS) projects (especially in bioinformatics) that have limited resources.

We will discuss how we design, implement and integrate reusable JavaScript-based UI components (called *cyWidgets*) into Java-based Cytoscape apps, and how we build, deploy and access non-Java services across the web.


Cytoscape
(Java Desktop Application)
cellular lipid metabolic process
Reuse!
Component -based Web Application
DeepCell v1.0
activity
binding
heterocyclic compound binding
organic cyclic compound binding
Molecular Function
Component 1
DNA repair
RAD23
RAD59
Component 3
Component 2

The SPOT ontology toolkit : semantics as a service

Olga Vrousgou[1], Simon Jupp[1], Thomas Liener[1], Tony Burdett, Helen Parkinson[1]

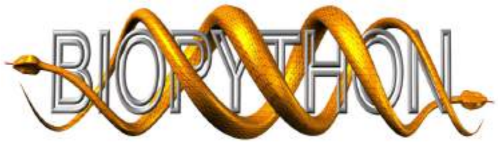
[1] European Bioinformatics Institute (EMBL-EBI), European Molecular Biology Laboratory, Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, UK: olgavrou@ebi.ac.uk

Project Website: https://ebispot.github.io/
Web Pages and Source Code:
Ontology Lookup Service: http://www.ebi.ac.uk/ols, https://github.com/EBISPOT/OLS
Zooma: http://www.ebi.ac.uk/spot/zooma, https://github.com/EBISPOT/zooma
OxO: http://www.ebi.ac.uk/spot/oxo, https://github.com/EBISPOT/OLS-mapping-service
Webulous: https://www.ebi.ac.uk/efo/webulous, https://github.com/EBISPOT/webulous
BioSolr: https://ebispot.github.io//BioSolr, https://github.com/EBISPOT/BioSolr
License: http://www.apache.org/licenses/LICENSE-2.0
Mailing List: ontology-tools-support@ebi.ac.uk

Annotating data with ontologies adds value and results in data that is more readily interoperable, discoverable and reusable. Working with multiple ontologies and their nuances creates overhead that can be readily addressed with tooling that encapsulates common use-cases. Certain activities that are common to ontology-aided data curation include querying ontologies, mapping data to and between ontologies, and creating or requesting new ontology terms. The Samples Phenotypes and Ontologies Team (SPOT) work with multiple databases to develop tools that support high-throughput, semi-automated data curation with ontologies. These tools form a toolkit of open-source software that can be deployed and run anywhere, and can be configured to work with data and ontologies from any domain.

The SPOT ontology toolkit aims to reduce the barriers to entry in the annotation of data with ontologies. The Ontology Lookup Service (OLS) provides a repository for accessing and visualising multiple ontologies. Zooma provides a repository of curated annotation knowledge that can be used to predict ontology annotations with a measure of confidence that enables the automated curation of unseen data with ontologies. OxO provides access to curated ontology cross-references and provides services for mapping from one ontology standard to another. Webulous supports ontology development from spreadsheets through a google sheet add-on. OLS, Zooma, OxO and Webulous come with integrated RESTful APIs that allow for scalable programmatic access and the development of data curation pipelines. Finally BioSolr, an ontology expansion plugin for Solr and Elasticsearch, demonstrates how advanced ontology-powered search can be achieved over data enriched with ontology annotation.

OLS, Zooma, OxO and Webulous combined address a specific need for a suite of tools that can automate more of the process of data curation with ontologies. Here we will present this toolkit, along with a suggested ontology annotation workflow designed to lower the cost of life sciences data curation.



Biopython Project Update 2017

Sourav Singh[a], Christian Brueffer[b], Peter Cock[c],
and the Biopython Contributors[d]

18th Bioinformatics Open Source Conference (BOSC) 2017, Prague, CZ

Website: http://biopython.org
Repository: https://github.com/biopython/biopython
License: Biopython License Agreement (BSD like, see http://www.biopython.org/DIST/LICENSE)

The Biopython Project is a long-running distributed collaborative effort, supported by the Open Bioinformatics Foundation, which develops a freely available Python library for biological computation [1]. We present here details of the Biopython releases since BOSC 2016, namely Biopython 1.68, 1.69 and 1.70. Together these had 82 named contributors including 51 newcomers which reflects our policy of trying to encourage even small contributions.

Biopython 1.68 (August 2016) was a relatively small release, with the main new feature being support for RSSB's new binary Macromolecular Transmission Format (MMTF) for structural data.

Biopython 1.69 (April 2017) represents the start of our re-licensing plan, to transition away from our liberal but unique *Biopython License Agreement* to the similar but very widely used *3-Clause BSD License*. We are reviewing the code base authorship file-by-file, in order to gradually dual license the entire project.

Major new features include: a new parser for the ExPASy Cellosaurus cell line database, catalogue and ontology; support for the UCSC Multiple Alignment Format (MAF), FSA sequencing files, version 4 of the Affymetrix CEL format; updates to the REBASE February 2017 restriction enzyme list; Bio.PDB.PDBList now can download more formats including MMTF; enhanced PyPy support by taking advantage of NumPy and compiling most of the Biopython C code modules.

Biopython 1.70 (July 2017) has internal changes to better support the now standard `pip` tool for Python package installation. Major new features include: support for Mauve's eXtended Multi-FastA (XMFA) file format, updates to our BLAST XML and MEME parsers, ExPASy support, and phylogenetic distance matrices. This release is noteworthy for our new logo (below), contributed by Patrick Kunzmann. This draws on our original double helix logo (above), and the blue and yellow colors of the current Python logo:

All releases fixed miscellaneous bugs, enhanced the test suite, and continued efforts to follow the PEP8 and PEP257 coding style guidelines which is now checked automatically with GitHub-integrated continuous integration testing using TravisCI. We now also use AppVeyor for continuous integration testing under Windows. Current efforts include improving the unit test coverage, which is easily viewed online at CodeCov.io.

References

[1] Cock, P.J.A., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., de Hoon, M.J. (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**(11) 1422-3. doi:10.1093/bioinformatics/btp163

[a] Dept. of Computer Engineering, VIIT, Pune, India. Email: ssouravsingh12@gmail.com

[b] Department of Clinical Sciences, Lund University, Lund, SE

[c] Information and Computational Sciences, James Hutton Institute, Invergowrie, Dundee, UK

[d] See contributor listing on GitHub.

BeerDeCoded: exploring the beer metagenome

Jonathan Sobel[1,2] , Luc Henry[1,3], Gianpaolo Rando[1,4]

1 Hackuarium, % UniverCité, chemin du Closel 5, 1020 Renens, Switzerland
2 Université de Lausanne (UNIL), Department of neuroscience, Bugnon 9, 1005 Lausanne, Switzerland
3 Ecole Polytechnique Fédérale de Lausanne (EPFL), Route Cantonale, 1015 Lausanne, Switzerland
4 SwissDeCode Sàrl, chemin du Closel 5, 1020 Renens, Switzerland

BeerDeCoded is a project carried out by members of Hackuarium, a Swiss not-for-profit association that supports unconventional research ideas and promote the public understanding of science by making laboratory tools more accessible. The goal of BeerDeCoded is to extend scientific knowledge about beer, while discussing issues related to personal genomics, food technology, and their role in society with the general public. Two years ago, a crowdfunding campaign provided funding for the first stage of the project. Reaching out through this channel also allowed to collect 120 beer samples from 20 countries.

We have now obtained the metagenomic profiles for 39 of these beers using a targeted approach (ITS). We have demonstrated that it is possible to extract DNA directly from bottled beer using low cost methodologies available to citizen scientists. DNA sequenced from these samples identified a variety of wild yeast species in commercial beers. For example, some brews contain traces of more than 30 different fungal species. Brewing is a complex process and it is unclear if the beer microbiome can directly affect the final beer taste or texture and how it could be controlled to create beers with a specific character.

To answer these questions, we are collecting information about the brewing process to correlate the metagenomic profiles with metadata related to the brewing process. For instance, using a hierarchical clustering approach, we built a proximity tree of the different beers. This analysis revealed that two stouts brewed in the same city have a similar wild yeast content. However, there is currently limited access to information about the recipe of commercial beers. A notable exception is BrewDog, a craft brewery from Scotland that recently released all its brewing recipes. We would like to use this resource and pair it with a new protocol based on a portable DNA sequencer to build the proof of concept for a beer metagenome analysis pipeline that could be used in high schools, citizen science laboratories, craft breweries or industrial plants.

Altogether, we demonstrated that coupling simple laboratory procedures with DNA sequencing and metagenomic analysis can allow the detection of the microbial content in commercial beer and can easily trigger a tripartite conversation between the general public, the scientists and professional brewers. The next step is to set up an open repository where these parties can add the metagenomic profiles of their favourite beers, expanding the database and allowing researchers to test new analyses, brewers to try new recipes and the general public to discover more science and more beers.

Supporting curation communities & collecting technical dividends

Monica C Munoz-Torres[1], Seth Carbon[1], Nathan A Dunn[1], Deepak Unni[2], Eric Yao[3], Christine E Elsik[2], Ian Holmes[3], Suzanna E Lewis[1]

[1] Lawrence Berkeley National Laboratory, Environmental Genomics and Systems Biology Division, Berkeley, CA. Email: MCMunozT@lbl.gov
[2] University of Missouri, Divisions of Plant and Animal Sciences, Columbia, MO.
[3] University of California Berkeley, Department of Bioengineering, Berkeley, CA.

Project Website: http://genomearchitect.org/
Source Code: https://github.com/GMOD/Apollo
License: Berkeley Software Distribution (BSD-3).

Scientific research is inherently a collaborative task; for curation of genome annotations, this means a dialog within and between communities of researchers to reach a shared understanding of the biology underlying the sequences. Here we describe our experience developing software to support an increasing number of genome curation communities, emphasizing the collection of technical contributions as unexpected dividends in the process.

Annotation tools help improve curation quality, and in many research groups, they constitute a means to introducing human curation to the annotation process for the first time. To facilitate the dialog between and within communities of curators, our team has developed a number of web-based tools, among them **Apollo**, a widely-adopted genome annotation editor designed to support structural curation of gene models in a collaborative, real-time fashion. Apollo, built on top JBrowse (http://jbrowse.org/), presents a set of tooling for data extraction and integration that allows users to generate analysis-ready data and progress reports; it also offers a secure web-service API to programmatically perform annotations, to integrate Apollo with workflow tools (e.g. Galaxy https://galaxyproject.org/, via Docker https://www.docker.com/), etc.

We have dedicated considerable efforts to spread the word about Apollo (i.e. talks and workshops to train the community on installation and use), and to continuously make improvements to support the needs of the growing research community that forms our user base. It is currently used in more than one hundred genome annotation projects around the world and across the tree of life. Recently, these improvements have included technical contributions from developers outside our core group, ranging from productive discussions, to organizing public hackathons, to welcoming code contributions to Apollo's main development branch. This means that a portion of our target communities both have an interest in creating an ideal curation environment for their research objectives using Apollo, and are choosing to prioritize investing their resources to achieve it. It is also evidence that our efforts to support curators with an open-source, community-based bioinformatics tool have been so far successful. And as a result, we are receiving extensive technical contributions from these communities, an unexpected return on these efforts that both supports our mission and improves the workflow for our community of users.

Journal of Open Source Software (JOSS)

Pjotr Prins[*], Vignesh Sekar[†], Roman Valls Guimera[‡]& Arfon Smith[§]

May 24, 2017

Website: http://joss.theoj.org/papers/popular
Repository: https://github.com/openjournals/joss
License: MIT

Journal of Open Source Software (JOSS)

JOSS is a free open-access journal for software developers in academic research. JOSS reviewed and published 100 papers in its first year and the list of authors, reviewers and editors is growing.

Publishing a paper with JOSS gives you a citeable publication for your software project. JOSS provides a cross reference Digital Object Identifier (DOI) and is indexed on Google Scholar. With enough bioinformatics papers we aim to get JOSS automatically indexed on Pubmed.

As a journal we believe that the source code is the story. When the source code is published under an open software initiative (OSI) approved license, anyone can access and use it. *If you already wrote all the source code, documentation and tests, why would you need to write another full-length paper?*

With JOSS, the write-up can be a short piece of text (up to one page) with references, see for example the GeneNetwork paper:

http://joss.theoj.org/papers/10.21105/joss.00025

You can see the review process uses the github issue tracker. Review is transparent, open and designed to be collaborative between the authors, reviewers, and editor. The full workflow is explained at

https://doi.org/10.6084/m9.figshare.4688911

Note: JOSS runs on github and is an OSI affiliate.

Next to JOSS we will also discuss ongoing work on a new **Journal of Open Data**. This journal will allow people to publish their data for free in a content addressable public space and get a DOI. Not only will this data be citeable, it will also be FAIR and help reproducible analysis. Metadata can be mined through SPARQL and JSON filters.

[*] University Medical Center Utrecht, The Netherlands, Email: pjotr.public34@thebird.nl
[†] NIIT University, India
[‡] Center for Cancer Research, University of Melbourne, Australia
[§] Data Science Mission Office, Space Telescope Science Institute, Baltimore, MD, USA

Building an open, collaborative, online infrastructure for bioinformatics training

Bérénice Batut[1], Galaxy Training Network[2], Dave Clements[3], Bjoern Gruening[1]

[1]Bioinformatics group, University of Freiburg, Freiburg, Germany. Email: berenice.batut@gmail.com

[2]https://galaxyproject.org/teach/gtn/

[3]Johns Hopkins University, Baltimore, USA

Project Website: http://galaxyproject.github.io/training-material/
Source Code: https://github.com/galaxyproject/training-material/

Abstract

With the advent of high-throughput platforms, analysis of data in life science is highly linked to the use of bioinformatics tools, resources, and high-performance computing. However, the scientists who generate the data often do not have the knowledge required to be fully conversant with such analyses. To involve them in their own data analysis, these scientists must acquire bioinformatics vocabulary and skills through training.

Unfortunately, data analysis and its training are particularly challenging without computational background. The Galaxy framework addresses this problem by offering a web-based, intuitive and accessible user interface to numerous bioinformatics tools. It enables sophisticated bioinformatic analysis without requiring life scientists to learn programming, command line interfaces, or systems administration.
Trainings based on Galaxy generates significant interest and the number of such events is ever growing, with more than 70 events in 2016. To federate these events and the involved people, the Galaxy Training Network (GTN) was created in 2014. GTN now has 32 member groups and almost 100 individuals.

Recently, GTN set up a new open, collaborative, online model for delivering high-quality bioinformatics training material: http://galaxyproject.github.io/training-material. Each of the current 12 topics provides tutorials with hands-on, slides and interactive tours. Tours are a new way to go through an entire analysis, step by step inside Galaxy in an interactive and explorative way. All material is openly reviewed, and iteratively developed in one central repository by 40 contributors. Content is written in Markdown and, similarly to the Software/Data Carpentry, the model separates presentation from content. It makes easier for community contributions and enables bulk updates and style changes independently of the training content. This is beneficial for the sustainability of this project. In addition, the technological infrastructure needed to teach each topic is also described with an exhaustive list of needed tools. The data (citable via DOI) required for the hands-on, time and resource estimations and flavored Galaxy Docker images are also provided.

This material is automatically propagated to Elixir's TeSS portal. With this community effort, the GTN offers an open, collaborative, FAIR and up-to-date infrastructure for delivering high-quality bioinformatics training for scientists.

Software and social strategies for community sourced biological networks and ontologies

Dexter Pratt[1]

[1] Dexter Pratt, UC San Diego, 9500 Gilman Drive, La Jolla CA. Email: depratt@ucsd.edu

Project Website: http://www.ndexbio.org
Source Code: https://github.com/ndexbio
License: BSD 3-Clause (http://www.home.ndexbio.org/disclaimer-license/)

Main Text of Abstract

We present community-sourcing social strategies and supporting software infrastructure to promote the creation, maintenance, and dissemination of computable biological networks and ontologies, coupled with practical, ready-to-use facilities in the Network Data Exchange (NDEx) framework. Network resources of many types, from pathway models authored by experts to data-driven ontologies and patient similarity networks, are valuable to researchers both as human-readable references and as input to analyses in applications and scripts. There are, however, significant challenges in providing high quality network resources: curation efforts are expensive and difficult to sustain; data-driven networks may be isolated as supplemental information to publications without support for revision and reuse. Moreover, network resources are not typically subject to peer review: even when a network is associated with a peer-reviewed publication, its content is unlikely to be reviewed in detail. To address these challenges, we present a strategy to create incentives for sustainable community-sourced development by (1) enabling organizations such as research consortia or editorial boards of subject matter experts to publish reviewed collections of networks, (2) facilitating workflows integrating network data with academic publication, and (3) preserving author attribution in derivative works. The effort required by participants is reduced by targeted infrastructure and interfaces incorporated in the NDEx framework for sharing, accessing, and publishing networks. The use of standards by authors is facilitated by streamlined annotation interfaces in NDEx and encouraged by rewards such as preferential search rankings for well-annotated networks. Reproducible science is promoted by enabling researchers to readily disseminate reusable networks, accessible via stable identifiers and discoverable via multi-parameter search. Finally, we make a call to action for researchers to participate in creating diverse, high-quality, and sustainable resources of biological networks and ontologies as authors, reviewers, community organizers, and thought leaders.

Distance-based, online bioinformatics training in Africa: the H3ABioNet experience

Kim Gurwitz[1,5], Shaun Aron[2,5], Sumir Panji[1,5], Suresh Maslamoney[1], Pedro L. Fernandes[3], David P. Judge[4], Nicola Mulder[1,5]

[1]Computational Biology Division, Department of Integrative Biomedical Sciences, IDM, University of Cape Town, South Africa.

[2]Sydney Brenner Institute for Molecular Bioscience, University of the Witwatersrand, South Africa.

[3]Instituto Gulbenkian de Ciencia, Portugal.

[4]Independent Bioinformatics training specialist, United Kingdom.

[5]Members of the H3ABioNet Consortium's Education and Training Working Group, part of the H3Africa consortium.

Email of presenting author: kim.gurwitz@uct.ac.za

Project Website: http://training.h3abionet.org/IBT_2016/
Source Code: NA

A distance-based Introduction to Bioinformatics course is run by H3ABioNet - a pan African Bioinformatics network for H3Africa. In its second iteration this year, the free, 3-month, skills-based course teaches the basics of various bioinformatics analyses. It makes use of multiple education delivery methods, namely: face-to-face learning; distance learning; and open educational resources, in order to increase access across Africa. Local classrooms - 27 classrooms hosting roughly 600 participants, in total, across 12 African countries (in 2017) - are attended face-to-face for additional support where there is interaction with volunteer, teaching assistants and with peers. During these face-to-face sessions, classrooms watch open access, pre-recorded and downloaded lecture recordings, prepared by expert African bioinformatics trainers. Classrooms also sign in to a virtual classroom that links them all to each other and to the trainer during biweekly contact sessions. Further, course participants and volunteer local staff engage via online forums hosted on the course management platform. Additional features of the course this year, developed out of an extensive review of last year's iteration, include: staff training at local classrooms; promoting previous year attendees as trainee teaching assistants; consolidation sessions; and encouraging engagement within and across classrooms as well as with local bioinformatics communities. The unique course design ensures that many common challenges in Africa, namely: access to bioinformatics expertise; access to bioinformatics training; and Internet access instability, are not barriers to accessing education. Although developed for a resource limited setting, the learning model employed could easily be adapted to other settings.

Recent object formation in the core of Galaxy

Martin Čech[1], and the Galaxy Team and Community

[1]Department of Biochemistry and Molecular Biology, Penn State University, University Park, PA 16801, USA

Contact: marten@bx.psu.edu
Code: github.com/galaxyproject/galaxy
Project: galaxyproject.org
License: Academic Free License version 3.0

Galaxy pursues accessibility, transparency and reproducibility in data-intensive science. It is a web-based framework that automatically tracks analyses and enables researchers to use advanced computational tools and resources while focusing on the research questions rather than the supporting compute infrastructure. In this report we highlight recently added and enhanced features and discuss future directions. The most prominent recent developments are:

Conda for dependency resolution. *Galaxy* has deprecated its own solution (Tool Shed package recipes) for the software package management and embraced the Conda manager for these purposes. This makes tool dependencies more reliable and stable, and they are also easier to test and faster to develop.
Interactive Environments integrated into Galaxy interface are powerful tools for interactive *ad hoc* analyses as demonstrated in a recent paper (doi.org/10.1101/075457). In addition to the Jupyter notebook and RStudio, the Galaxy community has expanded the number of available IEs to include Phinch for metagenomic data visualization, Ethercalc for tabular and csv data, and Neo4j for graph database manipulation.
Galaxy Webhooks are a system for adding plugins allowing for customization of individual instances. Webhooks are admin-enabled and often community-contributed pieces of code that alter the interface and offer extra features. Popular webhooks will likely become part of the core UI in the future.
Dataset Collections are becoming increasingly powerful - you can now directly create them when uploading datasets, flatten and import them to data libraries, and use many new and improved collection operations tools. Various toolkits were enhanced to handle collections natively.
Other notable advancements: You can start up an independent **chat server** and connect it to Galaxy enabling users to share and collaborate without leaving the analysis interface. Galaxy also supports **compressed FASTQ** formats allowing to save storage and remove unnecessary steps from workflows. Tool cache and **'hot reload'** functionalities have also been added, enabling administrators to update tools without a server restart. The tool cache has also made Galaxy startup much faster, especially for instances with many tools. Histories now allow for **drag&drop** of datasets as tool inputs and also can **propagate dataset tags** through tool executions. In the last year **3 new IUC members** joined our ranks totalling 15 now. They handled 401 pull requests in last 12 months and added numerous contributions to Bioconda and other connected projects.

Reproducibility of computational workflows is automated using continuous analysis

Brett K. Beaulieu-Jones[1], Casey S. Greene[2]

[1] Genomics and Computational Graduate Group, Perelman School of Medicine, University of Pennsylvania, Philadelphia, Pennsylvania, USA. Email: brettbe@med.upenn.edu

[2] Department of Systems Pharmacology and Translational Therapeutics, Perelman School of Medicine, University of Pennsylvania, Philadelphia, Pennsylvania, USA.

Project Website: http://www.nature.com/nbt/journal/v35/n4/full/nbt.3780.html
Source Code: https://github.com/greenelab/continuous_analysis/
License: (example) BSD 3-clause - https://github.com/greenelab/continuous_analysis/blob/master/LICENSE

Reproducibility is central to science. The ability to trust, validate, and extend previous works drives scientific advancement. In a recent survey conducted by Nature, ninety-percent of researchers acknowledged a "reproducibility crisis". Concerns about reproducibility are beginning to change funding and publishing decisions – in short science that lacks rigor can impact you immediately. Funders are moving towards early release of research products – data, software, protocols, preprints and publishers are emphasizing data availability, statistical rigor and so on. This talk will highlight the impact of the "reproducibility crisis" on different stakeholders and suggest potential routes forward including "Continuous Analysis" –

Reproducing computational biology experiments, which are scripted, should be straightforward. But reproducing such work remains challenging and time consuming. In the ideal world, we would be able to quickly and easily rewind to the precise computing environment where results were generated. We would then be able to reproduce the original analysis or perform new analyses. We introduce a process termed "continuous analysis" which provides inherent reproducibility to computational research at a minimal cost to the researcher. Continuous analysis combines Docker, a container service akin to virtual machines, with continuous integration, a popular software development technique, to automatically re-run computational analysis whenever relevant changes are made to the source code. This allows results to be reproduced quickly, accurately and without needing to contact the original authors. Continuous analysis also provides an audit trail for analyses that use data with sharing restrictions. This allows reviewers, editors, and readers to verify reproducibility without manually downloading and rerunning any code.

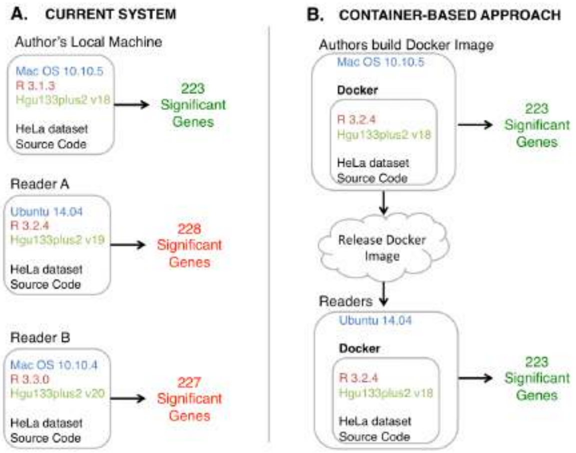


Figure 1. Research computing versus container-based approaches for differential gene expression analysis of HeLa cells. (a,b) Numbers of significantly differentially expressed genes identified using different versions of software packages (a) and a container-based approach with a defined computing environment (b). n = 3 biological replicates per group (wild-type or double-knockdown HeLa cells).

Full-stack genomics pipelining with GATK4 + WDL + Cromwell

Kate Voss[1], Jeff Gentry[2], Geraldine Van der Auwera[3]

[1] Broad Institute, Cambridge, MA, USA. Email: kvoss@broadinstitute.org

[2] Broad Institute, Cambridge, MA, USA. Email: jgentry@broadinstitute.org

[3] Broad Institute, Cambridge, MA, USA. Email: vdauwera@broadinstitute.org

Project Websites:
http://software.broadinstitute.org/gatk
http://software.broadinstitute.org/wdl

Source Code:
https://github.com/broadinstitute/gatk/
https://github.com/broadinstitute/cromwell
https://github.com/broadinstitute/wdl

License: BSD 3-clause (see https://github.com/broadinstitute/gatk/blob/master/LICENSE.TXT)

Main Text of Abstract

GATK4 is the new major version of the Genome Analysis Toolkit (GATK), one of the most widely used software toolkits for germline short variant discovery and genotyping in whole genome and exome data. For genomics analysts, this new version greatly expands the toolkit's scope of action within the variant discovery space and provides substantial performance improvements with the aim of shortening runtimes and reducing cost of analysis. But it also offers significant new advantages for developers, including a completely redesigned and streamlined engine that provides more flexibility, is easier to develop against, and supports new technologies such as Apache Spark and cloud platform functionalities (e.g. direct access to files in Google Cloud Storage).

WDL and Cromwell are a Workflow Definition Language and a workflow execution engine, respectively. The imperative that drives WDL's development is to make authoring analysis workflows more accessible to analysts and biomedical scientists, while leaving as much as possible of any runtime complexity involved to the execution engine.

GATK4, WDL and Cromwell are all developed by the Data Sciences Platform (DSP) at the Broad Institute and released under a BSD 3-clause license. For more information on GATK's recent licensing change, please see https://software.broadinstitute.org/gatk/blog?id=9645.

Taken together, these three components constitute a pipelining solution that is purposely integrated from the ground up, although they can each be used independently and in combination with other packages through features that maximize interoperability. This principle of integration applies equally to development, to deployment in production at the Broad, and to support provided to the external community.

ToolDog - generating tool descriptors from the ELIXIR tool registry

Kenzo-Hugo Hillion[1], Ivan Kuzmin[2], Hedi Peterson[2], Jon Ison[3], Hervé Ménager[1]

[1]Bioinformatics and Biostatistics HUB, C3BI, Institut Pasteur, France. Email: kehillio@pasteur.fr

[2] Institute of Computer Science, University of Tartu, Estonia.

[3]Center for Biological Sequence Analysis Department of Systems Biology, Technical University of Denmark, Denmark.

Over the last years, the use of bioinformatics tools has been eased by the use of workbench systems such as Galaxy [1] or frameworks that use the Common Workflow Language (CWL) [2]. Still, the integration of these resources in such environments remains a cumbersome, time consuming and error-prone process. A major consequence is the incomplete description of tools that are often missing information such as some parameters, a description or metadata.

ToolDog (Tool DescriptiOn Generator) is the main component of the Workbench Integration Enabler service of the ELIXIR bio.tools registry [3, 4]. The goal of this tool is to guide the integration of tools into workbench environments. In order to do that, ToolDog is divided in two main parts: the first part analyses the source code of the bioinformatics software with language dedicated tools and generates a Galaxy XML or CWL tool description. Then, the second part is dedicated to the annotation of the generated tool description using metadata provided by bio.tools. This annotator can also be used on its own to enrich existing tool descriptions with missing metadata such as the recently developed EDAM annotation.

References:

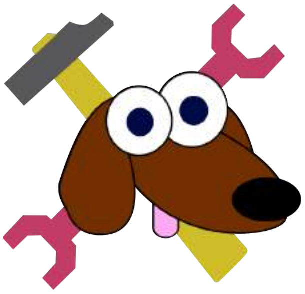
[1] Enis Afgan et al. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. Nucleic Acids Research (2016) doi: 10.1093/nar/gkw343

[2] Amstutz, Peter et al. (2016): Common Workflow Language, v1.0. figshare. https://doi.org/10.6084/m9.figshare.3115156.v2. Retrieved: 15 37, Mar 09, 2017 (GMT)

[3] Jon Ison et al. Tools and data services registry: a community effort to document bioinformatics resources. Nucleic Acids Research, 44(D1):D38–D47, January 2016. ISSN 0305-1048, 1362-4962. doi: 10.1093/nar/gkv1116.

[4] https://bio.tools

Project Website: https://github.com/bio-tools/tooldog/
Source Code: https://github.com/bio-tools/tooldog/
License: MIT



BioThings SDK: a toolkit for building high-performance data APIs in biology

Cyrus Afrasiabi[1§], Sebastien LeLong[1§], Jiwen Xin[1], Ginger Tsueng[1], Andrew I. Su[1], Chunlei Wu[1*]

[1] The Scripps Research Institute, 10550 N Torrey Pines Rd, La Jolla, CA 92037

[*] Email: cwu@scripps.edu

[§] Contributed equally to this work

Project Website: http://biothings.io
Source Code: https://github.com/biothings/biothings.api
License: Apache License v2

The accumulation of biological knowledge and the advance of web and cloud technology are growing in parallel. The latest computation technology allows us to modernize the way we collect, organize and disseminate the growing biological knowledge. Building web-based APIs (Application Programming Interfaces) becomes more and more popular for accessing data in a simple and reliable manner. Comparing to the traditional raw flat-file downloads, web APIs allow users to request specific data such as a list of genes of interest without having to download the entire data file. Web APIs typically return data in a format conforming to common standards (e.g. JSON or XML). This means that developers can spend less time on wrangling data, and more time on analysis and discovery.

We previously developed two high-performance and scalable web APIs for gene and genetic variant annotations, accessible at MyGene.info and MyVariant.info. These two APIs are a tangible implementation of our expertise and collectively serve over 6 million requests every month from thousands of unique users. Crucially, the underlying design and implementation of these systems are in fact not specific to genes or variants, but rather can be easily adapted to other biomedical data types such drugs, diseases, pathways, species, genomes, domains and interactions, collectively, we refer them as "**BioThings**".

BioThings SDK is a generalized software development kit (aka **SDK**) for building the same high-performance APIs for other BioThings data types. This SDK enables other developers to build their own BioThings API for their specific data types. Users can take advantage of the abstracted technical layers we built into the SDK, and produce a high-performance API, which follows the best practice and community standards. We also adopted JSON-LD (JSON for Linked Data) as a mechanism to form the semantic connections between different data types, so that the set of BioThings APIs will form a network of linked programmatic-accessible biological knowledge. BioThings SDK now becomes the common backend of both MyGene.info and MyVariant.info APIs, and was also used to create two new members of BioThings APIs: one for chemical compound and drug data (http://c.biothings.io), and the other for taxonomy data (http://t.biothings.io). A generic *biothings* Python client (https://github.com/biothings/biothings_client.py) has also been created to provide a ready-to-use Python client for any BioThings API without the need of any extra code, and yet it still remains extensible for any data-type specific customization.

Integrating cloud storage providers for genomic analyses

Ted Liefeld[1], Marco Ocana[2], Michael Reich[1], Helga Thorvaldsdottir[2], Jill P Mesirov[1]

[1] The University of California San Diego, San Diego, CA, USA. Email: jliefeld@cloud.ucsd.edu

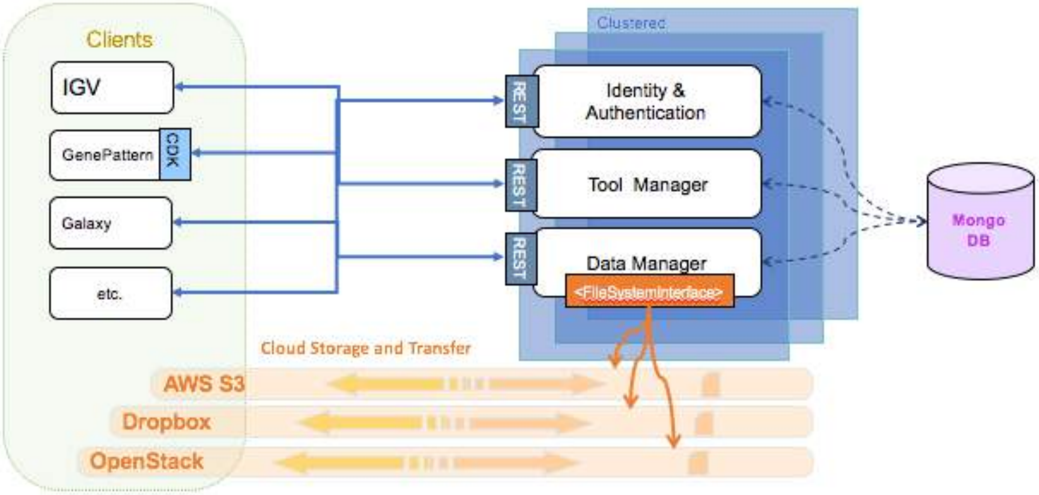
[2] The Broad Institute of MIT and Harvard, Cambridge, MA, USA.

Project Website: http://www.genomespace.org/
Source Code: https://bitbucket.org/GenomeSpace/combined
License: LGPL version 2.1 (http://www.opensource.org/licenses/lgpl-2.1.php)

Cloud storage is being used ever more frequently for genomics analysis due to its extreme scalability and constantly declining costs. Numerous cloud storage providers have been used in projects including Google, Amazon AWS, Dropbox, and hybrid clouds using tools such as openStack Swift. Existing genomic analysis tools offer disparate support for these cloud storage platforms. Some, but not all, of these tools run on one of the common cloud providers. To improve analysis throughput we would like the analysis and tools to be as closely co-located as possible, or failing that, for there to be wide bandwidth transfer between the location of the data and the location of the analysis.

To address this problem, and simultaneously avoid the m*n solutions necessary to integrate storage providers with the analysis providers, we have developed GenomeSpace (www.genomespace.org), a free and open source cloud-based environment that provides interoperability between best-of-breed computational tools. GenomeSpace includes an interface (API) for analysis tools to securely access data in the cloud as well as a platform that makes it easy to add additional tools and cloud storage providers. The GenomeSpace FileSystemInterface and StorageSpec API provide a structured way to connect cloud storage provider accounts with their GenomeSpace account, and make its contents available to all GenomeSpace tools through the GenomeSpace DataManager REST-ful API. The FileSystemInterface uses familiar file metaphors, but works equally well object stores such as Amazon S3 and OpenStack Swift.

In this talk we will describe the details of the data management architecture and interfaces in GenomeSpace that facilitate the connection of multiple cloud storage systems with a large and growing collection of analysis tools.



Title	Fighting Superbugs with Open Source Software
Author	*Kai Blin*, Marnix Medema, Sang Yup Lee, Tilmann Weber
Affiliation	Technical University of Denmark, Novo Nordisk Foundation Center for Biosustainability
Contact	kblin@biosustain.dtu.dk
URL	https://bitbucket.org/antismash/antismash
License	GNU Affero General Public License (AGPL) version 3.0

Antibiotics are one of the most important discoveries in medical history. They form the foundation of many other fields of modern medicine, from cancer treatments to transplantation medicine. Unfortunately, antibiotics are liable to misuse, and have been widely misused, giving rise to an ever-growing number of resistant bacteria, often called superbugs. Medical professionals all over the world are increasingly encountering superbug infections that are untreatable by any available antibiotics. New classes of antibiotics that can sidestep the common resistance mechanisms are desperately needed. Unfortunately, the pipeline for discovering new antibiotics has all but dried up. From 1935 until 1968, 14 new classes of antibiotics were discovered. Since then, only five further classes have been added to our arsenal. The superbugs seem to be winning the arms race.

Fortunately, the cloud has a silver lining. About 70% of the clinically used antibiotics are produced by a group of bacteria, the actinomycetes. With the recent surge in genome sequencing technology, is is becoming clear that many actinomycetes - as well as other bacteria and fungi - carry a large, untapped reservoir of further potential antibiotics. In order to assist lab scientists in discovering these new antibiotics while lowering the re-discovery rate of already known substances, new search strategies are needed.

antiSMASH is a fully Open Source tool to assist life scientists in the discovery of new drug leads. Since its initial release in 2011, it has become one of the most popular tool in the area of antibiotics discovery. Standing on the shoulder of giants, antiSMASH in turn leverages many Open Source life science tools to do its job, providing state-of-the-art bioinformatics analyses via an accessible, easy to use web interface.

This talk presents how Open Source software powers the hunt for new antibiotics to fight the rising threat of superbugs.

Users, Communication, and a Light Application-Level API: A Request for Comments

Seth Carbon[*]

17th Bioinformatics Open Source Conference (BOSC) 2017, Prague, Czech Republic

Website: https://github.com/kltm/lala
Repository: https://github.com/kltm/lala

Much thought and paper has been put into sharing information between resources, from the use of common identifiers to use of common data stores and APIs. We would like to explore an area that has not, to the authors' knowledge, been as well explored: the sharing of simple information directly between web-based applications, without the use of a non-application central server or API as an intermediary.

More concretely, the use cases that we wish to look at are how to manage user authentication, authorization, and how to obtain small packets of specific information from an external resource that has its own associated web application. As a way to explore this space, we have created applications that accomplish this by passing tokens and round-tripping a packet of information (acting as a "black box" or "piggy bank") through the external application using basic HTTP methods, allowing an easy high-level "federation". In an environment where much effort is being spent on the creation of rich web applications tailored to the habits of specific groups of scientists and curators, users and engineers would have much to gain by being able to reuse functionality from external applications as-is in their workflows and stacks, as well as having the added benefit of driving traffic to external web applications that implement such a common specification.

Through these explorations, we have decided that a path forward should capture the following qualities:

- Easy to implement: complexity can be a barrier to adoption and implementation
- Basic HTTP tooling: any system should have easy access to the tools necessary
- "Stateless": simplifying debugging and implementation
- Minimal need for initial coordination
  Beyond what data is to be returned, the external application does not need to understand the transiting packet
- No need for calling application to coordinate changes to own API after initial coordination: as the calling application is responsible for decoding the information it initially sent and the location it is sent to, major API changes can occur without the need to coordinate with any other resource
- Have the ability to perform operations "remotely" while still logged-in to the calling application, without the need to coordinate cross-site logins: this can be done by placing an authorization token in the transiting packet

Taking inspiration from the methods used by Galaxy [1] to pull data in from external applications, we'd like to discuss this potential protocol and get feedback from the community about general patterns for passing users and relatively simple data directly between applications. Variations of this proposal have been implemented or explored in applications such as: Noctua, Textpresso Central, AmiGO, and PubAnnotation.

References

[1] The Galaxy platform for accessible, reproducible and collaborative biomedical analyses, 2016 update *Nucleic Acids Research* 44(W1): W3-W10, doi:10.1093/nar/gkw343

[*] Berkeley Bioinformatics Open-source Projects, Lawrence Berkeley National Lab, Berkeley, CA, USA.

RADAR-CNS - Research Infrastructure for processing wearable data to improve health

Julia Kurps[1], Maxim Moinat[1], Joris Borgdorff[1], Francesco Nobilia[2], Maximilian Kerz[2], Nivethika Mahasivam[1], Irina Pulyakhina[1], Matthias Dümpelmann[4], Herculano Campos[5], Mark Begale[6], Richard Dobson[2,3], Amos Folarin[2,3]

1 The Hyve, Arthur van Schendelstraat 650, 3511 MJ Utrecht, The Netherlands
2 Department of Biostatistics and Health Informatics, Institute of Psychiatry Psychology & Neuroscience, King's College London, Box P092, 16 De Crespigny Park, SE5 8AF, UK
3 Farr Institute of Health Informatics Research, UCL Institute of Health Informatics, University College London, London WC1E 6BT, UK.
4 Center of Epilepsy, University Hospital Freiburg, Breisacher Str. 64, 79106 Freiburg, Germany
5 Goldenarm, 20 Jay Street, Suite 840, Brooklyn, NY, 11201
6 Vibrent Health, 12015 Lee Jackson Memorial Highway, Suite 13, Fairfax, VA, 2203, United States

Corresponding E-mail: julia@thehyve.nl, amos.folarin@kcl.ac.uk

Project Website: http://www.radar-cns.org/
Source Code: https://github.com/RADAR-CNS
License: Apache2.0

Remote **A**ssessment of **D**isease **A**nd **R**elapse – **C**entral **N**ervous **S**ystem (**RADAR-CNS**) is an innovative collaborative research project to evaluate the potential of wearable devices and smartphone technology to improve quality of life for people with epilepsy, multiple sclerosis (MS) and major depression disorder (MDD).

RADAR-CNS is built upon close collaboration of patient organizations, clinical partners, research institutes and industry to develop new strategies for treatment of patients with brain disorders. The aim of RADAR-CNS is to evaluate how to best leverage innovative technologies like wearable devices and smartphone-based applications for remote measurement and potential relapse prediction. Together with our data processing partners, The Hyve is building an open source infrastructure to capture, process, manage and analyse data from wearable devices and facilitate the integration with data from multiple other sources like clinical and -omics data. Our clinical partners will use this data processing pipeline in multiple clinical trials. Sustainability is a focus point during the development of the RADAR platform. Therefore, we develop a generic platform, which will not be limited to brain disorder applications, but will be applicable for subsequent RADAR projects like RADAR Diabetes. Furthermore, we are actively facilitating a striving open source community around the RADAR platform to ensure longevity of the research infrastructure and encourage cross-infrastructure efforts. Goal of the pilot study is the evaluation of wearable device data for passive remote measurement of patients in a hospital epilepsy monitoring unit (EMU). We developed an Android application, which captures wearable data via a Bluetooth connection and streams data to an internal hospital server. Integration of SDKs for multiple wearable devices allows us to investigate and compare the quality of data collected from different devices (e.g., Empatica E4 and Angel Sensor) as well as device specifications, such as battery life and signal stability/range. Besides data quality assessment, we will investigate whether data from wearable devices has potential for seizure detection. Parallel video and EEG monitoring in monitoring units of specialized epilepsy centers are used as a Gold Standard for seizure detection evaluation. We will investigate the potential of wearable devices as clinically valuable alternatives to complement or even replace hospital-based technologies; a prerequisite for ambulatory passive remote measurement of patients in their home environment.

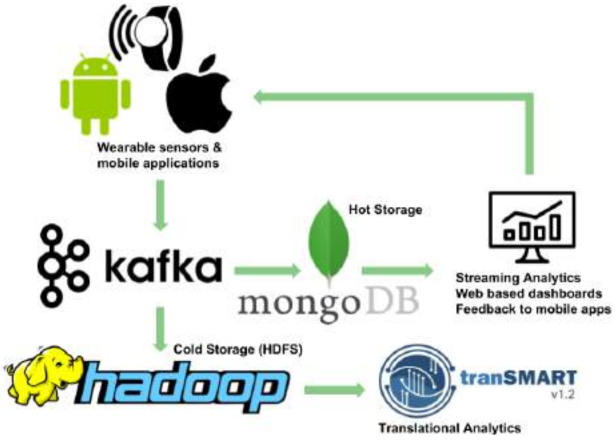


Figure1. **Technology stack of processing infrastructure.** Data is collected from wearable sensors, standardized to data schemata and processed with Apache kafka, which allows streaming analytics. Raw data is stored using Hadoop for further advanced analytics.

The RADAR-CNS platform is adding an additional dimension to the research infrastructure for personalised medicine and health by allowing new ways to leverage innovative technologies for better data integration.

References

1. http://www.radar-cns.org
2. https://github.com/RADAR-CNS

Using Wikidata as an open, community-maintained database of biomedical knowledge

Andrew I. Su[1], Sebastian Burgstaller-Muehlbacher[1], Timothy Putman[1], Andra Waagmeester[2], Gregory S. Stupp[1], Julia Turner[1], Elvira Mitraka[3], Matthew Jacobson[4], Núria Queralt-Rosinach[1], Paul Pavlidis[4], Lynn Schriml[3], Benjamin M. Good[1]

[1] The Scripps Research Institute, La Jolla, CA, USA; asu@scripps.edu.

[2] Micelio, Antwerp, Belgium.

[3] University of Maryland Baltimore, Baltimore, MD, USA.

[4] University of British Columbia, Vancouver, Canada.

Project Website: https://www.wikidata.org/wiki/User:ProteinBoxBot
Source Code: https://github.com/SuLab/GeneWikiCentral (and repos linked therein)
License: MIT

The sum total of biomedical knowledge is accumulating at an explosive rate. One metric of this rapid progress is the exponential growth in the biomedical literature. There are now over 1.2 million new articles published every year, averaging to one new article every 26 seconds. Unfortunately, however, the entirety of that knowledge is not easily accessible. In most cases, biomedical knowledge is locked away in free-text research articles, which are very difficult to use for querying and computation. In some cases, that knowledge has been deposited in structured databases, but even then the fragmented landscape of such databases is a barrier to knowledge integration.

Here, we describe the use of Wikidata (wikidata.org) as an open, community-maintained biomedical knowledge base. Wikidata is a sister project of Wikipedia -- both are run by the Wikimedia Foundation and both are entirely populated by community contributions. In contrast to Wikipedia's emphasis on encyclopedic content, Wikidata focuses on organizing structured knowledge. Wikidata currently contains 140 million individual statements on over 25 million items encoded by 1.8 billion triples. Within that set, we have seeded Wikidata with data on key biomedical entities, including genes, proteins, diseases, drugs, genetic variants, and microbes. To ensure source databases are properly credited, we have implemented a standardized model for referencing and attribution. These data, combined with other data sets imported by the broader Wikidata community, enable powerful integrative queries that span multiple domain areas via the Wikidata SPARQL endpoint.

The emphasis of this abstract is on Wikidata as resource for biomedical Open Data that can serve as a foundation for other bioinformatics applications and analyses. In addition, the code developed to execute this project is also available as Open Source software. This suite of code includes modules for populating Wikidata, for automatically synchronizing with source databases, and for creating domain-specific applications to engage specific user communities.

Emerging public databases of clinical genetic test results: Implications for large scale deployment of precision medicine

Stephen Lincoln[1], Shan Yang[1], Benedict Paten[2], Melissa Cline[2], Yuya Kobayashi[1], Can Zhang[2], Scott Topper[1], David Haussler[2], Robert Nussbaum[1,3]

[1] Invitae, San Francisco, California. [2] University of California, Santa Cruz. [3] University of California, San Francisco. Corresponding author email: steve.lincoln@me.com

Implementing precision medicine requires that reliable and consistent clinical interpretations of molecular test results be available to physicians. In germline genetic testing the process of determining which variants in a patient are pathogenic (disease causing) and which are benign is crucial to the clinical utility of these tests. This classification process involves expert review of complex and sometimes conflicting evidence, after which some variants must be considered of uncertain significance. To promote quality control, collaboration and consensus building, many (not all) clinical laboratories contribute classifications to open, public databases, notably ClinVar.

We examined 74,065 classifications of 27,224 clinically observed genetic variants in ClinVar, and found inter-laboratory concordance to be high (96.7%) although this varied considerably among specialties: cancer genes had the highest concordance (98.5%) with cardiology (94.2%) and metabolic disease (95.1%) the lowest. Unsurprisingly, data from research labs were 6-times more likely to disagree with clinical test results, and old classifications often disagreed with newer ones. More interestingly, evidence supporting pathogenicity appears to be more consistently used than evidence against pathogenicity (data to be shown). Low penetrance genes, which confer a modest risk of disease, were 7-times more likely to harbor disagreements compared to high penetrance genes. Technically challenging variant types (e.g. large indels and mutations in low complexity or highly homologous sequences) were under-represented in ClinVar even though independent data from an 80,000 patient cohort shows that 9% of pathogenic variants in patients are of these types.

We further examined variants in BRCA1 and BRCA2, genes which can confer a substantial lifetime risk of breast, ovarian, and other cancers. While BRCA1/2 testing is common it remains controversial, because (a) proposals to implement population scale BRCA1/2 testing as part of routine physical exams have been raised, (b) irreversible preventive options (e.g. prophylactic surgery) are offered to otherwise healthy individuals found to carry germline mutations, and (c) the largest laboratory with data to inform such testing (Myriad) refuses to share that data or provide it for detailed community review. We analyzed physician provided Myriad results along with those from other ClinVar laboratories, finding high agreement (98.5%) in a data set representing over 20,000 tested patients. Moreover, the few discordant variants were all very rare, suggesting that 99.8% of patients would receive concordant tests from any of these labs in routine practice. This confirms a similar result of ours in a prospective 1000-patient clinical trial.

Open data sharing via ClinVar is a powerful mechanism which has already helped to uncover critical factors which must be addressed as new precision medicine approaches evolve in various medical specialties. ClinVar also provides a common mechanism to drive consensus and evaluate inter-laboratory performance, as exemplified by this study.

Note: All data from these studies are publicly available without restriction for analysis by the community. Some of the results described in this abstract are currently in press. Prior published work mentioned includes Lincoln et al. J Mol Diag 2015 and Yang et al, PSB 2016.

Discovering datasets with DATS in DataMed

Alejandra Gonzalez-Beltran[1], Philippe Rocca-Serra[1], Susanna-Assunta Sansone[1], George Alter[2], Jeffrey Grethe[3], Hua Xu[4], Ian Fore[5], Jared Lyle[2], Anupama Gururaj[4], Xiaoling Chen[4], Hyeon-eui Kim[3], Nansu Zong[4], Yueling Li[3], Ruiling Liu[4], Burak Ozyurt[4], Lucila Ohno Machado[3]

[1] University of Oxford, UK. Email: alejandra.gonzalezbeltran@oerc.ox.ac.uk

[2] University of Michigan, USA.

[3] University of California San Diego, USA

[4] The University of Texas Health Science Center at Houston, USA

[5] National Institutes of Health, USA

Project Website: http://biocaddie.org/
Source Code: http://github.com/biocaddie/WG3-MetadataSpecifications

Among the research outputs in the life sciences, datasets are fast becoming the new currency for evaluating research output. As they underlie publications, accessing, mining and integrating them is the way forward for verifying, reproducing results, as well as propelling new discoveries. Thus, traditional literature searches via PubMed need to be complemented by the capability to find and access the related datasets, distributed through an array of generalist or highly specialized data repositories.

DataMed, a prototype, funded by the National Institutes of Health's Big Data to Knowledge initiative, provides a data discovery index, currently indexing over 60 data repositories including a wide range of data types and granularity scales across domains of life sciences. Behind DataMed index, a model known as the DAta Tag Suite (DATS) [1], provides the constructs allowing harmonization of dataset descriptions across repositories. Akin to the Journal Article Tag Suite (JATS) used in PubMed, the DATS model enables the submission of datasets metadata to DataMed. DATS is divided in two layers. DATS core whose elements are generic and applicable to any type of dataset, from any topic and non-specific to the bio-domain. The extended DATS element, on the other hand, can accommodate more advanced use cases and more finely grained needs, befitting the biomedicine domain.

Designed and developed in an entirely open and collaboratively fashion, DATS considered existing well-established schemas and models designed for describing dataset and/or used in biodata repositories. In addition, search cases were compiled. The DATS model is expressed as JSON schemas with schema.org JSON-LD context files. The schema.org model, is widely used by major search engines such as Google, Microsoft, Yahoo and Yandex and therefore caters for relevant indexing use cases. Our mapping of DATS into schema.org identified gaps and has contributed to its extension to better support the biomedical use cases. Work is ongoing to refine and improve ways to implement DATS and optimize information indexing in DataMed.

[1] Pre-preprint of manuscript accepted for publication in 2017 at https://doi.org/10.1101/103143

Bioschemas for life science data

Carole Goble[1], Rafael Jimenez[2], Alasdair Gray[3], Niall Beard[1], Giuseppe Profiti[4], Norman Morrison[2]

[1] The University of Manchester, UK / ELIXIR-UK, Email: carole.goble@manchester.ac.uk

[2] ELIXIR-Hub, Hinxton Genome Campus, UK.

[3] Heriot-Watt University, UK / ELIXIR-UK.

[4] Università di Bologna, Italy / ELIXIR-Italy.

Project Website: http://bioschemas.org/
Source Code: https://github.com/BioSchemas/bioschemas

Abstract

Schema.org provides a way to add semantic markup to web pages. It describes 'types' of information, which then have 'properties'. For example, 'Event' is a type that has properties like 'startDate', 'endDate' and 'description'. Bioschemas aims to improve data interoperability in life sciences by encouraging people in life science to use schema.org markup. This structured information then makes it easier to discover, collate and analyse distributed data. Bioschemas reuses and extends schema.org in a number of ways: defining a minimum information model for the datatype being described using as few concepts as possible and only where necessary adding new properties, and the introduction of cardinalities and controlled vocabularies. The main outcome of Bioschemas is a collection of specifications that provide guidelines to facilitate a more consistent adoption of schema.org markup within the life sciences for the "Find" part of the FAIR (Findable, Accessible, Interoperable, Reusable) principles.

In 2016 Bioschemas successfully piloted with training materials and events to enable the EU ELIXIR Research Infrastructure Training Portal (TeSS) to rapidly and simply harvest metadata from community sites. Encouraged by this in March 2017 we launched a 12 month project to pilot Bioschemas for data repositories and datasets. Specifically we are working on:

- General descriptions for datasets and data repositories
- Specific descriptions for prioritised datatypes: Samples, Human Beacons, Plant phenotypes and Protein annotations
- Facilitating discovery by registries and data aggregators, and by general search engines
- Facilitate tool development for annotation and validation of compliant resources

All work is grounded on describing real data resources for real use cases: to this end large and small dataset are part of the project: Pfam, Interpro, PDBe, UniProt, BRENDA, EGA, COPaKB, and Gene3D. Data aggregators participating include: InterMine, BioSamples and OmicsDI. Registries include Identifiers.org, DataMed, Biosharing and the Beacon Network. Bioschemas operates as an open community initiative, sponsored by the EU ELIXIR Research Infrastructure and is supported by the NIH BD2K programme and Google.

Introducing the Brassica Information Portal: Towards integrating genotypic and phenotypic Brassica crop data

Annemarie Eckes[1], Tomasz Gubala[1,6], Piotr Nowakowski[1,6], Tomasz Szymczyszyn[1], Rachel Wells[2], Judith Irwin[2], Carlos Horro[1], John Hancock[1], Graham King[4], **Wiktor Jurkowski[1,*]**

[1]Earlham Institute, Norwich Research Park, Norwich NR4 7UZ, UK
[2]Academic Computer Centre CYFRONET, AGH University of Science and Technology, Kraków, Poland
[3]John Innes Centre, Norwich Research Park, Norwich, NR4 7UH, UK
[4]Southern Cross Plant Science, Southern Cross University, Lismore, NSW 2480, Australia
[*] correspondence to: wiktor.jurkowski@earlham.ac.uk

A new community resource, the Brassica Information Portal (BIP), provides an open access web repository for *Brassica* phenotyping data (https://bip.earlham.ac.uk). It facilitates crop improvement by filling the gap for standardised trait data and is already integrated with tools to perform on-the-fly phenotype-genotype association analysis online.

In collaboration with the UK *Brassica* Research Community, we have build a tool that aims to assist global research on *Brassica* crops: Users can store and publish their own study results in the Portal thanks to advanced data submission capabilities. Similarly, user-friendly interfaces and programmatic download mechanisms further facilitate work with the data. This makes the database content readily comparable and cross-linkable to that of other tools and resources for further downstream analysis. Integrated with the GWAS analysis tool GWASSER, it now enables the user to perform simple GWAS analysis with selected data on BIP.

We also aim to implement community-wide phenotypic trait definition standards to encourage trait data re-use. To ensure comparability, we are working on creating brassica trait ontology terms based on the Crop Ontology model suitable for association analysis. This creates the opportunity to carry out meta-analysis on datasets generated across multiple studies. To make BIP data more comparable and reusable, we are collaborating with ELIXIR to implement MIAPPE standards to the BIP schema. To make data more interoperable and accessible, we are working on being BrAPI (Breeding API) - compliant, as well as on extension of our own BIP-API.

There are huge benefits from making organised trait data available and accessible to the *Brassica* community and we hope that it becomes common practice to store brassica trait data systematically and share it with the wider research community in a similar way as has become routine for sequence, transcript, metabolite and protein structure data.

We will present the resource and a GWAS use-case to demonstrate the value and utility of BIP, both for single research groups and global *Brassica* research and breeding community.

Source code: https://github.com/TGAC/brassica
Open Source License: GNU General Public License 3.0

Poster Abstracts

BOSC 2017 Posters

Poster	Authors	Title
A-107	Ismail Moghul, Suresh Hewapathirana, Nazrath Nawaz, Anisatu Rashid, Marian Priebe, Bruno Vieira, Fabrizio Smeraldi and Conrad Bessant	GeoDiver: Differential Gene Expression Analysis & Gene-Set Analysis for GEO Datasets
A-108	Alberto Riva, Richard L Bennett and Jonathan D Licht	DAMON, an open source framework for reliable and reproducible analysis pipelines
A-109	Uros Sipetic	Workflow for processing standard bioinformatics formats with SciClone to infer tumor heterogeneity
A-110	Nikola Tesic and Marko Kalinic	Microsatellite instability profiling of TCGA colorectal adenocarcinomas using a Common Workflow Language pipeline
A-111	Monica-Andreea Drăgan, Ismail Moghul, Anurag Priyam, Claudio Bustos and Yannick Wurm	GeneValidator: identify problems with protein-coding gene predictions
A-112	Sanja Mijalkovic, Milan Domazet	Somatic Variant Calling Benchmarking
A-114	Wei Wang, Yichao Zhou, Patrick Tan, Vincent Kyi, Xinxin Huang, Chelsea Ju, Justin Wood and Peipei Ping	Aztec: Automated Biomedical Tool Index with Improved Information Retrieval System
A-115	Malvika Sharan, Toby Hodges, Julia Ritzerfeld and Georg Zeller	Building a local bioinformatics community: challenges and efforts
A-116	Aleksandra Gruca	CHARME - harmonising standardisation strategies to increase efficiency and competitiveness of European life-science research
A-117	Vincenzo Belcastro, Stephane Cano, Diego Marescotti, Carine Poussin, Ignacio Gonzales-Suarez, Florian Martin, Filipe Bonjour, Nikolai Ivanov, Julia Hoeng	High Content Screening data storage and analysis platform - An open source solution
A-118	Laura Huerta, Elisabet Barrera, Wojciech Bazant, Nuno A. Fonseca, Anja Fullgrabe, Maria Keays, Suhaib Mohammed, Alfonso Munoz-Pomer Fuentes, Amy Tang, Irene Papatheodorou, Robert Petryszak, Ugis Sarkans and Alvis Brazma	Expression Atlas: exploring gene expression results across species under different biological conditions
A-119	Jorge Miguel Martins Boucas	Microservices in data, code, and project management.
A-120	Pieter Lukasse, Fedde Schaeffer, Oleguer Plantalech Casals, Sander Tan and Sjoerd van Hagen	Gene Set Variation Analysis in cBioPortal

A-121	Tazro Ohta	Databases to support reanalysis of public high-throughput DNA sequencing data
A-122	Chunlei Wu, Cyrus Afrasiabi, Sebastien Lelong, Jiwen Xin, Ginger Tsueng and Andrew I. Su	BioThings SDK: a toolkit for building high-performance data APIs in biology
A-123	Evanthia Kaimaklioti, Robert P Davey and Ian Mulvany	Reproducing computational experiments in situ as an interactive figure in a journal article
A-124	Mark Robinson, Stian Soiland-Reyes, Michael Crusoe and Carole Goble	CWL Viewer: The Common Workflow Language Viewer
A-125	Thomas Cokelaer, Dimitri Desvillechabrol, Rachel Legendre and Mélissa Cardon	Sequanix: a standalone application to expose Snakemake pipelines to end-users
A-126	Rickard Hammarén, Philip Ewels and Max Käller	NGI-RNAseq - a best practice analysis pipeline in Nextflow
A-127	Ngoc-Vinh Tran, Bastian Greshake and Ingo Ebersberger	PhyloProfile: an interactive and dynamic visualization tool for multi-layered phylogenetic profiles
A-128	Nikita Moshkov, Daria Iakovishina and Dimitri Nikogosov	CueSea: quality control tool for Illumina genotyping microarray data, with correction on intensity, clusterization and biological specificity.
A-129	Rishi Nag	Bio::DB::HTS – accessing HTSlib from Perl
A-130	Luis Pedro Coelho, Paulo Monteiro, Renato Alves, A.T. Freitas and Peer Bork	NGLESS: Perfectly understandable and reproducible metagenomics pipelines using a domain-specific language
A-131	Keiran Raine, Adam Butler, Peter Clapham, Jon Teague, Peter Campbell	CGP as a Service (CGPaaS) - From data submission to results using your web-browser
A-132	Lacey-Anne Sanderson, Reynold Tan, Carolyn Caron, Kirstin Bett	Large-scale genotypic and phenotypic data support for Tripal: Chado optimization by utilizing modern PostgreSQL functionality
A-133	Michael Reich, Thorin Tabor, Helga Thorvaldsdóttir, Barbara Hill, Ted Liefeld, Jill Mesirov, Pablo Tamayo	GenePattern Notebooks: An integrative analytical environment for genomics research
A-134	John Fonner, Rion Dooley, Jacquelyn Turcinovic and Matthew Vaughn	BioContainers for supercomputers: 2,000+ accessible, discoverable Singularity apps

A-135	Felix Shaw, Anthony Etuk, Rob Davey, Alejandra Gonzalez-Beltran, David Johnson, Phillipe Rocca-Serra and Susanna Sansone	Collaborative Open Plant Omics: A platform for “FAIR” data for plant science
A-136	Farah Khan, Andrew Lonie, Richard Sinnott and Stian Soiland-Reyes	CWL+Research Object == Complete Provenance
A-137	Steffen Moeller	Workflow-ready bioinformatics packages for Debian-based distributions and this Linux distribution’s infrastructure for low-friction reproducible research
A-138	Yo Yehudi, Daniela Butano, Matthew Chadwick, Justin Clark-Casey, Sergio Contrino, Josh Heimbach, Rachel Lyne, Julie Sullivan and Gos Micklem	Forever in BlueGenes: a next-generation genomic data interface powered by InterMine
A-139	Silvia Di Giorgio and Konrad Förstner Dr.	GRADitude: A computational tool for the analysis of Grad-seq data
A-140	Stefan Popa and Radu Dobrescu	Enabling the optimization of open-source biological computational tools with scripting languages
A-141	Rabie Saidi, Hanna Papkova, Tunca Dogan and Maria Martin	Protein Inpainter: a Message-Passing-based Predictor using Spark GraphX
A-142	Ricardo Wurmus and Altuna Akalin	Reproducible and user-controlled software management in HPC with GNU Guix
A-143	Monther Alhamdoosh, Milica Ng and Matthew Ritchie	An ensemble approach for gene set testing analysis with reporting capabilities
A-144	Julia Kurps, Maxim Moinat, Joris Borgdorff, Fanscesco Nobilia, Maximilian Kerz, Nivethika Mahasivam, Irina Pulyakhina, Matthias Dümpelmann, Herculano Campos, Mark Begale, Richard Dobson and Amos Folarin	RADAR-CNS - Research Infrastructure for processing wearable data to improve health
A-145	Kevin Sayers, Paolo Di Tommaso, Maria Chatzou, Evan Floden and Cedric Notredame	Workflows interoperability with Nextflow and Common WL
A-146	Carole Goble, Rafael Jimenez, Alasdair Gray, Niall Beard, Giuseppe Profiti and Norman Morrison	Bioschemas for life science data
A-147	Denis Yuen, Brian O'Connor, Andrew Duncan, Solomon Shorser, Vincent Chung, Xiang Kun Liu, Janice Patricia, Han Yuan Cao, Gary Luu, Vincent Ferretti and Lincoln Stein	The GA4GH Tool Registry Service (TRS) and Dockstore - Year One

A-148	Annemarie H. Eckes, Tomasz Gubała, Piotr Nowakowski, Tomasz Szymczyszyn, Rachel Wells, Judith A. Irwin, Carlos Horro, John M. Hancock, Graham King, Sarah C. Dyer and Wiktor Jurkowski	Introducing the Brassica Information Portal: Towards integrating genotypic and phenotypic Brassica crop data
A-149	Anil S. Thanki, Nicola Soranzo, Wilfried Haerty and Robert P. Davey	Discovery and visualisation of homologous genes and gene families using Galaxy
A-150	Olga Vrousgou, Simon Jupp, Thomas Liener, Tony Burdett and Helen Parkinson	The SPOT ontology toolkit : semantics as a service
A-151	Aditya Bharadwaj, Divit P. Singh, Anna Ritz, Allison N. Tegge, Christopher L. Poirel, Pavel Kraikivski, Neil Adames, Kurt Luther, Shiv D. Kale, Jean Peccoud, John J. Tyson and T. M. Murali	GRAPHSPACE: Stimulating interdisciplinary collaborations in network biology
A-152	Keiichiro Ono, Eric Sage and Barry Demchak	Revitalizing a classic bioinformatics tool using modern technologies: the case of the Cytoscape Project
A-153	Kieran O'Neill, Benjamin Decato, Alexander Goncearenco, Azhar Khandekar, Benjamin Busby and Aly Karsan	Screw: tools for building reproducible single-cell epigenomics workflows
A-154	Stephen Lincoln, Shan Yang, Benedict Paten, Melissa Cline, Yuya Kobayashi, Scott Topper, Can Zhang, David Haussler and Robert Nussbaum	Emerging public databases of clinical genetic test results: Implications for large scale deployment of precision medicine
A-155	Alexander S. Rose and Stephen K. Burley	NGL – a molecular graphics library for the web
A-156	Kenzo-Hugo Hillion, Ivan Kuzmin, Hedi Peterson, Jon Ison and Hervé Ménager	ToolDog - generating tool descriptors from the ELIXIR tool registry
A-157	Ted Liefeld, Marco Ocana, Michael Reich, Helga Thorvaldsdottir and Jill P Mesirov	Integrating cloud storage providers for genomic analyses
A-159	Kate Voss, Jeff Gentry and Geraldine Van der Auwera	Full-stack genomics pipelining with GATK4 + WDL + Cromwell
A-160	Matúš Kalaš, Sveinung Gundersen, László Kaján, Hervé Ménager, Jon Ison, Steve Pettifer, Christophe Blanchet, Rodrigo Lopez, Kristoffer Rapacki and Inge Jonassen	BioXSD | BioJSON | BioYAML - Towards unified formats for sequences, alignments, features, and annotations
A-161	Matúš Kalaš, Hervé Ménager, Veit Schwämmle, Jon Ison and Edam Contributors	EDAM - The ontology of bioinformatics operations, types of data, topics, and data formats (2017 lightning update)

A-107

Geodiver: Differential Gene Expression Analysis & Gene-Set Analysis for GEO Datasets

Ismail Moghul[1,2], Suresh Hewapathirana[1], Nazrath Nawaz[1], Anisatu Rashid[1], Marian Priebe[1], Bruno Vieira[1], Fabrizio Smeraldi[3] and Conrad Bessant[1]

[1]School of Biological and Chemical Sciences, Queen Mary University of London, UK; [2]UCL Cancer Institute, University College London, London, WC1E 6DD, UK; [3]School of Electronic Engineering and Computer Science, Queen Mary University of London, UK

Website: http://www.geodiver.co.uk

Source Code: https://github.com/geodiver/geodiver

Abstract

The growing genomic data repository GEO (Gene Expression Omnibus) has become the popular public database for microarray data from gene expression experiments. GeoDiver is an online web application (http://www.geodiver.co.uk) for performing Differential Gene Expression Analysis (DGEA) and Generally Applicable Gene-set Enrichment Analysis (GAGE) on GEO datasets. GeoDiver allows researchers to fully take advantage of the growing GEO resource and perform powerful analyses without the need for downloading or installing additional software, learning command-line skills or having prior programming knowledge. Users can easily run both DGEA and GSA within a few minutes of accessing the web application. The output produced includes numerous high-quality interactive graphics, allowing users to easily explore and examine complex datasets instantly and understand the data they have analysed (see Figure 1). Furthermore, the results produced can be reviewed at a later date and shared via a URL. GeoDiver is therefore not only designed to facilitate the analysis of gene-expression data but also ensures that users can fully explore the results of their analysis. This is important as the ability to use such powerful analytical tools has to be paired with the corresponding ability to interpret the output.

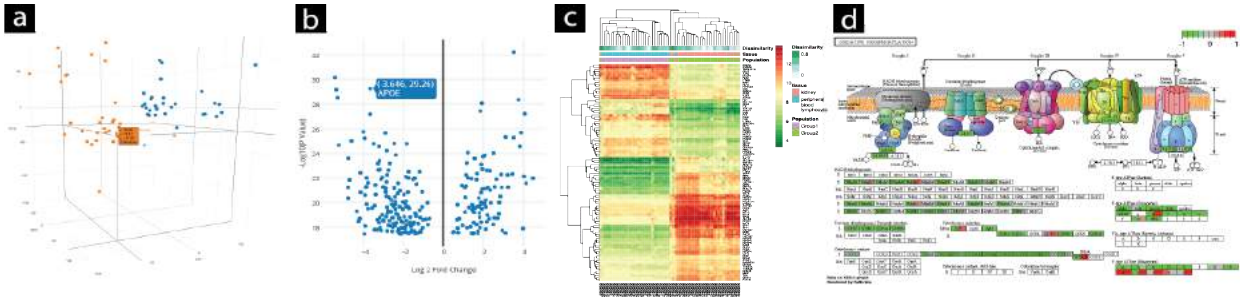


Figure 1: Exemplar GeoDiver Visualisations: a) An Interactive 3D PCA plot showing data points for the first three components, b) an interactive volcano plot showing significant genes c) a heatmap of the top differentially expressed genes and d) a colour-coded interaction network (oxidative phosphorylation). The above exemplar visualisations have been produced with GEO dataset GDS724, comparing Peripheral Blood Lymphocyte (as Group A) and Kidney (as Group B) expression levels.

DAMON, an open source framework for reliable and reproducible analysis pipelines

Alberto Riva[1], Richard L. Bennett[2], Jonathan D. Licht[2]
ariva@ufl.edu
[1]ICBR Bioinformatics Core, University of Florida, Gainesville, FL, USA
[2]UF Health Cancer Center, University of Florida, Gainesville, FL, USA

Source code: https://github.com/albertoriva/damon
License: MIT

DAMON (Distributed Analysis MONitor) is an open-source, object-oriented Python framework for the development of computational pipelines, specifically designed for NGS data analysis in a cluster computing environment. Pipelines are built out of reusable objects implementing analysis steps (e.g. short-read alignment, transcript quantification) and are controlled by simple configuration files detailing the experimental setup, the input data, and the steps to be performed. DAMON is able to handle any number of experimental conditions, biological replicates, and technical replicates, easily supporting complex experimental designs with no changes to the pipeline structure.

Pipeline execution is controlled by a *Director* object. Given an *Actor* object representing the pipeline, the Director performs the necessary setup and executes all required steps. Each step is represented by a *Line* object, which provides standard methods for Setup, Verification, Pre-Execution, Execution, Post-Execution, and Reporting. Thanks to this standard API, steps can be freely combined: for example, changing the short-read aligner from Bowtie to STAR only requires swapping one Line object for another in the pipeline definition. The framework's high modularity makes it easy to build new pipelines or extend existing ones by simply defining new Line objects. This increases reusability of modules, reducing the potential for coding errors and increasing analysis reproducibility.

DAMON automatically handles submission and management of jobs to the cluster, ensuring proper job sequencing and coordination. It supports both slurm and PBS environments through the use of a custom job submission script. Finally, DAMON pipelines automatically generate an HTML report of their execution: within a standard template (customizable by specializing the Actor object) each step may add one or more sections containing text, tables, figures, links to downloadable files. Reports and downlodable files can be automatically packaged in a compressed file for easy download and distribution.

We describe the DAMON framework discussing its advantages in terms of analysis reliability and reproducibility, and we outline pipelines developed with it (including RNA-seq, ChIP-seq, ATAC-seq, methylation analysis, variant discovery, genome annotation) that are currently in routine use by our group.

A-109

Workflow for processing standard bioinformatics formats with SciClone to infer tumor heterogeneity

Uros Sipetic

Tumor heterogeneity refers to the notion of tumors showing distinct cellular morphological and phenotypic properties which can affect disease progression and response. Next-gen sequencing results from bulk whole exome or whole genome samples can be expanded to infer tumor structure (heterogeneity) using a few different methods. One of these methods, sciClone, was developed for detecting tumor subclones by clustering Variant Allele Frequencies (VAFs) of somatic mutations in copy number neutral regions. Presented in the poster is a portable and reproducible Common Workflow Language (CWL) implementation of a sciClone based tumor heterogeneity workflow. The results of sciClone can be directly fed into additional software like ClonEvol and Fishplot, which produce additional visualizations, e.g. phylogenetic trees showing clonal evolution. The main idea of the workflow was to build a pipeline that can process standard bioinformatics file types (VCF, BAM) and produce a comprehensive set of outputs describing tumor heterogeneity. Additionally, the workflow was designed to be robust regardless of the number of samples provided.

Microsatellite instability profiling of TCGA colorectal adenocarcinomas using a Common Workflow Language pipeline

Nikola Tešić, Marko Kalinić

May 2017

Microsatellite instability (MSI) is an important factor for classifying certain cancer types, it has been shown to be associated with prognosis, and it can be used as a biomarker for treatment selection. Here, we present a portable and reproducible Common Workflow Language (CWL) implementation of *MSIsensor* tool and *lobSTR* toolkit, we will show how to use *MSIsensor* and *lobSTR*, and we will see how *lobSTR* deals with PCR noise and how it can silence it. Finally, we present results obtained using the aforementioned tools on more than 600 TCGA colorectal adenocarcinoma cases, as well as our further analysis and interpretation of the results. Out of 592 cases that had MSI classification on GDC portal, *MSIsensor* predicted the MSI status correctly for 580 of them with F-score 0.93 using tumor and normal samples, while results obtained using *lobSTR* enabled us to correctly predict the MSI status in 584 cases with F-score 0.96 using tumor and normal samples, and 581 cases with F-score 0.94 using tumor only.

GeneValidator: identify problems with protein-coding gene predictions

Monica-Andreea Drăgan[1], Ismail Moghul[2], Anurag Priyam[2], Claudio Bustos[3], Yannick Wurm[2,*]

[1] Department of Computer Science, ETH Zürich

[2] School of Biological and Chemical Sciences, Queen Mary University of London, UK

[3] Departamento de Psiquiatría y Salud Mental, University of Concepción, Chile

[*] Contact: y.wurm@qmul.ac.uk

Project Website: https://wurmlab.github.io/tools/genevalidator
Source Code: https://github.com/wurmlab/genevalidator
License: AGPL

Genomes of emerging model organisms are now being sequenced at very low cost. However, obtaining accurate gene predictions remains challenging: even the best gene prediction algorithms make substantial errors and can jeopardize subsequent analyses. Therefore, many predicted genes must be time-consumingly visually inspected and manually curated. We developed GeneValidator to automatically identify problematic gene predictions and to aid manual curation. For each gene, GeneValidator performs multiple analyses based on comparisons to gene sequences from large databases. The resulting report identifies problematic gene predictions and includes several statistics and graphs for each prediction. These can be used to eliminate problematic gene models from a set of annotations, compare two sets of annotations, or to guide manual curation efforts. GeneValidator thus accelerates and enhances the work of researchers and biocurators who need accurate gene predictions from newly sequenced genomes.

GeneValidator can be used through a web interface or in the command-line. It is open-source (AGPL), available at https://wurmlab.github.io/tools/genevalidator.

Somatic Variant Calling Benchmarking

- Sanja Mijalkovic, Seven Bridges Genomics, Serbia
- **Milan Domazet, Seven Bridges Genomics, Serbia**

Short Abstract: The increased number of sequenced cancer genomes since the completion of the human genome project, and the importance of correctly identifying somatic mutations, which can influence treatment or prognosis, is driving forward the development of novel somatic variant calling tools (somatic callers). A lack of best practices algorithm for identifying somatic variants, however, requires constant testing, comparing and benchmarking these tools. The absence of truth set further hinders the effort for the evaluation. By comparing widely used open source somatic callers, such as Strelka, VarDict, VarScan2, Seurat and LoFreq, through analysis of in-house generated synthetic data, we found complex dependencies of somatic caller parameters relative to coverage depth, allele frequency, variant type, and detection goals. Next, we normalised and filtered the output data such that it can be appropriately compared to the truth set. The acquired benchmarking results were automatically and efficiently structured and stored. All of the tools used for the analysis have been implemented in Common Workflow Language which makes them portable and reproducible.

Aztec: Automated Biomedical Tool Index with Improved Information Retrieval System

Wei Wang, Yichao Zhou, Patrick Tan, Vincent Kyi, Xinxin Huang, Chelsea Ju, Justin Wood and Peipei Ping

Advances in bioinformatics, especially in the field of genomics, have greatly accelerated as development of powerful computational systems and new distributed platforms enabled rapid processing of massive amounts of datasets. Accordingly, this has led to an explosion of public software, databases, and other resources for biological discovery.

Modern biomedical research requires the comprehension and integration of multiple types of data and tools; specifically, understanding biomedical phenotypes requires analysis of molecular data (genomics, proteomics), imaging data (sonography; computed tomography, CT), and textual data (case reports, electronic health records). Researchers require software tools in a common platform that can analyze and integrate data from each of these domains. However, many of the existing resource and tool repositories are narrowly focused, fragmented, or scattered and do not support the multidimensional data analysis required for advancing precision medicine. There is a lack of unified platform collection software applications across various disciplines, leaving many users unable to locate the tools critical to their research.

To manage this influx of digital research objects (tools and databases), the FORCE11 community put forth a set of guiding principles to make data Findable, Accessible, Interoperable, and Reusable (FAIR) (Wilkinson et al., 2016). While these issues have been addressed in the data elements domain, they are equally important to but thus far neglected by the biomedical informatics software domain.

Here, we introduce Aztec (**A to Z technology;** https://aztec.bio), an open-source online software discovery platform that empowers biomedical researchers to find the software tools they need. Aztec currently hosts over 10,000 resources, spanning 17 domains including imaging, gene ontology, text mining, data visualization, and various omic analyses (i.e., genomics, proteomics, and metabolomics). Aztec serves as a common portal for biomedical software across all of these domains, enabling modern multi-dimensional characterization of biomedical processes.

Building a local bioinformatics community: challenges and efforts

Malvika Sharan[1,3,*], Toby Hodges[1], Julia Ritzerfeld[2,3], Georg Zeller[1,3]

[1] *European Molecular Biology Laboratory (EMBL)*, Heidelberg, Germany
[2] *German Cancer Research Institute (DKFZ),* Heidelberg, Germany
[3] *Heidelberg Center for Human Bioinformatics (HD-HuB)* within the *German Network for Bioinformatics Infrastructure (de.NBI),* hosted by the *University of Heidelberg*, Germany
[*] **Presenting author's email**: malvika.sharan@embl.de

Project Websites:
HD-HuB: https://www.hd-hub.de/
EMBL Bio-IT:
https://www.embl.de/research/interdisciplinary_research/bioinformatics/community/bio-it/
de.NBI: https://www.denbi.de/
Bio-IT de.NBI event page: https://www.hd-hub.de/de-nbier-technical-sessions

Main Text for Abstract:

The German Network for Bioinformatics Infrastructure (de.NBI) is a national network funded by the Federal Ministry of Education and Research. de.NBI provides bioinformatics services to life-science and biomedicine researchers, constituting at the same time the German ELIXIR node. Through one of its service centers, the Heidelberg Center for Human Bioinformatics (HD-HuB), we host a variety of bio-computational resources and offer courses and training opportunities to scientists in a wide range of topics. Additionally, we organize events that provide a platform to assist researchers in the Heidelberg area to network with each other and exchange their knowledge. HD-HuB unites the bioinformatics expertise of three distinguished research institutions: the European Molecular Biology Laboratory (EMBL), the German Cancer Research Center (DKFZ), and Heidelberg University, and facilitates an environment for establishing an effective and efficient bioinformatics community in Heidelberg.

EMBL Heidelberg provides a local bioinformatics support platform, Bio-IT, to their ~600 life-scientists across >50 groups. EMBL has reported a consistent increase in the fraction of scientists who devote ≥50% of their time to computational-based research, seeing a growth from 39% to 46% between 2009 and 2015. Bio-IT is an initiative established to support the development and technical capability of this diverse bio-computational community. The community has grown organically to tackle challenges together, providing training and support for scientific computing, creating and maintaining tools and resources for reproducible science, and providing opportunities for discussion and collaboration.

Here, we share some lessons learned while establishing these communities and discuss the efforts required to maintain it thereafter.

CHARME - harmonising standardisation strategies to increase efficiency and competitiveness of European life-science research

Aleksandra Gruca [1,2]
[1]*EU COST Action CHARME CA15110*
[2]*Institute of Informatics, Silesian University of Technology, Gliwice, Poland*
aleksandra.gruca@polsl.pl

In the last few years, the issues related to reproducibility problems in life sciences and biomedical research are gaining a lot of interest and traction in the scientific community. These problems are not limited to laboratory procedures and protocols but are also related to computational analysis of experimental data. Results of computational findings often cannot be reproduced because of problems associated with software versioning, packaging, installation and execution, and because of lack of proper documentation.

By enabling the reuse of research assets, research becomes considerably more efficient and economical. This can only be achieved reliably and efficiently if these are generated according to standards and Standard-Operating-Procedures (SOPs). Thus, standards represent important drivers in the life sciences because they guarantee that data become accessible, shareable and comparable along the value chain. Several initiatives launched the development and implementation of standards. Unfortunately these efforts remain fragmented and largely disconnected, therefore efficient exchange of data and results is often not possible or is at least hindered.

The COST [1] Action CHARME (harmonising standardisation strategies to increase efficiency and competitiveness of European life-science research) is focused on the problems of standardization in systems biology and computational modelling. Currently 29 countries are involved in this non-commercial initiative, which makes it one of the most widely recognized actions among all COST initiatives. CHARME consists of five interactively working groups: (i) community/platform-building, (ii) innovation transfer, (iiI) design, implementation, and incorporation of software solutions, (iv) development of a strategic dissemination and (v) educational advertising.

The main objective of CHARME is to create a common platform for sustainable and efficient cooperation on standardization. The CHARME aims to bridge and combine the fragmented areas of the various moves in the development of new norms and standards with a particular reference to systems biology and computational modelling. This will be done by identifying needs and gaps, and teaming up with other initiatives to avoid duplication and overlap of standardisation activities. The CHARME efforts are focused on increasing the awareness for the need of standards, promoting FAIR data, and data stewardship in scientific community. By active involvement of all stakeholders (from research, industry and policy), a common, long-term strategy will be developed to successfully assimilate standards into the daily workflow. Finally, CHARME will provide a common ground for researchers from academia, research institutes, SMEs and multinational organizations. Your contribution participation is welcome. Please contact us via our website http://cost-charme.eu/.

[1] (COoperation in Science and Technology)

High Content Screening data storage and analysis platform - An open source solution

- **Vincenzo Belcastro, Philip Morris International, Switzerland**
- Stephane Cano, Philip Morris International, Switzerland
- Diego Marescotti, Philip Morris International, Switzerland
- Carine Poussin, Philip Morris International, Switzerland
- Ignacio Gonzales-Suarez, Philip Morris International, Switzerland
- Florian Martin, Philip Morris International, Switzerland
- Filipe Bonjour, Philip Morris International, Switzerland
- Nikolai Ivanov, Philip Morris International, Switzerland
- Julia Hoeng, Philip Morris International, Switzerland

Short Abstract: High Content Screening (HCS) is a powerful tool that can quantify a large number of biological readouts in a short period of time. At Philip Morris International (PMI), HCS is routinely used as part of the toxicological risk assessment of products with the potential to present less risk of harm than continued smoking. Each HCS-based assay is a complex, multi-step procedure that starts with an adequate experimental design, followed by the data generation phase (exposure, image acquisition and quantification) and finally data analysis and reporting. Testing of various endpoints in combination with multiple experimental conditions can easily result in hundreds of plates that need to be managed. Therefore, the use of a self-contained platform becomes necessary in order to ease the process. Here, we present an open source platform that implements a Graphical User Interface (GUI) to manage the entire HCS procedure. The platform includes (i) a GUI to design an HCS experiment; (ii) an automatic distribution of samples on plates; (iii) automatic fetching of raw images quantifications (the platform implements a web-service layer that allows multiple (proprietary) systems to be easily integrated); (iv) data analysis functionalities via pmitcpl R package that include multiple dose-response fitting and best fit selection, various statistics (e.g., Min Effective Concentration and Max Tolerated Dose) and historical data analysis. The HCS data storage and analysis platform represents an all-in-one, open source solution for HCS experimental processes. The platform is suitable for the overall management (storage, processing and reporting) of High Content Screening experimental data in BioMedical Research, thereby increasing the consistency, reliability, flexibility, speed of interpretation and traceability of the HCS data flow.

Expression Atlas: exploring gene expression results across species under different biological conditions

Laura Huerta, Elisabet Barrera, Wojciech Bażant, Nuno A. Fonseca, Anja Fullgrabe, Maria Keays, Suhaib Mohammed, Alfonso Munoz-Pomer, Amy Tang, Irene Papatheodorou, Robert Petryszak, Ugis Sarkans, Alvis Brazma

EMBL-EBI, Wellcome Genome Campus, Hinxton, Cambridgeshire, CB10 1SD, UK
E-mail: lauhuema@ebi.ac.uk

Project Website: https://www.ebi.ac.uk/gxa
Source code: https://github.com/gxa/gxa
https://github.com/nunofonseca/irap/
License: Expression Atlas Licence (http://www.ebi.ac.uk/gxa/licence.html)

Expression Atlas is a database and web-service at EMBL-EBI that curates, re-analyses and displays gene expression data across species and biological conditions such as different tissues, cell types, developmental stages and diseases among others. Currently, it provides gene expression results on more than 3,000 experiments (based on microarray and RNA-sequencing) from 40 different organisms, including metazoans and plants. Expression profiles of tissues from Human Protein Atlas, GTEx and FANTOM5, and of cancer cell lines from ENCODE, CCLE and Genentech datasets can be explored in Expression Atlas. All experiments within Expression Atlas are manually curated to accurately represent gene expression data, and annotated to Experimental Factor Ontology (EFO) terms. The use of EFO annotations allows efficient search via ontology-driven query expansion and facilitates data integration. All RNA-sequencing experiments are analysed in a uniform way using iRAP, our standardised pipeline. It is possible to search and download datasets into R for further analysis via the R package 'ExpressionAtlas'.

Expression Atlas visualises gene expression results using heatmaps showing gene expression levels across different biological conditions. Novel analysis and visualisations include: exploration of gene co-expression across tissues, cell lines or other conditions within a baseline experiment and enrichment analysis of user provided sets of genes in a given organism against differentially expressed genes in all available comparisons.

Finally, our baseline gene expression results in different tissues are integrated within other specialised resources to contribute to the understanding of pathways (Reactome, http://www.reactome.org/) and disease targets (Target Validation Platform by Open Targets, https://www.opentargets.org/).

A-119

Microservices in data, code, and project management.

In the increasing world of data, the mingles of data, code, and project management have become crucial for a successful and agile processing of data into meaning.

While several proficient tools have emerged along the last years for each of the individual tasks, a unified tool is still in the need for many.

Avoiding the costs of developing and deploying an unified tool, using existing tools in a microservices architecture is at the reach of any, allows refactoring and is infrastructure independent.

bit is a git-based tool for the management of code and data. It uses git for code versioning and for example ownCloud for storing and exchanging data. It avoids the costs of data versioning by logging in git the changes made on the data storage platform (eg. ownCloud).

Data can be transferred over curl to avoid the risks of mounted drives and the need for ftp access. Every time data is uploaded to ownClowd, the upload is registered on the respective wiki on github together with an hyperlink to the respective folder. Project metadata and data can therefore be kept in different instances.

Using rsync in the backend, it allows seamless multiple rsync calls for efficient synchronisation of projects between different HPCs or local machines.

It is primarily developed for multiple analysts working on shared projects on an HPC cluster but it can be easily used on local/cloud/non-HPC machines without changes to the code.

A-120

Gene Set Variation Analysis in cBioPortal

Pieter Lukasse[1], Fedde Schaeffer[1], Oleguer Plantalech Casals[1], Sander Tan[1], Sjoerd van Hagen[1]

[1] The Hyve, Arthur van Schendelstraat 650, 3511 MJ Utrecht, The Netherlands. Email: office@thehyve.nl

[1] The Hyve, Cambridge Innovation Center, 1 Broadway, 14th Floor, Cambridge, MA 02142, USA

Project Website: http://www.cbioportal.org
Source Code: https://github.com/cbioportal
License: GNU Affero General Public License v3

Abstract

cBioPortal is an open source application for interactive analysis and visualization of large scale cancer genomics datasets, originally developed by Memorial Sloan Kettering Cancer Center, New York, and since 2015 by a larger community including The Hyve, The Netherlands. Here we present the recent development we (The Hyve) did for supporting Gene Set Variation Analysis (GSVA) in cBioPortal, across the different views. We show how the new GSVA data is displayed in the oncoprint, with support for hierarchical clustering, and how it can be used in a variety of other plots and analyses.

This functionality is new in cBioPortal and could be useful to any of the users of this popular open source cancer genomics platform. It enables a new dimension of exploratory analysis in cBioPortal, allowing researchers to search and find patterns at the level of molecular processes where multiple genes that are known to work in concert, instead of exploring the data at the level of individual genes.

In our presentation we aim to share the details of this important update to the cBioPortal platform, also highlighting the way we worked with a large Pharma customer and other cBioPortal developers to implement this new feature.

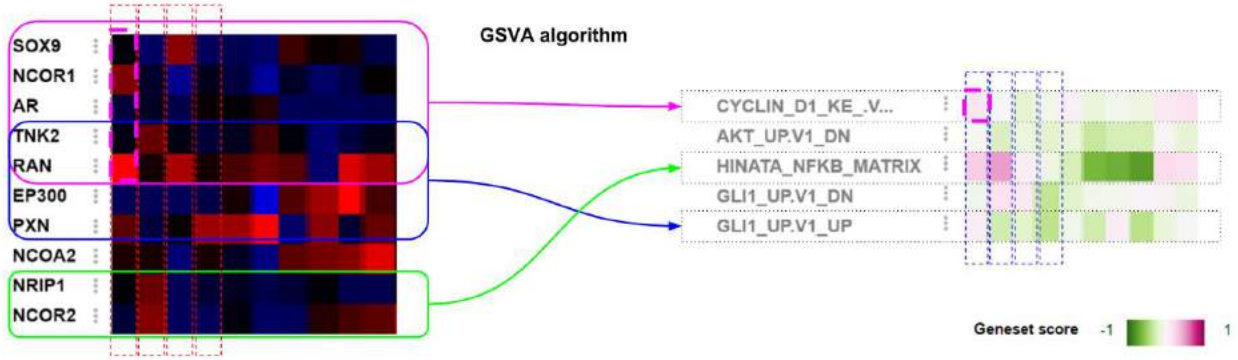


Figure 1: overview of GSVA algorithm. The goal is to highlight each gene set / sample combinations where the **genes in the given gene set** are coordinately up- or down-regulated.

Databases to support reanalysis of public high-throughput data

Tazro Ohta
Database Center for Life Science (DBCLS), Research Organization of Information and Systems (ROIS), Japan

The advance of the high-throughput DNA sequencing technologies is producing a tremendous amount of primary sequencing data which are going to be submitted to the public data repository such as the Sequence Read Archive. The cost of data archive, however, is not decreasing for years in comparison to that of data production [1]. Thus, it is necessary to organize archived data so that the suitable archiving strategy such as the different compression levels is applied to the data.

In this presentation, we would like to introduce two projects which aim to provide additional data and information calculated based on the public high-throughput sequencing data. The Quanto project is to provide quality information of public high-throughput DNA sequencing data by calculating sequencing quality. The repository users can use the quality information such as number of reads or base call accuracy to obtain suitable subset satisfying one's quality requirement. The summary of calculated sequencing quality has published via the FigShare [2] and the GigaDB [3]. Another project called ChIP-Atlas (http://chip-atlas.org) is providing the database which provides re-analyzed data of public ChIP-Seq and DNase-Seq experiments. ChIP-Atlas is not only providing the calculated peak-call data, but also providing curated metadata. These data and database can help to classify archived data by a likelihood of data reuse.

Since it is becoming usual to reanalyze large amount of public high-throughput data, the open data and its reuse should be discussed with the computational environment. We also would like to show how we handle those data using DDBJ supercomputer system and commercial cloud environment.

1. Cochrane G, Cook CE, Birney E. The future of DNA sequence archiving. GigaScience. 2012 Jul 12;1(1):1.
2. Ohta, Tazro A summary of sequencing quality of data archived in Sequence Read Archive Quality information of sequencing data archived in Sequence Read Archive calculated by FastQC Sequencing quality 2017-01-05 http://dx.doi.org/10.6084/m9.figshare.4498907.v2
3. Ohta, T; Nakazato, T; Bono, H (2017): Supporting data for "Calculating quality of public high-throughput sequencing data to obtain suitable subset for reanalysis from the Sequence Read Archive" GigaScience Database. http://dx.doi.org/10.5524/100304

BioThings SDK: a toolkit for building high-performance data APIs in biology

Cyrus Afrasiabi[1§], Sebastien LeLong[1§], Jiwen Xin[1], Ginger Tsueng[1], Andrew I. Su[1], Chunlei Wu[1*]

[1] The Scripps Research Institute, 10550 N Torrey Pines Rd, La Jolla, CA 92037

[*] Email: cwu@scripps.edu

[§] Contributed equally to this work

Project Website: http://biothings.io
Source Code: https://github.com/biothings/biothings.api
License: Apache License v2

The accumulation of biological knowledge and the advance of web and cloud technology are growing in parallel. The latest computation technology allows us to modernize the way we collect, organize and disseminate the growing biological knowledge. Building web-based APIs (Application Programming Interfaces) becomes more and more popular for accessing data in a simple and reliable manner. Comparing to the traditional raw flat-file downloads, web APIs allow users to request specific data such as a list of genes of interest without having to download the entire data file. Web APIs typically return data in a format conforming to common standards (e.g. JSON or XML). This means that developers can spend less time on wrangling data, and more time on analysis and discovery.

We previously developed two high-performance and scalable web APIs for gene and genetic variant annotations, accessible at MyGene.info and MyVariant.info. These two APIs are a tangible implementation of our expertise and collectively serve over 6 million requests every month from thousands of unique users. Crucially, the underlying design and implementation of these systems are in fact not specific to genes or variants, but rather can be easily adapted to other biomedical data types such drugs, diseases, pathways, species, genomes, domains and interactions, collectively, we refer them as "**BioThings**".

BioThings SDK is a generalized software development kit (aka **SDK**) for building the same high-performance APIs for other BioThings data types. This SDK enables other developers to build their own BioThings API for their specific data types. Users can take advantage of the abstracted technical layers we built into the SDK, and produce a high-performance API, which follows the best practice and community standards. We also adopted JSON-LD (JSON for Linked Data) as a mechanism to form the semantic connections between different data types, so that the set of BioThings APIs will form a network of linked programmatic-accessible biological knowledge. BioThings SDK now becomes the common backend of both MyGene.info and MyVariant.info APIs, and was also used to create two new members of BioThings APIs: one for chemical compound and drug data (http://c.biothings.io), and the other for taxonomy data (http://t.biothings.io). A generic *biothings* Python client has also been created to provide a ready-to-use Python client for any BioThings API without the need of any extra code, and yet it still remains extensible for any data-type specific customization.

Reproducing computational experiments *in situ* as an interactive figure in a journal article

Evanthia Kaimaklioti[1,2], Robert P Davey[1], Ian Mulvany.

[1] The Earlham Institute, Norwich Research Park, Norwich NR4 7UH, UK; [2] The University of East Anglia, Norwich; [3] eLife Publishing Journal, Cambridge, CB2 1JP, UK.

Project Repositories: [node server: https://github.com/code56/node_web_server.git]; [node-web-server-formula: https://github.com/code56/node-web-server-formula.git]; [node web server with box-plot interactive figure: https://github.com/code56/nodeServerSimpleFig.git].
License: MIT [node web server] GPL V3 [box-plot interactive figure]
Contact: evanthia.kaimaklioti@earlham.ac.uk

Irreproducible experiments are a problem across all disciplines of Life Sciences[1], posing a strong financial burden on society; an estimated $28 billion was spent on irreproducible biomedical science in 2015 in the USA alone[2]. There has been a vibrant movement towards Open Access and reproducible science, however there are still a lot of problems with making research reproducibility easy, effective and adopted by all scientists. Here, we present the prototype of an *Interactive Figure* (IF) residing inside a research article serving as a platform for reproducing computational experiments *in situ* as a means of improving the reproducibility of experiments.

The concept of an IF is a technical solution to allow the user to have access to raw data, data analysis steps (including open source code underlying that analysis), and an interactive web page through which results can be interrogated. As such, the reader will be able to see the effects of incorporating their own data and changing the analysis steps in real time, as well as modifying code and see how the presentation of the results will be altered. This project is in collaboration with eLife, and aims to produce an environment conducive to hosting interactive figures. Published code and data is provided as a virtual machine (Vagrant) to download and be spun up in a user's system, as well as within eLIFE's servers. Users will also be able to add their own data into the IF for use within computational steps presented in the paper. The Vagrant technology is preferred as it is a lightweight and portable means of compartmentalising code for anyone to reproduce experiments in their own systems, irrespective of operating system. The Vagrant box will be available to access as a web-page as it will be deployed via an Amazon Web Service (AWS) server. The project is written in Javascript and Jinja (for the Salt Vagrant provisioning), and is fully open source and available in GitHub.

We provide prototypes of the IF architecture, comprising a Javascript node server running an example article page that contains a wheat expression viewer from the Expression Visualisation and Integration Platform (expVIP). ExpVIP integrates a large amount of wheat RNA-seq data, allowing the comparison of results from parallel studies by interactively visualising them[3]. We also provide a second example comprising a node server with an interactive BioJS bar plot[4].

The use and need for this platform has been confirmed by a survey that we have conducted to scientists in the Norwich Research Park and a cohort (139) of eLife authors. The prototype of this platform will be distributed to eLife authors to receive user feedback on whether the experiments presented were more easily reproduced. We will then use these surveys to influence the development of a production IF article platform at eLIFE.

References

1. Grant, B., 2012. Science's Reproducibility Problem [Web Document]. TheScientist. url: (http://www.the-scientist.com/?articles.view/articleNo/33719/title/Science-s-Reproducibility-Problem/) (accessed 12.2.15).
2. Freedman, L.P., Cockburn, I.M., Simcoe, T.S., 2015. The Economics of Reproducibility in Preclinical Research. PLOS Biol. 13, e1002165.
3. Borrill, P., Ramirez-Gonzalez, R., Uauy, C., 2016. expVIP: a customisable RNA-seq data analysis and visualisation platform. Plant Physiol. pp.01667.2015.
4. http://biojs.io/d/biojs-vis-box-plot

CWL Viewer: The Common Workflow Language Viewer

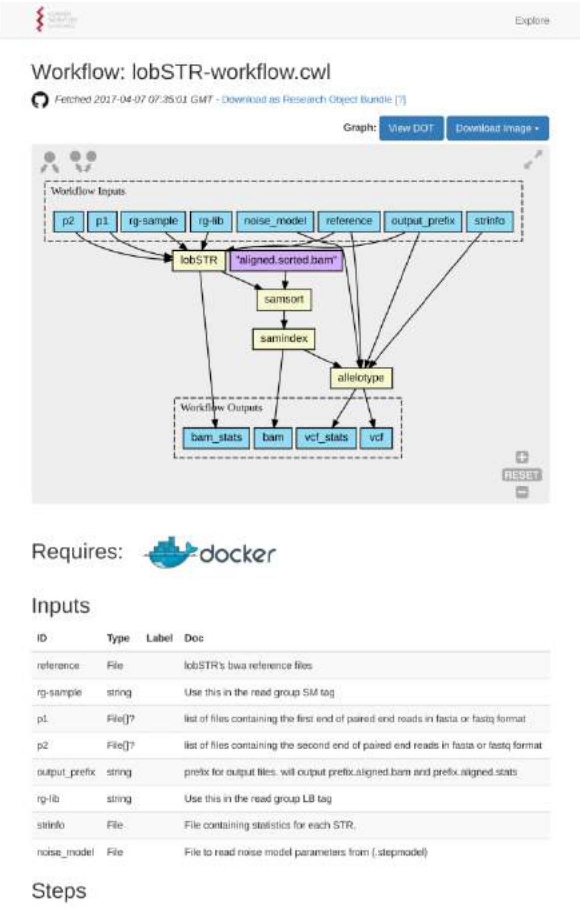
Mark Robinson[1], Stian-Soiland-Reyes[1], Michael Crusoe[2], Carole Goble[1]

[1] The University of Manchester, UK. Email: soiland-reyes@manchester.ac.uk

[2] Common Workflow Language project.

Project Website: https://view.commonwl.org/
Source Code: https://github.com/common-workflow-language/cwlviewer

Abstract



Workflow: lobSTR-workflow.cwl

Fetched 2017-04-07 07:35:01 GMT - Download as Research Object Bundle [?]

Requires:

Inputs

ID	Type	Label	Doc
reference	File		lobSTR's bwa reference files
rg-sample	string		Use this in the read group SM tag
p1	File[]?		list of files containing the first end of paired end reads in fasta or fastq format
p2	File[]?		list of files containing the second end of paired end reads in fasta or fastq format
output_prefix	string		prefix for output files. will output prefix.aligned.bam and prefix.aligned.stats
rg-lib	string		Use this in the read group LB tag
strinfo	File		File containing statistics for each STR.
noise_model	File		File to read noise model parameters from (.stepmodel)

Steps

The Common Workflow Language (CWL) project emerged from the BOSC 2014 Codefest as a grassroots, multi-vendor working group to tackle the portability of data analysis workflows. It's specification for describing workflows and command line tools aims to make them portable and scalable across a variety of computing platforms.

At its heart CWL is a set of structured text files (YAML) with various extensibility points to the format. However, the CWL syntax and multi-file collections are not conducive to workflow browsing, exchange and understanding: for this we need a visualization suite.

CWL Viewer is a richly featured CWL visualization suite that graphically presents and lists the details of CWL workflows with their inputs, outputs and steps. It also packages the CWL files into a downloadable Research Object Bundle including attribution, versioning and dependency metadata in the manifest, allowing it to be easily shared. The tool operates over any workflow held in a GitHub repository. Other features include: path visualization from parents and children nodes; nested workflows support; workflow graph download in a range of image formats; a gallery of previously submitted workflows; and support for private git repositories and public GitHub including live updates over versioned workflows.

The CWL Viewer is the de facto CWL visualization suite and has been enthusiastically received by the CWL community.

Sequanix: a standalone application to expose Snakemake pipelines to end-users.

Thomas Cokelaer*†, Dimitri Desvillechabrol‡, Rachel Legendre*, Mélissa Cardon‡

Bioinformatics Open Source Conference (BOSC) 2017, Prague, Czech

Website: http://sequana.readthedocs.io
Repository: https://github.com/sequana/sequana
License: BSD3 new clause

Sequana (http://sequana.readthedocs.io) is a Python-based software dedicated to the development of New Generation Sequencing (NGS) pipelines. The first motivation for this project was to provide NGS pipelines to a sequencing platform (Biomics pole, Institut Pasteur, Paris, France), which produces more than 200 runs a year combining MiSeq and HiSeq 2500 technologies but also long reads technology (Pacbio).

For production, we decided to use Snakemake framework to design our NGS pipelines. We currently have 6-7 pipelines covering quality control, variant calling, long-reads quality, de-novo and RNA-seq analysis [2]. Some of the pipelines provide original tools that are also available as standalone applications. For instance, a fast taxonomic analysis based on Kraken as well as a tool to perform exhaustive coverage analysis [1] have been developed.

For our end-users, we recently developed a GUI called **Sequanix** [3] that is developed with the PyQt framework. Sequanix exposes all Sequana pipelines (Snakemake pipelines) within an easy-to-use interface. Within the graphical interface, the configuration file used by Snakemake are automatically loaded and can be edited by end-users with dedicated widgets. We made the interface generic enough so that not only Sequana pipelines can be run interactively but also any Snakemake pipelines. Consequently, Sequanix should be useful for all Snakemake developers. Although the interface is simple enough for end-users, it allows developers to set any Snakemake parameters, to stop/restart the analysis, and remains 100% Snakemake-compatible. Since Sequanix is part of the Sequana project, it is also an open source project (see https://github.com/sequana/sequana).

References

[1] Dimitri Desvillechabrol, Christiane Bouchier, Sean Kennedy, Thomas Cokelaer Detection and characterization of low and high genome coverage regions using an efficient running median and a double threshold approach. http://biorxiv.org/content/early/2016/12/08/092478

[2] Desvillechabrol D, Bouchier C, Cokelaer T and Kennedy S. Sequana: a set of flexible genomic pipelines for processing and reporting NGS analysis [v1; no peer reviewed]. F1000Research 2016, 5:1767 (poster) (doi: 10.7490/f1000research.1112656.1)

[3] Desvillechabrol et al. (2017) Sequanix: A Dynamic Graphical Interface for Snakemake Workflows, in preparation

[4] Köster, J., and Rahmann, S. (2012). Snakemake - a scalable bioinformatics workflow engine. Bioinformatics, **28**(19), 2520-2522.

[*] Institut Pasteur - Bioinformatics and Biostatistics Hub - C3BI, USR 3756 IP CNRS - Paris, France.

[†] Email:thomas.cokelaer@pasteur.fr

[‡] Institut Pasteur - Biomics Pole - CITECH - Paris, France

A-126

NGI-RNAseq - a best practice analysis pipeline in Nextflow

Rickard Hammarén[1], Philip Ewels[2], Denis Moreno[2] , Max Käller[3]

[1]Science for Life Laboratory, Department of Biosciences and Nutrition, Karolinska Institute, Solna, 17121, Sweden. Email: rickard.hammaren@scilifelab.se
[2] Science for Life Laboratory, Stockholm University, Stockholm 106 91, Sweden
[3] Science for Life Laboratory, School of Biotechnology, Division of Gene Technology, Royal Institute of Technology, Stockholm, Sweden

Project Website: https://github.com/SciLifeLab/NGI-RNAseq
Source Code: https://github.com/SciLifeLab/NGI-RNAseq
Licence: MIT

The National Genomics Infrastructure is a national sequencing platform for Sweden, based at the Science for Life Laboratory. We process more RNA-sequencing samples than any other application, handling nearly 11 thousand during 2016 alone. In order to facilitate handling the quality control of this many samples we must rely on an automated workflow. The pipeline must be reliable, robust and scalable. In order to solve this problem we built the NGI-RNAseq pipeline.

The pipeline is written using Nextflow which is configured to handle job submission to our HPC through SLURM by default. Running the pipeline is now as simples as:

```
nextflow run SciLifeLab/NGI-RNAseq --reads '*_R{1,2}.fastq.gz'
```

As long as Nextflow is installed and has an internet connection it will clone the pipeline github repository and run the pipeline automatically.

The pipeline has been successfully run on Swedish national computing resources UPPMAX and C3SE and we are currently developing support for AWS. It is also possible to run the pipeline locally on your own laptop or server without having to install all of the required software, by using docker. The pipeline aligns using STAR by default, which is very fast and performs well, but it needs a lot of memory (~ 32 GB) for human samples. Thus we have implemented support for HISAT2 as well, which has more modest requirements.

The docker integration and low-memory HISAT2 option allow us to run automatic testing with Travis CI. It downloads a small example (subsampled data with yeast reference genome), runs the entire pipeline and notifies us of any bugs or issues that the new code has introduced.

We deliver interactive MultiQC reports as a part of the pipeline, making it easy for us and our users to visualise the results. Our custom MultiQC plugin also pulls user sample names into the report and generates a heatmap and MDS plot based on euclidean sample distances.

NGI-RNAseq

PhyloProfile: an interactive and dynamic visualization tool for multi-layered phylogenetic profiles

Ngoc-Vinh Tran[1*], Bastian Greshake[1], and Ingo Ebersberger[1]

[1]Dept. for Applied Bioinformatics, Inst. of Cell Biology and Neuroscience, Goethe University, Frankfurt am Main

*Email: tran@bio.uni-frankfurt.de

Project Website: https://phyloprofile.shinyapps.io/phyloprofile/
Source Code: https://github.com/trvinh/phyloprofile/
License: The MIT License

Newly sequenced genomes, even of species from the remotest corner of the organismic phylogeny, emerge almost on a daily basis. Phylogenetic profiles, capturing the presence and absence of orthologs across species, form the basal information layer for annotating and integrating this novel data. However, these profiles are increasingly overlaid with complementary data such as the extent of sequence similarity between orthologs, similarities and differences of their domain architectures, their Gene Ontology-term-based semantic similarities, or their distribution in a particular phylogenetic clade. A comprehensive tracing of proteins and their functionalities across species and through time requires an intuitive integration optimally of all these various layers. Yet, there are few, if any, visualization tools that aid in this process.

Here, we present PhyloProfile, an intuitive tool for integrating, visualizing and exploring multi-layered phylogenetic profiles. Alongside the presence/absence pattern of orthologs across large taxon collections, the current version of PhyloProfile allows the integration of any two additional information layers. Taxa can be dynamically collapsed into higher order systematic groups, as defined by the NCBI taxonomy. By that the user can rapidly change the resolution from the comparative analyses of proteins in individual species to that of entire kingdoms or even domains without the need of input data modification. Profiles can be filtered dynamically according to various criteria. For example, setting a minimal threshold for the fraction of species in a systematic group having a particular ortholog present can reduce the impact of spurious ortholog identification on evolutionary interpretations. Likewise, filtering collections of orthologs based on the pre-computed similarity of their domain architectures – if provided as an information layer – can either highlight or blend out orthologs with deviating architectures. Subsequent to filtering, the data can be downloaded to serve as input for downstream analyses. Profile plots can be exported as ready-to-publish PDF images with many options for editing their presentation like color, size, position of plot's axis and legend.

We exemplify the versatility of PhyloProfile by analyzing the phylogenetic profiles of 1605 Microsporidia proteins in 230 species from all three domains of life. We demonstrate how the tool can be applied for (i) estimating gene ages, (ii) tracing changes in protein function approximated by changes in their domain architecture, and (iii) extracting a phylogenomic data set to explore deep splits in the eukaryotic tree of life.

PhyloProfile is provided as an online tool at https://phyloprofile.shinyapps.io/phyloprofile/. The open source code can be downloaded at https://github.com/trvinh/phyloprofile.

CueSea: quality control tool for Illumina genotyping microarray data, with correction on intensity, clusterization and biological specificity.

Nikita Moshkov[1,2], Dimitri Nikogosov[2], Daria Iakovishina[2].
[1] Higher School of Economics, Moscow. Email: nikitam851@gmail.com
[2] Atlas Biomed Group, Moscow

Project Website: https://github.com/Arkkienkeli/CueSea
Source Code: https://github.com/Arkkienkeli/CueSea
License: GNU GPL v.3

Genotyping microarrays are widely used for studying various genomic questions. Nevertheless there was no tool thus far that covered all aspects of SNP quality control. Most quality control approaches (e.g., in GWAS studies) use PLINK and GenomeStudio. PLINK only works with genotype data, and miss steps related to clustering and intensity (i.e., raw data directly from the sequencing machine). On the other hand, GenomeStudio performs clustering and evaluation of SNPs with its GenTrain algorithm, but on a low number of samples (i.e., less than 100), default clustering can work ambiguously. For instance, two samples from the same person can have different genotypes in some SNPs, even though the raw data does not vary for both alleles. In cases when there are a large number of data samples, re-clustering is performed manually, which may also lead to ambiguity.

We introduce a new quality control tool for Illumina genotyping microarrays, which does not require a large number of data samples. It takes into account limitations of the raw data biological specificity, based on the GenomeStudio final report. This tool takes into account sample intensity and automatically performs SNP clusterization. If genotype given by GenomeStudio does not correspond to the expected cluster, CueSea reports mis-clustered. Also it checks genotype calls for sex chromosomes (sex is also detected automatically) and mitochondria, assesses the compliance with the Hardy-Weinberg equilibrium.

CueSea was tested on 346 samples for 5 000 SNPs each and 8 batches of 48 samples with 550 000 SNPs each. We validated CueSea on samples received from the same individual. These samples had 474 different genotype SNPs out of 550 000. CueSea filtered 410 SNPs by quality, and changed genotype for 62 SNPs. We received approximately the same results on 5 more samples (also twice genotyped).

Bio::DB::HTS – accessing HTSlib from Perl

Rishi Nag[1], Keiran M. Raine[2], Christopher J. Fields[3], Alex Hodgkins[4], Andrew Yates[1], David Jones[2], James Gilbert[2], Fergal Martin[1], Bronwen Aken[1]

[1]European Molecular Biological Laboratory, European Bioinformatics Institute, Wellcome Genome Campus, Hinxton, UK. Email rishi@ebi.ac.uk
[2]Wellcome Trust Sanger Institute, Wellcome Genome Campus, Hinxton, UK
[3]University of Illinois, Urbana, IL, USA
[4]Congenica, Hinxton, UK

Project Website: https://github.com/Ensembl/Bio-DB-HTS
Source Code: https://github.com/Ensembl/Bio-DB-HTS
License: Apache License 2.0

We have developed Bio::DB::HTS, a Perl library which uses the Perl XS framework to create an API for Perl scripts to access the functions and data structures of the HTSlib C library for reading and writing high-throughput sequencing data. This has the benefit of providing access to file formats in Perl with the speed and memory improvements offered by a C implementation and without introducing errors as would be possible in a new native Perl implementation.

This library addresses a pressing need from those who use Perl to access high-throughput sequencing data. In particular, many were experiencing the limits of Bio-SamTools - a Perl API which provides an API into 0.x versions of SAMtools. At Version 1.0, SAMtools was split into a command-line based front end (also named SAMtools) and a back end (HTSlib). Bio-SamTools has not yet been updated to reflect the separation of HTSlib and Samtools, and was hence unable to use post version 1.0 functionality such as the ability to read CRAM files.

Currently Bio::DB::HTS offers the ability to read BAM, CRAM, BCF, FASTA (uncompressed or block zipped), FASTQ and TABIX file formats in Perl. As an example of a project accessing big data using Perl, Ensembl has benefited from the ability to read CRAM files for storing and distributing an increased volume of RNA-seq alignments. Ensembl has also used Bio::DB::HTS to improve management of the large number of human variants resulting in speed increases for the Ensembl Variant Effect Predictor (VEP) and add the ability to display CRAM data in the Ensembl genome browser. Bio::DB::HTS is open source software and can be obtained via CPAN or GitHub. The accompanying scripts allow the installation of Bio::DB::HTS in a variety of configurations.

NGLESS: Perfectly understandable and reproducible metagenomics pipelines using a domain-specific language

Luis Pedro Coelho (coelho@embl.de)[1], Paulo Monteiro[2], Renato Alves (ralves@embl.de)[1], Ana Teresa Freitas[2], Peer Bork[13]

[1] European Molecular Biology Laboratory (EMBL), Heidelberg, Germany
[2] INESC-ID, Instituto Superior Técnico, University of Lisbon, Portugal
[3] Max Delbrück Centre for Molecular Medicine, Berlin, Germany

Project Website: http://ngless.readthedocs.io
Source Code: https://github.com/luispedro/ngless
License: MIT

We present ngless, a tool for metagenomics analysis, which can produce functional and taxonomic profiles from sequence files in an understandable and reproducible fashion. Ngless is based on the concept of a domain specific language for next-generation sequence processing. This language and tool are designed from the ground-up to enable perfect computational reproducibility, while being easily understandable.

With ngless, users define their metagenomics pipeline in a short script written in a domain specific language. The built-in knowledge of the problem domain allows for best-practices to be automatic: our tool always collects quality control statistics, for example. Debugging time, that is often a significant fraction of development time, is curtailed by using ngless since it performs several parameter checks before interpretation begins. This includes traditional computer language elements such as names and types, but also elements such as file permissions. This benefits the user whenever they have made a mistake as errors are detected immediately.

Unlike tools which are based on traditional programming languages, ngless is designed with reproducibility as a goal. Documenting the version of all the tools used is considered a best practice, but is not always followed in practice and it is error-prone as it is difficult to exhaustively capture all dependencies and a dependency list can quickly diverge from the versions used in analysis. With ngless, every script must include a version declaration and every imported module specifies the particular version which is being imported. By making it a requirement within the script, ngless ensures that this best practice is followed. At the same time, ngless lowers the effort required. For example, the versions of internally used tools (such as bwa or samtools) are implicitly fixed by specifying the ngless version. Furthermore, ngless is implemented so that its results do not depend on the environment in which it is being run.

Additionally, ngless has built-in support for several datasets that are useful for analysing metagenomes. As with other dependencies, the exact versions of these resources is defined to ensure reproducibility.

Currently, ngless supports the following processing steps natively: (1) data preprocessing, (2) assembly, (3) mapping (short read alignment), (4) filtering alignment results, and (5) profiling (computation of alignment summary statistics). It has built-in knowledge of model organisms and metagenomics gene catalogs. The focus of ngless has been metagenomics processing, but it can be used in other NGS domains. To facilitate this, the tool can be extended with user-supplied scripts by adding a YaML formatted description of inputs and outputs.

CGP as a Service (CGPaaS) - From data submission to results using your web-browser

- **Keiran Raine, Wellcome Trust Sanger Institute, United Kingdom**
- Adam Butler, http://keiranmraine.github.io, United Kingdom
- Peter Clapham, Wellcome Trust Sanger Institute, United Kingdom
- Jon Teague, Wellcome Trust Sanger Institute, United Kingdom
- Peter Campbell, Wellcome Trust Sanger Institute, United Kingdom

Short Abstract: The Cancer Genome Project (CGP) has been heavily involved in the work of the ICGC PanCancer Analysis Of Whole Genomes (PCAWG) project to characterise 2,800 cancers. CGP provided one of the three core somatic calling pipelines. As part of this effort we have successfully produced a codebase that is entirely portable, open-source, and available as a PCAWG workflow on www.dockstore.org. This workflow generates Copy-Number, Substitution, Insertion, Deletion and Structural Variant results optimised for tumour-normal paired somatic NGS data. CGP are now looking to provide an updated version of this workflow within a cloud enabled framework. One of the key issues that faces investigators when working with large sequence data is the difficulty in transferring large datasets without the need to install dedicated software. In order to address this issue we plan to implement an in-browser, drag and drop process for data submission and retrieval. Following successful validation of data, mapping and analysis through the standard Whole Genome Sequence (WGS) somatic calling pipeline will be triggered. Here we present the current state of this work along with the current road-map for the next 2 years development.

Large-scale genotypic and phenotypic data support for Tripal: Chado optimization by utilizing modern PostgreSQL functionality

- Lacey-Anne Sanderson, University of Saskatchewan, Canada
- Reynold Tan, University of Saskatchewan, Canada
- **Carolyn Caron, University of Saskatchewan, Canada**
- Kirstin Bett, University of Saskatchewan, Canada

Short Abstract: Present-day breeding programs demand evaluation of large-scale phenotypic and genotypic data to assist with selections. We have extended Tripal, an open-source toolkit for developing biological data web portals, to manage and display large-scale genotypic and phenotypic data. Tripal utilizes the GMOD Chado database schema to achieve flexible, ontology-driven storage in PostgreSQL. The ND Genotypes module (https://github.com/UofS-Pulse-Binfo/nd_genotypes) supports loading of a variety of file formats, including the common Variant Call Format (VCF). The Chado schema was extended to include an associative table, connecting the genotype call with the marker and stock assayed. Meta-data, including quality metrics, is supported as a JSONB array for maximum flexibility. Use of materialized views and indexing in PostgreSQL improves performance such that summary charts and tables can be generated dynamically. Researchers may download the data for further analysis with choice of various software-friendly formats. Similarly, the Analyzed Phenotypes module (https://github.com/UofS-Pulse-Binfo/analyzedphenotypes) alters the Chado schema through use of a single associative phenotype table, and dynamically produces phenotype distribution charts for the user to compare phenotypic measurements between site-years. At present, both of these modules have been tested with 5 billion data points with no noticeable reduction in performance. The utility and performance of these extension modules has been benchmarked on our own breeder-focused web portal for legumes, KnowPulse (http://knowpulse.usask.ca).

A-133

GenePattern Notebooks: An integrative analytical environment for genomics research

- **Michael Reich, University of California, San Diego, USA**
- Thorin Tabor, Broad Institute of MIT and Harvard, USA
- Helga Thorvaldsdóttir, Broad Institute, USA
- Barbara Hill, Broad Institute, USA
- Ted Liefeld, UCSD, USA
- Jill Mesirov, UC San Diego, USA
- Pablo Tamayo, UC San Diego Moores Cancer Center, USA

Short Abstract: As the availability of genetic and genomic data and analysis tools from large-scale cancer initiatives continues to increase, the need has become more urgent for a software environment that supports the entire “idea to dissemination” cycle of an integrative cancer genomics analysis. Such a system would need to provide access to a large number of analysis tools without the need for programming, be sufficiently flexible to accommodate the practices of non-programming biologists as well as experienced bioinformaticians, and would provide a way for researchers to encapsulate their work into a single “executable document” that included not only the analytical workflow but also the associated descriptive text, graphics, and supporting research. To address these needs, we have developed GenePattern Notebook, based on the GenePattern environment for integrative genomics and the Jupyter Notebook system. GenePattern Notebook presents a familiar lab notebook format that allows researchers to build a record of their work by creating “cells” containing text, graphics, or executable analyses. Researchers add, delete, and modify cells as the research evolves. When an analysis is ready for publication, the same document that was used in the design and analysis phases becomes a research narrative that interleaves text, graphics, data, and executable analyses, serving as the complete, reproducible, in silico methods section for a publication.

BioContainers for supercomputers: 2,000+ accessible, discoverable Singularity apps

John M. Fonner[1], Rion Dooley[1], Jacquelyn Turcinovic[1], Matthew W. Vaughn[1]

[1] The University of Texas at Austin, Austin, TX, USA. Email: jfonner@tacc.utexas.edu

Project Website: https://togo.agaveapi.co
Source Code: https://bitbucket.org/agaveapi/agave-togo
License: BSD-3Clause
(https://bitbucket.org/agaveapi/agave/src/6af91b4d4046d7148d448087ac182efc502bc931/LICENSE)

Containerization of scientific applications has done a lot to make software more portable and reproducible. Application catalogues like BioContainers[1] have done a lot to make thousands of apps more accessible and discoverable. Unfortunately, almost all high performance computing (HPC) systems, such as those provided to researchers through the NSF's XSEDE program, currently do not support Docker containers due to kernel requirements and security policy. Thus, despite significant progress toward reproducible scientific software, there is a debilitating disconnect between the substantial computing power of HPC systems available for academic computing and the container technologies required to populate those systems with apps.

To reconcile this disconnect, we have fostered the adoption of Singularity[2] on XSEDE academic supercomputing resources at the Texas Advanced Computing Center, and we have ported over 2,000 computational biology applications from the BioContainers project to Singularity images that run effectively on HPC systems. These images have been made available to the scientific community through a freely accessible web interface and API. These containers are actively synced with the BioContainers repository to ensure new apps are added to the catalogue. We have created a web interface for executing any of these apps through the Cyverse[3] platform (also freely accessible and open source), and the scripts used to generate the containers themselves are available alongside documentation. We are actively working with the developers of Singularity Hub to make these containers available through their APIs, further improving discoverability and ease of access.

This work represents a significant leap forward for computational biology on the national supercomputing cyberinfrastructure, and the catalogue of applications now available is an order of magnitude greater than what it was. Container technologies like Docker and community-lead efforts like BioContainers have radically improved portability and reproducibility for scientific computing on cloud and virtual machine resources. Similarly, we hope that the tools developed by the Singularity team alongside a catalogue of thousands of community-curated containers will extend these advances to the many world-class and institutional HPC systems that can benefit from them.

[1] Leprevost, F. D., et al. "BioContainers: An open-source and community-driven framework for software standardization." *Bioinformatics (Oxford, England)* (2017).
[2] Kurtzer, Gregory M., Vanessa Sochat, and Michael W. Bauer. "Singularity: Scientific containers for mobility of compute." *PloS one* 12, no. 5 (2017): e0177459.
[3] *Merchant, Nirav, et al., "The iPlant Collaborative: Cyberinfrastructure for Enabling Data to Discovery for the Life Sciences," PLOS Biology (2016), doi: 10.1371/journal.pbio.1002342.*

Collaborative Open Plant Omics: A platform for “FAIR” data for plant science

Felix Shaw[1], Anthony Etuk[1], Alejandra Gonzalez-Beltran[2], David Johnson[2], Philippe Rocca-Serra[2], Susanna Sansone[2], Robert P Davey[1]

[1] Earlham Institute, Norwich, UK

[2] Oxford e-Research Centre, Oxford, UK

Project Website: http://copo-project.org
Source Code: https://github.com/collaborative-open-plant-omics/COPO
License: Creative Commons

Abstract

COPO is a brokering service for the aggregation, annotation, sharing and deposition of research objects allowing plant scientists to provide a richer context to their work when preparing it for dissemination. As well as published manuscripts plant scientists may, for instance, produce sequence data, images, figures, posters, presentations, source code, workflows, etc. COPO allows these disparate objects to be aggregated in a single place, where they can be labelled with required metadata before being automatically deposited to the appropriate internet repository. As such, COPO is not a repository itself and objects are deleted from our servers after deposition, however repository accessions and any other unique identifiers (UUIDs, URIs) are stored. COPO assigns the whole research collection a UUID which can be used to refer to the aggregation of object accessions. It then becomes possible to read about an experiment, follow the COPO ID and find the original data, workflows and results all in the same place, enabling reproducible science. Authentication and access is provided by Single Sign-on via ORCiD, allowing integration with other services that support this method, and enables the import of any work/publication history stored in the user’s ORCiD account. Users create a COPO “Profile”, and into this they can upload sequence data, pdfs, spreadsheets and images. Wizards then guide the user through the process of annotation with the minimum required metadata for deposition, enforcing the collection of standardised descriptions allowing datasets deposited through the system to be used elsewhere and searched in predictable ways by third party tools. Much of this annotation is powered using the EBI Ontology Lookup Service therefore allowing ontological context to be applied to many fields in the metadata. Other less highly-curated data types such as images, spreadsheets and free text data use metadata fields in common with Dublin Core and Figshare. Relying on the ISA-API (https://github.com/ISA-tools/isa-api), the ISAconverter functions enable the translation of COPO’s user-friendly web forms into the required JSON/XML deposition formats, without the user needing to do this themselves.

Deposition to ENA for sequence based ‘omics data is supported as well as deposition to Figshare or Dataverse for images, pdfs, figures etc, and we are currently implementing the MIAPPE standard. We will integrate with analysis platforms such as CyVerse and Galaxy so that workflows and data stored in COPO can be reproduced in the same or different configurations by users. Since the essence of COPO is a large network of ontologically labelled metadata (stored as Linked Data in a MongoDB database), we will start mining this data for patterns to aid users in metadata attribution and to look for hidden trends which may provide important insight into future research directions.

CWL + Research Object == Complete Provenance

Farah Zaib Khan[1], Stian-Soiland-Reyes[2], Andrew Lonie[1], Richard Sinnott[1]

[1] The University of Melbourne, Australia. Email: farah.khan@unimelb.edu.au

[2] The University of Manchester, UK.

Source Code: https://github.com/common-workflow-language/common-workflow-language/wiki/Research-Object-Proposal

Abstract

The term **Provenance** is referred to as 'The beginning of something's existence; something's origin' Or 'A record of ownership of a work of art or an antique, used as a guide to authenticity or quality'. Provenance tracking is crucial in scientific studies where workflows have emerged as an exemplar approach to mechanize data-intensive analyses. Gil et al. analyze challenges of scientific workflows and concluded that formally specified workflow helps 'accelerate the rate of scientific process' and facilitates others to reproduce the given experiment provided that provenance of end-to-end process at every level is captured .

We have implemented exemplar ***GATK variant calling workflow*** using three approaches to workflow definition namely ***Galaxy***, ***CWL*** and ***Cpipe*** to identify assumptions implicit in these approaches. These ***assumptions*** lead to limited or no understanding of reproducibility requirements due to lack of documentation and comprehensive provenance tracking and resulted in identification of provenance information crucial for genomic workflows as shown in Figure 1.

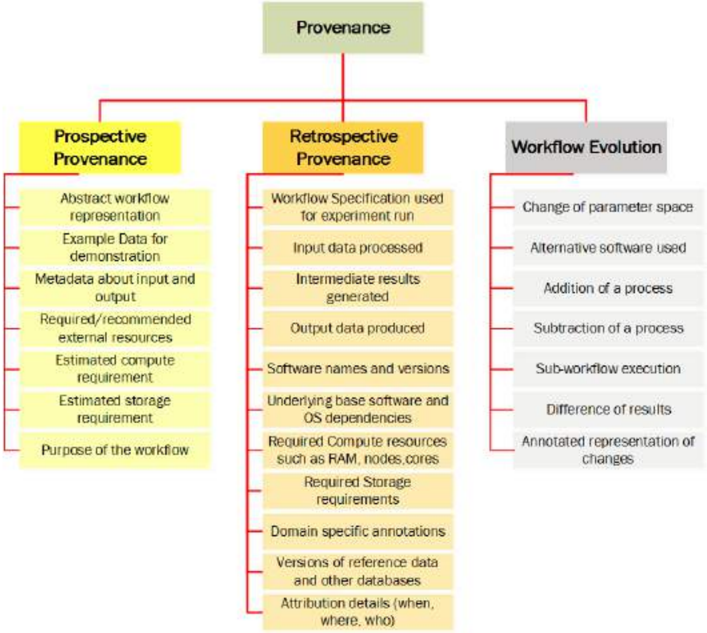

Provenance
Prospective Provenance
Retrospective Provenance
Workflow Evolution
Abstract workflow representation
Example Data for demonstration
Metadata about input and output
Required/recommended external resources
Estimated compute requirement
Estimated storage requirement
Purpose of the workflow
Workflow Specification used for experiment run
Input data processed
Intermediate results generated
Output data produced
Software names and versions
Underlying base software and OS dependencies
Required Compute resources such as RAM, nodes,cores
Required Storage requirements
Domain specific annotations
Versions of reference data and other databases
Attribution details (when, where, who)
Change of parameter space
Alternative software used
Addition of a process
Subtraction of a process
Sub-workflow execution
Difference of results
Annotated representation of changes

Figure 1 Provenance Requirements for Typical Genomic Workflow

CWL provides a declarative approach to workflow declaration making minimal assumptions about precise software environment, base software dependencies, configuration settings, alteration of parameters and software versions. It aims to provide an open source extensible standard to build flexible and customized workflows including intricate details of every process. It facilitates capture of information by supporting declaration of requirements, `cwl:tool` and checksums etc. Currently, there is no mechanism to gather the produced information as a result of a workflow run into one bundle for future use. We propose to demonstrate the implementation of a module for CWL as shown in Figure 2.

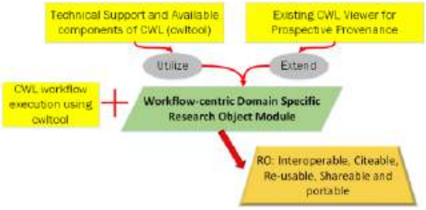

Technical Support and Available components of CWL (cwltool)
Existing CWL Viewer for Prospective Provenance
Utilize
Extend
CWL workflow execution using cwltool
Workflow-centric Domain Specific Research Object Module
RO: Interoperable, Citeable, Re-usable, Shareable and portable

Figure 2 Implementation of a module for CWL resulting in RO generation after a workflow Run

A-137

Title: Workflow-ready bioinformatics packages for Debian-based distributions and this Linux distribution’s infrastructure for low-friction reproducible research
Author: Contributors to the Debian Med Blend

Abstract: Debian is one of the first GNU/Linux distributions and has always invited users from all technical and scientific backgrounds to contribute. For over 15 year with contributors from all around the globe the Debian Med special interest group organised packaging of platforms and toolkits for computational biology and medical informatics. Together with Debian Science and Debichem groups, additional packages of shared interest with neighbouring disciplines are maintained.

Since Debian automatically builds all software from source code, it allows bioinformatics workflows to become consistently auto-deployable across platforms, e.g. 32bit ARM7, 64bit ARM8, x86, IBM POWER, or mainframes. It is also offered as a base for all prominent virtualisation technologies and cloud infrastructure service providers. Compiler flags for Debian builds are strict, as expected for security-sensitive parts of a Linux distribution. Fixes to the source tree are communicated back. Builds are checked for consistency by the http://reproducible-builds.org initiative, and automated testing has become an integral part of Debian packaging shown on http://ci.debian.net.

The packages in Debian are in their latest versions, or otherwise updated quickly upon request. Similarly important, http://snapshot.debian.org hosts packages from the past to facilitate assessments of scientific advancements or to consistently continue or reproduce older studies. Requests for new packages are typically how the number of contributors grows - we actively help with our expertise and curiously welcome yours - be it with your package, your cloud template, your workflow, or your participation to our next meeting.

Forever in BlueGenes: a next-generation genomic data interface powered by InterMine

Yo Yehudi[1,2], Daniela Butano[1,2], Matthew Chadwick[1,2], Justin Clark-Casey[1,2], Sergio Contrino[1,2], Josh Heimbach[1,2], Rachel Lyne[1,2], Julie Sullivan[1,2], Gos Micklem[1,2]

[1]Department of Genetics, University of Cambridge, Cambridge, United Kingdom
[2]Cambridge Systems Biology Centre, University of Cambridge, Cambridge, United Kingdom

Project Website: http://bluegenes.apps.intermine.org/
Source Code: https://github.com/intermine/bluegenes
License: LGPL (see https://github.com/intermine/bluegenes/blob/dev/LICENCE)

Main Text of Abstract

The plight of the computational biologist, or indeed any data scientist, often boils down to data: data may be badly formatted, missing information, hard to access and hard to integrate with other sources. Similarly, user interfaces designed for data analysis may not be easy to use, hindering scientific analysis rather than assisting it.

BlueGenes is designed to facilitate biological data discovery and analysis in a user-friendly and enjoyable way, without requiring that scientists write code or wrangle multiple datasets themselves. Designed as a modern replacement for InterMine's[1] JSP-powered interface, which traces its origin as far back as 2003, BlueGenes focuses on enhancing user experience using in-page updates, colour-coded data types, and mobile / touch friendliness where possible. Iterative user testing and community feedback is used to enhance usability.

The data available in BlueGenes is sourced via web services from individual InterMine instances, which are typically comprised of multiple high-quality integrated datasets. Using InterMine as a data source allows BlueGenes to source genomic and proteomic data from the array of biological databases worldwide that run InterMine instances, including model organisms such as the mouse, rat, budding yeast, fruit fly, nematode worm, and thale cress as well as cattle, chinese hamster ovary cells, and many plants.

BlueGenes is browser-based and runs client-side. The decoupled architecture separates the user interface from the data server, allowing the same front-end instance to seamlessly switch between InterMine data sources. Whilst the core of BlueGenes is programmed in ClojureScript for rapid iterative prototyping purposes, a plugin architecture is currently under development that will allow data analysis and visualisation tools to be seamlessly integrated in native JavaScript. BlueGenes runs on Re-Frame[2], a ClojureScript library based on React.js.

[1] Smith RN, Aleksic J, Butano D, Carr A, Contrino S, Hu F, Lyne M, Lyne R, Kalderimis A, Rutherford K, Stepan R, Sullivan J, Wakeling M, Watkins X, Micklem G. InterMine: a flexible data warehouse system for the integration and analysis of heterogeneous biological data. Bioinformatics. 28(23):3163-5 (2012)

[2] Thompson, M. Re-Frame: A Reagent Framework For Writing SPAs, in Clojurescript. Zenodo. http://doi.org/10.5281/zenodo.801613 (2015)

GRADitude: A computational tool for the analysis of Grad-seq data

Silvia Di Giorgio[1] , Konrad U. Förstner[1 2].

[1]Institute of Molecular Infection Biology (IMIB), University of Würzburg, 97080 Würzburg, Germany
[2]Core Unit Systems Medicine, University of Würzburg, 97070 Würzburg, Germany

Grad-seq is a high-throughput profiling method for the organism-wide detection of RNA-RNA and RNA-protein interactions in which molecular complexes are separated in a gradient by shape and size (Smirnov *et al.*, 2016, *PNAS*). After this physico-chemical separation the gradient fractions are sequenced using 2nd generation sequencing protocols. The Grad-seq approach offers a way to study the role of different RNA components in various macromolecular assemblies.

GRADitude is a computational tool for the analysis of Grad-seq in-gradient profiling. This open source tool (ISC license), developed in Python will perform all required steps to translate sequencing data of a Grad-seq experiment into a list of potential molecular complexes. It offers different methods to normalize read counts of fractions and generates numerous statistics. GRADitude is going to integrate several machine learning approaches to predict interactions and create interactive visualizations to explore a given data set.

GRADitude is still at an early stage of development and further functionalities will be added in the future.

SOURCE CODE URL:

https://github.com/konrad/GRADitude

REFERENCES:

Alexander Smirnov et al., 2016, PNAS - Grad-seq guides the discovery of ProQ as a major small RNA-binding protein

Enabling the optimization of open-source biological computational tools with scripting languages

Stefan Popa [1]

[1] Intel Inc. / University POLITEHNICA Bucharest Romania. Email: stefan.popa@acm.org / stefan.a.popa@intel.com

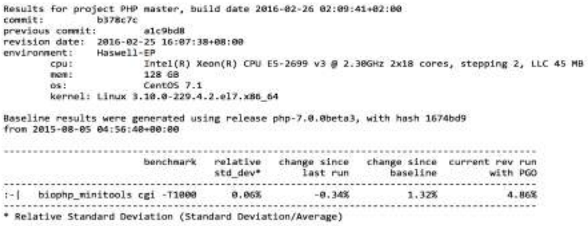
Project Website: http://languagesperformance.intel.com
Source Code: https://github.com/php/php-src, https://hg.python.org/cpython/
License: PHP License Agreement (GPL-compatible--see https://github.com/php/php-src/blob/master/LICENSE), Python License Agreement (GPL-compatible--see https://hg.python.org/cpython/file/tip/LICENSE)

Main Text of Abstract

In this presentation it is described software optimizations for open-source biological computational tools like BioPHP or BioPython through Intel Architecture software optimizations of the open-source scripting languages used, like PHP or Python.

Today's compilers are providing various optimization techniques, such as loop unrolling, code layout optimizations, inlining and many other. All these techniques have a common point, they make their decisions based on the analysis made to the application source code. Even though this approach provides good results for general case, compilers can optimize furthermore an application if we know the workload, the input data or any other information about runtime by applying sampling or instrumentation Profile Guided Optimizations (PGO).

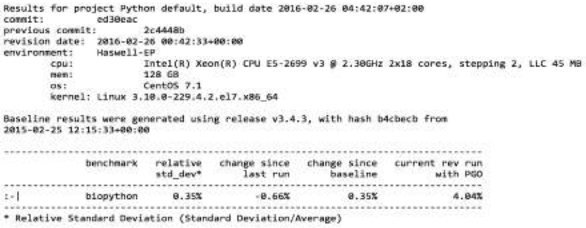
In Bio-related Cloud Computing applications, knowing in advance the workload or how the application must behave at runtime, can lead to a great degree of improvement and better end-user experience. In this paper we talk about how can we apply PGO on ones of the most used open-source biological computational tools, BioPHP and BioPython. The presentation will detail the way we used PGO to optimize PHP and Python interpreters (https://github.com/php/php-src/commit/7dac4d449f72d7eb029aa1a8ee87aaf38e17e1c5, https://hg.python.org/cpython/rev/7fcff838d09e).



```
Results for project PHP master, build date 2016-02-26 02:09:41+02:00
commit:         b378c7c
previous commit:        a1c9bd8
revision date:  2016-02-25 16:07:38+08:00
environment:    Haswell-EP
        cpu:            Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz 2x18 cores, stepping 2, LLC 45 MB
        mem:            128 GB
        os:             CentOS 7.1
        kernel: Linux 3.10.0-229.4.2.el7.x86_64

Baseline results were generated using release php-7.0.0beta3, with hash 1674bd9
from 2015-08-05 04:56:40+00:00

-------------------------------------------------------------------------------------------
                        benchmark   relative   change since   change since   current rev run
                                    std_dev*       last run       baseline          with PGO
-------------------------------------------------------------------------------------------
:-|   biophp_minitools cgi -T1000      0.06%         -0.34%          1.32%             4.86%
-------------------------------------------------------------------------------------------
* Relative Standard Deviation (Standard Deviation/Average)
```



```
Results for project Python default, build date 2016-02-26 04:42:07+02:00
commit:         ed30eac
previous commit:        2c4448b
revision date:  2016-02-26 00:42:33+00:00
environment:    Haswell-EP
        cpu:            Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz 2x18 cores, stepping 2, LLC 45 MB
        mem:            128 GB
        os:             CentOS 7.1
        kernel: Linux 3.10.0-229.4.2.el7.x86_64

Baseline results were generated using release v3.4.3, with hash b4cbecb from
2015-02-25 12:15:33+00:00

-------------------------------------------------------------------------------
                benchmark   relative   change since   change since   current rev run
                            std_dev*       last run       baseline          with PGO
-------------------------------------------------------------------------------
:-|             biopython      0.35%         -0.66%          0.35%             4.04%
-------------------------------------------------------------------------------
* Relative Standard Deviation (Standard Deviation/Average)
```

Profile Guided Optimizations brings more than 4% performance increase for open-source biological computational tools like BioPHP and BioPython. The initiative to have optimized open-source scripting languages is on-going. The large and very active communities behind each of these open-source scripting languages like PHP and Python generate daily commits, bug fixes, new language features, and performance optimizations. The overall performance status of PHP and Python language can be tracked daily using https://01.org/lp and http://languagesperformance.intel.com.

Protein Inpainter: a Message-Passing-based Predictor using Spark GraphX

Rabie Saidi[1,*], Hanna Papkova[1], Tunca Dogan[1], Maria J. Martin[1]
1 European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI),
Cambridge CB10 1SD, UK
*To whom correspondence should be addressed: rsaidi@ebi.ac.uk

Project Website: http://www.ebi.ac.uk/~rsaidi/pi
Source Code: https://bitbucket.org/rabie_saidi/pi
License: Apache License 2.0

ABSTRACT

As the data tsunami continues to surge, many Big Data technologies such as the famous Hadoop, have been developed and are still pulling improvements from many contributors. These technologies have matured from their earlier emerging stage and have helped in bridging the gap between the increasing data volumes and the computational potential. In the last few years, a new open source technology termed Apache Spark, finally surpassed Hadoop in interest and popularity. One of the main reasons behind this is that Spark can run programs much more quickly than Hadoop as it supports in-memory parallel processing. Beyond the hype, Spark has real assets that can be efficiently used to address many Data Science issues, like machine learning, interactive SQL, and real-time processing. The GraphX component is one of these assets for graphs and graph-parallel computation. It provides an API on top of Spark for graph analysis and benefits from the various data transformation and data analysis features provided by Spark and its different components. However, as far as we know, neither the official website nor the various practitioners or contributors to the Spark project have provided guidance to use GraphX in Java whilst many Java developers are continuously searching for this.

In this scope, we have leveraged Spark and GraphX to address a known issue in Bioinformatics that can be represented in the graph paradigm, which is the prediction of protein enzymatic functions. We introduce our prediction system Protein Inpainter (PI) that can be used to enhance the quality of predicted functions as well as labelling proteins with unknown functions. PI is inspired from the techniques of digital image recovery (or inpainting) and uses the Java API of Spark GraphX to parse data and perform parallel computation. PI translates proteins into a bipartite graph. Nodes are either proteins or protein attributes (e.g., protein signatures) that can be used to infer functions. Edges are built between each protein and its corresponding attributes. After building the graph, PI iteratively performs message passing from labelled proteins to attributes and then from attributes to unlabelled proteins which will eventually become labelled. After each message passing wave from a side of the graph to the other, merging procedures are performed at each node, while taking into account the weights of sending and receiving nodes. For protein nodes, merging is followed by another procedure of labelling which is based on a collaborative filtering approach. The PI software has not completely matured yet nor was it benchmarked against conventional prediction methods in bioinformatics like machine learning and alignment. However, it is functional and comes with two main advantages that could attract further contributions. To begin with, it is the first open source software written in Java and using GraphX, as all available GraphX-based programs are written either in Scala or Python since they are easier to code in, whereas Java is more typed and might require more instructions to be explicitly written. Secondly, PI is an example of how message passing can be used to perform prediction. It can be adapted to tackle other cases with different types of data than proteins.

Reproducible and user-controlled software management in HPC with GNU Guix

Ricardo Wurmus[*], Altuna Akalin[†]

18th Bioinformatics Open Source Conference (BOSC) 2017, Prague, CR

Website: https://gnu.org/software/guix
Repository: https://git.savannah.gnu.org/cgit/guix.git
License: GNU General Public License version 3 (or later)

Reproducibility is the corner stone of science, and there is growing awareness among researchers of the problem of reproducibility in the field of bioinformatics research[1]. Computational research fields such as bioinformatics crucially depend on the reproducibility of software packages and the computational pipelines that are made up of them. Unfortunately, traditional software deployment methods generally do not take reproducibility into account.

On high-performance computing (HPC) systems two conflicting approaches to managing software collide: system administrators manage these large systems in a highly conservative manner, whereas the researchers using these systems may require up-to-date tool chains as well as libraries and scientific software in countless variations. Users often fall back to *ad-hoc* software deployment to satisfy their immediate requirements. As a result, HPC system users often have no guarantee that they will be able to reproduce results at a later point in time, even on the same system, and they have little hope of being able to reproduce the same software environment elsewhere.

We present GNU Guix and the functional package management paradigm and show how it can improve reproducibility and sharing among researchers[2]. Functional package management differs from other software management methodologies in that reproducibility is a primary goal. With GNU Guix users can freely customize their own independent software profiles, recreate workflow-specific application environments, and publish a package set to enable others to reproduce a particular workflow, without having to abandon package management or sharing. Profiles can be rolled back or upgraded at will by the user, independent from system administrator-managed packages.

We will introduce functional package management with GNU Guix, demonstrate some of the benefits it enables for research, such as reproducible software deployment, workflow-specific profiles, and user-managed environments, and share our experiences with using GNU Guix for bioinformatics research at the Max Delbrck Center. We will also compare the properties and guarantees of functional package management with the properties of other application deployment tools such as Docker or Conda.

References

[1] Peng, R.: Reproducible Research in Computational Science. Science, 02 Dec 2011: Vol. 334, Issue 6060, pp. 1226-1227. DOI: 10.1126/science.1213847

[2] Courts, L., Wurmus, R.: Reproducible and User-Controlled Software Environments in HPC with Guix. In: Hunold, Sascha, et al., eds. Euro-Par 2015: Parallel Processing Workshops: Euro-Par 2015 International Workshops, Vienna, Austria, August 24-25, 2015, Revised Selected Papers. Vol. 9523. Springer, 2015.

[*] GNU Guix co-maintainer, System administrator, Berlin Institute of Medical Systems Biology, Max Delbrueck Center for Molecular Medicine, Berlin, Germany. Email: ricardo.wurmus@mdc-berlin.de

[†] Principal investigator, Berlin Institute of Medical Systems Biology, Max Delbrueck Center for Molecular Medicine, Berlin, Germany.

An ensemble approach for gene set testing analysis with reporting capabilities

Monther Alhamdoosh[1], Milica Ng[1], Matthew E. Ritchie[2,3]

[1]CSL Limited, Bio21 Institute, 30 Flemington Road, Parkville, Victoria 3010, Australia. Email: monther.alhamdoosh@csl.com.au

[2]Molecular Medicine Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3052, Australia.

[3]School of Mathematics and Statistics, The University of Melbourne, Parkville, Victoria 3010, Australia.

Project Website http://bioconductor.org/packages/EGSEA/
Source Code: https://github.com/Bioconductor-mirror/EGSEA/
License: GPL-2

Gene set enrichment (GSE) analysis allows researchers to efficiently extract biological insight from long lists of differentially expressed genes by interrogating them at a systems level. In recent years, there has been a proliferation of GSE analysis methods and hence it has become increasingly difficult for researchers to select an optimal GSE tool based on their particular data set. Moreover, the majority of GSE analysis methods do not allow researchers to simultaneously compare gene set level results between multiple experimental conditions. The ensemble of genes set enrichment analyses (EGSEA) is a method developed for RNA-sequencing data that combines results from twelve algorithms and calculates collective gene set scores to improve the biological relevance of the highest ranked gene sets. EGSEA's gene set database contains around 25,000 gene sets from sixteen collections. It has multiple visualization capabilities that allow researchers to view gene sets at various levels of granularity. EGSEA has been tested on simulated data and on a number of human and mouse data sets and, based on biologists' feedback, consistently outperforms the individual tools that have been combined. Our evaluation demonstrates the superiority of the ensemble approach for GSE analysis, and its utility to effectively and efficiently extrapolate biological functions and potential involvement in disease processes from lists of differentially regulated genes.

References

Alhamdoosh, M., Ng, M., Wilson, N.J., Sheridan, J.M., Huynh, H., Wilson, M.J., & Ritchie, M.E. (2017). Combining multiple tools outperforms individual methods in gene set enrichment analyses. *Bioinformatics*, 33 (3).

A-144

RADAR-CNS - Research Infrastructure for processing wearable data to improve health

Julia Kurps[1], Maxim Moinat[1], Joris Borgdorff[1], Francesco Nobilia[2], Maximilian Kerz[2], Nivethika Mahasivam[1], Irina Pulyakhina[1], Matthias Dümpelmann[4], Herculano Campos[5], Mark Begale[6], Richard Dobson[2,3], Amos Folarin[2,3]

1 The Hyve, Arthur van Schendelstraat 650, 3511 MJ Utrecht, The Netherlands
2 Department of Biostatistics and Health Informatics, Institute of Psychiatry Psychology & Neuroscience, King's College London, Box P092, 16 De Crespigny Park, SE5 8AF, UK
3 Farr Institute of Health Informatics Research, UCL Institute of Health Informatics, University College London, London WC1E 6BT, UK.
4 Center of Epilepsy, University Hospital Freiburg, Breisacher Str. 64, 79106 Freiburg, Germany
5 Goldenarm, 20 Jay Street, Suite 840, Brooklyn, NY, 11201
6 Vibrent Health, 12015 Lee Jackson Memorial Highway, Suite 13, Fairfax, VA, 2203, United States

Corresponding E-mail: julia@thehyve.nl, amos.folarin@kcl.ac.uk

Project Website: http://www.radar-cns.org/
Source Code: https://github.com/RADAR-CNS
License: Apache2.0

Remote **A**ssessment of **D**isease **A**nd **R**elapse – **C**entral **N**ervous **S**ystem (**RADAR-CNS**) is an innovative collaborative research project to evaluate the potential of wearable devices and smartphone technology to improve quality of life for people with epilepsy, multiple sclerosis (MS) and major depression disorder (MDD).

RADAR-CNS is built upon close collaboration of patient organizations, clinical partners, research institutes and industry to develop new strategies for treatment of patients with brain disorders. The aim of RADAR-CNS is to evaluate how to best leverage innovative technologies like wearable devices and smartphone-based applications for remote measurement and potential relapse prediction. Together with our data processing partners, The Hyve is building an open source infrastructure to capture, process, manage and analyse data from wearable devices and facilitate the integration with data from multiple other sources like clinical and -omics data. Our clinical partners will use this data processing pipeline in multiple clinical trials. Sustainability is a focus point during the development of the RADAR platform. Therefore, we develop a generic platform, which will not be limited to brain disorder applications, but will be applicable for subsequent RADAR projects like RADAR Diabetes. Furthermore, we are actively facilitating a striving open source community around the RADAR platform to ensure longevity of the research infrastructure and encourage cross-infrastructure efforts. Goal of the pilot study is the evaluation of wearable device data for passive remote measurement of patients in a hospital epilepsy monitoring unit (EMU). We developed an Android application, which captures wearable data via a Bluetooth connection and streams data to an internal hospital server. Integration of SDKs for multiple wearable devices allows us to investigate and compare the quality of data collected from different devices (e.g., Empatica E4 and Angel Sensor) as well as device specifications, such as battery life and signal stability/range. Besides data quality assessment, we will investigate whether data from wearable devices has potential for seizure detection. Parallel video and EEG monitoring in monitoring units of specialized epilepsy centers are used as a Gold Standard for seizure detection evaluation. We will investigate the potential of wearable devices as clinically valuable alternatives to complement or even replace hospital-based technologies; a prerequisite for ambulatory passive remote measurement of patients in their home environment.

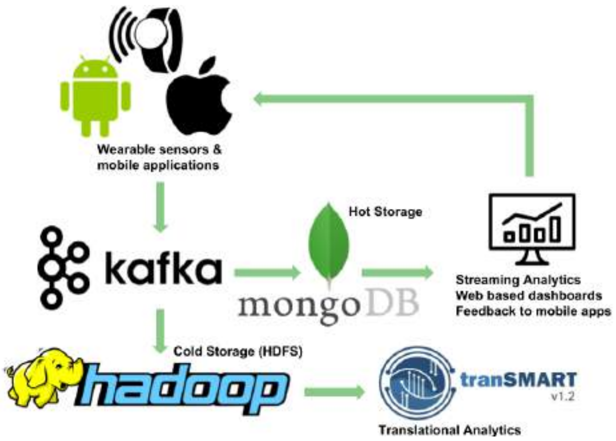


Figure1. **Technology stack of processing infrastructure.** Data is collected from wearable sensors, standardized to data schemata and processed with Apache kafka, which allows streaming analytics. Raw data is stored using Hadoop for further advanced analytics.

The RADAR-CNS platform is adding an additional dimension to the research infrastructure for personalised medicine and health by allowing new ways to leverage innovative technologies for better data integration.

References

1. http://www.radar-cns.org
2. https://github.com/RADAR-CNS

Workflows interoperability with Nextflow and Common WL.

Kevin Sayer[1], Paolo Di Tommaso[1], Evan Floden[1,2], Maria Chatzou[1], Cedric Notredame[1]

[1] Centre for Genomic Regulation (CRG), Dr. Aiguader 88, 08003 Barcelona, Spain.

[2] Universitat Pompeu Fabra (UPF), 08003 Barcelona, Spain.

Email: kevin.sayers@crg.eu.

Project Website: http://www.nextflow.io

Source Code: https://github.com/nextflow-io/nextflow

License: GPLv3

Reproducibility has become one of biology's most pressing issues. This impasse has been fuelled by the combined reliance on increasingly complex data analysis methods and the exponential growth of biological datasets. When considering the installation, deployment and maintenance of bioinformatic pipelines, an even more challenging picture emerges due to the lack of community standards. Moreover, the effect of limited standards on reproducibility is amplified by the very diverse range of computational platforms and configurations on which these applications are expected to be applied (workstations, clusters, HPC, clouds, etc.).

Nextflow[1] is a pipeline orchestration tool that has been designed to address exactly these issues. It provides a domain specific language (DSL) which streamlines the writing of complex distributed computational pipelines in a portable and replicable manner. It allows the seamless parallelization and deployment of any existing application with minimal development and maintenance overhead, irrespective of the original programming language. The built-in support for software containers guarantees numerical stability and enables truly replicable computational workflows across multiple execution platforms.

This talk will introduce the Nextflow support the Common Workflow Language[2]. CWL is a community driven specification for describing analysis workflows and tools in portable manner and that is being used by a number of institutions. The presentation will discuss how Nextflow can be used as the execution engine for workflows defined by using the CWL specification, the benefits and disadvantages of this approach, the current limitations and open challenges of this project.

As a proof of concept, we will show how community based CWL pipelines can be deployed by using the Nextflow execution runtime.

References

1. Nextflow enables reproducible computational workflows (Nature Biotech, April 2017, doi:10.1038/nbt.3820)
2. Common WL, https://dx.doi.org/10.6084/m9.figshare.3115156.v2

Bioschemas for life science data

Carole Goble[1], Rafael Jimenez[2], Alasdair Gray[3], Niall Beard[1], Giuseppe Profiti[4], Norman Morrison[2]

[1] The University of Manchester, UK / ELIXIR-UK, Email: carole.goble@manchester.ac.uk

[2] ELIXIR-Hub, Hinxton Genome Campus, UK.

[3] Heriot-Watt University, UK / ELIXIR-UK.

[4] Università di Bologna, Italy / ELIXIR-Italy.

Project Website: http://bioschemas.org/
Source Code: https://github.com/BioSchemas/bioschemas
License: Creative Commons Attribution-ShareAlike License (version 3.0)

Abstract

Schema.org provides a way to add semantic markup to web pages. It describes 'types' of information, which then have 'properties'. For example, 'Event' is a type that has properties like 'startDate', 'endDate' and 'description'. Bioschemas aims to improve data interoperability in life sciences by encouraging people in life science to use schema.org markup. This structured information then makes it easier to discover, collate and analyse distributed data. Bioschemas reuses and extends schema.org in a number of ways: defining a minimum information model for the datatype being described using as few concepts as possible and only where necessary adding new properties, and the introduction of cardinalities and controlled vocabularies. The main outcome of Bioschemas is a collection of specifications that provide guidelines to facilitate a more consistent adoption of schema.org markup within the life sciences for the "Find" part of the FAIR (Findable, Accessible, Interoperable, Reusable) principles.

In 2016 Bioschemas successfully piloted with training materials and events to enable the EU ELIXIR Research Infrastructure Training Portal (TeSS) to rapidly and simply harvest metadata from community sites. Encouraged by this in March 2017 we launched a 12 month project to pilot Bioschemas for data repositories and datasets. Specifically we are working on:

- General descriptions for datasets and data repositories
- Specific descriptions for prioritised datatypes: Samples, Human Beacons, Plant phenotypes and Protein annotations
- Facilitating discovery by registries and data aggregators, and by general search engines
- Facilitate tool development for annotation and validation of compliant resources

All work is grounded on describing real data resources for real use cases: to this end large and small dataset are part of the project: Pfam, Interpro, PDBe, UniProt, BRENDA, EGA, COPaKB, and Gene3D. Data aggregators participating include: InterMine, BioSamples and OmicsDI. Registries include Identifiers.org, DataMed, Biosharing and the Beacon Network. Bioschemas operates as an open community initiative, sponsored by the EU ELIXIR Research Infrastructure and is supported by the NIH BD2K programme and Google.

A-147

The GA4GH Tool Registry Service (TRS) and Dockstore - Year One

Denis Yuen[1], Brian O'Connor[2], Andrew Duncan[3] , Vincent Chung[3], Xiang Kun Liu[3], Janice Patricia[3], Han Yuan Cao[3], Gary Luu[3], Vincent Ferretti[3], Lincoln Stein[3]

[1] Ontario Institute for Cancer Research, MaRS Centre, Toronto, Ontario. Email: denis.yuen@oicr.on.ca

[2] UC Santa Cruz Genomics Institute, University of California, Santa Cruz, CA, USA

[3] Ontario Institute for Cancer Research, MaRS Centre, Toronto, Ontario.

Project Website: https://dockstore.org
Source Code: https://github.com/ga4gh/dockstore/
License: Apache License 2.0 https://www.apache.org/licenses/LICENSE-2.0.html

Workflows written for the PCAWG (Pan-Cancer Analysis of Whole Genomes) study created a challenge for the cloud projects team at OICR and our collaborators due to the highly heterogenous nature of our fourteen computing environments (cloud and HPC, commercial and academic, geographically distributed). We met the challenge by distributing our workflows in Docker containers described by a proprietary descriptor. As we wrapped up, we realized that this approach could be of use to others so we adopted CWL (Common Workflow Language) descriptors and split out the Dockstore project as its own open-source website and associated utilities. This project reached a 1.0 milestone in September 2016.

Tools registered in Dockstore are encouraged to include open-source build instructions for the Docker image, pulling in open-source binaries and/or source code into the image, being built on a publicly visible third-party service and accompanied by a programmatic descriptor of the tool including metadata. In practice, this has meant Dockerfiles and Common Workflow Language (CWL) or Workflow Description Language (WDL) files checked into GitHub in public repositories, built on Quay.io, and indexed on Dockstore.org.

We have also donated a Tool Registry Service (TRS) (https://goo.gl/zfhR4q) API to the Global Alliance for Genomics and Health (GA4GH) in the hope that like-minded groups can implement it in order to facilitate the sharing of Dockerised tools in the sciences. Currently, Dockstore indexes workflows for PCAWG, the workflows for the GA4GH-DREAM challenge, the UCSC Genomics Institute, and more while welcoming contributions from all bioinformaticians.

In this talk we also present our lessons learned creating CWL+Docker images, open-source and commercial platforms for running tools and workflows on Dockstore, and interoperability challenges when converting tools between tool registries. We also highlight new Dockstore features such as test parameters, pluggable file provisioning, private tool support, and workflow visualization.

Introducing the Brassica Information Portal: Towards integrating genotypic and phenotypic Brassica crop data

Annemarie Eckes[1], Tomasz Gubala[1,6], Piotr Nowakowski[1,6], Tomasz Szymczyszyn[1], Rachel Wells[2], Judith Irwin[2], Carlos Horro[1], John Hancock[1], Graham King[4], **Wiktor Jurkowski[1,*]**

[1]Earlham Institute, Norwich Research Park, Norwich NR4 7UZ, UK
[2]Academic Computer Centre CYFRONET, AGH University of Science and Technology, Kraków, Poland
[3]John Innes Centre, Norwich Research Park, Norwich, NR4 7UH, UK
[4]Southern Cross Plant Science, Southern Cross University, Lismore, NSW 2480, Australia
[*] correspondence to: wiktor.jurkowski@earlham.ac.uk

A new community resource, the Brassica Information Portal (BIP), provides an open access web repository for *Brassica* phenotyping data (https://bip.earlham.ac.uk). It facilitates crop improvement by filling the gap for standardised trait data and is already integrated with tools to perform on-the-fly phenotype-genotype association analysis online.

In collaboration with the UK *Brassica* Research Community, we have build a tool that aims to assist global research on *Brassica* crops: Users can store and publish their own study results in the Portal thanks to advanced data submission capabilities. Similarly, user-friendly interfaces and programmatic download mechanisms further facilitate work with the data. This makes the database content readily comparable and cross-linkable to that of other tools and resources for further downstream analysis. Integrated with the GWAS analysis tool GWASSER, it now enables the user to perform simple GWAS analysis with selected data on BIP.

We also aim to implement community-wide phenotypic trait definition standards to encourage trait data re-use. To ensure comparability, we are working on creating brassica trait ontology terms based on the Crop Ontology model suitable for association analysis. This creates the opportunity to carry out meta-analysis on datasets generated across multiple studies. To make BIP data more comparable and reusable, we are collaborating with ELIXIR to implement MIAPPE standards to the BIP schema. To make data more interoperable and accessible, we are working on being BrAPI (Breeding API) - compliant, as well as on extension of our own BIP-API.

There are huge benefits from making organised trait data available and accessible to the *Brassica* community and we hope that it becomes common practice to store brassica trait data systematically and share it with the wider research community in a similar way as has become routine for sequence, transcript, metabolite and protein structure data.

We will present the resource and a GWAS use-case to demonstrate the value and utility of BIP, both for single research groups and global *Brassica* research and breeding community.

Source code: https://github.com/TGAC/brassica
Open Source License: GNU General Public License 3.0

Discovery and visualisation of homologous genes and gene families using Galaxy

Anil S. Thanki, Nicola Soranzo, Wilfried Haerty, Robert P. Davey
Earlham Institute, Norwich Research Park, Norwich NR4 7UH, UK

Anil Thanki: Anil.Thanki@earlham.ac.uk

Source Code: https://github.com/TGAC/earlham-galaxytools

License: MIT License

The phylogenetic information inferred from the study of homologous genes helps us to understand the evolution of gene families and plays a vital role in finding ancestral gene duplication events as well as identifying regions that are under positive selection within species.

The Ensembl GeneTrees pipeline generates gene trees based on coding sequences and provides details about exon conservation, and is used in the Ensembl Compara project to discover homologous gene families. Since expertise is required to configure and run the pipeline via the command-line, we created GeneSeqToFamily, an open-source Galaxy workflow based on Ensembl GeneTrees. GeneSeqToFamily helps users to run potentially large-scale gene family analyses without requiring the command-line while still allowing tool parameters, configurations, and the tools themselves to be modified.

At present, we are using this workflow on a set of–vertebrate genomes (human, dog, chicken, kangaroo, macropod, opossum, mouse, platypus, and tasmanian devil), with some analyses comprising more than 13000 gene families. Gene families discovered with GeneSeqToFamily can be visualised using the Aequatus.js interactive tool, integrated within Galaxy as a visualisation plugin.

Handling this large number of input datasets became problematic for both Galaxy itself and certain tools such as T-Coffee which adversely affected memory allocation and file IO processes. We have made modifications to both the T-Coffee wrapper scripts and low-level changes to the Galaxy framework as a whole to help address these issues.

We are also working on integrating protein domain information from SMART (a Simple Modular Architecture Research Tool) to complement discovered gene families, as well as the incorporation of PantherDB into the workflow for validation of families.

The SPOT ontology toolkit : semantics as a service

Olga Vrousgou[1], Simon Jupp[1], Thomas Liener[1], Tony Burdett, Helen Parkinson[1]

[1] European Bioinformatics Institute (EMBL-EBI), European Molecular Biology Laboratory, Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, UK: olgavrou@ebi.ac.uk

Project Website: https://ebispot.github.io/
Web Pages and Source Code:
Ontology Lookup Service: http://www.ebi.ac.uk/ols, https://github.com/EBISPOT/OLS
Zooma: http://www.ebi.ac.uk/spot/zooma, https://github.com/EBISPOT/zooma
OxO: http://www.ebi.ac.uk/spot/oxo, https://github.com/EBISPOT/OLS-mapping-service
Webulous: https://www.ebi.ac.uk/efo/webulous, https://github.com/EBISPOT/webulous
BioSolr: https://ebispot.github.io//BioSolr, https://github.com/EBISPOT/BioSolr
License: http://www.apache.org/licenses/LICENSE-2.0
Mailing List: ontology-tools-support@ebi.ac.uk

Annotating data with ontologies adds value and results in data that is more readily interoperable, discoverable and reusable. Working with multiple ontologies and their nuances creates overhead that can be readily addressed with tooling that encapsulates common use-cases. Certain activities that are common to ontology-aided data curation include querying ontologies, mapping data to and between ontologies, and creating or requesting new ontology terms. The Samples Phenotypes and Ontologies Team (SPOT) work with multiple databases to develop tools that support high-throughput, semi-automated data curation with ontologies. These tools form a toolkit of open-source software that can be deployed and run anywhere, and can be configured to work with data and ontologies from any domain.

The SPOT ontology toolkit aims to reduce the barriers to entry in the annotation of data with ontologies. The Ontology Lookup Service (OLS) provides a repository for accessing and visualising multiple ontologies. Zooma provides a repository of curated annotation knowledge that can be used to predict ontology annotations with a measure of confidence that enables the automated curation of unseen data with ontologies. OxO provides access to curated ontology cross-references and provides services for mapping from one ontology standard to another. Webulous supports ontology development from spreadsheets through a google sheet add-on. OLS, Zooma, OxO and Webulous come with integrated RESTful APIs that allow for scalable programmatic access and the development of data curation pipelines. Finally BioSolr, an ontology expansion plugin for Solr and Elasticsearch, demonstrates how advanced ontology-powered search can be achieved over data enriched with ontology annotation.

OLS, Zooma, OxO and Webulous combined address a specific need for a suite of tools that can automate more of the process of data curation with ontologies. Here we will present this toolkit, along with a suggested ontology annotation workflow designed to lower the cost of life sciences data curation.

GRAPHSPACE: Stimulating interdisciplinary collaborations in network biology

Aditya Bharadwaj[1], Divit P. Singh[1], Anna Ritz[2], Allison N. Tegge[3], Christopher L. Poirel[4], Pavel Kraikivski[5], Neil Adames[6], Kurt Luther[1], Shiv D. Kale[7], Jean Peccoud[6], John J. Tyson[5], and T. M. Murali[1]

[1]Dept. of Computer Science, Virginia Tech, email: adb@vt.edu, [2]Biology Dept., Reed College, [3]Dept. of Statistics, Virginia Tech, [4]RedOwl Analytics, [5]Dept. of Biological Sciences, Virginia Tech, [6]Dept. of Chemical and Biological Engineering, Colorado State University, [7]Biocomplexity Institute, Virginia Tech

Project Website: `http://graphspace.org`
Source Code: `https://github.com/Murali-group/GraphSpace`
License: GNU General Public License v3

Computational analysis of molecular interaction networks has become pervasive in systems biology. Despite the existence of several software systems and interfaces to analyze and view networks, interdisciplinary research teams in network biology face several challenges in sharing, exploring, and interpreting computed networks in their collaborations.

GRAPHSPACE is a web-based system that provides a rich set of user-friendly features designed to stimulate and enhance network-based collaboration:

- Users can upload richly-annotated networks, irrespective of the algorithms or software used to generate them. GRAPHSPACE networks follow the JSON format supported by Cytoscape.js [1]. Users of Cytoscape [3] can export their networks and upload them directly into GRAPHSPACE.
- Users can create private groups, invite other users to join groups, and share networks with groups.
- A user may search for networks with a specific property or that contain a specific node or collection of nodes.
- A powerful layout editor allows users to efficiently modify node positions, edit node and edge styles, save new layouts, and share them with other users.
- Researchers may make networks public and provide a persistent URL in a publication, enabling other researchers to explore these networks.
- A comprehensive RESTFul API streamlines programmatic access to GRAPHSPACE features.
- A Python module called `graphspace_python` allows a user to rapidly construct a graph, set visual styles of nodes and edges, and then upload the graph, all within tens of lines of code. It is very easy to integrate this script into a user's software pipeline.

Currently, GraphSpace supports more than 100 users who have stored more than 21,000 graphs (most of them private) containing a total of over 1.4 million nodes and 3.8 million edges. Conceptually, GRAPHSPACE serves as a bridge between visualization and analysis of individual networks supported by systems such as Cytoscape [3] and the network indexing capabilities of NDex [2]. We anticipate that GRAPHSPACE will find wide use in network biology projects and will assist in accelerating all aspects of collaborations among computational biologists and experimentalists, including preliminary investigations, manuscript development, and dissemination of research.

References

[1] M. Franz, C. T. Lopes, G. Huck, Y. Dong, O. Sumer, and G. D. Bader. Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics*, 32:309–311, Sep 2015.

[2] D. Pratt, J. Chen, D. Welker, R. Rivas, R. Pillich, V. Rynkov, K. Ono, C. Miello, L. Hicks, S. Szalma, A. Stojmirovic, R. Dobrin, M. Braxenthaler, J. Kuentzer, B. Demchak, and T. Ideker. NDEx, the Network Data Exchange. *Cell Syst*, 1(4):302–305, Oct 2015.

[3] M. E. Smoot, K. Ono, J. Ruscheinski, P. L. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27(3):431–432, Feb 2011.

A-152

Revitalizing a classic bioinformatics tool using modern technologies: the case of the Cytoscape Project

Keiichiro Ono[1], Eric Sage[1], Barry Demchak[1]
[1] University of California, San Diego. La Jolla, CA USA. Email: {kono, edsage, bdemchak}@ucsd.edu

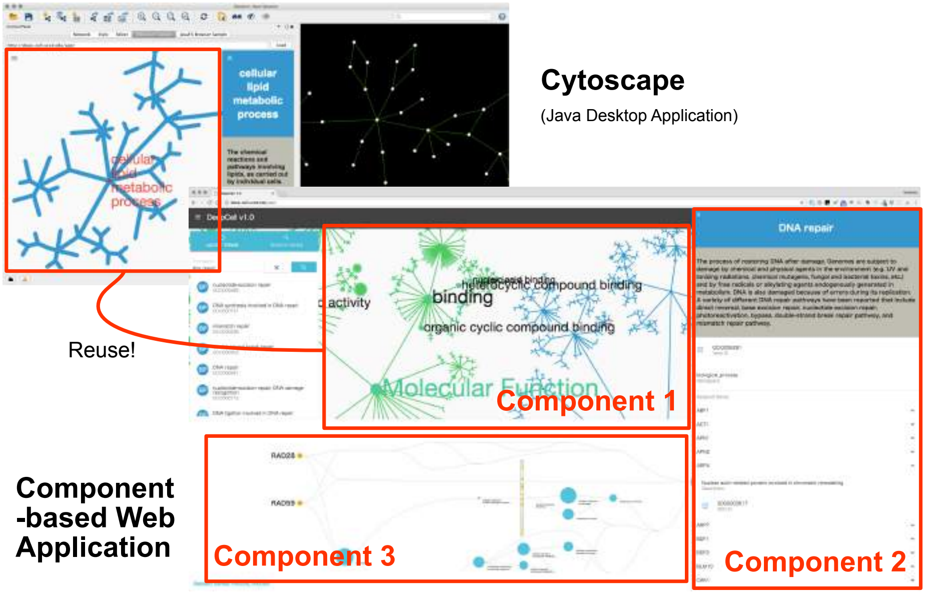
Project Website: http://www.cytoscape.org/
Source Code: https://github.com/cytoscape/
License: LGPL/MIT License

Abstract

To date, widely-used bioinformatics tools are often based on long established programming systems that are unable to effectively leverage or interoperate with new tools developed on modern programming systems running on modern web platforms. Abandoning and rewriting existing tools represents both risky and large investments of both calendar and monetary resources. In this presentation, we will discuss new approaches that enable small bioinformatics software developer teams to economically construct bridges between existing applications and modern web-based tools. We will highlight construction and use of portable and reusable UI components and computational services.

For our case study, we use Cytoscape, the de-facto standard network visualization platform in biology. Its first version was released in 2002, and it is still actively developed by the Cytoscape Consortium. It is a desktop Java-based plugin architecture, and third party developers have released over 300 Java-based apps – these apps represent a large and valuable ecosystem that cannot be abandoned even if Cytoscape itself were to be redeveloped (e.g., as a microservice architecture or single-page web application). Additionally, because Java is a relatively closed programming system, there is no easy way for Cytoscape core or apps to integrate emerging web technology that implements a complex user interface and or data visualization modules. This hurdle is often insurmountable for open source software (OSS) projects (especially in bioinformatics) that have limited resources.

We will discuss how we design, implement and integrate reusable JavaScript-based UI components (called *cyWidgets*) into Java-based Cytoscape apps, and how we build, deploy and access non-Java services across the web.


Cytoscape
(Java Desktop Application)
cellular lipid metabolic process
Reuse!
DeepCell v1.0
activity
binding
heterocyclic compound binding
organic cyclic compound binding
Molecular Function
Component 1
DNA repair
RAD23
RAD33
Component -based Web Application
Component 3
Component 2

Screw: tools for building reproducible single-cell epigenomics workflows

Kieran O'Neill[1,2], B Decato[3], A Goncearenco[4], A Khandekar[4], B Busby[4], and A Karsan[1,2]

[1] Pathology Department, University of British Columbia, Vancouver, Canada
[2] Michael Smith Genome Sciences Centre, BC Cancer Agency, Vancouver, Canada
[3] Molecular & Computational Biology Department, University of Southern California, Los Angeles, California, USA
[4] National Center for Biotechnology Information, National Institutes of Health, Bethesda, Maryland, USA

DNA methylation is a heritable epigenetic mark that shows a strong correlation with transcriptional activity. The gold standard for detecting DNA methylation is whole genome bisulfite sequencing (WGBS). Recently, WGBS has been performed successfully on single cells (SC-WGBS) [1]. The resulting data represents a fundamental shift in the capacity to measure and interpret DNA methylation, especially in rare cell types and contexts where subtle cell-to-cell heterogeneity is crucial, such as in stem cells or cancer. However, SC-WGBS comes with unique technical challenges which require new analysis techniques to address. Furthermore, although some software tools have been published, and several existing studies have tended to use similar methods, no standardized pipeline for the analysis of SC-WGBS yet exists.

Simultaneously, there has been a drive within bioinformatics towards improved reproducibility. Textual descriptions of bioinformatic analyses are deeply inadequate, and often require "forensic bioinformatics" to reproduce [2]. Recreating the exact results of a study requires not only the exact code, but also the exact software (down to version, compilation options, etc). Common Workflow Language (CWL) provides a framework for specifying complete workflows, while Docker allows for bundling of the exact software and auxiliary data used in an analysis within a container that can be executed anywhere. Together, these have the potential, via repositories such as Dockstore [3], to enable completely reproducible bioinformatics research.

Here we present Screw (Single Cell Reproducible Epigenomics Workfow). Screw is a collection of standard tools and workflows for analysing SC-WGBS data, implemented in CWL, with an accompanying Docker image. Screw provides the parts for constructing fully-reproducible SC-WGBS analyses. Tools provided include quality control visualization, clustering and visualisation of cells by pairwise dissimilarity measures, construction of recapitulated-bulk methylomes from single cells of the same lineage, generation of bigWig methylation tracks for downstream visualization, and wrappers around published tools such as DeepCpG [4] and LOLA [5]. Screw has the added benefit that CWL's compatibility with interactive GUI-based workflow tools such as Galaxy can lower the barriers to use for less-technical wet lab biologist users.

CWL sources for Screw are available under the MIT license at https://github.com/Epigenomics-Screw/Screw. Tools and workflows are available from Dockstore under Epigenomics-Screw namespace, for example https://dockstore.org/workflows/Epigenomics-Screw/Screw/screw-preprocess

1. Schwartzman, Tanay (2015) Nature Reviews Genetics 16:716–26.

2. Gentleman (2005) Statistical applications in genetics and molecular biology. doi: 10.2202/1544-6115.1034

3. O'Connor et al. (2017) F1000Research 6:52.

4. Angermueller, Lee, Reik, Stegle (2016) bioRxiv 055715.

5. Sheffield, Bock (2015) Bioinformatics 32:587–589.

Emerging public databases of clinical genetic test results: Implications for large scale deployment of precision medicine

Stephen Lincoln[1], Shan Yang[1], Benedict Paten[2], Melissa Cline[2], Yuya Kobayashi[1], Can Zhang[2], Scott Topper[1], David Haussler[2], Robert Nussbaum[1,3]

[1] Invitae, San Francisco, California. [2] University of California, Santa Cruz. [3] University of California, San Francisco. Corresponding author email: steve.lincoln@me.com

Implementing precision medicine requires that reliable and consistent clinical interpretations of molecular test results be available to physicians. In germline genetic testing the process of determining which variants in a patient are pathogenic (disease causing) and which are benign is crucial to the clinical utility of these tests. This classification process involves expert review of complex and sometimes conflicting evidence, after which some variants must be considered of uncertain significance. To promote quality control, collaboration and consensus building, many (not all) clinical laboratories contribute classifications to open, public databases, notably ClinVar.

We examined 74,065 classifications of 27,224 clinically observed genetic variants in ClinVar, and found inter-laboratory concordance to be high (96.7%) although this varied considerably among specialties: cancer genes had the highest concordance (98.5%) with cardiology (94.2%) and metabolic disease (95.1%) the lowest. Unsurprisingly, data from research labs were 6-times more likely to disagree with clinical test results, and old classifications often disagreed with newer ones. More interestingly, evidence supporting pathogenicity appears to be more consistently used than evidence against pathogenicity (data to be shown). Low penetrance genes, which confer a modest risk of disease, were 7-times more likely to harbor disagreements compared to high penetrance genes. Technically challenging variant types (e.g. large indels and mutations in low complexity or highly homologous sequences) were under-represented in ClinVar even though independent data from an 80,000 patient cohort shows that 9% of pathogenic variants in patients are of these types.

We further examined variants in BRCA1 and BRCA2, genes which can confer a substantial lifetime risk of breast, ovarian, and other cancers. While BRCA1/2 testing is common it remains controversial, because (a) proposals to implement population scale BRCA1/2 testing as part of routine physical exams have been raised, (b) irreversible preventive options (e.g. prophylactic surgery) are offered to otherwise healthy individuals found to carry germline mutations, and (c) the largest laboratory with data to inform such testing (Myriad) refuses to share that data or provide it for detailed community review. We analyzed physician provided Myriad results along with those from other ClinVar laboratories, finding high agreement (98.5%) in a data set representing over 20,000 tested patients. Moreover, the few discordant variants were all very rare, suggesting that 99.8% of patients would receive concordant tests from any of these labs in routine practice. This confirms a similar result of ours in a prospective 1000-patient clinical trial.

Open data sharing via ClinVar is a powerful mechanism which has already helped to uncover critical factors which must be addressed as new precision medicine approaches evolve in various medical specialties. ClinVar also provides a common mechanism to drive consensus and evaluate inter-laboratory performance, as exemplified by this study.

Note: All data from these studies are publicly available without restriction for analysis by the community. Some of the results described in this abstract are currently in press. Prior published work mentioned includes Lincoln et al. J Mol Diag 2015 and Yang et al, PSB 2016.

A-155

NGL – a molecular graphics library for the web

Alexander S. Rose[1,2], Stephen K. Burley[1,2,3]

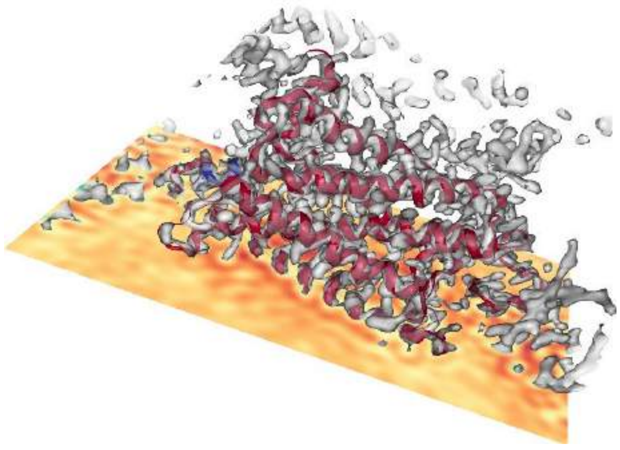
[1]RCSB Protein Data Bank [2]San Diego Supercomputer Center, UC San Diego [3]Rutgers, The State University of New Jersey

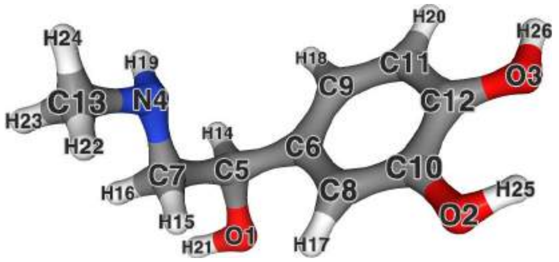
Source Code & Project Website: https://github.com/arose/ngl/
License: MIT license

Interactive visualization and annotation of large macromolecular complexes on the web is becoming a challenging problem as experimental techniques advance at an unprecedented rate. Integrative/Hybrid approaches are increasing being used to determine 3D structures of biological macromolecules by combining information from X-ray crystallography, NMR spectroscopy, and cryo-electron microscopy with data from diverse complementary experimental and computational methods. The wealth, size and complexity of available and future structures make scalable visualization and annotation solutions more important than ever. The web can provide easy access to the resulting visualizations for all interested parties, including colleagues, collaborators, reviewers and educators. Herein, we utilize the web-based NGL library to provide 3D visualization of experimental and computational data integrated with general molecular graphics.

The NGL library has a versatile API to control every aspect from data loading, processing and rendering. A distinguishing feature of NGL is its scalability to system with a million atoms and more. Further, the library supports many file formats for small molecules, macromolecular structures, molecular dynamics trajectories, maps for crystallographic, microscopy and general purpose volumetric data. Annotations can be loaded from text, json, msgpack or xml files. A wide array of customizable representations is available. Molecular data can be displayed as balls, sticks, cartoons, surfaces and labels or with specialized representations such as hyperballs and ropes. Volumetric data can be rendered as isosurfaces, point clouds or volume slices. Additional file parsers and data representations can be added through a plugin system.

The NGL library allows developers to create custom visualization solutions for specialized or novel 3D data derived from bioinformatics calculations and biophysical/biochemical experiments. The resulting interactive visualizations enable spatial understanding and exploratory analyses. Furthermore, these web-based tools simplify data exchange and foster collaborative analysis.





ToolDog - generating tool descriptors from the ELIXIR tool registry

Kenzo-Hugo Hillion[1], Ivan Kuzmin[2], Hedi Peterson[2], Jon Ison[3], Hervé Ménager[1]

[1]Bioinformatics and Biostatistics HUB, C3BI, Institut Pasteur, France. Email: kehillio@pasteur.fr

[2] Institute of Computer Science, University of Tartu, Estonia.

[3]Center for Biological Sequence Analysis Department of Systems Biology, Technical University of Denmark, Denmark.

Over the last years, the use of bioinformatics tools has been eased by the use of workbench systems such as Galaxy [1] or frameworks that use the Common Workflow Language (CWL) [2]. Still, the integration of these resources in such environments remains a cumbersome, time consuming and error-prone process. A major consequence is the incomplete description of tools that are often missing information such as some parameters, a description or metadata.

ToolDog (Tool DescriptiOn Generator) is the main component of the Workbench Integration Enabler service of the ELIXIR bio.tools registry [3, 4]. The goal of this tool is to guide the integration of tools into workbench environments. In order to do that, ToolDog is divided in two main parts: the first part analyses the source code of the bioinformatics software with language dedicated tools and generates a Galaxy XML or CWL tool description. Then, the second part is dedicated to the annotation of the generated tool description using metadata provided by bio.tools. This annotator can also be used on its own to enrich existing tool descriptions with missing metadata such as the recently developed EDAM annotation.

References:

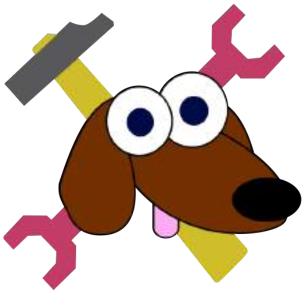
[1] Enis Afgan et al. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. Nucleic Acids Research (2016) doi: 10.1093/nar/gkw343

[2] Amstutz, Peter et al. (2016): Common Workflow Language, v1.0. figshare. https://doi.org/10.6084/m9.figshare.3115156.v2. Retrieved: 15 37, Mar 09, 2017 (GMT)

[3] Jon Ison et al. Tools and data services registry: a community effort to document bioinformatics resources. Nucleic Acids Research, 44(D1):D38–D47, January 2016. ISSN 0305-1048, 1362-4962. doi: 10.1093/nar/gkv1116.

[4] https://bio.tools

Project Website: https://github.com/bio-tools/tooldog/
Source Code: https://github.com/bio-tools/tooldog/
License: MIT



A-157

Integrating cloud storage providers for genomic analyses

Ted Liefeld[1], Marco Ocana[2], Michael Reich[1], Helga Thorvaldsdottir[2], Jill P Mesirov[1]

[1] The University of California San Diego, San Diego, CA, USA. Email: jliefeld@cloud.ucsd.edu

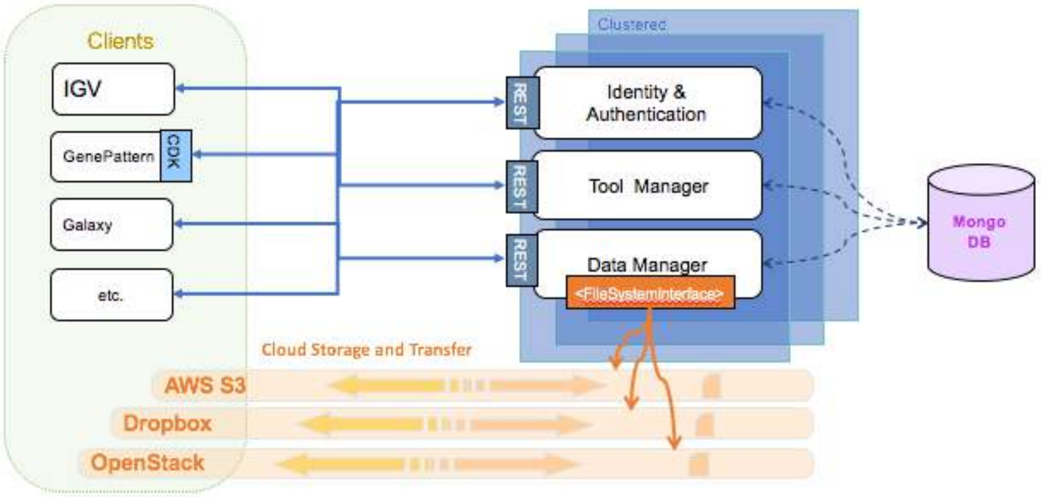
[2] The Broad Institute of MIT and Harvard, Cambridge, MA, USA.

Project Website: http://www.genomespace.org/
Source Code: https://bitbucket.org/GenomeSpace/combined
License: LGPL version 2.1 (http://www.opensource.org/licenses/lgpl-2.1.php)

Cloud storage is being used ever more frequently for genomics analysis due to its extreme scalability and constantly declining costs. Numerous cloud storage providers have been used in projects including Google, Amazon AWS, Dropbox, and hybrid clouds using tools such as openStack Swift. Existing genomic analysis tools offer disparate support for these cloud storage platforms. Some, but not all, of these tools run on one of the common cloud providers. To improve analysis throughput we would like the analysis and tools to be as closely co-located as possible, or failing that, for there to be wide bandwidth transfer between the location of the data and the location of the analysis.

To address this problem, and simultaneously avoid the m*n solutions necessary to integrate storage providers with the analysis providers, we have developed GenomeSpace (www.genomespace.org), a free and open source cloud-based environment that provides interoperability between best-of-breed computational tools. GenomeSpace includes an interface (API) for analysis tools to securely access data in the cloud as well as a platform that makes it easy to add additional tools and cloud storage providers. The GenomeSpace FileSystemInterface and StorageSpec API provide a structured way to connect cloud storage provider accounts with their GenomeSpace account, and make its contents available to all GenomeSpace tools through the GenomeSpace DataManager REST-ful API. The FileSystemInterface uses familiar file metaphors, but works equally well object stores such as Amazon S3 and OpenStack Swift.

In this talk we will describe the details of the data management architecture and interfaces in GenomeSpace that facilitate the connection of multiple cloud storage systems with a large and growing collection of analysis tools.



Full-stack genomics pipelining with GATK4 + WDL + Cromwell

Kate Voss[1], Jeff Gentry[2], Geraldine Van der Auwera[3]

[1] Broad Institute, Cambridge, MA, USA. Email: kvoss@broadinstitute.org

[2] Broad Institute, Cambridge, MA, USA. Email: jgentry@broadinstitute.org

[3] Broad Institute, Cambridge, MA, USA. Email: vdauwera@broadinstitute.org

Project Websites:
http://software.broadinstitute.org/gatk
http://software.broadinstitute.org/wdl

Source Code:
https://github.com/broadinstitute/gatk/
https://github.com/broadinstitute/cromwell
https://github.com/broadinstitute/wdl

License: BSD 3-clause (see https://github.com/broadinstitute/gatk/blob/master/LICENSE.TXT)

Main Text of Abstract

GATK4 is the new major version of the Genome Analysis Toolkit (GATK), one of the most widely used software toolkits for germline short variant discovery and genotyping in whole genome and exome data. For genomics analysts, this new version greatly expands the toolkit's scope of action within the variant discovery space and provides substantial performance improvements with the aim of shortening runtimes and reducing cost of analysis. But it also offers significant new advantages for developers, including a completely redesigned and streamlined engine that provides more flexibility, is easier to develop against, and supports new technologies such as Apache Spark and cloud platform functionalities (e.g. direct access to files in Google Cloud Storage).

WDL and Cromwell are a Workflow Definition Language and a workflow execution engine, respectively. The imperative that drives WDL's development is to make authoring analysis workflows more accessible to analysts and biomedical scientists, while leaving as much as possible of any runtime complexity involved to the execution engine.

GATK4, WDL and Cromwell are all developed by the Data Sciences Platform (DSP) at the Broad Institute and released under a BSD 3-clause license. For more information on GATK's recent licensing change, please see https://software.broadinstitute.org/gatk/blog?id=9645.

Taken together, these three components constitute a pipelining solution that is purposely integrated from the ground up, although they can each be used independently and in combination with other packages through features that maximize interoperability. This principle of integration applies equally to development, to deployment in production at the Broad, and to support provided to the external community.

BioXSD
BioJSON BioYAML

Towards unified formats for sequences, alignments, features, and annotations

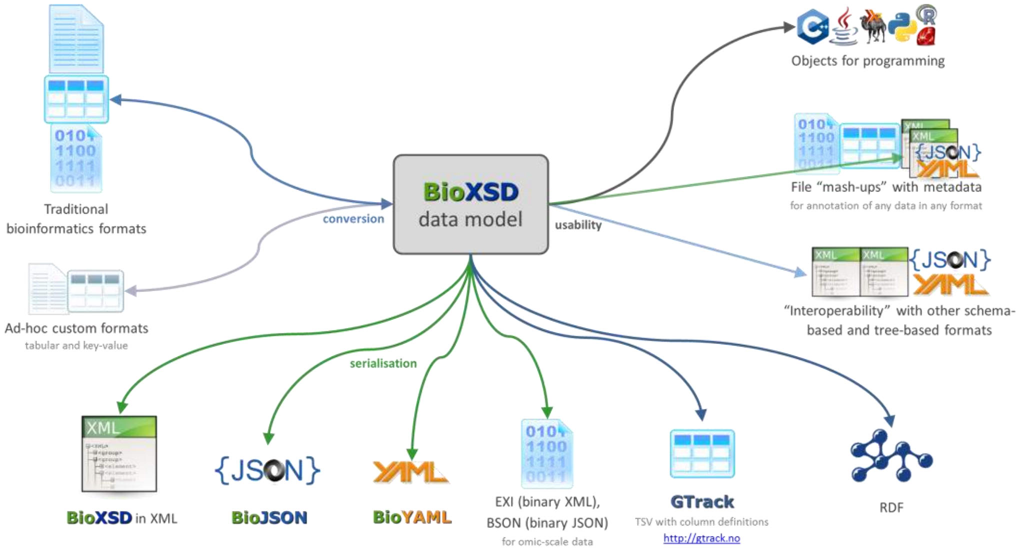
Matúš Kalaš[1], Sveinung Gundersen[2], László Kaján[3], Hervé Ménager[4], Jon Ison[5], Steve Pettifer[6], Christophe Blanchet[7], Rodrigo Lopez[8], Kristoffer Rapacki[5] and Inge Jonassen[1], and open for contributions

[1] Computational Biology Unit, Department of Informatics, University of Bergen, Bergen, Norway, **Email:** matus.kalas@uib.no; [2] Department of Informatics, University of Oslo, Oslo, Norway; [3] unaffiliated (previously Bioinformatics and Computational Biology Department, Technische Universität München, Garching, Germany); [4] Institut Pasteur, Paris, France; [5] Center for Biological Sequence Analysis, Department of Systems Biology, Technical University of Denmark, Lyngby, Denmark; [6] School of Computer Science, University of Manchester, Manchester, UK; [7] French Institute of Bioinformatics, Gif-sur-Yvette, France; [8] European Bioinformatics Institute, EMBL, Hinxton, UK.

Project website: http://bioxsd.org
Source code: https://github.com/bioxsd/bioxsd
Licence: Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).
Additional Code of Conduct for development and derived work is stated in BioXSD-1.1.xsd.
Contact: developers@bioxsd.org, @BioXSD

BioXSD has been developed as a tree-structured data model and exchange format for basic bioinformatics data, centered around bio-polymer sequence. BioXSD allows integration of diverse features, information, measurements, and inferred values about a biological molecule or its part or context, annotated with provenance and reliability metadata, ontology concepts, scientific remarks, and conclusions.

BioJSON and BioYAML are alternatives to the XML serialization of BioXSD, following the same data model. As tree-structured data formats, BioXSD, BioJSON, and BioYAML are particularly suitable for programming in object-oriented languages, and for use with web applications and web APIs (Web services), while at the same time allowing a reasonable level of human readability. BioXSD|BioJSON|BioYAML are now further developed together with GTrack (a universal tabular format for sequence features), under the umbrella of ELIXIR Norway. Work on convenient conversion, validation, and manipulation tools is ongoing (see figure).


Objects for programming
Traditional bioinformatics formats
conversion
BioXSD data model
usability
File "mash-ups" with metadata
for annotation of any data in any format
XML
{JSON}
YAML
Ad-hoc custom formats
tabular and key-value
"Interoperability" with other schema-based and tree-based formats
serialisation
XML
BioXSD in XML
{JSON}
BioJSON
YAML
BioYAML
EXI (binary XML), BSON (binary JSON)
for omic-scale data
GTrack
TSV with column definitions
http://gtrack.no
RDF

EDAM The ontology of bioinformatics operations, types of data, topics, and data formats (2017 update)

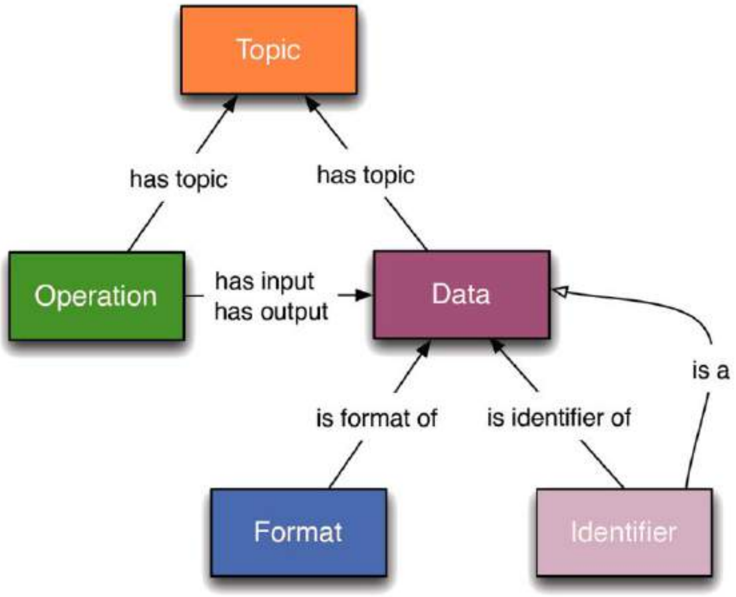
Matúš Kalaš[1], Hervé Ménager[2], Veit Schwämmle[3], Jon Ison[4], and the EDAM contributors[5]

[1] Computational Biology Unit, Department of Informatics, University of Bergen, Bergen, Norway, **Email:** matus.kalas@uib.no; [2] Institut Pasteur, Paris, France; [3] Department of Biochemistry and Molecular Biology, University of South Denmark, Ødense, Denmark; [4] Center for Biological Sequence Analysis, Department of Systems Biology, Technical University of Denmark, Lyngby, Denmark; [5] http://edamontologydocs.readthedocs.io/en/latest/contributors.html

Project website: http://edamontology.org
Source code: https://github.com/edamontology/edamontology
Licence: Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0)
Contact: edam-dev@elixir-dk.org, @edamontology

EDAM is an ontology of well established, familiar concepts that are prevalent within bioinformatics, including types of data and data identifiers, data formats, operations, and topics. EDAM has a relatively simple structure (see figure), and comprises a set of concepts with terms, synonyms, definitions, relations, links, and some additional information (especially for data formats).

8 consecutive stable versions of EDAM have been released since July 2015 (version 1.10), with version 1.17 being the current one at the time of the abstract update. EDAM is developed in a participatory and transparent fashion, with a growing community of contributors. The development of EDAM is coordinated with the development and curation of the Bio.Tools registry (http://bio.tools), the Debian Med community (http://debian.org/devel/debian-med), the Common Workflow Language (http://commonwl.org), SEQanswer SEQwiki (http://seqanswers.com), and other related communities and initiatives, including developers of various bioinformatics workbenches (including Galaxy, http://usegalaxy.org).


Topic
has topic
has topic
Operation
has input
has output
Data
is a
is format of
is identifier of
Format
Identifier