Welcome to BOSC 2019! The Bioinformatics Open Source Conference has been held annually since 2000, usually 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.

BOSC Sponsors

CollaborationFest Sponsors

Sponsorships from private companies and organizations help to defray some of our costs and enable us to grant more travel fellowships. We are grateful to our sponsors: Amazon Web Services (virtually unlimited infrastructure and fast networking for scalable HPC), Google Cloud (partnering with you to engineer a healthier world), eLife (cutting-edge technology for cutting-edge research), The Hyve (open source solutions for bioinformatics), GigaScience (an open access, open data, open peer-review journal), PLoS Computational Biology (making research open-access and freely available), and KNIME (software for end-to-end data science).

BOSC is a community effort. We thank all who made it possible, including the organizing committee, the review committee, the session chairs, our sponsors, the presenters and attendees, and the ISCB.

BOSC 2019 Organizing Committee

Nomi Harris (Chair)

Heather Wiencko (Co-Chair), Peter Cock, Chris Fields, Bastian Greshake Tzovaras, Michael Heuer, Karsten Hokamp, Monica Munoz-Torres, Bastian Rieck, Yo Yehudi

Review Committee

Kai Blin*, Christian Brueffer*, Brad Chapman, Anamaria Crisan, Arun Decano*, Gianluca Della Vedova*, Christopher Fields*, Konrad Förstner*, Bastian Greshake Tzovaras*, Nomi Harris*, Michael Heuer*, Karsten Hokamp*, Anisha Keshavan, Farah Z Khan, Radhika Khetani*, Aleix Lafita*, Jessica Maia, Hervé Ménager, Monica Munoz-Torres, Frank Nothaft, Kieran O'Neill, Konstantin Okonechnikov, Lorena Pantano*, Bastian Rieck, Surya Saha*, Malvika Sharan, Nicole Vasilevsky*, Heather Wiencko, Jason Williams, Yo Yehudi*

* reviewed both rounds

If you are interested in reviewing abstracts next year, 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 Data Crunching, Data Modeling and Formats, Open Data, Containers, Workflows, Open Science, and Building Open Source Communities, as well as a session of late-breaking lightning talks. The longer talks this year are 17 minutes (plus 3 minutes for questions); lightning talks are 5 minutes, with a short time allocated for questions at the end of each group of lightning talks.

Keynote: Nicola Mulder

Building infrastructure for responsible open science in Africa

Due to a history of exploitation and inequitable scientific partnerships, many African researchers are reluctant to fully embrace open science practices. Recent investment in genomics research on the continent and associated capacity development initiatives have enabled the development of research infrastructures and data related skills. This is helping to narrow the gap in expertise and access to data analysis capacity and facilitate more equitable engagement in international collaboration or more importantly, more independent research. H3ABioNet is a Pan African bioinformatics network that has been instrumental in building capacity for genomics data analysis on the continent. The network has an ethos of openness and is promoting open science practices among its members. This is exercised through many different activities, including open source software and workflow development, open science

training, and efforts to make our data, tools and training materials FAIR. Though the genomic data we work with is controlled access, H3ABioNet is working to ensure the data is findable, harmonized and interoperable to increase the value for both data providers and users who are granted access for responsible secondary use. In this talk I will describe some of our activities in data, tool and training material curation, standardization and dissemination. Our approach considers past inequities and tries to promote responsible openness that ensures protection of privacy and recognition of scientific contributions.

Prof. Nicola Mulder heads the Computational Biology Division at the University of Cape Town (UCT) in South Africa and leads H3ABioNet, a Pan-African Bioinformatics Network of 28 institutions in 16 African countries. H3ABioNet is developing bioinformatics capacity to enable genomic data analysis on the continent. Prior to her position at UCT, she worked at the European Bioinformatics Institute in Cambridge UK as a Team Leader for bioinformatics resources. At UCT her research focuses on genetic determinants of susceptibility to disease, African genome variation, microbiomes, genomics and infectious diseases from the host and pathogen perspectives. Prof Mulder is actively involved in training and education, including bioinformatics curriculum development. She is a member of the Board of Directors for the International Society for Computational Biology (ISCB); she co-chairs the Nominations and Education Committees.

Posters

Check https://www.open-bio.org/events/bosc/schedule/ for your poster number.

- Setup: Day 1 (Wednesday, 24-July), 7:30-10:00am
- Poster session: Day 1 (Wednesday, 24-July), 6:00-8:00pm
- Takedown: Posters should be removed by 2pm on the last day of BOSC/ISMB.

If you want to get some food before the poster session, some quick places nearby are listed on this map. The closest is Thai House (Clarastrasse 34), which is open for takeout even though the sit-down restaurant is closed. Note that many restaurants and stores in Basel are closed in July. The ISMB has a compiled a list of nearby restaurants, but they may not have checked whether they're open in July.

Birds of a Feather (BoFs)

BoFs are informal, self-organized meetups focused on specific topics. They're a great way to meet other like-minded community members and have in-depth discussions on a topic of interest.

Anyone is welcome to propose a BoF! All you need is a title, an organizer, and a brief description. BoFs are held during lunchtime each day as well as during an additional BOSC-only BoF session (4:40-5:40pm on the first day of BOSC, 24-July). Visit bit.ly/BOSC2019-bofs to see the schedule or propose your own BoF! Please note that last-minute BoFs will not get rooms; you will need to find a suitable lobby or cafe in which to meet.

Pay-your-own-way dinner

We invite you to join BOSC organizers and attendees at an optional pay-your-own-way dinner on the last evening of BOSC **(Thursday, 25 July) at 7:30pm**. We have reserved seating for 30 people at the Basel Markthalle at Steinentorberg 20 (the same place as the ISMB reception on Tuesday, July 23). The Markthalle is like a big food court: you order your food at one of the counters and then sit down. Our reserved seating area will be near the house bar--look for the BOSC sign.

Sign up to attend at http://bit.ly/BOSC2019-dinner. Dinner attendees are responsible for paying for their own food and drinks (cash preferred). Please order all drinks at the house bar.

OBF CollaborationFest (CoFest)

In conjunction with BOSC, the Open Bioinformatics Foundation runs a collaborative event (formerly called CodeFest, short for coding festival, and now called CollaborationFest, or CoFest for short). At these events, participants work together to contribute to bioinformatics software, documentation, training materials, and use cases.

This year's CoFest will take place the two days after BOSC, July 26-27, at The Swiss Innovation Hub for Personalized Medicine in Basel, Switzerland. Registration is free; sponsorships offset the cost of venue, coffee and snacks. Join the CoFest Gitter to be part of the action!

OBF Travel Fellowships

The Open Bioinformatics Foundation (OBF) sponsors a Travel Fellowship program aimed at increasing diverse participation at events (such as BOSC) that promote open source bioinformatics software development and open science in the biological research community. The next application deadline is August 15. You can apply for funding for events you plan to attend or that you recently attended.

Joining OBF

Anyone involved in open science or open source software or the life sciences is invited to join BOSC's parent organization, the Open Bioinformatics Foundation (OBF). You can find information on how to join OBF on our website, open-bio.org. Anyone who is involved in some way in open source or open science is welcome to join; there is no membership fee..

If you'd like to meet some of the OBF Directors and members, please join us at the "Welcome to BOSC" Birds of a Feather (BoF) session on the first day of the conference during lunch or the OBF Board Meeting BoF on the second day during lunch.

Stay in touch

Here are ways to stay in touch with the BOSC community before, during and after the meeting:

- Our website (https://www.open-bio.org/events/bosc/) -- check the schedule page for any updates to the program
- Our public Gitter room: https://gitter.im/OBF/BOSC_community
- Follow us on Twitter (@OBF_BOSC) and help us get the word out about BOSC by (re)tweeting (use hashtag #BOSC2019)
- Join the low-traffic bosc-announce mailing list

We look forward to meeting you in Basel! If you have any questions, you can contact the BOSC organizing committee at bosc@open-bio.org.

BOSC 2019 Schedule at a Glance

Day 1 (Wednesday, July 24)	
Time	**Session**
8:15-10:15	ISMB announcements & keynote; coffee break
10:15-12:40	*Session 1*
	BOSC opening remarks; OBF update; GSoC update
	Session: Data crunching
	Session: Data modeling and formats
12:40-14:00	Lunch, BoFs
14:00-16:00	*Session 2*
	Keynote: Nicola Mulder
	Session: Open data
16:00-16:40	Coffee break
16:40-17:50	*Session 3*
	Extra BoFs (bit.ly/BOSC2019-bofs)
18:00-20:00	Posters

Day 2 (Thursday, July 25)	
Time	**Session**
8:30-9:40	*Session 4*
	BOSC announcements
	Late-Breaking Lightning Talks
9:40-10:15	Coffee break
10:15-12:40	*Session 5*
	Session: Containers
	Session: Open science
12:40-14:00	Lunch, BoFs
14:00-16:40	Session 6
	Session: Workflows
	Session: Building Open Source Communities
	CoFest preview; BOSC closing remarks
16:40-18:00	Coffee break & ISMB keynote
19:30	BOSC dinner, Basel Markthalle

Complete schedule of talks

(Check https://www.open-bio.org/events/bosc/schedule/ for updates)

Day 1 (Wednesday, July 24, 2019)

Title	Speaker	Start time	End time	Session
Opening remarks	Nomi Harris	10:15	10:25	**Session: Introducing BOSC and the OBF**
The Open Bioinformatics Foundation	Heather Wiencko	10:25	10:33	**Session: Introducing BOSC and the OBF**
Google Summer of Code 2018	Kai Blin	10:33	10:40	**Session: Introducing BOSC and the OBF**
Session: Data crunching	Chair: Peter Cock	10:40	11:40	**Session: Data crunching**
elPrep 4: A multi-threaded tool for sequence analysis	Charlotte Herzeel	10:40	11:00	**Session: Data crunching**
Variant Transforms and BigQuery: Large scale data analytics in the cloud	Andrew Moschetti	11:00	11:05	**Session: Data crunching**
Forome Anfisa – an Open Source Variant Interpretation Tool	Michael Bouzinier	11:05	11:10	**Session: Data crunching**
Biotite: A comprehensive and efficient computational molecular biology library in Python	Patrick Kunzmann	11:10	11:15	**Session: Data crunching**
Q&A for lightning talks	-	11:15	11:20	**Session: Data crunching**
Portable Pipeline for Whole Exome and Genome Sequencing	Andrey Kokorev	11:20	11:25	**Session: Data crunching**
Epiviz File Server - Query, Compute and Interactive Exploration of data from Indexed Genomic Files	Jayaram Kancherla	11:25	11:30	**Session: Data crunching**
What does 1.0 take? MISO LIMS after 9 years of development	Morgan Taschuk	11:30	11:35	**Session: Data crunching**
Q&A for lightning talks	-	11:35	11:40	**Session: Data crunching**

Title	Speaker	Start time	End time	Session
Session: Data modeling and formats	Chair: Karsten Hokamp	11:40	12:25	**Session: Data modeling & formats**
BioLink Model - standardizing knowledge graphs and making them interoperable	Deepak Unni	11:40	12:00	**Session: Data modeling & formats**

pysradb: A Python package to query next-generation sequencing metadata and data from NCBI Sequence Read Archive	Saket Choudhary	12:00	12:05	**Session: Data modeling & formats**
Disq, a library for manipulating bioinformatics sequencing formats in Apache Spark.	Michael Heuer	12:05	12:10	**Session: Data modeling & formats**
A toolkit for semantic markup, exploration, comparison and merging of metadata models expressed as JSON-Schemas	Dominique Batista	12:10	12:15	**Session: Data modeling & formats**
A lightweight approach to research object data packaging	Stian Soiland-Reyes	12:15	12:20	**Session: Data modeling & formats**
Q&A for lightning talks	-	12:20	12:25	**Session: Data modeling & formats**
Lunch, BoFs, unofficial poster session		**12:40**	**14:00**	**ISMB and BOSC Birds of a Feather**

Title	Speaker	Start time	End time	Session
BOSC 2019 Keynote: Building infrastructure for responsible Open science in Africa	Nicola Mulder	14:00	15:00	**Keynote**
Session: Open data	Chair: Karsten Hokamp	15:00	16:00	**Session: Open data**
The (Re)usable Data Project	Seth Carbon	15:00	15:20	**Session: Open data**
The FAIR data principles and their practical implementation in InterMine	Sergio Contrino	15:20	15:40	**Session: Open data**
GA4GH: Developing Open Standards for Responsible Data Sharing	Rishi Nag	15:40	15:45	**Session: Open data**
The Commons Alliance: Building cloud-based infrastructure to support biomedical research in Data STAGE and AnVIL	Brian O'Connor	15:45	15:50	**Session: Open data**
Fake it 'til You Make It: Open Source Tool for Synthetic Data Generation to Support Reproducible Genomic Analyses	Adelaide Rhodes	15:50	15:55	**Session: Open data**
Q&A for lightning talks		15:55	16:00	**Session: Open data**
Coffee break		16:00	16:40	
BOSC Birds of a Feather (BoFs)		**16:40**	**17:40**	**Session: BOSC Birds of a Feather**
Poster Session		**18:00**	**20:00**	

Day 2 (Thursday, July 25, 2019)

Title	Speaker	Start time	End time	Session
Session: Late-Breaking Lightning Talks	Chair: Yo Yehudi	8:30	9:35	**Session: LBLTs**
BOSC announcements	Heather Wiencko	8:30	8:40	
Archaeopteryx.js: Web-based Visualization and Exploration of Annotated Phylogenetic Trees (JavaScript)	Christian Zmasek	8:40	8:45	**Session: LBLTs**
Sequenceserver: a modern graphical user interface for custom BLAST databases	Anurag Priyam	8:45	8:50	**Session: LBLTs**
Parallel, Scalable Single-cell Data Analysis	Ryan Williams	8:50	8:55	**Session: LBLTs**
Q&A for late-breaking lightning talks		8:55	9:00	**Session: LBLTs**
RAWG: RNA-Seq Analysis Workflow Generator	Zeyu Yang	9:00	9:05	**Session: LBLTs**
SAPPORO: workflow management system that supports continuous testing of workflows	Tazro Ohta	9:05	9:10	**Session: LBLTs**
Lazy representation and analysis of very large genomic data resources in R / Bioconductor	Qian Liu	9:10	9:15	**Session: LBLTs**
Q&A for late-breaking lightning talks		9:15	9:20	**Session: LBLTs**
The Monarch Initiative: Closing the knowledge gap with semantics-based tools	Monica Munoz-Torres	9:20	9:25	**Session: LBLTs**
DAISY: a tool for the accountability of Biomedical Research Data under the GDPR.	Pinar Alper.	9:25	9:30	**Session: LBLTs**
Q&A for late-breaking lightning talks		9:30	9:35	
Coffee break		9:40	10:15	

Title	Speaker	Start time	End time	Session
Session: Containers	Chair: Heather Wiencko	10:20	11:00	**Session: Containers**
Dockstore: Enhancing a community platform for sharing cloud-agnostic research tools	Louise Cabansay	10:20	10:40	**Session: Containers**
Bioconductor with Containers: Past, Present, and Future	Nitesh Turaga	10:40	11:00	**Session: Containers**
Mini-Break		11:00	11:15	
Session: Open science	Chair: Monica Munoz-Torres	11:15	12:22	**Session: Open science**

OpenEBench. The ELIXIR platform for benchmarking.	Salvador Capella-Gutierrez	11:15	11:35	**Session: Open science**
ELIXIR Europe on the Road to Sustainable Research Software	Mateusz Kuzak	11:35	11:55	**Session: Open science**
The Kipoi repository: accelerating the community exchange and reuse of predictive models for genomics	Julien Gagneur	11:55	12:15	**Session: Open science**
A method for systematically generating explorable visualization design spaces	Anamaria Crisan	12:15	12:22	**Session: Open science**
Lunch, BoFs, unofficial poster session		12:40	14:00	**ISMB and BOSC Birds of a Feather**

Title	Speaker	Start time	End time	Session
Session: Workflows	Chair: Michael Heuer	14:00	15:20	**Session: Workflows**
snakePipes enable flexible, scalable and integrative epigenomic analysis	Devon Ryan	14:00	14:20	**Session: Workflows**
nf-core: Community built bioinformatics pipelines	Alexander Peltzer	14:20	14:40	**Session: Workflows**
NGLess: a domain-specific language for NGS analysis (the NG-meta-profiler case study)	Luis Pedro Coelho	14:40	15:00	**Session: Workflows**
Benten: An experimental language server for the Common Workflow Language	Kaushik Ghose	15:00	15:05	**Session: Workflows**
Janis: an open source tool to machine generate type-safe CWL and WDL workflows	Richard Lupat	15:05	15:10	**Session: Workflows**
Collecting runtime metrics of genome analysis workflows by CWL-metrics	Tazro Ohta	15:10	15:15	**Session: Workflows**
Q&A for lightning talks		15:15	15:20	**Session: Workflows**
Mini-Break		15:20	15:30	
Session: Building Open Source Communities (BOSC)	Chair: Nomi Harris			**Session: BOSC**
Inclusiveness in Open Science Communities	Malvika Sharan	15:30	15:50	**Session: BOSC**
ECRcentral: An open-source platform to bring early-career researchers and funding opportunities together	Aziz Khan	15:50	16:10	**Session: BOSC**
The Data Carpentry Genomics Curriculum: Overview and Impact	Jason Williams	16:10	16:15	**Session: BOSC**

Impact of The African Genomic Medicine Training Initiative: a Community-Driven Genomic Medicine Competency-Based Training Model for Nurses in Africa	Vicky Nembaware	16:15	16:20	**Session: BOSC**
Biopython Project Update 2019	Peter Cock	16:20	16:25	**Session: BOSC**
Q&A for lightning talks		16:25	16:30	**Session: BOSC**
Introducing CoFest 2019 - the post-BOSC Collaboration Festival	Alexander Peltzer	16:30	16:35	**Session: CoFest 2019 and Closing Remarks**
Closing Remarks	Nomi Harris	16:35	16:40	**Session: CoFest 2019 and Closing Remarks**
Coffee break and ISMB closing keynote		16:40	18:20	
BOSC dinner (pay your own way)		19:30		

Any last-minute schedule updates will be posted at
https://www.open-bio.org/events/bosc/schedule/

Poster Numbers

#	Title	Talk / Poster	Poster Presenter	Talk Presenter
P-01	pysradb: A Python package to query next-generation sequencing metadata and data from NCBI Sequence Read Archive	Talk + Poster	Saket Choudhary	Saket Choudhary
P-02	snakePipes enable flexible, scalable and integrative epigenomic analysis	Talk + Poster	Devon Ryan	Devon Ryan
P-03	The FAIR data principles and their practical implementation in InterMine	Talk + Poster	Sergio Contrino	Sergio Contrino
P-04	Collecting runtime metrics of genome analysis workflows by CWL-metrics	Talk + Poster	Tazro Ohta	Tazro Ohta
P-05	GA4GH: Developing Open Standards for Responsible Data Sharing	Talk + Poster	Rishi Nag	Rishi Nag
P-07	ELIXIR Europe on the Road to Sustainable Research Software	Talk + Poster	Mateusz Kuzak	Mateusz Kuzak
P-09	A method for systematically generating explorable visualization design spaces	Talk + Poster	Anamaria Crisan	Anamaria Crisan
P-10	Biotite: A comprehensive and efficient computational molecular biology library in Python	Talk + Poster	Patrick Kunzmann	Patrick Kunzmann
P-14	NGLess: a domain-specific language for NGS analysis (the NG-meta-profiler case study)	Talk + Poster	Luis Pedro Coelho	Luis Pedro Coelho
P-16	Forome Anfisa – an Open Source Variant Interpretation Tool	Talk + Poster	Dmitry Etin	Michael Bouzinier
P-22	Janis: An open source tool to machine generate type-safe CWL and WDL workflows	Talk + Poster	Richard Lupat	Richard Lupat
P-25	Fake it 'til You Make It: Open Source Tool for Synthetic Data Generation to Support Reproducible Genomic Analyses	Talk + Poster	Adelaide Rhodes	Adelaide Rhodes
P-53	OpenEBench. The ELIXIR platform for benchmarking.	Talk + Poster	Salvador Capella-Gutiérrez	Salvador Capella-Gutiérrez
P-26	The Kipoi repository: accelerating the community exchange and reuse of predictive models for genomics	Talk + Poster	Ziga Avsec	Julien Gagneur
P-28	Dockstore: Enhancing a community platform for sharing cloud-agnostic research tools	Talk + Poster	Denis Yuen	Louise Cabansay

P-29	Disq, a library for manipulating bioinformatics sequencing formats in Apache Spark.	Talk + Poster	Michael Heuer	Michael Heuer
P-30	The Data Carpentry Genomics Curriculum: Overview and Impact	Talk + Poster	François Michonneau	François Michonneau
P-31	Epiviz File Server - Query, Compute and Interactive Exploration of data from Indexed Genomic Files	Talk + Poster	Jayaram Kancherla	Jayaram Kancherla
P-32	A lightweight approach to research object data packaging	Talk + Poster	Stian Soiland-Reyes	Stian Soiland-Reyes
P-34	ECRcentral: An open source platform to bring early career researchers and funding together	Talk + Poster	Aziz Khan	Aziz Khan
P-35	Impact of The African Genomic Medicine Training Initiative: a Community-Driven Genomic Medicine Competency-Based Training Model for Nurses in Africa	Talk + Poster	Victoria Nembaware	Victoria Nembaware
P-36	Parallel, Scalable Single-cell Data Analysis	Talk + Poster	Ryan Williams	Ryan Williams
P-40	RAWG: RNA-Seq Analysis Workflow Generator	Talk + Poster	Zeyu Yang	Zeyu Yang
P-42	What does 1.0 take? MISO LIMS after 9 years of development	Talk + Poster	Morgan Taschuk	Morgan Taschuk
P-44	SAPPORO: workflow management system that supports continuous testing of workflows	Talk + Poster	Hirotaka Suetake	Hirotaka Suetake
P-45	Lazy representation and analysis of very large genomic data resources in R / Bioconductor	Talk + Poster	Qian Liu	Qian Liu
P-47	Sequenceserver: a modern graphical user interface for custom BLAST databases	Talk + Poster	Anurag Priyam	Anurag Priyam
P-50	Archaeopteryx.js: Web-based Visualization and Exploration of Annotated Phylogenetic Trees (JavaScript)	Talk + Poster	Christian Zmasek	Christian Zmasek
P-51	The Monarch Initiative: Closing the knowledge gap with semantics-based tools	Talk + Poster	Monica C Munoz-Torres	Monica C Munoz-Torres
P-52	DAISY: a tool for the accountability of Biomedical Research Data under the GDPR.	Talk + Poster	Venkata Pardhasaradhi Satagopam	Venkata Pardhasaradhi Satagopam

#	Title	Talk / Poster	Poster Presenter
P-06	Creating a pluggable visualisation toolsuite with BlueGenes Tool API	Poster only	Yo Yehudi
P-08	Cellular Genetics Informatics support group: Nextflow and Jupyter on Kubernetes, Nextflow web interface	Poster only	Anton Khodak
P-11	OpenBio-C: An Online Social Workflow Management System and Research Object Repository	Poster only	Alexandros Kanterakis
P-12	CWLab: an open-source, platform-agnostic, and cloud-ready framework for simplified deployment of the Common Workflow Language using a graphical web interface	Poster only	Kersten Henrik Breuer
P-13	pdb-tools: a dependency-free cross-platform swiss army knife for PDB files.	Poster only	João Rodrigues
P-15	OmicsSIMLA: A multi-omics data simulation tool for complex disease studies	Poster only	Ren-Hua Chung
P-17	Tagging of disease names in biomedical literature	Poster only	Jeanette Prinz
P-18	WhatsHap: fast and accurate read-based phasing	Poster only	Peter Ebert
P-19	Recommendations and guidelines for tumor heterogeneity quantification using deconvolution of methylation data: data challenges as a tool for benchmarking studies	Poster only	Magali Richard
P-20	Sustainability of legacy software - Making the antiSMASH genome mining tool ready for the future	Poster only	Kai Blin
P-21	Methrix: An R package for efficient processing of bedGraph files from large-scale methylome cohorts	Poster only	Anand Mayakonda
P-23	Run Scanner: a tool for monitoring sequencer runs and accessing run information	Poster only	Morgan Taschuk
P-24	Ada Discovery Analytics: All-in-One Data Platform for Clinical and Translational Medicine with Scalable Machine Learning	Poster only	Peter Banda
P-27	CViTjs: Dynamic Whole Genome Visualisation	Poster only	Andrew Wilkey
P-33	Developing Python and Rust libraries to improve the ontology ecosystem	Poster only	Martin Larralde
P-37	Another point of view for the fast and accurate large MSA, the regressive approach	Poster only	Edgar Garriga Nogales
P-38	Analyzing protein structure and evolution using Julia with MIToS.jl	Poster only	Diego Zea
P-55	ImmPort: Ensuring FAIR Data through a Trustworthy Biomedical Data Repository	Poster only	Dawei Lin

P-39	Pedigree-based analysis pipeline version 2 (PBAP v.2): new features added	Poster only	Alejandro Nato
P-41	Crowdsourcing towards Antimicrobial Resistance & Open Source Drug Discovery	Poster only	Anshu Bhardwaj
P-43	10 recommendations to make your research software FAIRer	Poster only	Carlos Martinez Ortiz
P-46	Terra Open Science Contest	Poster only	Geraldine Van der Auwera
P-48	EDAM: the ontology of bioinformatics operations, types of data, topics, and data formats (2019 update)	Poster only	Hervé Ménager
P-49	Physlr: Construct a Physical Map from Linked Reads	Poster only	Shaun Jackman
P-56	java2script/SwingJS for bioinformatics: Reintroducing Jalview on the Web as JalviewJS	Poster only	Robert Hanson

Talk + Poster Abstracts

The Open Bioinformatics Foundation

Heather Wiencko

The Open Bioinformatics Foundation (https://www.open-bio.org/) is a non-profit, volunteer-run group that promotes open source software development and Open Science within the biological research community. The OBF helps to provide support and publicity for a variety of member projects. It also runs the annual Bioinformatics Open Source Conference (BOSC) and the associated CollaborationFest. Since 2016, the OBF has sponsored a Travel Fellowship program aimed at increasing diversity at community events.

The OBF is open to anyone who is interested in promoting open source bioinformatics / open science, which includes everyone at BOSC! We invite you to learn more about the OBF and engage in a two-way dialog with OBF Board members at the OBF BoF on 25-July at 12:45pm, which will include a vote on a new candidate for the Board.

Google Summer of Code 2018

Kai Blin

Google's Summer of Code program 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 eight times. In 2018, the Open Bioinformatics Foundation has acted as an umbrella organisation for four projects from the open source bioinformatics community, and five students successfully finished the program. In 2019, OBF is an umbrella for five open source bioinformatics projects. 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.

Introducing CoFest 2019 - the post-BOSC Collaboration Festival

Alexander Peltzer

In conjunction with the Bioinformatics Open Source Conference (BOSC), the Open Bioinformatics Foundation (OBF) runs a welcoming, self-organizing, non-competitive, and highly productive collaborative event called the CollaborationFest, or CoFest.

Everyone is welcome to attend. We will have a mix of experienced developers, users, trainers, and researchers, newcomers to experienced bioinformaticians, and everything in between. Attendees will self-organize into working groups based on shared interests like programming languages, open source projects, or biological questions.

CollaborationFest is not a competition; there are no prizes. Rather its goals are to grow and foster the contributor community for open source bioinformatics projects, and to extend, enhance, and otherwise improve open-source bioinformatics code and non-code artefacts, such as documentation and training materials.

This will be the 10th such event since 2010, which we originally called the Coding Festival or CodeFest. Communities such as Galaxy, Common Workflow Language, Nextflow, and others have found CollaborationFest a fun, rewarding, and highly productive experience. As in previous years, a summary of the results of the event will be included in the BOSC meeting report.

CollaborationFest 2019 will take place the two days after BOSC, July 26-27, at The Swiss Innovation Hub for Personalized Medicine in Basel, Switzerland. Registration is free; sponsorships offset the cost of the venue, coffee and snacks.

Day1 10:40-11:00

elPrep 4: A multi-threaded tool for sequence analysis

Charlotte Herzeel[1], Pascal Costanza[1]

[1]) imec, ExaScience Lab, Kapeldreef 75, 3001 Leuven, Belgium
Correspondence: charlotte.herzeel@imec.be

The use of next-generation sequencing (NGS) has increased dramatically over the last years, and the time spent on analyzing the terabytes of data generated for NGS analysis easily takes up hundreds of compute hours. It is therefore important to improve the efficiency of the computing process. The de-facto standard software packages for processing NGS data are GATK, SAMtools, and Picard, which are primarily developed by the Broad and Sanger Institutes. Our goal is to improve on these packages in terms of computational performance, which led us to the development of the elPrep software.

elPrep [1, 2, 3] is a multi-threaded tool for executing NGS pipelines. It can be uses as a drop-in replacement for standard tools such as GATK, Picard, and SAMtools, while producing identical results. elPrep has a unique software architecture that allows executing an NGS pipeline by making only a single pass through the data, no matter how many steps are used in the pipeline. elPrep is designed as a multi-threaded application, optimized for running in memory, avoids repeated file I/O, and merges the computation of multiple pipeline steps to significantly speed up the execution time.

In this talk, we present elPrep 4 [2], our latest release. elPrep 4 is a reimplementation from scratch of the elPrep framework [1] for processing sequencing alignment map files (SAM/BAM) in the Go programming language. elPrep 4 introduces multiple new features allowing us to process all of the preparation steps defined by the GATK Best Practices pipelines for variant calling. This includes new and improved functionality for sorting, (optical) duplicate marking, base quality score recalibration, BED and VCF parsing, and various filtering options.

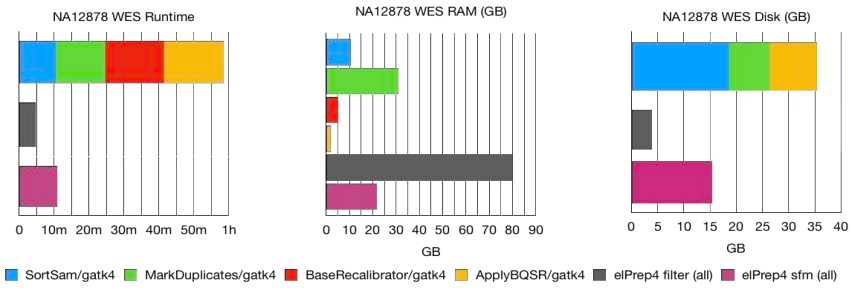


Figure 1: **WES benchmarks.** Runtime, RAM use, and disk use in GATK 4 vs. elPrep 4 (filter mode) vs. elPrep 4 (sfm mode). We see 5.4-13x speedup for 0.7-2.6x RAM use and 0.6-0.2x disk use when comparing elPrep 4 filter/sfm to GATK 4. The results, i.e. final BAM, metrics and recalibration files, are the same for all runs.

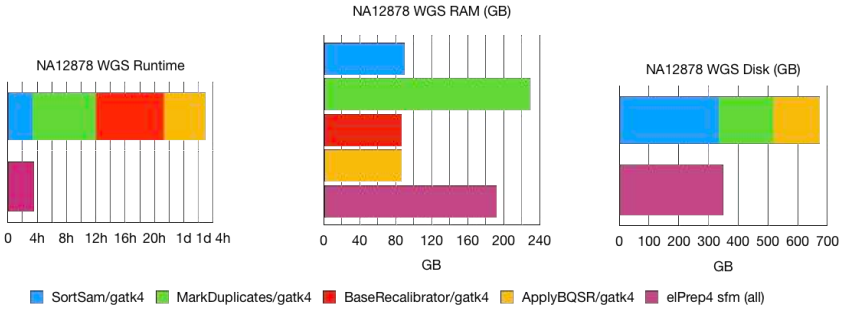


Figure 2: **WGS benchmarks.** Runtime, RAM use, and disk use in GATK 4 vs. elPrep 4 (sfm mode). elPrep 4 executes the pipeline 7.4x faster than GATK 4, using 0.84x of the RAM, and only 0.7x of the disk space. The final BAM, metrics, and recalibration files are the same for both runs.

The implementations of these options in elPrep 4 faithfully reproduce the outcomes of their counterparts in GATK 4, SAMtools, and Picard, even though the underlying algorithms are redesigned to take advantage of elPrep's parallel execution framework to vastly improve the runtime and resource use compared to these tools.

We present benchmarks that show that elPrep executes the preparation steps of the GATK Best Practices up to 13x faster on WES data, and up to 7.4x faster for WGS data compared to running the same pipeline with GATK 4, while using fewer compute resources (cf. Fig. 1-2). We also present a scaling experiment in the cloud using Amazon Web Services discussing performance in relation to cost.

We also show that elPrep achieves its superior performance while maintaining a modular, extensible design. The implementation of the individual pipeline steps is separated from the execution engine which parallelizes and merges their computations. This allows developers to add new pipeline steps without needing to touch the parallelization code itself. In fact, since its open-source release, we have welcomed multiple community contributions to extend elPrep.

elPrep is developed entirely in the Go programming language [3] and released as an open-source project at `https://github.com/ExaScience/elprep`

References

[1] Herzeel C, Costanza P, Decap D, Fostier J, Reumers J. elPrep: High-Performance Preparation of Sequence Alignment/Map Files for Variant Calling. PLoS ONE. 2015;10(7). doi:10.137/journal.pone.0138868.

[2] Herzeel C, Costanza P, Decap D, Fostier J, Verachtert W. elPrep 4: A multithreaded framework for sequence analysis PLoS ONE. 2019;14(2). doi:10.1371/journal.pone.0209523.

[3] Costanza P, Herzeel C, Verachtert W. A comparison of three programming languages for a full-fledged next-generation sequencing tool bioRxiv. 2019. doi:https://doi.org/10.1101/558056.

Day1 11:00-11:05

Variant Transforms and BigQuery: Large scale data analytics in the cloud

Author: Andrew Moschetti (moschi@google.com)
Author Affiliations: Google, LLC - Mountain View, CA.
Project URL: https://github.com/googlegenomics/gcp-variant-transforms
OSS License: Apache 2.0

Variant Transforms is an open source tool developed by Google Cloud to load variants from VCF files into BigQuery. BigQuery is a highly scalable and fully managed data warehouse provided as part of Google Cloud Platform (GCP). With Variant Transforms, users can load data from VCF files into BigQuery and write SQL queries that use the power of BigQuery to process large amounts of data in seconds. Variant Transforms allows users to merge data from millions of samples to facilitate easy and fast analysis. Joining with phenotypic, clinical, and other omics data allows users to optimize their research workloads.

Numerous improvements to Variant Transforms over the past year have increased the performance and added new use cases. Variant Transforms is able to import billions of records (terabyte to petabyte scale data) to a single table. Support for VEP annotation was added, and recently received a 10x speed improvement. Partitioning and clustering of data in BigQuery along with schema optimization results in faster queries that process less data. Importing gVCFs allows for joint genotyping across these large sample sets. Export back to VCF is supported to allow users to export a cohort from BigQuery for analysis with other existing tools.

Day1 11:05-11:10 (P-16)

Forome Anfisa – an Open Source Variant Interpretation Tool

Bouzinier M[1,2], Trifonov SI[2], Krier J[1,2], Etin D[2], Olchanyi D[2], Kargalov A[2],Ghazani AA[1], Sunyaev SR[1,3]

Forome Anfisa is a highly customizable suite of software for downstream genetic analysis, clinical variant interpretation, curation, and collaboration. It supports 3 real-life scenarios for effective WES/WGS and panel of genes variants analysis:

- the traditional clinical workflow for variant curation based on predefined guidelines;
- a workflow for design and development new guidelines for variant interpretation;
- a collaboration in variant interpretations.

Anfisa is developed under the Apache 2.0 license and is available on GitHub in the Forome Association repository: https://github.com/ForomePlatform/anfisa.

Anfisa is a modular system with three main components and a number of support modules. Main components are:

- annotation pipeline,
- backend database,
- frontend user interface.

Annotation pipeline starts with Ensembl VEP [1] and then adds annotations based on functional analysis, population genetics, clinical knowledge, epigenetics, etc. This is done by traversing databases such as gnomAD, ClinVar [2], HGMD [3], including results from spliceAI [4] and other sources. The backend stores the data in Druid Open Source OLAP [5] and metadata, such as user environment, curation notes and preferences are stored in MongoDB. The frontend is implemented using Vue JavaScript Framework [6] and Bootstrap toolkit [7]. Annotation Pipeline and Backend provide public REST API and technically can be used in standalone mode, integrated with other genomics tools and EMRs.

Anfisa is designed to forge collaboration between people with different goals and skills, and with different organizational roles, from treating physicians to clinical geneticists to researchers and bioinformaticians. The system is designed to efficiently operate with small and large genomic datasets. It is transparent for users.

A patient case consisting of the panel of genes loads several thousands variants directly into the main UI and offers filtering capabilities to quickly narrow the list down to a few dozens. It is then a workable amount of data to review manually.

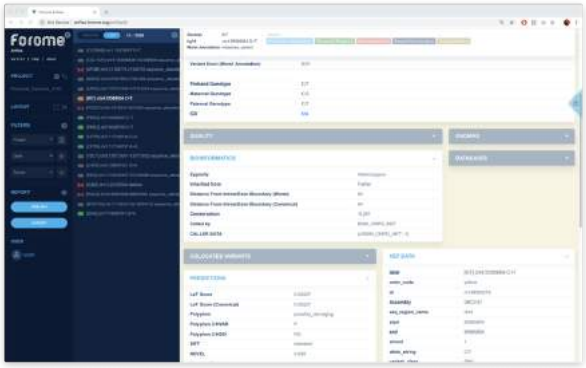
A case with WES/WGS data loads into the advanced filtering tool which would help users creating a custom workspace. The workspace is a combination of the various available filters and more complex rules (clinical guidelines), used by the user to reach a reasonable and meaningful amount of the variants to work in the manual mode. When the workspace is defined and applied to the case data, a list of variants is loaded into the main UI.

We have implemented two distinct scenarios to address the variety of users goals of the complexity of clinical research tasks:

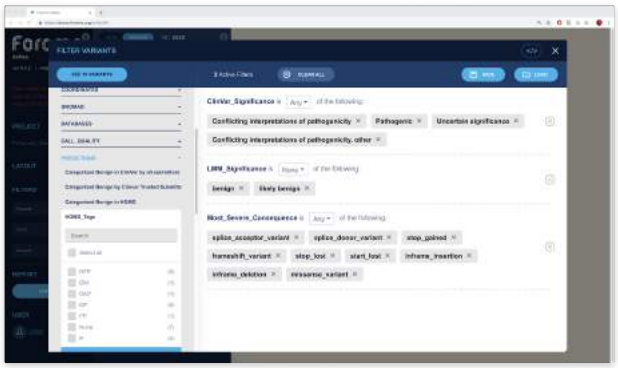
[1] Division of Genetics, Brigham & Women's Hospital
[2] Forome Association
[3] Department of Biomedical Informatics, Harvard Medical School

- the clinical use, where the users operate with predefined clinical rules (e.g. ACMG guidelines) and thus, assuring the standardized protocols in the clinical practice are being followed;
- the research scenario where the new rules are being evaluated and tested and eventually being promoted for use in the clinical scenario.

This approach implements our collaboration concept to bring the research findings quicker to the clinical practice. It is specifically valuable for WGS/WES diagnostics where the guidelines are still evolving.



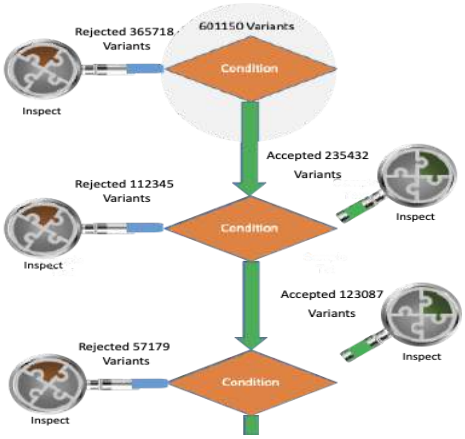
Interpretation mode



Research Mode

The clinical scenario implies working within the set of pre-defined clinical guidelines. It helps to focus on efficiency and collaboration. It allows to tag, comment and eventually to report on the candidate variants. This scenario is being used by SEQaBOO Project [8].

The researcher scenario allows a clinical geneticist or a researcher to apply various flexible criteria and create workspaces that can be shared with other collaborating researches. Collaborating researches can tag specific variants in the workspace that were shared with them and put textual notes explaining their reasoning.



A special workflow is designed for developing new clinical guidelines. This workflow allows the user to build a decision tree for variant classification.

Advanced workflows are being adopted by Brigham Genomics Medicine Project [9]

References

1. McLaren W, et al. The Ensembl Variant Effect Predictor. Genome Biol. 2016;17:122. doi: 10.1186/s13059-016-0974-4.
2. Landrum M.J., Lee J.M., Benson M. et al. (2018) ClinVar: improving access to variant interpretations and supporting evidence. Nucleic Acids Res., 46, D1062–D1067.
3. Stenson P. D., Ball E. V., Mort M., Phillips A. D., Shaw K., Cooper D. N. (2012). The Human Gene Mutation Database (HGMD) and its exploitation in the fields of personalized genomics and molecular evolution. Curr. Protoc. Bioinformatics 39 1.13.1–1.13.20.
4. K. Jaganathan, S. Kyriazopoulou Panagiotopoulou, J.F. McRae, S.F. Darbandi, D. Knowles, Y.I. Li, J.A. Kosmicki, J. Arbelaez, W. Cui, G.B. Schwartz, et al. Predicting Splicing from Primary Sequence with Deep Learning. Cell, 176 (2019), pp. 535-548.e24
5. Apache Druid is a high performance real-time analytics database http://druid.io/docs/latest/design/
6. Vue - The Progressive JavaScript Framework https://vuejs.org/
7. Bootstrap - an open source toolkit for developing with HTML, CSS, and JS. https://getbootstrap.com/
8. Haghighi A., Krier J.B., Toth-Petroczy A., Cassa C.A., Frank N.Y., Carmichael N., Fieg E., Bjonnes A., Mohanty A., Briere L.C.et al. An integrated clinical program and crowdsourcing strategy for genomic sequencing and Mendelian disease gene discovery. NPJ Genome Med. 2018
9. SEQaBOO - SEQuencing a Baby for an Optimal Outcome (http://seqaboo.bwh.harvard.edu/), under the NIH grant R01-DC015052-01

Day1 11:10-11:15 (P-10)

Biotite: A comprehensive and efficient computational molecular biology library in Python

Patrick Kunzmann, Kay Hamacher

April 11, 2019

1 Introduction

A typical computational molecular biology workflow consists of combining different programs in order to reach the desired goal. Each software is usually made for a very specific purpose, like sequence alignment or secondary structure annotation. Manually converting between the required file formats and adjusting the input data and parameters for these programs can be unhandy for the user. Furthermore, such a workflow can be inefficient due to an overhead of file read/write operations. These problems can be overcome by shifting the workflow to comprehensive computational biology library in a easy-to-learn scripting language like Python.

Biopython [1] is such a comprehensive library, however, its foundation was almost 20 years ago. Hence, it largely lacks modern scientific programming standards in Python. *NumPy* [2], for example, is only sparsely used.

We would like to present the open source Python package *Biotite*. It is a modern and comprehensive computational molecular biology library in the spirit of *Biopython*. Through extensive usage of *NumPy*, most operations in *Biotite* are C-accelerated. Originally published in BMC Bioinformatics [3] in October 2018, the package is in continuous development. With the presentation at the BOSC 2019 we hope to extend the userbase of the package and attract more developers, who like to contribute new functionalities.

2 Library

Biotite stores sequence and structure data internally as *NumPy* `ndarray` objects. These `ndarray` objects are directly accessible to the user. Hence, *Biotite* can be used as flexible library to build software upon, removing the need to implement basic functionality like file parsers. Furthermore, the extensive application of *NumPy* renders most operations efficient via vectorization. Additionally, *Cython* [4] code is used in places where vectorization is not feasible.

The package is divided into four subpackages: `sequence`, `structure`, `application` and `database`.

The `sequence` subpackage contains utilities to handle biological sequences. The base type for all sequences is the `Sequence` class, that stores a sequence encoded as a `ndarray` of integers [3]. These objects can be used to perform DNA translation, subsequence searches, alignments, etc. Additionally, this subpackage provides functions for visualization of sequence related objects, e.g. alignments (Fig. 1A).

The `structure` subpackage revolves around handling macromolecular structures, ranging from single models (`AtomArray` class) to entire trajectories (`AtomArrayStack` class). Both, the `AtomArray` and the `AtomArrayStack`, internally store the atom coordinates and the atom annotations (chain ID, residue name, etc.) as separate `ndarray` objects. In addition to a high performance, this has the advantage that `AtomArray` and `AtomArrayStack` objects can be indexed like an `ndarray`: The index is simply propagated to the coordinates and annotations. This subpackage offers a large variety of analysis functions ranging from simple geometric measurements to the calculation of the solvent accessible surface area.

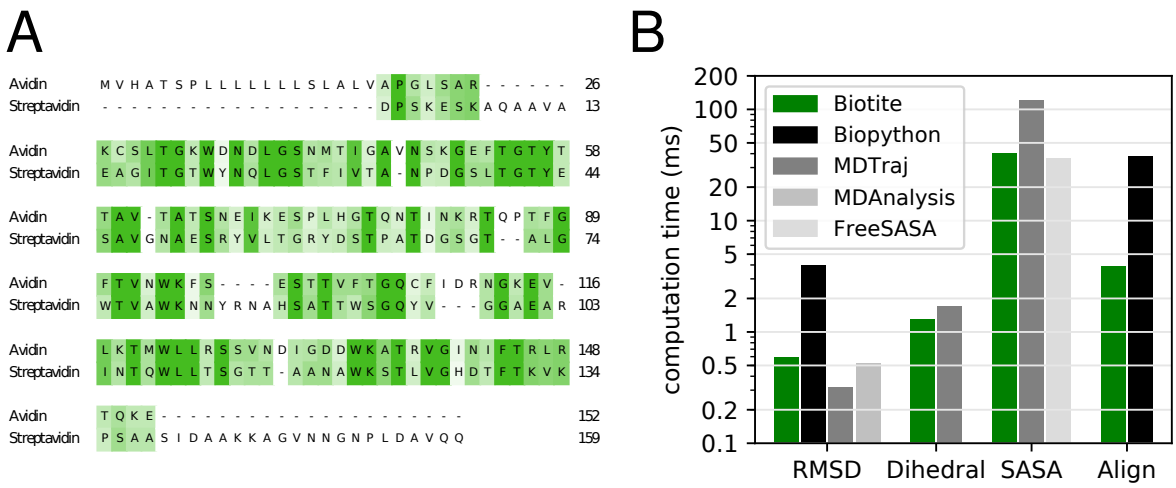


Figure 1: Figures are adapted from the original *Biotite* publication [3]. **A** Example visualization of a sequence alignment in *Biotite*. **B** Performance comparison of analysis algorithms. RMSD: Superimposition of a structure onto itself and subsequent RMSD calculation (PDB: 1AKI). Dihedral: Calculation of the backbone dihedral angles of a protein (PDB: 1AKI). SASA: Calculation of the SASA of a protein (PDB: 1AKI). Align: Optimal global alignment of two 1,000 residues long polyalanine sequences.

To round off the workflow, the `database` subpackage can be used to search in and fetch files from the RCSB PDB and NCBI Entrez database via HTTP requests.

The `application` subpackage provides seamless interfaces to external programs for analysis techniques that are not directly implemented in *Biotite*. At the moment these interface include a variety of multiple sequence alignment software, DSSP and NCBI BLAST.

3 Performance

Performance benchmarks demonstrate the computational efficiency of *Biotite* (Fig. 1B) [3]. In addition to *Biopython*, the performance of *Biotite* was compared to the Python packages *MDTraj*, *MDAnalysis* and *FreeSASA*. While the computation times of *Biotite* are comparable with the latter packages, *Biotite* is approximately one oder of magnitude faster than *Biopython*, although *Biopython* also uses C-acceleration.

4 Availability

Biotite is licensed under the *3-Clause BSD License*. The source code is hosted on GitHub (github.com/biotite-dev/biotite). The *Biotite* documentation is hosted at biotite-python.org, including a tutorial, the API reference and an example gallery, showing applications of *Biotite* on real and fictional biological problems.

References

[1] P. J. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. de Hoon, "Biopython: freely available Python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, 2009.

[2] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science and Engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[3] P. Kunzmann and K. Hamacher, "Biotite: A unifying open source computational biology framework in Python," *BMC Bioinformatics*, vol. 19, no. 1, 2018.

[4] S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith, "Cython: The best of both worlds," *Computing in Science and Engineering*, vol. 13, no. 2, pp. 31–39, 2011.

Day1 11:20-11:25

PORTABLE PIPELINE FOR WHOLE EXOME AND GENOME SEQUENCING

Timur Isaev
DBMI
HMS
tisaev@bwh.harvard.edu

Joel Krier
Division of Genetics
BWH
jkrier@bwh.harvard.edu

Patrick Magee
DNAStack

Heather Ward
DNAStack

Andrey Kokorev
Forome Association

Arezou A. Ghazani
Division of Genetics
BWH

Michael Bouzinier
Division of Genetics
BWH
mbouzinier@bwh.harvard.edu

1 Abstract

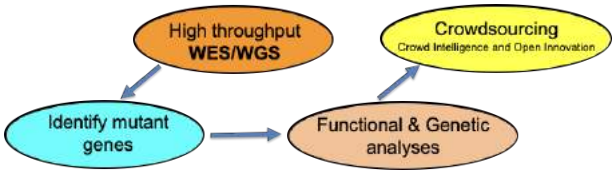
Genomic sequencing is an important part of modern healthcare, making bioinformatics pipelines an essential part of the diagnostic process.

With the advancement in technologies, it became possible to perform sequencing at a fast rate. Processing raw sequence data to find genomic variants plays a significant role in diagnostics, identifying optimal treatments, and understanding the nature of diseases. This makes bioinformatics pipelines an essential part of the genomic analysis. However, to be adopted for clinical practice, pipelines should comply with strict reproducibility requirements. Creating fully portable pipelines, that can be easily shared and used by different teams in various computational environments, allowing researchers and clinicians to reproduce each other's results, helps in moving moving towards standardizing this process. An additional benefit of portability is saving researchers time and costly investments for the development of their own pipelines. In recent years there has been a noticeable push to develop portable pipelines[1][2]. In particular GA4GH (Global Alliance for Genomics and Health) and it's Cloud Containers and Workflows workstream has played a significant role in advancing the principles of portability, interoperability and reproducibility of results[3]. Our fully open source pipeline is focused on processing human Whole Genome and Whole Exome cases and producing both clinical results and research data that is used for medical diagnostics through gene discovery.

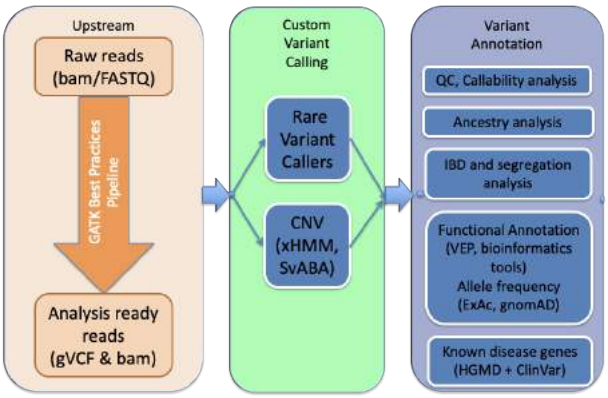
Originally, the pipeline was developed at Sunyaev Lab at the Department of Biomedical Informatics at Harvard Medical School and Genetics Division of Brigham & Women's Hospital and is being used by several clinical institutions[5]. As input, it takes raw sequencing data in FASTQ or BAM format. The pipeline consists of three distinct components:

1. Upstream, creating the aligned BAM files and calling common variants using GATK 3.x.
2. A set of custom variant callers identifies extremely rare and unknown variants in a pedigree-aware way. This component includes a De-Novo caller[7] for identifying de novo mutations (mutations that are present for the first time in one family member).
3. The downstream component annotates the variants with a wide range of information related to functional analysis, population genetics, clinical knowledge, epigenetics, etc. The resulting VCF is ready to be ingested by variant curation tools like xBrowse or Forome Anfisa. An optional last step in the pipeline can directly load the case into Anfisa.

While the pipeline was originally developed for Sun Grid Engine environment, the portable version of the pipeline that we present is written in WDL (Workflow Description Language) and exclusively uses GATK4. It can be executed with Cromwell[4] in any environment supported by Cromwell. We have successfully run cases on our local cluster and on Google Cloud using the DNAStack platform. Testing in the AWS version of Cromwell uncovered several issues in the AWS implementation that are currently being addressed by AWS and Broad Institute teams, prospectively making it a more stable release for the growing Cromwell community.



(a) Integrated Clinical Research Workflow



(b) Pipeline Components

To summarize, the pipeline provides a portable workflow for converting the raw sequencing data in FASTQ format to results usable in a clinical environment. In combination with the Forome Anfisa Variant Curation Tool, it can yield a clinically actionable report. It is used on a routine basis by BGM[5] and SEQaBOO[6] projects.

Our team was among the first to adopt WDL and Cromwell for portable bioinformatics pipelines from upstream to downstream and publish it for benefits of the open source community. The pipeline is being actively promoted within the Undiagnosed Disease Network bioinformatics community.

The WDL workflow for the pipeline comes under the Apache 2.0 license and is available on GitHub in the Forome Association repository: `https://github.com/ForomePlatform/pipeline`

References

[1] Fjukstad B, Bongo LA. A review of scalable bioinformatics pipelines. *Data Science and Engineering 2 (3)*, pages 245-251. 2017.

[2] Causey JL, Ashby C, Walker K, et al. DNAp: A Pipeline for DNA-seq Data Analysis. *Sci Rep. 2018;8(1):6793. doi:10.1038/s41598-018-25022-6*, 2018.

[3] O'Connor B. D., Yuen D, Chung V, Duncan A. G., Liu X. K, Patricia J, Paten B, Stein L, Ferretti V. The dockstore: enabling modular, community focused sharing of docker-based genomics tools and workflows. *F1000Research*, 2017.

[4] Voss K, Gentry J, Van dAG. Full-stack genomics pipelining with GATK4 + WDL + Cromwell 2017. *Available from: https://f1000research.com/posters/6-1379*, 2017.

[5] Haghighi A., Krier J.B., Toth-Petroczy A., Cassa C.A., Frank N.Y., Carmichael N., Fieg E., Bjonnes A., Mohanty A., Briere L.C.et al. An integrated clinical program and crowdsourcing strategy for genomic sequencing and Mendelian disease gene discovery. *NPJ Genome Med.*, 2018.

[6] SEQaBOO - SEQuencing a Baby for an Optimal Outcome. *http://seqaboo.bwh.harvard.edu/* under the NIH grant R01-DC015052-01

[7] Mohanty A, Vuzman D, Francioli L, Cassa C, Toth-Petroczy A, Sunyaev S. novoCaller: a Bayesian network approach for de novo variant calling from pedigree and population sequence data. August 2018.

Epiviz File Server - Query, Compute and Interactive Exploration of data from Indexed Genomic Files

The feasibility and reducing costs of running sequencing experiments has led to the generation of large amounts of genomic data. Genomic data repositories like The Cancer Genome Atlas (TCGA), Encyclopedia of DNA Elements (ENCODE), Bioconductor AnnotationHub and ExperimentHub etc., provide public access to large amounts of genomic data as files. Researchers often download a subset of data from these repositories and perform their data analysis. As these data repositories become larger, researchers often face bottlenecks in their data analysis and exploration. Increasing data size requires longer time to download, pre-process and load files into a database to run queries efficiently. Currently available genome browsers fall into two broad categories. One that uses a database management system to load genomic data from files into tables, create indexes/partitions for faster query of data by genomic intervals. The other category of genome browsers query data directly from indexed genomic file formats like bigbed, bigwig or tabix. Interactive visualization of data can be a powerful tool to enable visual exploration and generate insights. As users get familiar with the data and gain insights, it would be even more efficient to test, validate, visualize and compute the intermediate results of the analysis.

Based on the concepts of a NoDB paradigm, we developed Epiviz file server Python library, an in-situ data query system on indexed genomic files, not only for visualization but also for transformation. The library provides various modules to perform various tasks - Import, Query, Compute, Server API and Visualization. Using the file server, users will be able to explore data from publicly hosted files. We currently support various genomic file formats with indexing - BigBed, BigWig, HDF5 and any format that can be indexed using tabix. Once the data files are defined, users can also define summarizations and transformations on these data files using numpy functions. We use dask to manage, distribute and schedule various query and compute requests on files. Our cache implementation also makes sure we only access bytes not already cached locally. To make it easy for developers, we implemented a server module using the Python Sanic library to be able to make REST queries and access data. Once, the server is in place, We can use our Epiviz genome browser to visualize these results. The browser supports various types of visualizations, heatmap and scatter plots for gene expression, blocks (linear and stacked) tracks for visualizing peaks and line tracks (stacked, multi stacked) for visualizing signal (ChIP-seq, methylation etc). Hovering over a region in one visualization highlights this region in other tracks providing instant visual feedback to the user. These visualizations are developed using web component architecture, are highly customizable, reusable and can be integrated with most framework that support HTML. We also require the server hosting the data files to support HTTP range requests so that the file server's parser module can only request the necessary byte-ranges needed to process the query. A higher level architecture of how Epiviz browser and the file server library is shown in Figure 1.

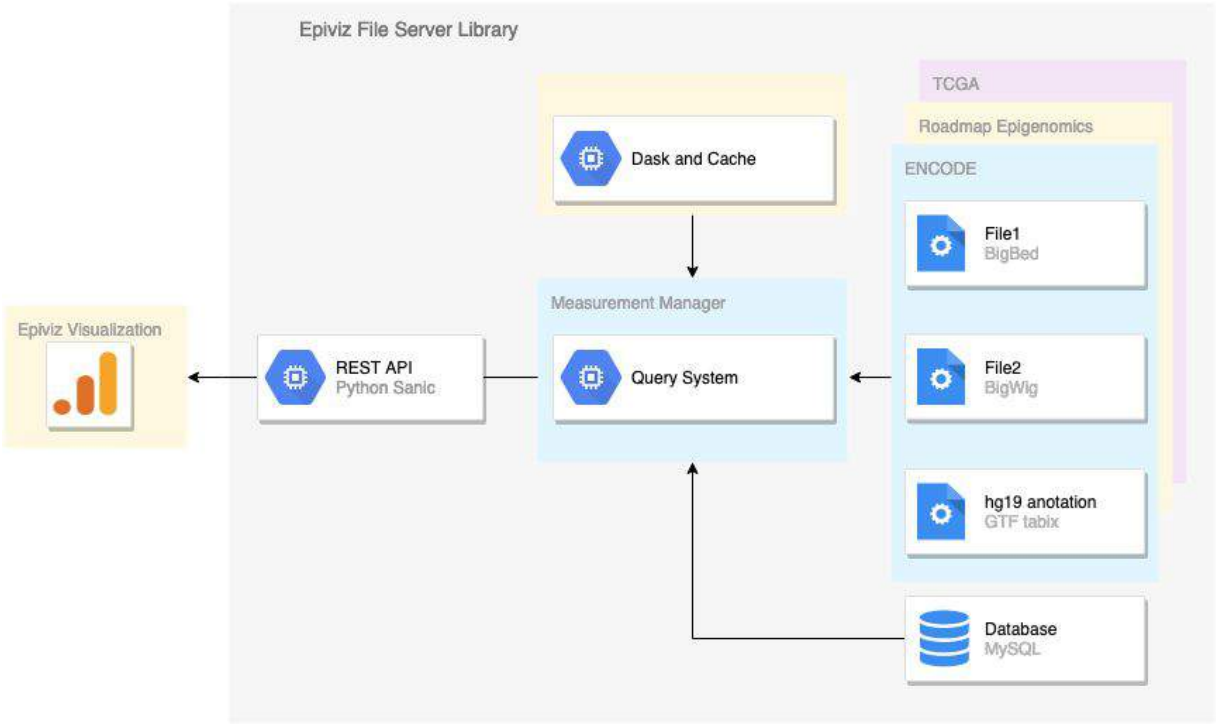


Figure 1. Architecture of the Epiviz File Server, various modules and its integration with the Epiviz browser for visual exploration.

MISO LIMS : managing information for sequencing operations

Morgan Taschuk[1], Heather Armstrong[1], Dillan Cooke[1], Andre Masella[1], Alexis Varsava[1], Lars Jorgensen[1]
1. Ontario Institute for Cancer Research, Toronto, Ontario, Canada

MISO is a laboratory information management system designed for eukaryotic sequencing operations. It supports genomic, exomic, transcriptomic, methyl-omic, and CHiP-seq protocols; long reads and short reads; and microarrays. MISO incorporates a wide feature set useful for both large and small facilities to track their lab workflows in great detail. MISO has two primary goals: 1) to allow laboratory technicians to record their work accurately, without having to adapt their protocols to match the system's model, with a minimum of data entry overhead and 2) to keep the associated metadata valid and structured enough to use for automation and other downstream applications.

History
The software was developed by Robert Davey's lab at the Earlham Institute in Norwich, UK, with the first release in 2011. By 2015, development had slowed to primarily maintenance activities. Our group was looking for a new LIMS and so approached the MISO development team with the intent of developing it to meet specific goals.

Goal 1: Record laboratory activities
MISO is flexible enough to keep up with rapidly evolving research protocols and methods while simultaneously providing input validation and reducing the pain of data entry. Our detailed sample hierarchy mimics actual laboratory processes for receiving tissues, creating stocks and aliquots, propagating to sequencing libraries, pooling and loading onto a sequencer. Quality control measures, volume, and concentrations can be recorded for each entity. Everything corresponding to a physical specimen is also barcoded, located by freezer and shelf, and tracked by changelogs. MISO supports new instruments like the Illumina NovaSeq, 10X Chromium, and Oxford Nanopore PromethION, added more extensive location tracking, and has improved overall performance.

We improved UI interfaces to simplify data entry by providing a spreadsheet-like bulk entry interface for every entity in MISO with functions like fill-down and auto-increment. If more power is necessary, MISO allows exporting and importing Excel spreadsheets. MISO also provides automatic name generators based on project names and templates to automatically fill in values for standard laboratory protocols.

Goal 2: Facilitate analysis automation
A major goal of MISO was to encourage laboratory technicians to enter information in enough detail and with sufficient rigor to automate downstream analysis, particularly base calling and alignment. MISO provides in-browser validation to ensure that controlled vocabulary stays

consistent. MISO also implements Pinery, which is a LIMS abstraction layer to allow read-only access to the metadata required for analysis and reporting.

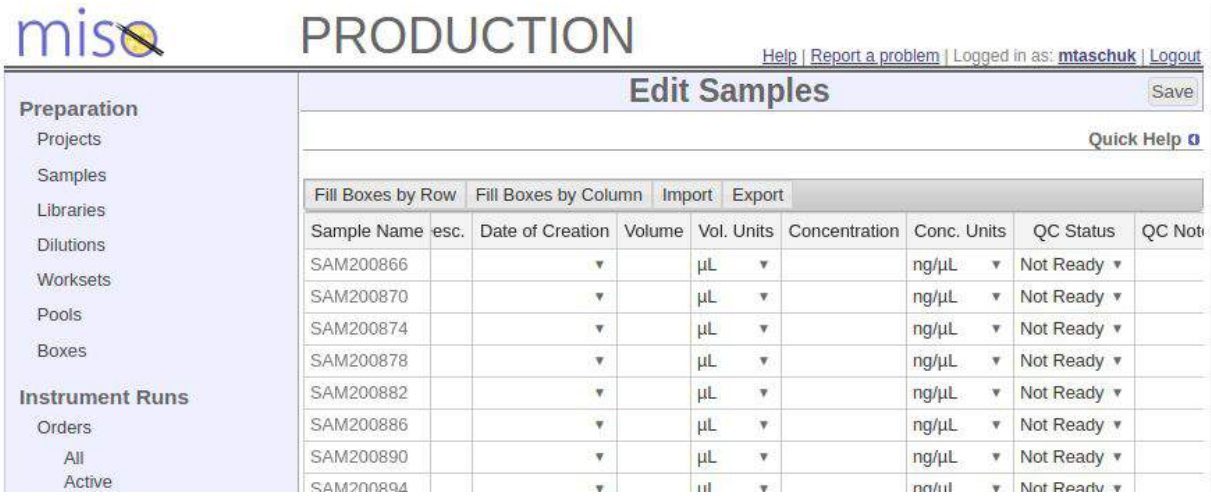


Figure 1: Simplifying data entry by providing a bulk data entry interface that has Excel-like features and templating for standard protocols

Other associated software

MISO is intended for laboratory technicians, and so does not have billing, detailed inventory control, or reporting functions built in. However, we've built other software to fill these needs.

- Pinery: a read-only view of MISO's sample, lane and library data that can be used for a variety of functions, including billing, reporting, and querying.
- Run Scanner: reads information from sequencing instrument directories and can report on the status of past and active sequencing runs in a computationally amenable format
- Shesmu: a decision-driven action launching system that can take information from MISO and Pinery and launch analysis jobs
- Ramen: a detailed inventory tracking system

Availability and Installation

The project is open-source and licensed under GPL-3.0 on Github: https://github.com/miso-lims/miso-lims. MISO is most easily installed using Docker Compose. Detailed instructions are available on the website to start MISO in a 'production' capacity within a few minutes with persistent storage, SSL encryption, and the features mentioned above. A detailed user manual and comprehensive walkthroughs are also available for training.

Future directions

MISO is used in production by several institutions, including the Ontario Institute for Cancer Research. After 8 years of development, we consider MISO feature-complete and are preparing a 1.0 release for late 2019.

Day1 11:40-12:00

Title: BioLink Model - standardizing knowledge graphs and making them interoperable.
Authors: The Biomedical Data Translator Consortium

Biological Knowledge Graphs (KGs) are an emerging way of connecting together and reasoning about entities such as genes, conditions, chemicals, pathways, tissues, and so on. However, as yet there is no agreed upon standard schema or data model for how such graphs should be constructed, resulting in siloed efforts. Creating such a standard is important but challenging due to the complexity of biology and the diversity of use cases.

The BioLink Model (BLMod) is a top-level ontology and a data model that aims to represent biological knowledge. It defines entities (e.g., gene, protein, disease, chemical substance) and enumerates associations between these entities (e.g., gene to disease association). Each entity type is defined as a class, and each class has mappings to other ontologies (e.g., OBO, SIO or WikiData) that enables modeling across ontologies. BLMod aims to serve as a way of standardizing how entities (nodes) and associations (edges) between these entities are represented. BLMod treats associations as first class entities, which enables expressive modeling of data and the ability to add additional properties that define the relationship (e.g., qualifiers, provenance, etc.). The model itself is agnostic to the technology used to build a KG.

We developed this model as part of the NCATS Biomedical Data Translator effort, which seeks to integrate and reason over multiple types of existing data sources, including objective signs and symptoms of disease, drug effects, and intervening types of biological data relevant to understanding pathophysiology. This effort is a collaboration between multiple teams, each of which has their own knowledge graph technology and data integration pipelines. KGs can be layered on different database systems with their own formalisms and metamodels. Within the Translator Consortium, we have RDF triple stores, graph databases such as Neo4J and datalog-based graph databases (miniKanren). The underlying metamodel imposes certain freedom and constraints on biological modeling. For example, RDF graphs are composed of 3-ary tuples (aka triples), whereas graph databases such as Neo4J utilize a property graph model, which allows information to be attached to edges. This diversity of representations posed particular challenges for defining a unifying schema. For example, OWL can be used to define a formalism that applies to RDF graphs, but not property graphs. Emerging standards such as schema.org, aimed at search engine optimization, also do not utilize the expressive features of property graphs.

To bridge across these differences we devised our own data modeling language, and defined the BLMod using this language. The two core parts of BLMod are:

- A hierarchical classification of core entity types, e.g., gene, disease, chemical substance. Each entity is mapped to multiple external resources such as OBO, SIO, WikiData and UMLS.
- A hierarchical classification of association types that constrain how entities can be connected together, and what additional properties are to be expected. For example, a disease can be connected to a phenotype via the disease to phenotypic feature association, and may have additional properties such as severity, onset and frequency. All associations are typed using ontologies such as the OBO Relation Ontology.

Although the core formalism is based on property graphs, we have standardized translation to RDF and are leveraging technology such as ShEx (Shape Expressions) for validation and profiling. We also provide translations of the schema to SQL DDL, JSON Schema, GraphQL, and OWL. One unique feature of BLMod, in contrast to schemas such as BioSchemas, is the concept of an association as a first class entity. In BioSchemas, a gene can be connected directly with a disease, but there is no way to annotate this linkage with provenance, evidence, or additional biological details about the nature of the connection.

Additionally, we developed a lightweight Python toolkit (KGX; BSD-3 License) for aggregating, validating, and merging different KGs. This is being used in the context of the Translator project to combine the works of different group's integration and reasoning efforts. We have used KGX to integrate sources such as the Monarch Knowledge Graph, the Semantic Medline database, and the Data2Services Knowledge Graph. We are also using KGX to build a large combined 'uber' knowledge graph for machine learning.

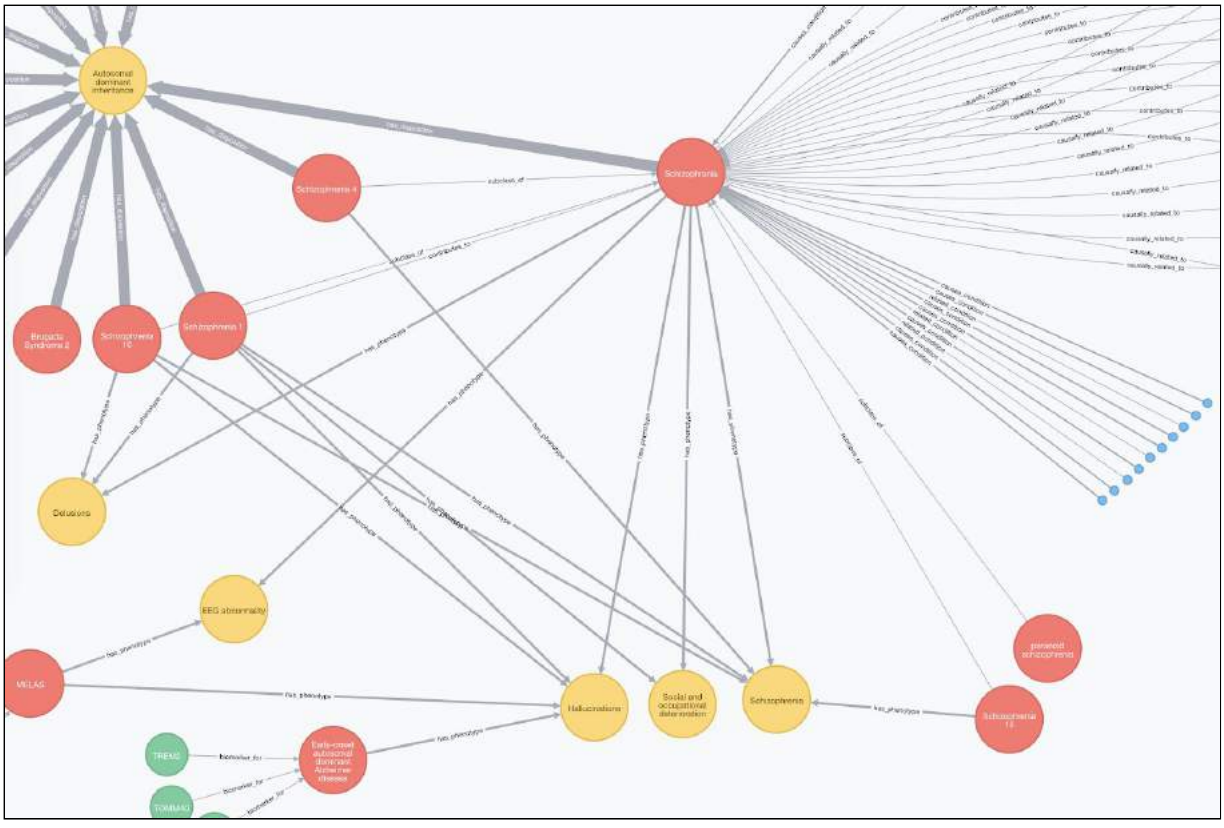


Figure 1: A Neo4j browser view of a Knowledge Graph modeled using the BioLink Model. Nodes in red, yellow, green and blue represent 'disease', 'phenotypic feature', 'gene' and 'sequence variant' entities, respectively. Links between the nodes represent the relationships between entities.

Using BLMod ensures that all the KGs are using the same dialect to represent knowledge. This reduces the burden of source-specific data access logic from agents (either users or machines) while improving discoverability. The model itself is open source (CC0 1.0 Universal License) and available on GitHub.

pysradb: A Python package to query next-generation sequencing metadata and data from NCBI Sequence Read Archive

Saket Choudhary*

Website : http://saketkc.github.io/pysradb/
Repository : https://github.com/saketkc/pysradb
Example : https://github.com/saketkc/pysradb/blob/master/docs/usage_scenarios.rst
License : BSD 3-Clause

NCBI's Sequence Read Archive (SRA) is the primary archive of next-generation sequencing datasets. SRA makes metadata and raw sequencing data available to the research community to encourage reproducibility, and to provide avenues for testing novel hypotheses on publicly available data. However, methods to programmatically access this data are limited. NCBI's SRA toolkit [1] provides utility methods to download raw sequencing data, while the metadata can be obtained by querying the website or through the Entrez `efetch` command line utility [2]. Most workflows analyzing public data rely on first searching for relevant keywords in the metadata either through the command line utility or the website and then downloading these. A more streamlined workflow can enable doing both these steps at once.

We introduce a Python package `pysradb` that provides a collection of command line methods to query and download metadata and data from SRA utilizing the curated metadata database available through the SRAdb [3] project.

`pysradb` package builds upon the principles of `SRAdb` providing a simple and user-friendly command-line interface for querying metadata and downloading datasets from SRA. It obviates the need for the user to be familiar with any programming language for querying and downloading datasets from SRA. Additionally, it provides utility functions that will further help a user perform more granular queries, that are often required when dealing with multiple datasets at large scale. By enabling both metadata search and download operations at the command-line, `pysradb` aims to bridge the gap in seamlessly retrieving public sequencing datasets and the associated metadata.

`pysradb` is written in Python and is currently developed on Github under the open-source BSD 3-Clause License. Each sub-command of `pysradb` contains a self-contained help string, that describes its purpose and usage example with additional documentation available on the project's website. In order to simplify the installation procedure for the end-user, it is also available for download through both PyPI and bioconda [4].

References

[1] SRA Toolkit Development Team, "Sra toolkit." https://ncbi.github.io/sra-tools/, Dec 2018. [Online; accessed 10-December-2018].

[2] J. Kans, "Entrez direct: E-utilities on the unix command line," 2018.

[3] Y. Zhu, R. M. Stephens, P. S. Meltzer, and S. R. Davis, "Sradb: query and use public next-generation sequencing data from within r," *BMC bioinformatics*, vol. 14, no. 1, p. 19, 2013.

[4] B. Grüning, R. Dale, A. Sjödin, B. A. Chapman, J. Rowe, C. H. Tomkins-Tinch, R. Valieris, J. Köster, and T. Bioconda, "Bioconda: sustainable and comprehensive software distribution for the life sciences.," *Nature methods*, vol. 15, no. 7, p. 475, 2018.

*Computational Biology and Bioinformatics, University of Southern California, Los Angeles, USA. Email: skchoudh@usc.edu

Day1 12:05-12:10 (P-29)

Disq, a library for manipulating bioinformatics sequencing formats in Apache Spark.

Code repository: https://github.com/disq-bio/disq
Software license: The MIT License (MIT)
Software license in code repository: https://github.com/disq-bio/disq/blob/master/LICENSE.txt

ADAM and GATK have independently developed parallel and distributed genomic applications on Apache Spark.

To access flat file formats such as BAM, CRAM, SAM, and VCF, both depend on the htsjdk library, which provides low-level codecs, and the Hadoop-BAM library, which extends these for parallel and distributed access.

Hadoop-BAM was found to have correctness (invalid BAM file splits, leading to corrupt read data) and performance (sequential implementation of some parallelizable tasks) issues. The Spark-BAM project demonstrated these issues could be addressed, and developed a comprehensive benchmark.

Thus members of the ADAM, Hadoop-BAM, htsjdk, GATK, Spark-BAM, and ViraPipe projects identified an opportunity to collaborate on a replacement library. Discussion between collaborators began virtually, then in-person at OpenBio Winter Codefest 2018 in Boston, and continued at GCCBOSC Collaboration Fest 2018 in Portland. A new project Disq was started in 2018, and has since made at least three releases (most recently version 0.3.0, released 19 March 2019).

Benchmarks show that Disq is faster and more accurate than Hadoop-BAM, and at least as fast as Spark-BAM.

Disq also adds significant new features, such as support for writing sharded files for efficiency, for taking advantage of index files while reading (e.g. .sbi index files to find splits between BAM records, .crai index files to find record boundaries in CRAM files), and for writing index files where appropriate.

In addition to unit tests, Disq includes integration tests that run against real-world files (multi-GB in size). SAMtools and BCFtools are used to verify files written with Disq can be read successfully.

Disq has been incorporated into ADAM and GATK, and will provide a convenient venue for further collaboration between those project teams. We also welcome new collaborators seeking correct and performant access to flat file formats on Apache Spark.

References

ADAM code repository, https://github.com/bigdatagenomics/adam

BCFtools code repository, https://github.com/samtools/bcftools

Disq benchmarks, https://github.com/tomwhite/disq-benchmarks

GATK code repository, https://github.com/broadinstitute/gatk

Hadoop-BAM code repository, https://github.com/HadoopGenomics/Hadoop-BAM

Htsjdk code repository, https://github.com/samtools/htsjdk

Maarala, Altti Ilari, et al. "ViraPipe: scalable parallel pipeline for viral metagenome analysis from next generation sequencing reads." Bioinformatics 34.6 (2017): 928-935.

Massie, Matt, et al. "Adam: Genomics formats and processing patterns for cloud scale computing." University of California, Berkeley Technical Report, No. UCB/EECS-2013 207 (2013): 2013.

McKenna, Aaron, et al. "The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data." Genome research 20.9 (2010): 1297-1303.

Niemenmaa, Matti, et al. "Hadoop-BAM: directly manipulating next generation sequencing data in the cloud." Bioinformatics 28.6 (2012): 876-877.

Nothaft, Frank Austin, et al. "Rethinking data-intensive science using scalable analytics systems." Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015.

Li, Heng, et al. "The sequence alignment/map format and SAMtools." Bioinformatics 25.16 (2009): 2078-2079.

SAMtools code repository, https://github.com/samtools/samtools

Spark-BAM code repository, https://github.com/hammerlab/spark-bam

ViraPipe code repository, https://github.com/NGSeq/ViraPipe

Day1 12:10-12:15

A toolkit for semantic markup, exploration, comparison and merging of metadata models expressed as JSON-Schemas.

Dominique Batista, Alejandra N. Gonzalez-Beltran, Susanna-Assunta Sansone and Philippe Rocca-Serra.
University of Oxford e-Research Centre, Department of Engineering Science, University of Oxford, 7 Keble Road, OX1 3QG, Oxford, United Kingdom.

There is no shortage of data models and annotation checklists representing the various domains of life sciences and their associated data (https://fairsharing.org/collection/MIBBI). It is becoming increasingly hard for developers to choose among those many standards and create data complying with those specifications and which can be explored by a range of systems, services as well human agents. Here, we introduce a set of tools aimed at assisting knowledge engineers who rely on JSON schema technology to define semantic anchoring of schema elements, compare and combine those with other elements or existing schemas, as well as visualize and present such schemas and their comparisons in an aesthetically pleasing interface. The core Python library provides functions to support the semantic annotation of JSON schema: more specifically, given a set of schemas and vocabularies, the tool will generate the required JSON-LD context files. The library also offers a comparison algorithm that makes use of the aforementioned semantic annotations to compare sets of schemas (i.e. schemas and their dependencies). Built on top of this functionality, a merge function enables developers to combine components defined by existing data representation standards and make providers' content compatible with several standards at a time. In addition, the suite of tools also contains two client-side web applications that are used as visualisation tools. The first one, called *JSON-Schemas Documenter*, resolves a set of schemas and presents the properties of each element (see figure 1). The second one reads comparison files created by the python library and outputs pairwise comparison reports, with dedicated visual cues to home in on problematic elements (see figure 2). To demonstrate the usefulness and applicability of the software component, several minimal requirements checklists (MIACA, MIACME...) and models (MiFlowCyt and DATS) have been expressed in JSON-Schemas semantically enhanced from their original forms (see http://github.com/fairsharing/mircat). These JSON-Schemas were annotated with ontology terms registered in associated JSON-LD context files, thus providing models understandable by both humans and machines. The combination of these tools has proved extremely helpful i.) when verifying that sets of schemas are correctly annotated, ii.) when computing the overlaps between complex sets of schemas such as the ones above and iii.) when visually verifying the validity of the merge options. Ongoing efforts are under way to integrate the tools with FAIRsharing services as well as expand the set of supported models and minimum information checklists in a push to promote reuse.

Open source code (license included in the repositories):

- Python library: https://github.com/FAIRsharing/jsonldschema
- Documenter: https://github.com/FAIRsharing/JSONschema-documenter
- Comparison viewer: https://github.com/FAIRsharing/JSONschema-compare-and-view

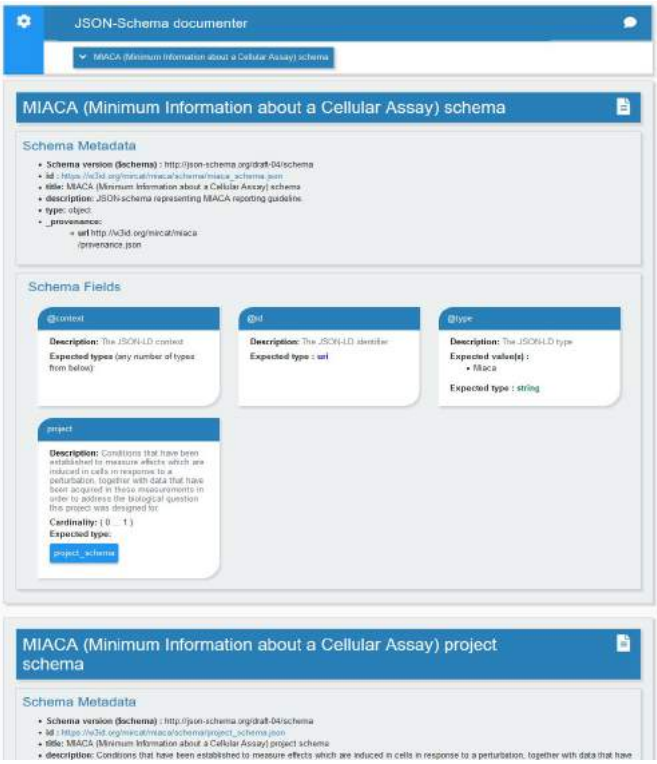


Figure 1: Screenshot of the JSON-Schema documenter loaded with the MIACA checklist.

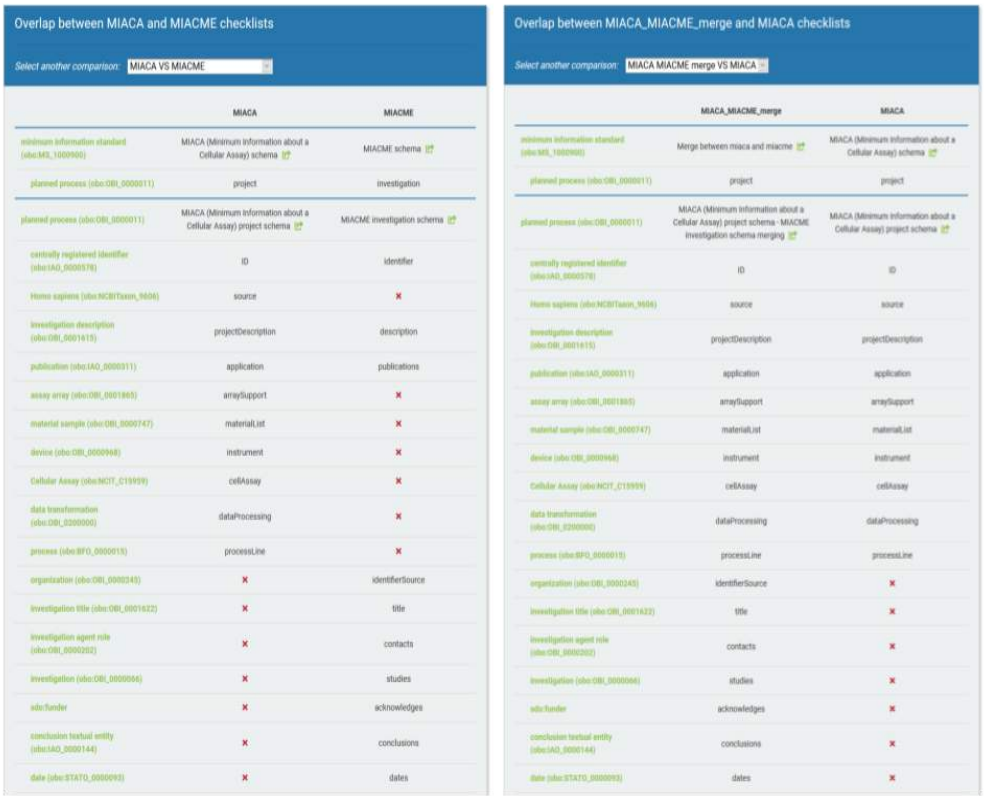


Figure 2: Sample comparison between MIACA AND MIACME checklists (left) and the merge of MIACA and MIACME checklists with MIACA checklist (right) based on ontology values obtained in context files.

A lightweight approach to Research Object data packaging

http://www.researchobject.org/ro-crate/
http://github.com/researchobject/ro-crate
Apache License, version 2.0

Eoghan Ó Carragáin, *Carole Goble*, *Peter Sefton*, *Stian Soiland-Reyes*

A **Research Object** (RO) provides a machine-readable mechanism to communicate the diverse set of digital and real-world resources that contribute to an item of research. The aim of an RO is to replace traditional academic publications of static PDFs, to rather provide a complete and structured archive of the items (such as people, organisations, funding, equipment, software etc) that contributed to the research outcome, including their identifiers, provenance, relations and annotations. This is increasingly important as researchers now rely heavily on computational analysis, yet we are facing a *reproducibility crisis* [1] as key components are often not sufficiently tracked, archived or reported.

We propose **Research Object Crate** (or **RO-Crate** for short), an emerging lightweight approach to package research data with their structured metadata, based on schema.org annotations in a formalized JSON-LD format that can be used independent of infrastructure to encourage FAIR sharing of reproducible datasets and analytical methods.

Background

Earlier work introduced the notion of *Research Objects* [2]. Their formalization combines existing *Linked Data* standards: W3C RDF, JSON-LD, OAI-ORE, W3C Web Annotations, PROV, Dublin Core Terms, ORCID. The RO ontologies [3] combined these to describe ROs, but do not themselves formalize how ROs are saved or transmitted. Multiple formats have since been realized: the portal RO Hub [4] use RDF REST resources; while workflow provenance make RO Bundle ZIP files [5] or Big Data BagIt archives [6, 7]. Each of these require RO support in the packaging infrastructure.

Multiple *data packaging* initiatives have recently emerged, within Research Data Alliance, Force11, DataOne and elsewhere; like Frictionless data [8] for table-like files, BioCompute Objects for regulatory science [9], CodeMeta for software, Psych-DS for psychology studies, and DataCrate [10] for datasets. RDA has surveyed a large variety of data packaging formats across different domains.

Common among these is *structured metadata*, e.g. with a single JSON file that refer to neighbouring data files and scripts maintained and published together, e.g. in GitHub. Many of these initiatives use schema.org [11] as basis for common metadata. With JSON-LD this offers a developer-friendly experience and interoperability with web conventions outside of the research domain.

Data packaging principles

At a RDA meeting on data packaging we concluded that many initiatives arrive at similar principles: simple folder structure; JSON-LD manifest; schema.org for core metadata; BagIt for fixity; OAI-ORE for aggregation. This points to: a) appetite for general package/folder-oriented approach in different contexts; b) a generic solution won't work for all and needs to be domain-extensible; c) a tendency to re-invent the wheel, leading to sub-optimal interoperability and duplication of effort.

Cite as: https://doi.org/10.5281/zenodo.3250687

We have identified a gap for a solid base format for data packaging that also allow communities to build domain-specific solutions. Frictionless data [8] could arguably fill this gap, with mature specifications and a strong design philosophy, however as an independent JSON format it does not fully apply Linked Data principles, and would be harder to use in FAIR integrations and extensions.

Our proposal is to build on DataCrate [10] to evolve **RO-Crate**, based around these principles: a) metadata as Linked Data, using schema.org as much as possible; b) extensible for different domains; c) retain the core Research Object principles *Identity, Aggregation, Annotation*; d) inferred metadata rather than repetition; e) "just-enough" provenance; f) layered validation; g) archivable with BagIt; h) hooks to reuse existing domain formats; i) lightweight programmatic generation and consumption. Similar to the approach of BioSchemas, rather than building new specifications from scratch, we aim to build best-practice guides and validatable profiles for building rich research data packages with existing standards, without requiring expert knowledge for developing producers and consumers.

Building community consensus

RO-Crate is a fresh initiative, bringing together data archive and repository maintainers with existing Research Object, workflow and provenance communities. Starting as a small cross-domain group, organically formed to build the core principles and first sketches of their use, we are now expanding to collect use cases and reaching out to other packaging initiatives to build common ground.

One emerging use of RO-Crate is for capturing workflows and tools in a federated *workflow repository* being built in EOSC-Life, a large European Open Science Cloud project across 13 research infrastructures in the life science domain. However RO-Crate is also aiming to be usable by individual scientists with no particular infrastructure beyond Jupyter notebook, who may not have the time or motivation to use a cascade of metadata vocabularies and research data management tools [12].

RO-Crate development and discussion is done openly in a GitHub repository by volunteers, with monthly telcons to synchronize the effort. Anyone can join to help form the RO-Crate approach.

References

[1] Monya Baker (2016): **1,500 scientists lift the lid on reproducibility**. *Nature* **533**. https://doi.org/10.1038/533452a

[2] Sean Bechhofer et al (2013): **Why Linked Data is Not Enough for Scientists**, *Future Generation Computer Systems* **29**(2) https://doi.org/10.1016/j.future.2011.08.004

[3] Khalid Belhajjame et al (2015): **Using a suite of ontologies for preserving workflow-centric research objects**. *Web Semantics*: Science, Services and Agents on the World Wide Web, https://doi.org/10.1016/j.websem.2015.01.003

[4] Jose Manuel Gomez-Perez et al (2017): **Towards a Human-Machine Scientific Partnership Based on Semantically Rich Research Objects**. *IEEE 13th International Conference on e-Science (e-Science 2017)*. https://doi.org/10.1109/eScience.2017.40 [preprint available]

[5] Stian Soiland-Reyes, Pinar Alper, Carole Goble (2016): **Tracking workflow execution with TavernaProv**. At *ProvenanceWeek 2016; PROV: Three Years Later.* 6 Jun 2016, Washington DC, US. https://doi.org/10.5281/zenodo.51314

[6] Ravi K Madduri et al (2018): **Reproducible big data science: A case study in continuous FAIRness**. https://doi.org/10.1101/268755

[7] Farah Zaib Khan et al (2018): **Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv**. Submitted to *GigaScience*. https://doi.org/10.5281/zenodo.1966881

[8] Jo Barratt, Serah Rono (2018): **Frictionless Data and Data Packages** At *Workshop on Research Objects (RO 2018)*, 29 Oct 2018, Amsterdam, Netherlands. https://doi.org/10.5281/zenodo.1301152

[9] Gil Alterovitz et al (2018): **Enabling Precision Medicine via standard communication of NGS provenance, analysis, and results**. *PLOS Biology.* **16**(12):e3000099 https://doi.org/10.1371/journal.pbio.3000099

[10] Peter Sefton et al (2018): **DataCrate: a method of packaging, distributing, displaying and archiving Research Objects** At *Workshop on Research Objects (RO 2018)*, 29 Oct 2018, Amsterdam, Netherlands. https://doi.org/10.5281/zenodo.1445817

[11] R. V. Guha, Dan Brickley, Steve Macbeth (2016): **Schema.org: evolution of structured data on the web**. *Communications of the ACM* **59**(2). https://doi.org/10.1145/2844544

[12] Cameron Neylon (2017): **As a researcher…I'm a bit bloody fed up with Data Management.** Blog *Science in the Open.* http://cameronneylon.net/blog/as-a-researcher-im-a-bit-bloody-fed-up-with-data-management/ [archived 2019-04-12]

Part of this work has been done at the BioExcel CoE (www.bioexcel.eu), a project funded by the European Union contracts H2020-INFRAEDI-02-2018-823830, H2020-EINFRA-2015-1-675728.

Day1 15:00-15:20

The (Re)usable Data Project

Seth Carbon*

E-mail: sjcarbon@lbl.gov

19th Bioinformatics Open-Source Conference (BOSC) 2019, Basel, Switzerland

Website: http://reusabledata.org
Repository: https://github.com/reusabledata/reusabledata

The goal of the (Re)usable Data Project[1] (RDP, http://reusabledata.org) is to draw attention to the licensing issues that make the reuse of valuable biomedical data challenging and complicated. The RDP is meant to provide an explorable resource that looks at some of the issues around the reuse of scientific data and open a conversation about how to deal with them. The RDP exists as an element in the environment of the FAIR Data Principles and the FAIR-TLC evaluation framework; it is focused on licensing issuesnarrowly scoped to reusabilityand seeks to draw attention to the pervasiveness of current practice failures and their effects. As the current centerpiece of this project, the RDP has put together a rubric that attempts the objective evaluation of a data resources license and basic data accessibility from the perspective of reuse on a linear scale.

While the criteria can be quite detailed (http://reusabledata.org/criteria), we have attempted to balance many needs (credit, mutability, commercialization, redistribution, etc.) and focus on trying to objectively evaluate how licenses can interact across resources. The RDP assesses licenses for the following criteria:

- **Clearly stated**
  A clearly stated, unambiguous, and hopefully standard, license for data use is critical for any (re)use of data: if there is no license to be found, then rights are unclear and one needs to assume the default: all rights reserved.
- **Comprehensive and non-negotiated**
  Data that is mixed under different licenses, only partially available, or must be in some way negotiated creates barriers to the (re)use of data.
- **Accessible**
  Data must be accessible in a reasonable and manner to be useful to the broader community.
- **Avoid restrictions on kinds of (re)use**
  Data should be able to be copied, built upon, edited, and modified as freely as possible.
- **Avoid restrictions on who may (re)use**
  Data should be available to as many people as possible for their (re)use.

Over the past couple of years, the RDP has processed and evaluated about sixty resources and their licenses using the developed rubric. We will present the rubric that we created, highlight some of the results of this examination, and explore license choices in the current licensing landscape.

Overall, the results of the RDPs examination indicate that there are ongoing issues with how resources license and present their data. While a great number and variety of publicly-funded biomedical

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

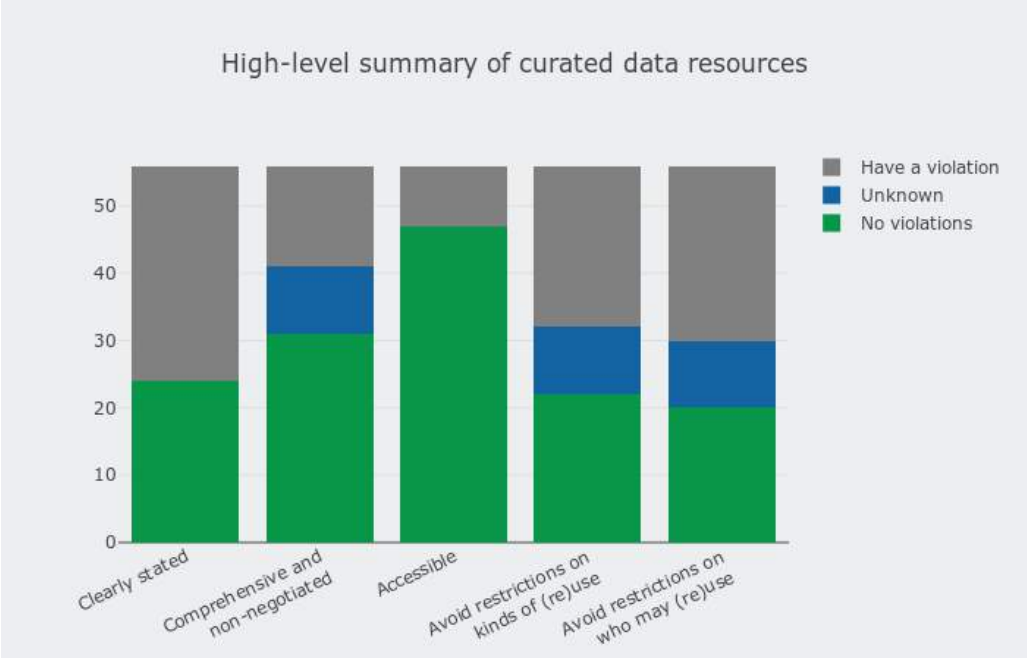


Figure 1: Number of resources that pass or fail criteria in each of the five categories of the rubric.

datasets are ostensibly open, complex licensing issues may be hindering them from being put to their best use.

A lack of licensing rigor and standardization can force downstream consumers to laboriously seek, essential reuse and redistribution permissions. Issues we observed include missing licenses, nonstandard licenses, and license provisions that are restrictive or incompatible. The legal interpretation of, and compliance with, database license and reuse agreements has become a significant burden and expense for many fields in the scientific community, where a complex and lengthy set of legal negotiations may be required for a data integration project to legally and freely redistribute all of its relevant data. This landscape does not benefit data providers, users, or scientific progress, especially from the perspective of reuse.

We hope that the efforts of the RDP will encourage the community to work together and generate discussions that help improve licensing practices, facilitating broad and long-term access to reusable scientific resources. Reusing data, especially when aggregating and synthesizing, comes with numerous challenges that we believe can be overcome with community engagement in these critical issues. The rapidly evolving landscape of funder and journal data sharing policies offers an opportunity to provide explicit direction to upstream data contributors about licensing and other practices that are consistent with the FAIR data principles and positively impact reuse.

References

[1] An Analysis and Metric of Reusable Data Licensing Practices for Biomedical Resources. *PLOS ONE* 14, no. 3 (March 27, 2019): e0213090, https://doi.org/10.1371/journal.pone.0213090.

The FAIR data principles and their practical implementation in InterMine

D. Butano[1], J. Clark-Casey[1], S. Contrino[1], J. Heimbach[1], R. Lyne[1], J. Sullivan[1], Y. Yehudi[1] and G. Micklem[1]
[1]*Department of Genetics, University of Cambridge, Cambridge, United Kingdom*

Abstract

The FAIR Data Principles [1] are a set of guidelines which aim to make data findable, accessible, interoperable and reusable. The principles are gaining traction, especially in the life sciences. We will present our experience of the practical implementation of the FAIR principles in InterMine [2], a platform to integrate and access life sciences data. We will cover topics such as the design of persistent URLs, standards for embedding data descriptions into web pages, describing data with ontologies, and data licences.

Introduction

Science is generating ever more data, faster than ever before. Reliably storing and retrieving this data isn't enough; integration between different datasets, from different sources is becoming equally important. Wider adoption of the FAIR principles will make this process easier and facilitate data use by machine and humans.

InterMine is a platform to integrate and access life sciences data, providing flexible querying through a user-friendly web interface as well as RESTful web services [3] . Whilst InterMine comes with a core data model for common biological entities, and loaders for popular data sources and file types, different deployments can extend these components to publish any type of data.

InterMine is an established platform first released in 2006, and already includes some FAIR principles such as search and structured query functionalities, web services, and cross-references to other InterMine instances and resources. We will describe here how we are improving InterMine adherence to FAIR principles.

Generating persistent URLs for web pages

InterMine already has unique URLs to identify the report pages for biological entities, but these are based on internal InterMine IDs that change at every database build and are therefore not persistent.

To achieve data **findability** and **accessibility**, we have generated new URLs based on the InterMine class names combined with local IDs provided by the data resource providers. For example, in HumanMine, the URL of the report page for the protein MYH7_HUMAN, with UniProt accession P12883, will be *www.humanmine.org/protein:P12883*.

Describing data with ontologies

The InterMine system is based on a core data model, described in an XML file, which defines classes (the entities in the model) and the relationships between them. The core model can be

extended with any number of additions files, which define new classes or fields, in order to represent the data types stored in a specific InterMine instance.

InterMine already automatically applied terms from the Sequence Ontology to its data model. To improve data **interoperability** we have now added more ontologies to its core data model and provided InterMine instance administrators with the ability to apply any ontologies to their data model extension. We selected the most appropriate terms from popular ontologies: Sequence Ontology, Semantic Science, EDAM, MeSH, Dublin Core, National Cancer Institute Thesaurus (US NIH).

The ontologies applied are available in the data model and will be used in the generation of RDF.

Marking up web pages

We have applied structured data in JSON-LD format to InterMine web pages, using Bioschemas.org types and profiles to improve **findability** so search engines can give more relevant results to users.

Publishing Data Licences

To improve data **reusability**, InterMine has updated its model, adding the attribute *licence* to include the licences that govern the data sets that have been integrated. We changed the data parsers to add the data licences where available.
We discovered that only a minority of data sets have a licence: of the 26 core dataset types that InterMine supports, only 9 have a data set licence, although 14 had some text about fair use.

We display licence information in the data sets report pages and in query results. We will propagate the licence information when generating RDF.

References

1. Wilkinson MD, Dumontier M, Aalbersberg IJJ, Appleton G, Axton M, Baak A, et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016;3: 160018.

2. Smith RN, Aleksic J, Butano D, Carr A, Contrino S, Hu F, et al. InterMine: a flexible data warehouse system for the integration and analysis of heterogeneous biological data. Bioinformatics. 2012;28: 3163–3165.

3. Kalderimis A, Lyne R, Butano D, Contrino S, Lyne M, Heimbach J, et al. InterMine: extensive web services for modern biology. Nucleic Acids Res. 2014;42: W468–72.

Day1 15:40-15:45 (P-05)

Global Alliance for Genomics & Health

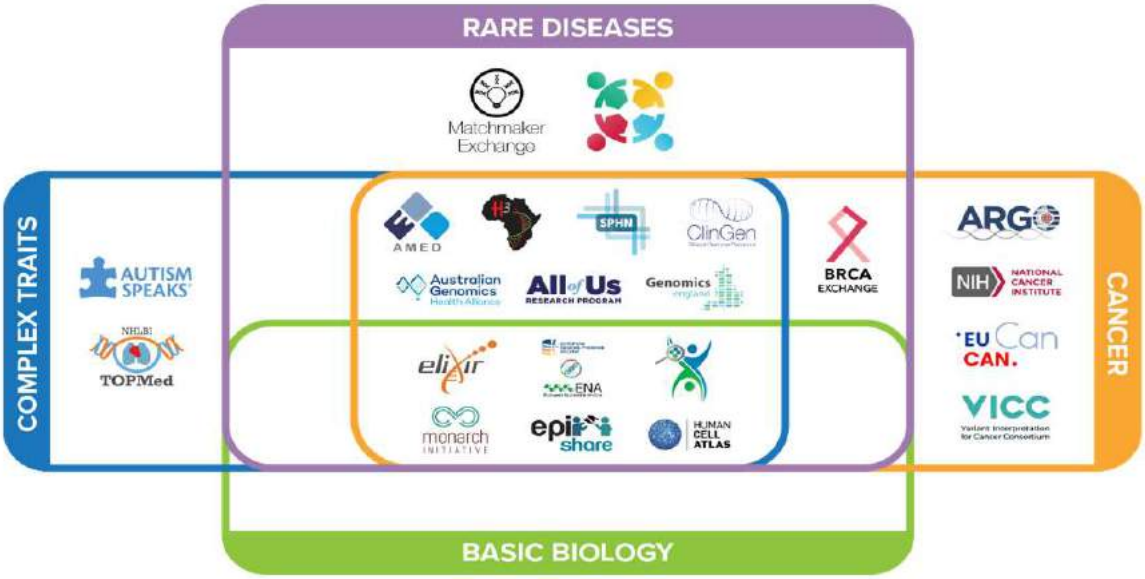
Collaborate. Innovate. Accelerate.

GA4GH: Developing Open Standards for Responsible Data Sharing

The Global Alliance for Genomics and Health (GA4GH) is creating frameworks and standards to enable the responsible, voluntary, and secure sharing of genomic and health-related data. This talk will introduce these specifications, the community based process in which they are developed, and show how you can contribute to this process.

GA4GH Development Process

There are benefits in being able to process large cohorts that range from improved understanding of basic biology to the areas of diagnosis and treatment of rare genetic disease and cancer. As genomic data collections are built outside of the research community technical and regulatory challenges are presented to researchers. By bringing together those building large data collections such as AllOfUS, the European Genome-Phenome Archive and GEnome Medical Alliance (GEM) Japan, and those working in specialist areas such as Matchmaker Exchange and ICGC-ARGO, GA4GH identifies and builds the open standards that are needed to cross bridges and boundaries in efforts to harness large cohorts of human genetic data. The development process is open and participation from members outside these projects are part of this cycle.

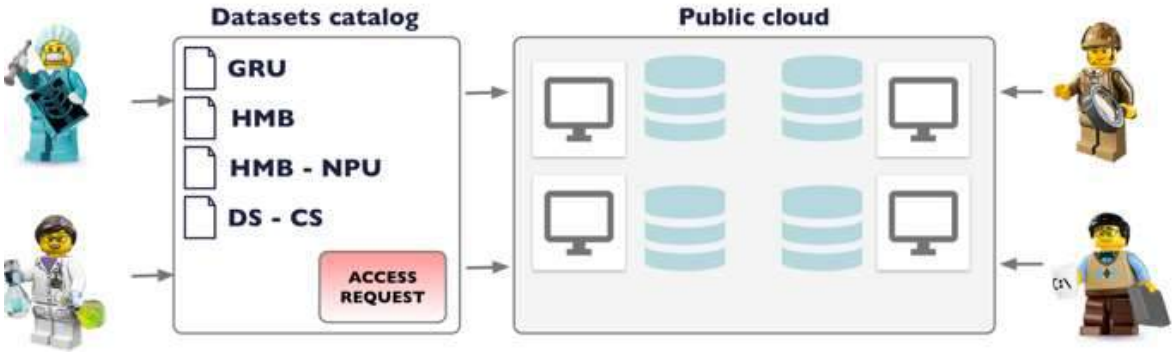


GA4GH Standards

The existing GA4GH Toolkit includes standard file formats such as CRAM and BCF, the Beacon API for querying allele presence in a dataset, and Cloud compute enabling standards such as Workflow Execution Service (WES), and the Data Use Ontology (DUO) and htsget detailed below.

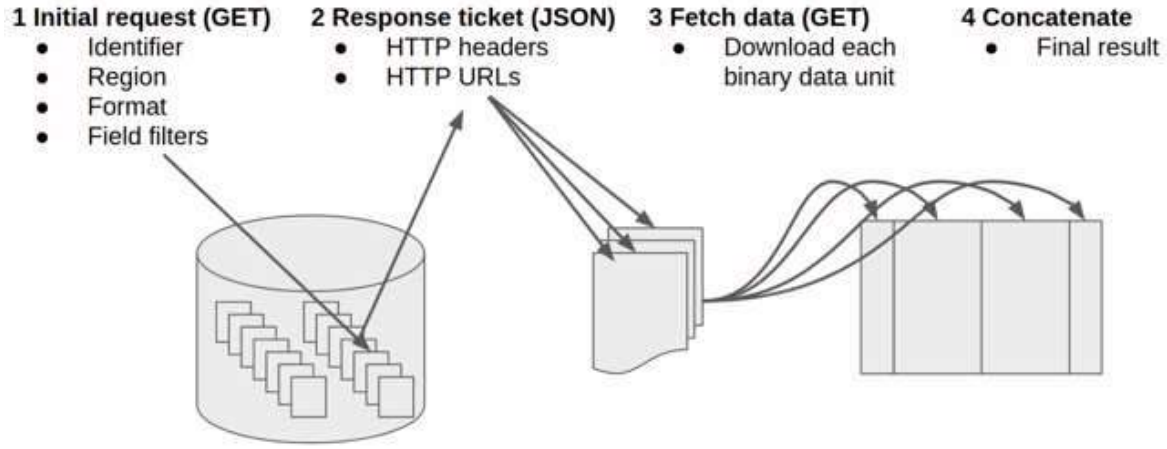
Data Use Ontology

Allows users to semantically tag genomic datasets with usage restrictions, allowing them to become automatically discoverable based on a health, clinical, or biomedical researcher's authorization level or intended use.



Htsget

Allows users to download read data for subsections of the genome in which they are interested



GA4GH Roadmap

To complete the GA4GH vision of accessing data and transferring the results back into the clinical space, further specifications are on the way! These shall be outlined here.

Participation

GA4GH is an organisation with many organisational members and individual contributors. By GA4GH standards being incorporated into software and standing up endpoints providing data in GA4GH formats an enabling ecosystem for genomic data sharing is being built upon. Contributions are welcome to the standards themselves by participation in meetings or via GitHub channels.

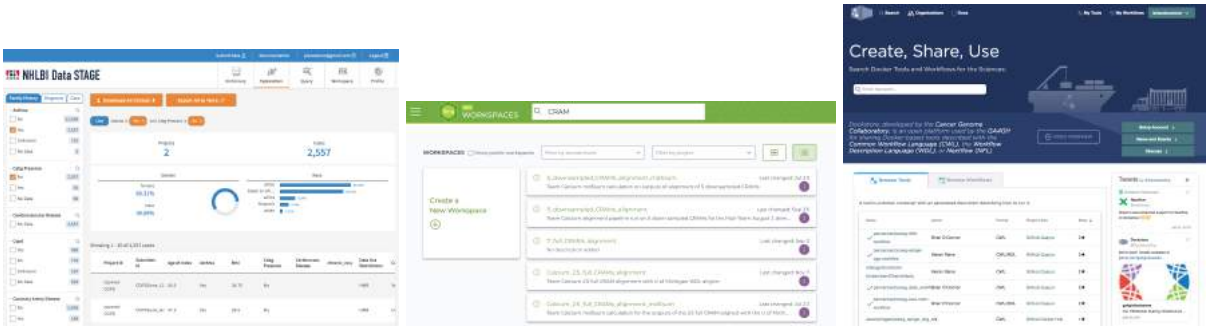
Day1 15:45-15:50

Title: The Commons Alliance: Building cloud-based infrastructure to support biomedical research in Data STAGE and AnVIL

Abstract:

Modern biomedical research datasets -- derived from diverse technologies such as genome sequencing, gene-expression analysis, proteomics, and imaging assays -- consist of copious amounts of data that many researchers struggle to leverage. While the ability to generate data is a massive opportunity for the biomedical research community, the infrastructure and skill set required to handle terabytes - even petabytes - of data is relatively rare. Researchers need to be simultaneously well versed in information technology and Linux server administration, deeply familiar with computer science and programming, and able to understand experimental design and hypothesis generation in their research area of focus. For many, these simultaneous requirements for expertise in such diverse areas is extremely challenging. This underscores the need to streamline and simplify the skill sets needed in order to successfully analyze data. A shift to cloud environments, despite solving many of the problems around data access and infrastructure, does not address this skill requirement issue. We need to provide researchers with approachable and easy-to-use cloud-based tooling for asking scientific questions of these datasets without getting bogged down in infrastructure challenges.

Here we present the work of the Commons Alliance, a collaboration between the Broad Institute, UCSC, the University of Chicago, and Vanderbilt to build cloud-based infrastructure and services for the biomedical research community. These components include the Gen3 platform (https://gen3.org/) for core data and authentication/authorization services, the Terra workspace platform (https://terra.bio/) for batch and interactive analysis, and the Dockstore registry (https://dockstore.org/) for workflow and tool sharing (co-developed with OICR). Each component shares a core design feature of a high-quality, web-based graphical user interface and focuses on making powerful cloud-based systems approachable and usable by a wide range of researchers. As part of the Commons Alliance, the groups behind Gen3, Terra, and Dockstore have endeavoured to work together to ensure that the whole range of tasks needed to analyze biomedical research data on the cloud are accessible. Gen3 allows for data onboarding, metadata searching, and a compelling web-based interface (Windmill) that enables synthetic cohort creation. Terra allows researchers to then use these synthetic cohorts of data in their research, running user or community created analytical workflows at scale through an easy-to-use workspace environment. And, finally, Dockstore provides both a rich library of workflows and tools ready-to-run in Terra but also a platform for publishing a researcher's own tools and workflows for use by a wider community.



A. **B.** **C.**

Figure: (A) Gen3, from the University of Chicago, provides data and metadata services, authentication and authorization, and a data search and synthetic cohort creation tool. (B) Terra, from the Broad Institute, provides workspaces where researchers can import synthetic cohorts from Gen3 and perform batch and interactive analysis on them using WDL workflows and Jupyter notebooks respectively. (C) Dockstore, from UCSC and OICR, is a tool and workflow registry that can provide batch workflows for execution in Terra.

The Commons Alliance is using the Gen3, Terra, and Dockstore products as part of the NHLBI Data STAGE and NHGRI AnVIL projects. In this context, we are working with the projects to onboard cloud-based data files, for example the University of Chicago recently ingested over 1.6PB of TOPMed data into the Gen3 stack. This was complimented by loading metadata into Gen3 and including several key “facets” that are presented to the end user in the Windmill browser, allowing the researcher to search for data of interest. By onboarding cloud-based data and metadata into Gen3, datasets are accessible to the Terra environment using community standard APIs. For these projects we created Terra workspaces for Data STAGE and AnVIL researchers to use the data and metadata from Gen3. Finally, workflows from TOPMed and other projects have been onboarded and are accessible in Dockstore. This allows users, for example, to bring their own data to the workspace environments in Terra, access TOPMed workflows from Dockstore, analyze their data in Terra using these workflows, and then compare their results with the larger TOPMed datasets. This whole flow, which is typical for the sort of analysis performed by researchers in the Data STAGE and AnVIL projects, is presented through convenient, simple-to-use web interfaces. This greatly democratizes access to the cloud by making the STAGE and AnVIL platforms easy to use without the need for specialized skills. By working together, the partner organizations of the Commons Alliance are lowering the barriers to entry just at a time when many of the most comprehensive and important research datasets are transitioning to the cloud. We feel the biomedical research community will significantly benefit from these efforts and look forward to Data STAGE and AnVIL’s upcoming general availability.

Day1 15:50-15:55 (P-25)

Fake it 'til You Make It: Open Source Tool for Synthetic Data Generation to Support Reproducible Genomic Analyses

Abstract:

The lack of readily accessible large scale public genomic data sets currently limits the reproducibility of published biomedical research to a subset of authorized users. Tool developers, educators, journal editors and researchers alike are affected by the lack of open access genomic datasets appropriate for reproducing biologically meaningful analysis at scale. We will present a prototype pipeline that promotes reproducible analysis by making it easy to generate publicly shareable custom synthetic datasets. The prototype workflow links existing tools into a consolidated community resource for generating synthetic data cheaply and efficiently. We will demonstrate how to use this workflow on Broad Institute's open access Terra platform, to reproduce someone else's analysis and make your own work reproducible. The workflow, as written, is portable to any cloud platform that runs the Cromwell Engine, an open source scientific Workflow Management System.

Case Study:

We presented the first prototype of this workflow at American Society of Human Genetics(ASHG) 2018 as part of a template for reproducible research. The workflow was used to generate the data needed to reproduce work by Matthieu Miossec and collaborators described in a biorXiv preprint titled "Deleterious genetic variants in *NOTCH1* are a major contributor to the incidence of non-syndromic Tetralogy of Fallot" (ToF). In the original study, the authors analyzed high-throughput exome sequence data from 829 cases and 1252 controls, identifying 49 rare deleterious variants within the *NOTCH1* gene that appeared associated with this congenital heart disease. Other researchers had previously identified *NOTCH1* in families with congenital heart defects, including ToF; however Miossec *et al*. were the first to scale variant analysis of ToF to nearly a thousand case samples and show that *NOTCH1* is a significant contributor to ToF risk. Preprint: https://www.biorxiv.org/content/10.1101/300905v1. Final publication: https://doi.org/10.1161/CIRCRESAHA.118.313250

Overall Approach:

We used information from the preprint and its Supplemental Materials to reconstruct the main phases of the work, including Data Input, Processing and Analysis (Figure 1). For the Data Input phase, we created a synthetic dataset to get around the lack of appropriate public data at the time of publication. For the Processing Phase, we applied a variant discovery workflow that we judged equivalent to the original study. For the Analysis phase, we collaborated with Dr. Miossec to reimplement them in two parts: the prediction of variant effects as a workflow in WDL (Workflow Description Language) and the clustering analysis as R code in a Jupyter notebook. We did all the work in the Broad Institute's Terra platform - a cloud-native platform for biomedical researchers to access data, run analysis tools, and collaborate.

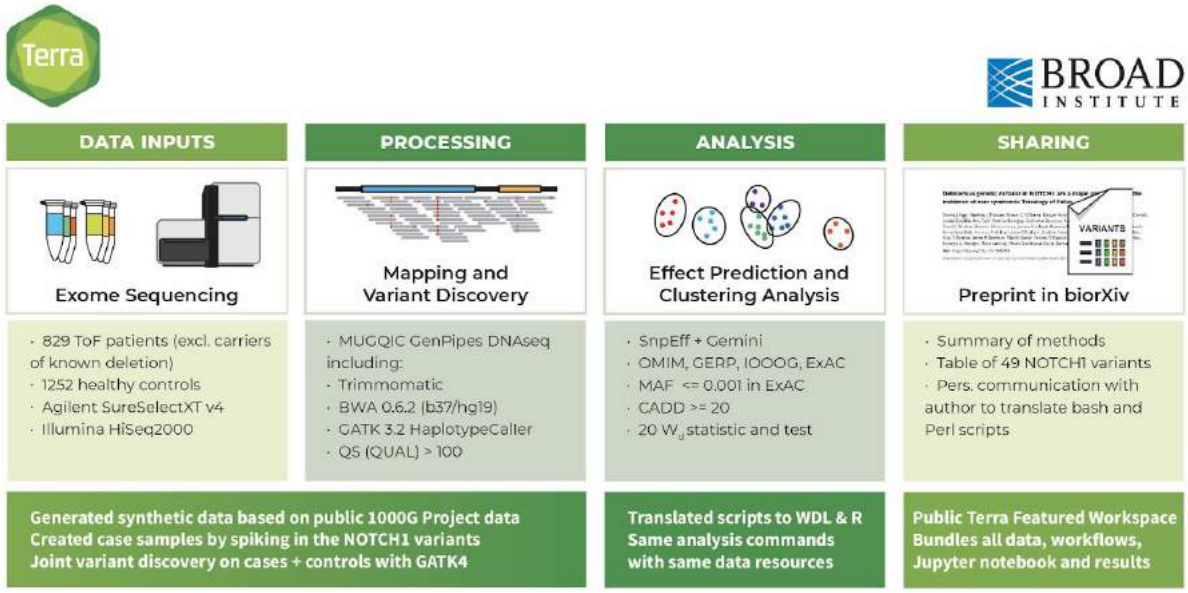


Figure 1: Case Study Reproducing a Paper on Tetralogy of Fallot using Synthetic Data

Recent Improvements from BioIT Hackathon 2019:

We were successfully able to reproduce original analysis with some compromises. In the process, we identified several areas for possible improvement in the realm of synthetic data. One was a need to be more cost efficient in order to generate synthetic data sets of sufficient size. A second was a need to identify high quality original data available as a template for generating synthetic data.

We developed an expanded second prototype with the following objectives during the 2019 Boston BioIT Hackathon:

1. **Data in demand:** Define the most common genomic data needs and set goals for developing freely available repositories of synthetic data for these scenarios.
2. **Method optimization:** Increase workflow efficiency in order to decrease costs and runtime.
3. **Quality control:** Verify that synthetic data replicates the original data set in key areas. For example, we should be able to pull the target mutation out of the synthetic data.
4. **Versatility:** Extend to variant types beyond single nucleotide variants.

Perspectives:

The open source Terra workspace that bundles all materials for reproducing this work is an important step in developing a scalable and reproducible tool for generating synthetic data (https://app.terra.bio/#workspaces/help-gatk/Reproducibility_Case_Study_Tetralogy_of_Fallot). In the future, we propose to develop a more cost effective and computationally tractable workflow. We will disseminate this tool through the Terra platform in order to promote the development of public repositories for large scale synthetic datasets for genomics that will serve as community resources,

Day2 08:40-08:45 (P-50)

Select Unique Features of Archaeopteryx.js

Available at: https://www.npmjs.com/package/archaeopteryx

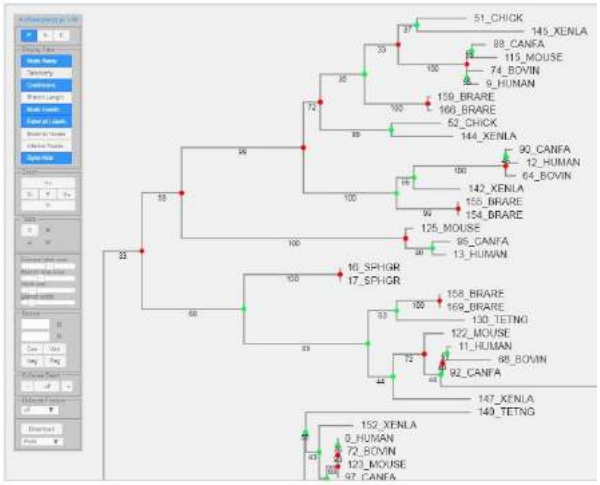


Figure 1: Archaeopteryx.js displaying gene duplications.

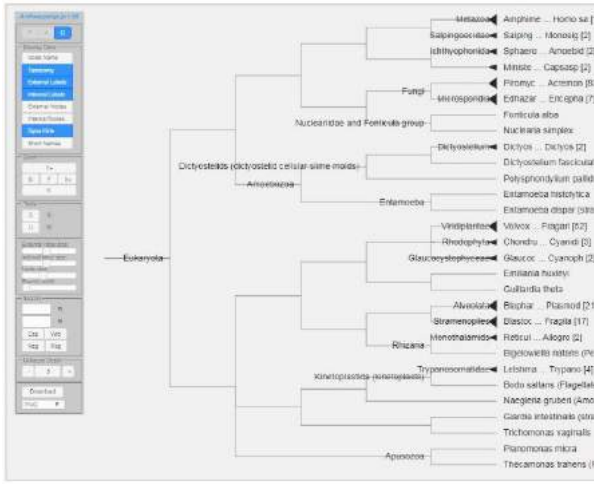


Figure 2: Demonstration of auto-collapse feature.

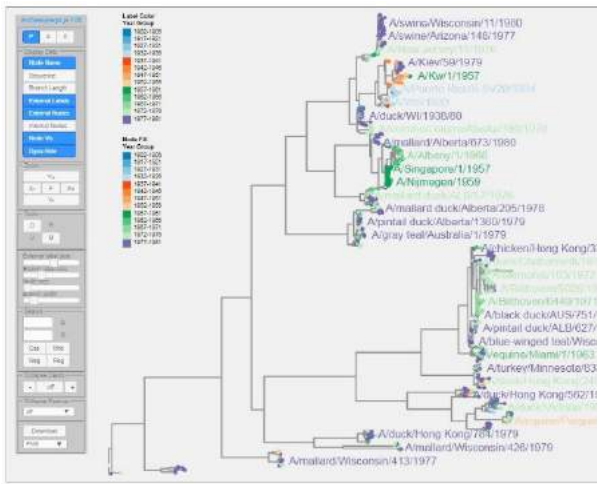


Figure 3: Example of visualization of meta-data: year as label color and node fill.

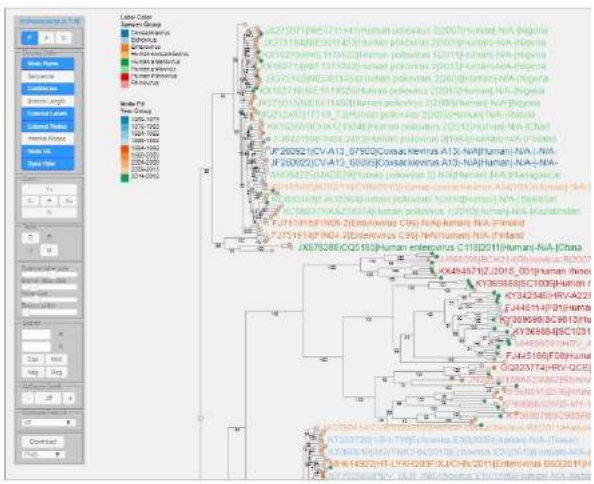


Figure 4: Example of visualization of meta-data: virus species as label color and year as node fill.

Day2 08:45-08:50 (P-47)

Sequenceserver: a modern graphical user interface for custom BLAST databases

Anurag Priyam, Yannick Wurm

Repository : https://github.com/wurmlab/sequenceserver
License : AGPL

The advances in DNA sequencing technologies have created many opportunities for novel research that require comparing newly obtained and previously known sequences. This is commonly done with BLAST, either as part of an automated pipeline, or by visually inspecting the alignments and associated meta-data. We previously reported Sequenceserver to facilitate the latter. Our software enables a user to rapidly setup a BLAST server on custom datasets and presents an interface that is modern looking and intuitive to use. However, interpretation of BLAST results can be further simplified using visualisations.
We have integrated three existing visualisations into Sequenceserver with the aim to facilitate comparative analysis of sequences. First, we provide a circos plot to rapidly check for conserved synteny, identify duplications and translocation events, or to visualise transposon activity. Second, we provide a histogram of length of all hits of a query to quickly reveal if the length of a predicted protein sequence matches that of its homologs. Finally, for each query-hit pair, the relative length and position of matching regions are shown. This is helpful to identify large insertion or deletion events between two genomic sequences, can reveal putative exon shuffling, and help confirm a priori knowledge of intron lengths.

Day2 08:50-08:55 (P-36)

Parallel, Scalable Single-cell Data Analysis

Ryan Williams, Tom White, Uri Laserson

Repository : https://github.com/lasersonlab/ndarray.scala
License : Apache 2

Single-cell sequencing generates a new kind of genomic data, promising to revolutionize understanding of the fundamental units of life. The Human Cell Atlas is a multi-year, multi-institution effort to develop and standardize methods for generating and processing this data, which poses interesting storage and compute challenges.
I'll talk about recent work parallelizing analysis of single-cell data using a variety of distributed backends (Apache Spark, Dask, Pywren, Apache Beam). I'll also discuss the Zarr format for storing and working with N-dimensional arrays, which several scientific domains have recently gravitated toward in response to challenges using HDF5 in parallel and in the cloud.

Day2 09:00-09:05 (P-40)

RAWG: RNA-Seq Analysis Workflow Generator

Alessandro Pio Greco, Patrick Hedley-Miller, Filipe Jesus, Zeyu Yang

May 15, 2019

The rapid pace of innovation in the field of sequencing has meant an explosion in the number of tools available for analysis. This creates problems when interpreting differences of downstream analyses between different RNA-Seq pipelines because there are multiple junctures at which discrepancies can occur. This issue is compounded since there are numerous parameters within each step of the pipeline that a user can manually adjust. The result is that inter-pipeline comparisons of RNA-seq analysis are difficult to interpret and users need to ensure a consistent set of parameters are used for all samples.

We developed a data analysis framework, RNA-Seq Analysis Workflow Generator (RAWG), which can act as a one stop shop for anyone wanting to preform RNA-Seq differential expression analysis. RAWG consists of a webportal, a set of server-side scripts, and a collection of command-line tool wrappers in Common Workflow Language (CWL). The webportal is the end-user's primary interface and is used to upload RNA-Seq reads and define analysis pipelines. The server-side scripts, written in Python, dynamically generate CWL workflows base on user-selected tools from the webportal and execute the workflows. Together, we achieved an user-friendly and easy to use data analysis framework which is also extensible so that developers can integrate tools into RAWG easily.

The main advantage of RAWG is that users are liberated from writing workflows manually as all the connections between tools are handled automatically. RAWG is also capable of performing multiple pipelines in one workflow, which means common steps in different pipelines are merged into a single step, hence saves computational resources. Leveraging container technology, researchers are freed from setting up complex software environments and the analysis workflow is more reproducible and portable. A demo server and the user guide is available here: https://github.com/rawgene/rawg/blob/master/doc/userguide.md

Application

To showcase the ability of using RAWG to make top quality scientific discoveries, we present two application examples. A differential expression analysis on neuroblastoma data and a comparison between RNA-Seq analysis pipelines.

Neuroblastoma Data Analysis

Previous studies have linked neuroblastoma progression and development to p53, NGF and TrkA expression. This study aims to find features which are differentially expressed solely due to NGF-independent TrkA activation via interactions with exogenous TrkA and mutant p53. Note that Fig. 1(d) is plotted by the webportal's visualisation section.

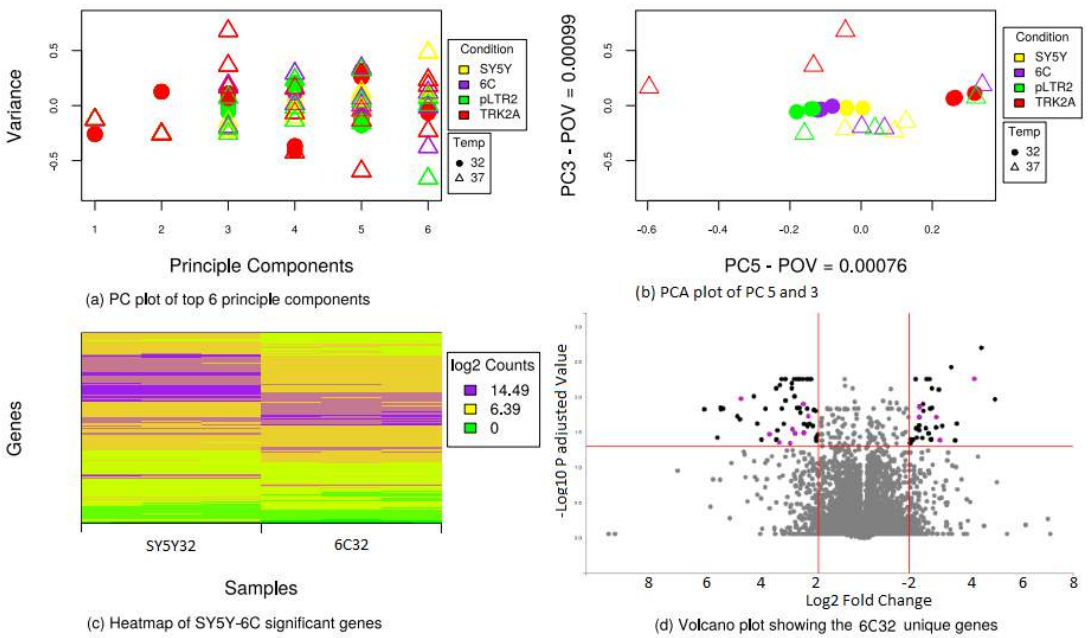


Figure 1: Data structure visualisations (a) top 6 PCs, (b) PCA plot for PC5 and PC3, (c) heatmap of significant genes (d) volcano plot of SY5Y-6C: insignificant genes are shown in grey cut-offs are shown in red. (Figures were not produced by RAWG except (d))

Comparison Between Pipelines

We simulated RNA-Seq data from *Escherichia Coli*'s genome and fed them to all the pipelines in RAWG that generate differential gene expression results. The fold changes of the 20 randomly selected transcripts are set to 4 so the log2 fold change should be as close to 2 as possible.

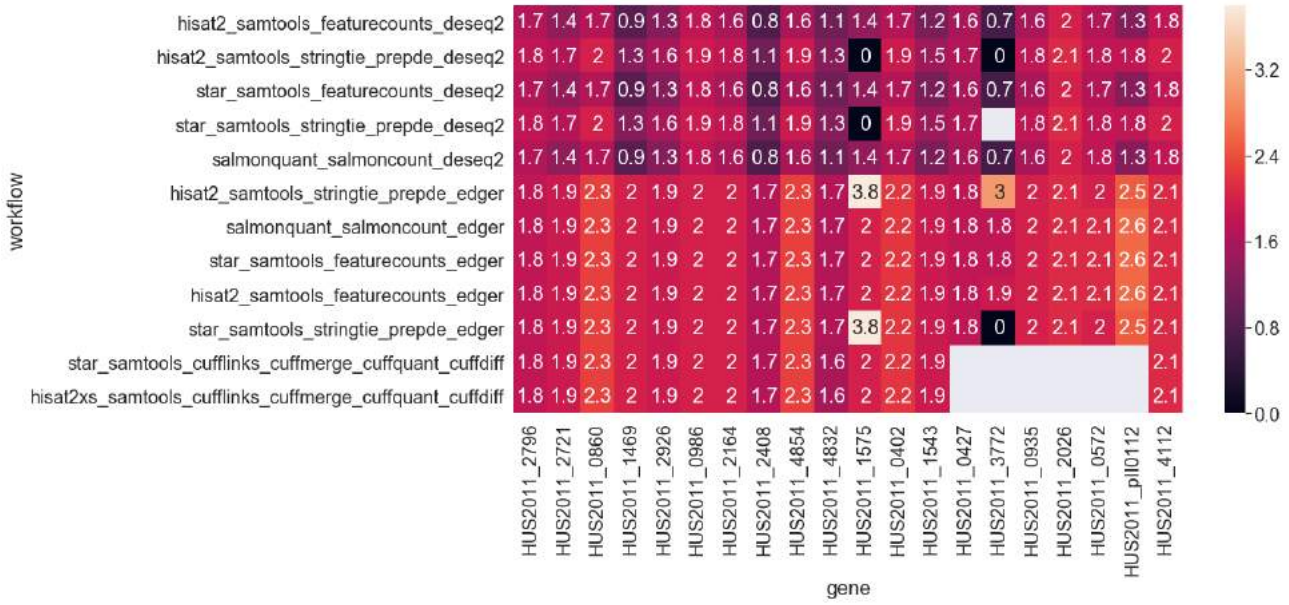


Figure 2: A heatmap of log2 fold change from DGE results with different workflows for twenty randomly chosen genes. (Figure was not produced by RAWG)

Day2 09:05-09:10 (P-44)

SAPPORO: workflow management system that supports continuous testing of workflows

Hirotaka Suetake[1], Tazro Ohta[2]

1. The University of Tokyo, Department of Creative Informatics, Graduate School of Information Science and Technology, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
2. Database Center for Life Science, Joint Support-Center for Data Science Research, Research Organization of Information and Systems, Shizuoka 411-8540, Japan

Project Website: https://suecharo.github.io/SAPPORO/
Source Code: https://github.com/suecharo/SAPPORO/
License: Apache2.0

Sharing personal genome data is critical to advance medical research. However, sharing data including personally identifiable information requires ethical reviews which usually takes time and often has limitations of computational resources that researchers can use. To allow researchers to analyze such data in controlled access efficiently, DNA Data Bank of Japan (DDBJ) developed a new workflow execution system called SAPPORO (Figure1). We designed the system to allow users to execute workflows with controlled access data without touching them. Users select a workflow on the SAPPORO's web interface to run it on a node for personal genome data analysis in the DDBJ's high-performance computing (HPC) platform. The system supports the Common Workflow Language (CWL) as its primary format to describe workflows; thus it can import the workflows developed by different institutes as long as they are described in CWL [1]. We implemented the workflow run service component by following the Workflow Execution Service API standard developed by the Cloud working group of Global Alliance for Genomics and Health (GA4GH) [2]. This highly flexible and portable system can be an essential module on data and workflow sharing in biomedical research.

1. Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., ... & Scales, M. (2016). Common Workflow Language, v1. 0.
2. GA4GH Cloud Work Stream https://github.com/ga4gh/wiki/wiki

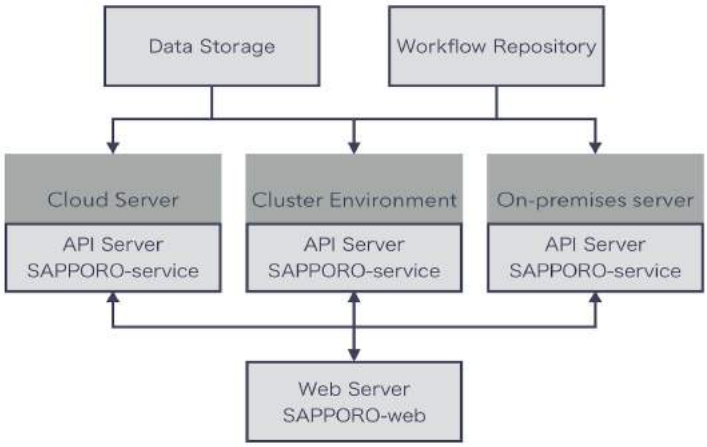


Figure 1. The system overview of SAPPORO

Response to reviewers

We would like to thank the organizers of BOSC 2019 and the reviewers for spending time to review our abstract. We agree with the reviewers that the project is still immature and has to be improved especially in usability. We updated README and released stable version(v0.3.5). We plan to release more new versions this month. Although it still has some issues to be solved, we would like to have a chance to give a lightning talk on BOSC to introduce our idea and ask the community to participate the project. Responses to each reviewer are below.

Reviewer 1:

> SAPPORO is a workflow management system that provides a platform to run CWL-defined workflows, using the GA4GH Workflow Execution Service standard. It is based on three components, a simple web-based interface, an execution server, and a data storage server. One of the important points of SAPPORO is the modularity, for instance through the use of the GA4GH API between the web server and the execution server. It would be interesting to know if this modularity allows for instance to run workflows from the web server on other GA4GH execution systems.

All queries from the web-based interface to the execution server use only the GA4GH API. Thus, this allows you to run workflows on the GA4GH execution system. However, there are not many GA4GH execution systems at present.

Reviewer 2:

> SAPPORO is an open source workflow management system for researchers to analyze raw data without actually having access to the data, I think. The abstract lacks the details I'd need for a more thorough review, so I am giving it a borderline score. In the future, I think the authors should make the motivation more clear, and explain in more detail the method (e.g. what does "controlled access data without touching them" mean?).

Access to raw data such as fastq by researchers requires an ethical review. Conversely, a researcher can skip an ethical review by executing a workflow and receiving only the result files without seeing the contents of raw data. DBCLS provides several workflows that can be used with SAPPORO. These workflows process raw data until there is no personally identifiable information. SAPPORO does not internally manage the contents of input and output data. Therefore, researchers can’t access the raw data.

Reviewer 3:

> SAPPORO is an Apache 2.0 licensed software package designed to run batch workflow jobs. I liked the CI/CD element mentioned in the README, which will help to ensure the workflows don't grow stale and break e.g. perhaps due to deprecated or channging dependencies.
>
> I tried to run each of the three sub-packages. Results are as follows:
>
> SAPPORO-web: Good install instructions, although I had to run them with sudo to make them work. I could load the server and log on to it. All the tabs were effectively blank apart from headers. Perhaps this would be different if I had the other two components running?

SAPPORO-service: Instructions were clear, and again had to run as sudo to prevent errors. In this case the server never seemed to start up and just printed "manifest for suecharo/sapporo-service:v0.3.5 not found" followed by "Starting uWSGI process..." to the terminal indefinitely.
SAPPORO-fileserver: This package had a dependency on a package called Minio. I downloaded and installed Minio, but wasn't sure where I needed to get the secret key and access key from mentioned in the readme for SAPPORO-fileserver, so was unable to proceed.

Finally, while docker is great for bootstrapping the application quickly, I wondered if this is the method one would use when developing the software? If not, it would be nice to add development build instructions as well as the docker running instructions.

Overall, the package sounds very interesting, but needs slightly better running instructions in order to be useful to others. I would love to re-assess this package if they are updated! Given the previous comments and the number of contributors to this package, I would recommend a poster.

We have updated the README and released a stable version. Thank you for your feedback!
There are two reasons to run SAPPORO in a docker environment. First, it is intended for use in a disposable environment such as cloud environment. The compatibility between cloud environments and container technology is very high. Second, it is intended for use in environments where software cannot be easily installed, such as on supercomputers. In fact, we use SAPPORO by using singularity(this does not require sudo privileges) on our supercomputer environment.

Day2 09:10-09:15 (P-45)

With increasing challenges in understanding very large and complex cancer genomic data, this abstract presents robust, scalable R/Bioconductor software data representations and statistical methods to help tackle significant problems in cancer biology. Bioconductor is a widely used, highly respected open-source environment for statistical analysis and comprehension of high-throughput genomic data, so these developments are useful to many researchers.

With large archives of publicly available genomic data implementing FAIR (findable, accessible, interoperable, reusable) principles, urgent demands are present for computational and bioinformatics tools to efficiently translate the data into clinical important insights. In order to reduce memory usage and optimize performance, Bioconductor has developed different data structures and interfaces to lazily represent big data sets either in “array” format (e.g., count data from single cell RNA sequencing (scRNA-seq) experiment), or in “data.frame” format (e.g., the feature or sample annotation information with clinical characteristics and relevance). Lightweight, lazy containers provide easy data manipulation within familiar R/Bioconductor paradigms, and support scalability and interoperability with existing bioinformatics tools available in R/Bioconductor.

The DelayedArray has been developed to represent very big genomic data sets, such as count data from scRNA-seq and variant data from high-throughput DNA-seq experiments. DelayedArray allows users to perform common array operations on it without loading the object in memory. Operations on the DelayedArray objects are either recorded but delayed, or executed using a block processing mechanism. Bioconductor has made DelayedArray easily extendable to different 'back end' representations of data, such as Hierarchical Data Format (HDF) (available in Bioconductor as HDF5Array), the Genomic Data Structure (GDS) (available in Bioconductor as GDSArray) and the Variant Call Format (VCF) (available in Bioconductor as VCFArray). The scalability offered by DelayedArray has enabled computational biologists and bioinformaticians to take advantage of rich programming semantics and diverse big data solutions. DelayedArray builds on familiar R / Bioconductor programming paradigms, and requires little effort on the part of the bioinformatician to learn new technologies.

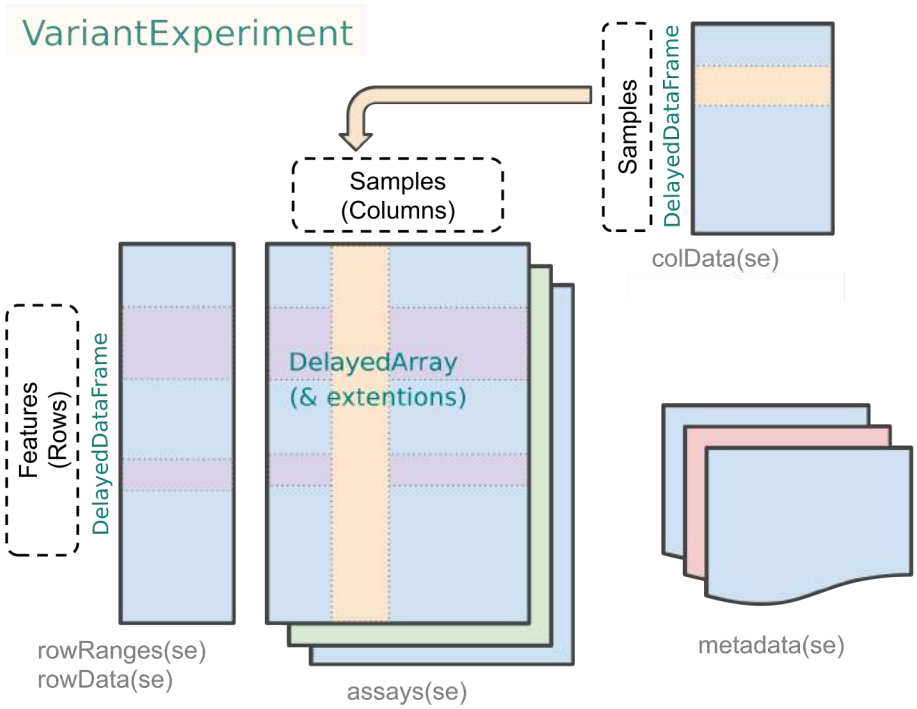
In addition to the lazy representation of assay data obtained from biological experiments, Bioconductor has developed a data structure called DelayedDataFrame to represent the metadata for features (e.g., gene symbols, tests of gene-wise statistical significance) or samples (e.g., clinical characteristics) with a DataFrame-like metaphor. DelayedDataFrame accommodates DelayedArray (and direct extension) objects in the columns. Operations on DelayedDataFrame are recorded but delayed until a specific realization call is invoked. Lazy sample and feature metadata can be combined with lazy assay data to be analyzed in R with common and familiar methods to bioinformaticians. These operations do not require loading the whole data into memory.

SQLDataFrame is another Bioconductor package that has been developed to lazily represent and efficiently analyze SQL-based tables in R. SQLDataFrame supports common and familiar 'DataFrame' operations such as “[” subsetting, rbind, cbind, etc.. The internal implementation is based on the widely adopted dplyr grammar and SQL commands. This provides advanced

users with the flexibility to directly use certain dplyr functions and raw SQL queries on SQLDataFrame objects.

We have developed VariantExperiment, an extension of the SummarizedExperiment data structure, to take advantage of the rich semantics of, and interoperate with, R / Bioconductor packages that are widely used for scRNA-seq and other analyses. VariantExperiment supports the DelayedArray (and extensions, such as HDF5Array, GDSArray, and VCFArray, etc.) objects as 'assay' data, and DelayedDataFrame for row (feature) and column (sample) annotations. With the on-disk representation of both assay data and annotation data, VariantExperiment has become a lightweight container for very large genomic data resources represented as a complete experiment. VariantExperiment uses significantly less memory than in-memory R alternatives. VariantExperiment implements the 'SummarizedExperiment' interface, enabling easy and common manipulations for high-throughput genetic/genomic data with familiar R / Bioconductor paradigms.

These Bioconductor data structures have implemented advanced approaches to data representation and computation on large-scale genomic datasets, emphasizing analysis and comprehension of domain-specific data types. By implementing scalable computational strategies to enhance performance and expressivity, these data structures considerably improve the acquisition, management, analysis and dissemination of big genomic datasets in R / Bioconductor. This benefits the broad community of bioinformatic software developers as well as the domain-specific cancer researchers.



The Monarch Initiative: Closing the knowledge gap with semantics-based tools

Monica Munoz-Torres[1], Melissa Haendel[1,2], Chris Mungall[3], Peter N. Robinson[4], David Osumi-Sutherland[5], Damian Smedley[6], Julius Jacobsen[7], Sebastian Köhler[8], Julie McMurry[1,2], and the members of The Monarch Initiative.

[1]Oregon State University, Corvallis, OR, USA. [2]Oregon Health & Science University, Portland, OR, USA. [3]Lawrence Berkeley National Laboratory, Berkeley, CA, USA. [4]The Jackson Laboratory for Genomic Medicine, Farmington, CT, USA. [5]European Bioinformatics Institute, Hinxton, UK. [6]Genomics England, Cambridge, UK. [7]Queen Mary University of London, London, UK. [8]Charité Universitätsmedizin, Berlin, Germany.

The Monarch Initiative is a consortium that seeks to bridge the space between basic and applied research, developing tools that facilitate connecting data across these fields using semantics-based analysis. The mission of the Monarch Initiative is to create methods and tools that allow exploration of the relationships between genotype, environment, and phenotype across the tree of life, deeply leveraging semantic relationships between biological concepts using ontologies. These tools include Exomiser, which evaluates variants based on the predicted pathogenicity, amongst many others. The goal is to enable complex queries over diverse data and reveal the unknown. With the semantic tools available at www.monarchinitiative.org, researchers, clinicians, and the general public can gather, collate, and unify disease information across human, model organisms, non-model organisms, and veterinary species into a single platform. Monarch defines phenotypic profiles, or sets of phenotypic terms, which are associated with a disease or genotype recorded using a suite of phenotype vocabularies (such as the Human Phenotype Ontology and the Mondo Ontology). Our niche is computational reasoning to enable phenotype comparison both within and across species. Such explorations aim to improve mechanistic discovery and disease diagnosis. We deeply integrate biological information using semantics, leveraging phenotypes to bridge the knowledge gap.

The Monarch Initiative GitHub Organization
https://github.com/monarch-initiative

Human Phenotype Ontology
Website: https://hpo.jax.org
License: https://hpo.jax.org/app/license

Exomiser
Website: https://github.com/exomiser/Exomiser
License: https://hpo.jax.org/app/license

Mondo
https://github.com/monarch-initiative/mondo
License: CC-BY

DAISY: a tool for the accountability of Biomedical Research Data under the GDPR

Regina Becker, Pinar Alper, Valentin Grouès, Sandrine Munoz, Yohan Jarosz, Jacek Lebioda, Kavita Rege, Christophe Trefois, Venkata Pardhasaradhi Satagopam, Reinhard, Schneider

Repository : https://github.com/elixir-luxembourg/daisy

GDPR requires the documentation of any processing of personal data, including data used for research and to be prepared for information provision to the data subjects. For institutions this requires a data mapping exercise to be performed and to keep meticulously track of all data processings. While there is no formal guidance on how data mapping should be done, we're seeing the emergence of some commercial "GDPR data mapping" tools and academic institutions creating registers with those tools. When it comes to mapping data in biomedical research, we observe that commercial tools may fall short as they do not capture the complex project-based, collaborative nature of research that leads to many different scenarios.

In this poster we describe a Data Information System (DAISY), our data mapping tool, which is specifically tailored for biomedical research institutions and meets the record keeping and accountability obligations of the GDPR. DAISY is open-source and is actively being used at the Luxembourg Centre for Systems Biomedicine and the ELIXIR-Luxembourg data hub.

Day2 10:20-10:40 (P-28)

Dockstore: Enhancing a community platform for sharing cloud-agnostic research tools

Denis Yuen[1], Louise Cabansay[2], Charles Overbeck[2], Andrew Duncan[1], Gary Luu[1], Walt Shands[2], Natalie Perez[2], David Steinberg[2], Cricket Sloan[2], Brian O'Connor[2], Lincoln Stein[1]

[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

Project Website : https://dockstore.org/

Source Code : https://github.com/ga4gh/dockstore/

Background

Dockstore was created in response to the many challenges faced during the PCAWG (Pan-Cancer Analysis of Whole Genomes) study—an international collaboration to identify common patterns of mutation in more than 2,800 cancer whole genomes from the International Cancer Genome Consortium. The study involved fourteen highly heterogeneous computing environments that were not only geographically distributed, but spanned different cloud and HPC machines that encompassed both academic and commercial varieties. The scale and complexity of modern biomedical science efforts like these have driven a rethink of bioinformatics infrastructure to leverage big data, containerization, and cloud technologies that increase the mobility, interoperability, and reproducibility of research. To address these goals, we continued extending and developing Dockstore for use by the wider research community. This report highlights the key updates to the platform since its 1.2.5 release last presented at BOSC in 2017, as well as the future work going forward.

Platform

Dockstore is a platform for sharing Docker-based resources that allow bioinformaticians to bring together tools and workflows into a centralized location.

By packaging software into portable containers and utilizing popular descriptor languages, Dockstore standardizes computational analysis, making workflows precisely reproducible and runnable in any environment that supports Docker. Supported descriptors now include Nextflow in addition to the Common Workflow Language (CWL) and Workflow Description Language (WDL). These descriptor documents and test parameter files can now be stored directly on Dockstore.org or through external registries like GitHub, Bitbucket, Quay.io, and Docker Hub.

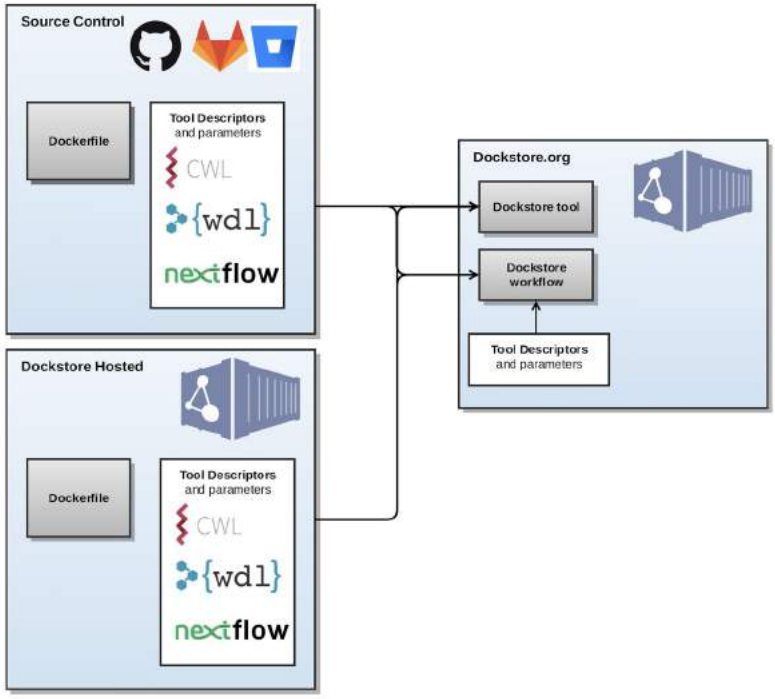


Figure: Overview of hosting for the Dockstore platform.

Integrations

We have combined hosting features with integrated deployment to cloud platforms and analysis environments. By registering workflows on Dockstore, developers can now provide the ability for users to run their workflows directly through a variety of launch-with partners like FireCloud/Terra, DNAnexus, and DNAstack. They can also share and launch workflows programmatically on compatible GA4GH WES-compatible platforms. When developing workflows locally, a handy Dockstore CLI also includes implementations for file-provisioning that support a variety of protocols including HTTP(S), S3, GS, FTP, ICGC Score, and a plugin that resolves the location of files using the GA4GH Data Registry Service (DRS) standard.

Usability

For end users, the Dockstore community has expanded the searchable catalogue of available tools and workflows. To further enhance this experience, we have added improvements such as the ability to link to specific versions of tools and workflows, expanded search and sorting options such as by descriptor language or author, and integrated community-provided DAG visualizations for WDL and CWL. Support for descriptor metadata rendering into markdown now allows tool and workflow info pages to display enhanced documentation.

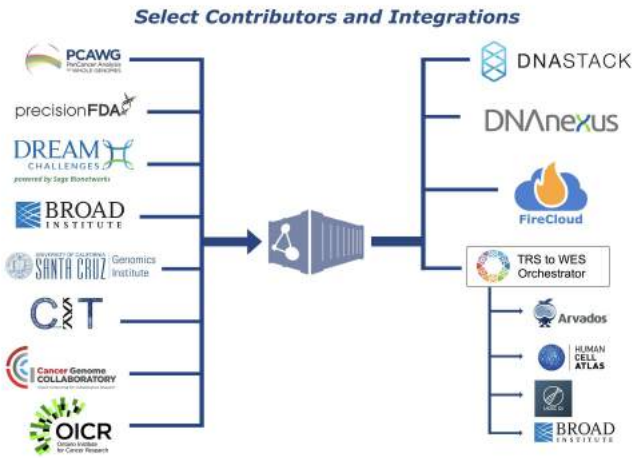
Users can now also aid one another with social features such as starring, labels, and discussion threads. Introduced collaboration features allow for permissions based sharing and enable groups to create organization pages that describe and highlight collections of workflows and tools.

Community

Dockstore is thankful to its many contributors, users, and partners. This community has pulled together a library of over 450 tools and workflows. In the diagram below we've highlighted a few select contributors to give a sense of what has been occuring in this space.

Our community and our team has also verified 28 high quality tools and workflows known to have run (reproduced) by someone outside the original team. In addition, 17 versions of tools and workflows are regularly tested by our team to maintain compatibility with Dockstore updates.

Finally, 8 workflows on 4 platforms (for a total of 32 combinations) are run via the GA4GH workflow testbed, a collaborative effort to run workflows pulled from Dockstore via the TRS API and submitted to environments that implement the WES platform.



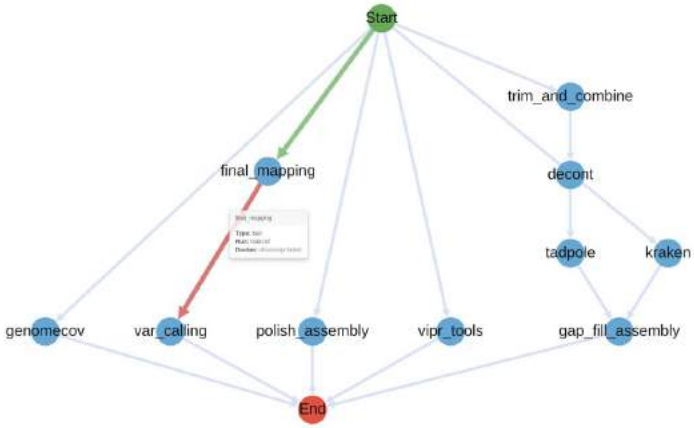


Figure: Workflow visualization with Docker image for step highlighted.

Future Work

Soon we will add support for long-lived services such as genome browsers, notebooks, and reference data providers by packaging these into containers and launching instances through Docker Compose. To improve security and integrity of container images, we plan to implement signing of Dockstore packages. Dockstore also plans to further expand integrations to include more launch-with partners, containerization technologies, and workflow languages.

References

O'Connor BD, Yuen D, Chung V *et al.* The Dockstore: enabling modular, community-focused sharing of Docker-based genomics tools and workflows [version 1; referees: 2 approved]. *F1000Research* 2017, **6**:52 (doi: 10.12688/f1000research.10137.1)

Denis Yuen, Andrew Duncan, Victor Liu, Brian O'Connor, Gary Luu, Charles Overbeck, … Abraham. (2019, April 5). ga4gh/dockstore: 1.6.0 (Version 1.6.0). Zenodo. http://doi.org/10.5281/zenodo.2630727

Gary Luu, Andrew Duncan, Denis Yuen, Kitty Cao, JWKaiqi, Charles Overbeck, … angular-cli. (2019, April 3). dockstore/dockstore-ui2: 2.3.0-rc.4 (Version 2.3.0-rc.4). Zenodo. http://doi.org/10.5281/zenodo.2626656

Amstutz, Peter; Crusoe, Michael R.; Tijanić, Nebojša; Chapman, Brad; Chilton, John; Heuer, Michael; Kartashov, Andrey; Leehr, Dan; Ménager, Hervé; Nedeljkovich, Maya; Scales, Matt; Soiland-Reyes, Stian; Stojanovic, Luka (2016): Common Workflow Language, v1.0. figshare. https://doi.org/10.6084/m9.figshare.3115156.v2

Voss K, Gentry J and Van der Auwera G. Full-stack genomics pipelining with GATK4 + WDL + Cromwell [version 1; not peer reviewed]. *F1000Research* 2017, **6**(ISCB Comm J):1379 (poster) (https://doi.org/10.7490/f1000research.1114631.1)

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., & Notredame, C. (2017). *Nextflow enables reproducible computational workflows.* Nature Biotechnology, 35(4), 316–319. doi:10.1038/nbt.3820

Day2 10:40-11:00

Bioconductor with Containers: Past, Present, and Future

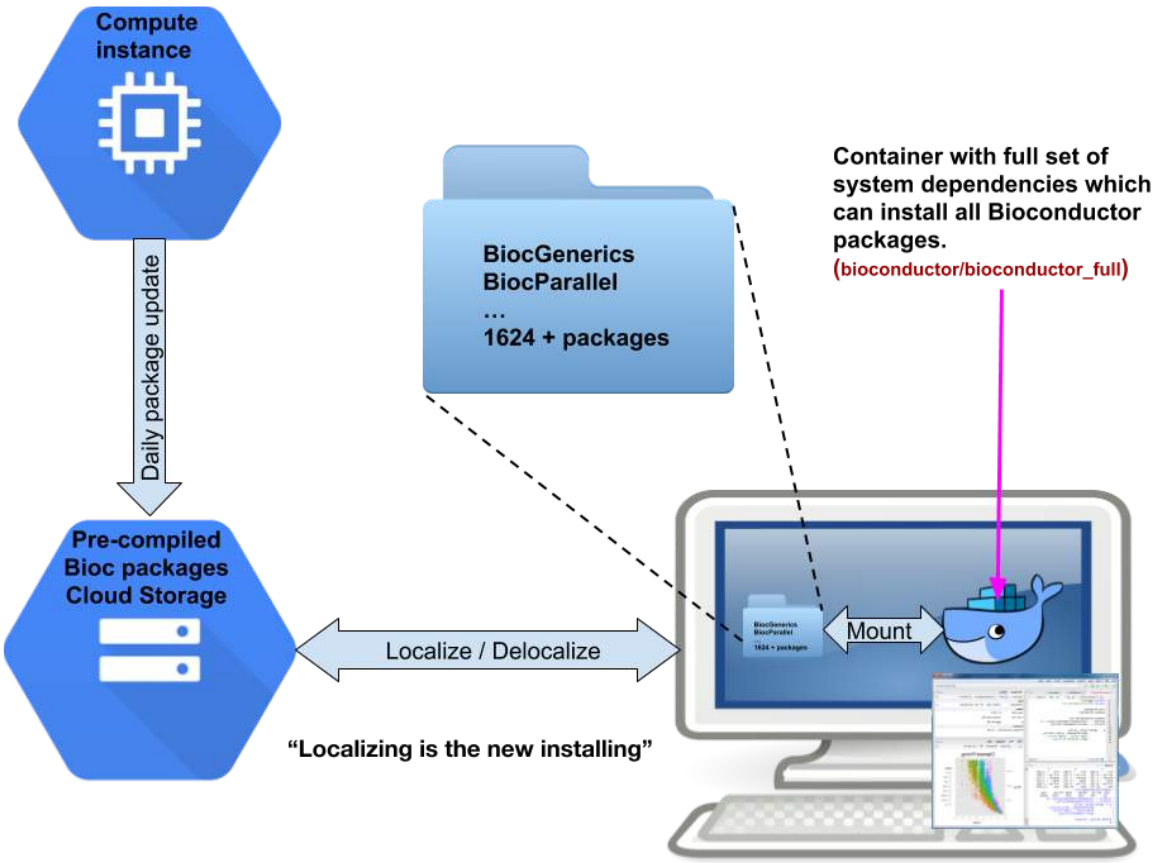
Bioconductor provides a range of options for containers to allow users and developers to perform their analysis. These containers take away the burden of installing dependencies and system requirements, allowing software to be used in a reproducible, reliable and isolated fashion. Bioconductor now produces containers which replicate the nightly linux build machines used by the project, allowing developers to test their changes more aggressively before pushing to production. We have seen that containers are now a common way to publish the "environment" used to perform data analysis.

Previously, containers were delivered to be used on a machine, and these came with a certain number of pre-installed Bioconductor packages and their dependencies delivering a “flavor” i.e, representing the type of analysis. While this option is still available, Bioconductor now provides more robust containers. The newer “bioconductor_full” containers contain system dependencies that allow installation of almost all of the 1600+ packages. R/Bioconductor users install arbitrary packages in a familiar way (install packages on the fly as needed at that moment) either on their local machine or in the cloud. The containers include RStudio and Jupyter notebook as front-end for interactive computing.

An additional development is the “package installation” method for these containers. Instead of distributing very large containers with pre-installed Bioconductor/R packages, the user mounts volumes where pre-compiled packages can be localized from cloud-based repositories. These pre-compiled packages can be localized at much faster speeds as compared to installing packages from scratch, which we think will mean “*Localizing is the new installing*”. Localization ensures that the user gets the latest versions of packages for the current release or development version of Bioconductor. This approach is particularly advantageous when used in cloud environments, where localization rates are much faster. We are developing pre-compiled package repositories from AWS and GCS.

While much of our effort has focused on docker-based containers, it is also possible to use Singularity-based Bioconductor containers. This is particularly helpful when deployed on high performance clusters within institutions, and when used in conjunction with BiocParallel and cluster management and job scheduling software. Users can pull these Bioconductor images and install packages as needed, without the help of a system administrator.

This talk will focus on why users should start thinking about containers for their Bioconductor based analysis needs, and the different flavors of containers being provided by Bioconductor. It will highlight the newer paradigms on how to use these Bioconductor containers which allow cloud and HPC usage. The goal of this work is to make your software development and data analysis reproducible and "machine-independent".


Compute instance
Daily package update
Pre-compiled Bioc packages Cloud Storage
BiocGenerics
BiocParallel
...
1624 + packages
Container with full set of system dependencies which can install all Bioconductor packages.
(bioconductor/bioconductor_full)
Localize / Delocalize
Mount
"Localizing is the new installing"

Day2 11:15-11:35 (P-54)

OpenEBench. The ELIXIR platform for benchmarking.

Benchmarking is intrinsically referred to in many aspects of everyday life from assessing the quality of stock market predictions to weather forecasting to predictions in the life sciences, such as 3D protein structure predictions or functional annotations. On an abstract level, benchmarking is comparing the performance of software under controlled conditions. Benchmarking encompasses the technical performance of individual tools, servers and workflows, including software quality metrics, as well as their scientific performance in predefined challenges. Scientific communities are responsible for defining reference datasets and metrics, reflecting those scientific challenges (Capella-Gutierrez et al. bioRxiv, 2017). In the context of ELIXIR, we have developed the OpenEBench platform aiming at transparent performance comparisons across life sciences. OpenEBench supports scientific communities by assisting in setting up emerging benchmarking efforts, foster exchange between communities and ultimately aims at making benchmarking not only more transparent, but also more efficient.

We will present the current OpenEBench and a preview on the upcoming implementations, which will be strongly focused on assisting communities to join the platform. OpenEBench is composed of two major sections. On one side, the OpenEBench tools monitoring section aims to provide a comprehensive observatory of bioinformatics software in terms of quality, FAIRness, and performance. At present OpenEBench collects data from more than 15,000 bioinformatics tools. The main target users would be researchers seeking to choose the most appropriate tool for a given analysis, considering availability, hardware requirements, software quality including documentation and/or deployment options, and comparative performance. We collect data from the ELIXIR Tools and Services registry, bio.tools; BioConda, Galaxy, BioContainers, software repositories e.g. Github, and perform text mining analysis on web sites and documentation to collect assessment items. In addition, several widgets as sites availability (including response time), and bibliographic citation history for each tool are provided.

On the other side, OpenEBench is dedicated to scientific benchmarking. The aim is to provide an infrastructure to support community-led scientific benchmarking initiatives at different levels of maturity. Target users are i) researchers seeking to choose the most appropriate tool for a specific scientific case; ii) developers aiming to test new software in the context of accepted benchmarking efforts; iii) communities aiming either to disseminate their existing results, and/or needing a technical infrastructure to perform the challenges, and iv) funders aiming to gather a collective view of a specific field. At OpenEBench we are working in three levels of operation (see Figure 1): level 1 (available) aims to collect and distribute data from established benchmarking communities; level 2 (beta state) is based on providing a technical infrastructure for computing benchmarking metrics, and to design benchmarking challenges; level 3 will extend the existing OpenEBench platform to execute benchmarkable workflows (provided as software containers) using controlled conditions to ensure an unbiased technical and scientific assessment. Level 1 complements the activities already done by benchmarking communities e.g. Quest for Orthologs (Altenhoff et al, Nat Meth 2016, CAMEO (Haas et al. Database 2013), TCGA (Bailey et al. Cell 2018); providing alternative views of benchmarking results adequate for the non-experts, and connected to the tools monitoring section. At level 2 OpenEBench provides a Virtual Research Environment (Codó et al. bioRxiv 2019) (https://openebench.bsc.es/submission) where two types of users are defined. On one hand, community managers may use the workspace to design new challenges, distribute reference datasets, and test and execute metrics on participant's providers data. On the other hand, developers can access the workspace to test their own tools using the available

benchmarking data and the community accepted metrics to assess the quality of their tools. Level 3 (expected 2020), will also provide the necessary infrastructure to benchmarking participants to upload their tools, packaged in software containers. Hence, the whole benchmarking event would be possible within the infrastructure. The availability of execution data in a controlled environment would add the possibility to monitor the software performance and computational requirements, to complete the technical monitoring of the tools. Moreover, it will also contribute to identify suitable public and private cloud instances where a given workflow can be executed to solve end-users specific scientific cases. Importantly, the three-level architecture allows different and independent levels of implementation. This allows to capture users feedback and incorporate cross-platform functionality as the platform evolves over time.

Overall, OpenEBench provides an integrated platform to orchestrate benchmarking activities, from the deposition of reference data to test software tools, to the provision of results employing metrics defined by scientific communities, and the overall accepted standards for software quality assessment.

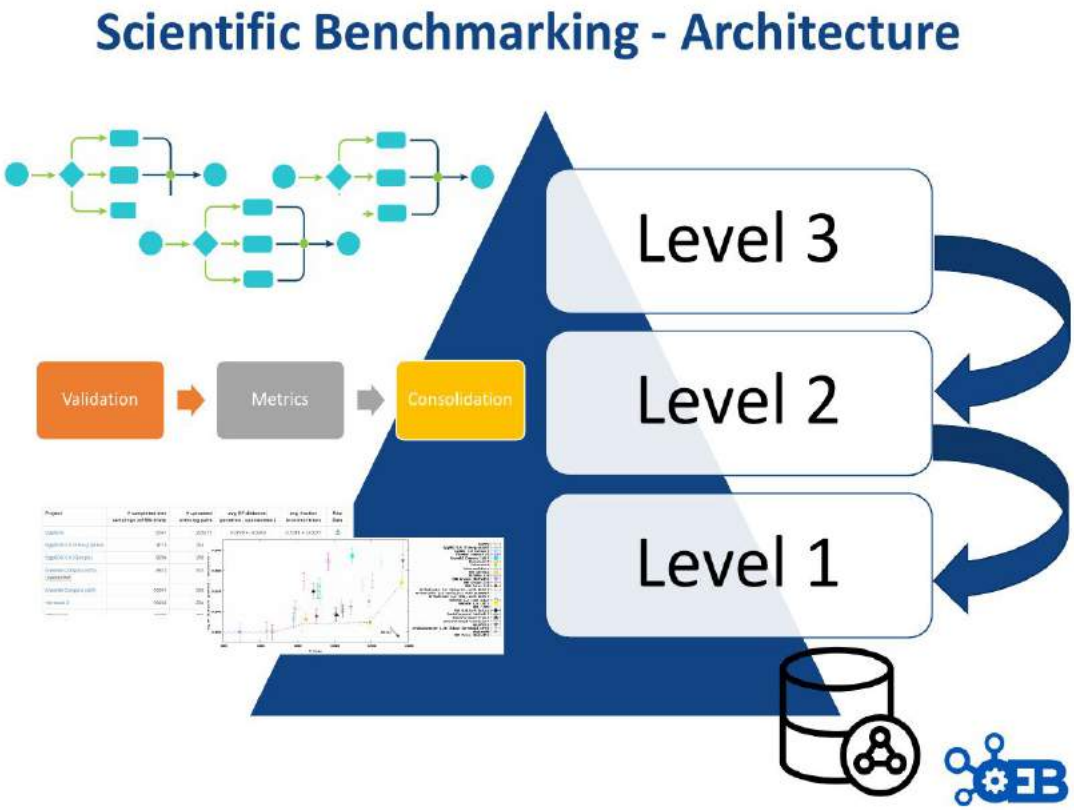


Figure 3. OpenEBench three-level architecture where each level depends on the previous one. First level is designed to host already generated benchmarked data by mature communities that want to preserve their results and reach a broader audience including non-expert users. Second level allows communities to design and implement metrics workflows that can be used to measure the level of agreement of participants for a given set of reference data. Second level data is then transformed into first level data for its archive and display in the OpenEBench website. Third level will allow participants to send their containerized workflows to be executed in a controlled environment. This will contribute to have a fairer technical comparison among participants. Workflows data will be then used as input for the metrics workflows at level two.

ELIXIR Europe on the Road to Sustainable Research Software

Mateusz Kuzak
Dutch Techcentre for Life Sciences
ELIXIR Netherlands
the Netherlands
mateusz.kuzak@dtls.nl

Jen Harrow
ELIXIR Hub Wellcome Genome Campus
Hinxton, UK

Rafael C. Jimenez
ELIXIR Hub Wellcome Genome Campus
Hinxton, UK

Paula Andrea Martinez
ELIXIR Belgium

Fotis E. Psomopoulos
Institute of Applied Biosciences, Centre for Research and Technology Hellas
Thessaloniki, Greece

Allegra Via
ELIXIR Italy, National Research Council of Italy (CNR)
Institute of Molecular Biology and Pathology (IBPM)
Italy
allegra.via@uniroma1.it

***Index Terms*—training, Open Source, software guidelines, best practices, recommendations, Open Science, Reproducible Research, sustainability, FAIR**

I. Introduction

ELIXIR [1] is an intergovernmental organization that brings together life science resources across Europe. These resources include databases, software tools, training materials, cloud storage, and supercomputers. One of the goals of ELIXIR is to coordinate these resources so that they form a single infrastructure. This infrastructure makes it easier for scientists to find and share data, exchange expertise, and agree on best practices. ELIXIR's activities are divided into the following five areas Data, Tools, Interoperability, Compute and Training known as "platforms". The ELIXIR Tools Platform works to improve the discovery, quality and sustainability of software resources. Software Best Practices task of the Tools Platform aims to raise the quality and sustainability of research software by producing, adopting, promoting and measuring information standards and best practices applied to the software development life cycle. We have published four (4OSS) simple recommendations to encourage best practices in research software [2] and the Top 10 metrics for life science software good practices [3].

II. Four Simple recommendations

The 4OSS simple recommendations are as follows:

1) Develop publicly accessible open source code from day one. Start a project as open source from the very first day, in a publicly accessible, version controlled repository (e.g. github.com, gitlab.com and bitbucket.org). The longer a project is run in a closed manner, the harder it is to open source it later.
2) Make software easy to discover by providing software metadata via a popular community registry. Facilitate the discoverability of the open source software projects by registering metadata related to the software in a popular community registry (e.g. bio.tools [4]), making your source code more discoverable. Metadata might include information such as source code location, contributors, license, references and how to cite the software.
3) Adopt a license and comply with the licence of third-party dependencies. Provide instructions and guidelines for other projects and software to use, modify and redistribute the software and the source code. Adopt a suitable Open Source license, include it in a publicly accessible source code repository, and ensure the software complies with the licenses of all third party dependencies.
4) Have a clear and transparent contribution, governance and communication processes. Open sourcing your software does not mean the software has to be developed in a publicly collaborative manner. Although it is desirable, the OSS recommendations do not mandate a strategy for collaborating with the community. However projects should be clear and transparent about how to contribute to them as well as, their governance model, and their communication channels.

III. Building Sustainabilty

IIn order to encourage researchers and developers to adopt the 4OSS recommendations and build FAIR (Findable, Accessible, Interoperable and Reusable) software, best practices

group in partnership with the ELIXIR Training platform, The Carpentries [5], [6], and other communities is creating a collection of training materials [7].The next step is to adopt, promote, and recognise these information standards and best practices, by developing comprehensive guidelines for software curation, and through workshops for training researchers and developers towards the adoption of software best practices and improvement of the usability of Tools Platform products. Additionally, a direct outcome of this task will be a software management plan template, connected to a concise description of the guidelines for open research software. Produce a white paper for the software development management plan for ELIXIR which can be consequently used to produce training material. We will work with the newly formed ReSA (Research Software Alliance) to facilitate the adoption of this plan to the broader community.

REFERENCES

[1] "ELIXIR | A distributed infrastructure for life-science information" Internet: www.elixir-europe.org, [Apr. 9, 2019]

[2] Jiménez RC, Kuzak M, Alhamdoosh M et al. (2017) "Four simple recommendations to encourage best practices in research software" *F1000Research* [Online]. 6:876. Available: http://dx.doi.org/10.12688/f1000research.11407.1 [Apr. 9, 2019]

[3] Haydee A, Chue Hong N, Corpaz M et. al. "Top 10 metrics for life science software good practices" *F1000Research*[online]. Available: https://doi.org/10.12688/f1000research.9206.1 [Apr. 9, 2019]

[4] "bio.tools Bioinformatics Tools and Services Discovery Portal" Internet: bio.tools, [Apr. 9, 2019]

[5] "carpentries.org" Internet: carpentries.org, Apr. 9, 2019 [Apr. 9, 2019]

[6] "ELIXIR teams up with The Carpentries to boost its training programme | ELIXIR", Internet: https://www.elixir-europe.org/news/elixir-carpentries-agreement, Aug. 17, 2018 [Apr. 9, 2019]

[7] "SoftDev4Research 4OSS-lesson" Internet: https://doi.org/10.5281/zenodo.2565040 Feb. 14, 2019 [Apr. 9, 2019]

The Kipoy repository: accelerating the community exchange and reuse of predictive models for genomics

Machine learning models trained on large-scale genomics datasets hold the promise to be major drivers for genome science. However, lack of standards and limited centralized access to trained models have hampered their practical impact. To address this, we present Kipoi, an initiative to define standards and to foster reuse of trained models in genomics[1]. The Kipoi repository currently hosts over 2,000 trained models from 21 model groups that cover canonical prediction tasks in transcriptional and post-transcriptional gene regulation (Fig. 1). The Kipoi model standard enables automated software installation and provides unified interfaces to apply models and interpret their outputs. Use cases include model benchmarking, variant effect prediction (Fig. 2), transfer learning and building new models from existing ones (Fig. 3). By providing a unified framework to archive, share, access, use, and extend models developed by the community, Kipoi will foster the dissemination and use of ML models in genomics.

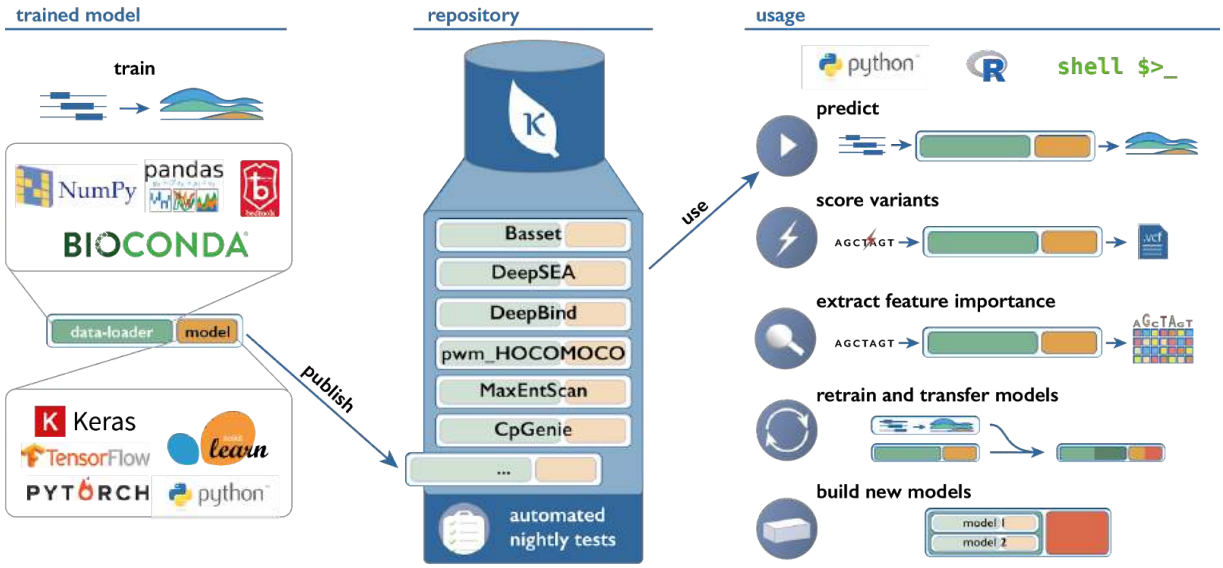


Figure 1 Overview. Kipoi (https://kipoi.org) defines an API for data-loaders and predictive models. Data-loaders translate genomics data types into numeric representation and enforce that all models can be applied to standard file format (fasta, bed, vcf, etc.). Kipoi models can be implemented using a broad range of ML frameworks. The models are automatically versioned, nightly tested and systematically documented with examples for their use. They can be accessed through unified interfaces from python, R, and command line. All models and their software dependencies get installed in a fully automatic manner. Kipoi streamlines the application of trained models to make predictions on new data, to score variants stored in the standard genetic variant file format, and to assess the effect of variation in the input to model predictions (feature importance score). Moreover, Kipoi models can be adapted to new tasks by either retraining them, or by combining existing ones.

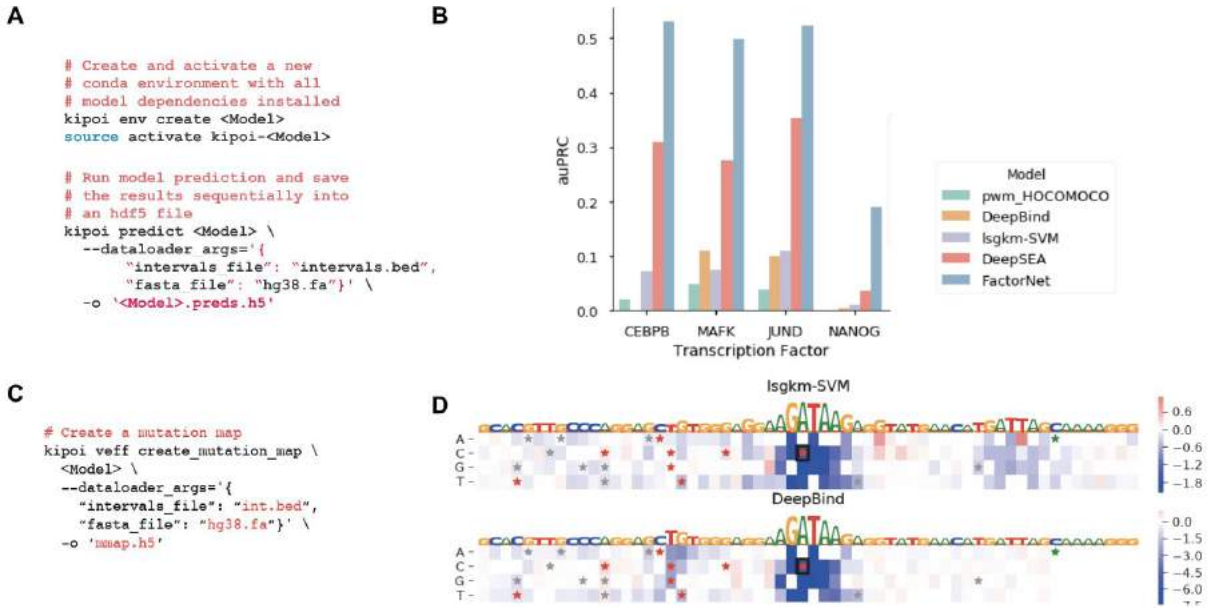


Figure 2 Simple and standard usage. (**A**) Use of Kipoi from the command line to install software dependencies, download the model, extract and pre-process the data, and write predictions to a new file. Results shown in (B) are obtained using these generic commands by varying the placeholder <Model>. (**B**) Benchmarking for TF binding prediction of several models (HOCOMOCO position weight matrix scanning[2], two sequence-based DL models DeepBind[3] and DeepSEA[4], support vector machine[5] and a DL model that further takes DNA accessibility data as input[6]). (**C**) Generic command for variant effect prediction. (**D**) Heatmap results of (C) for 2 models.

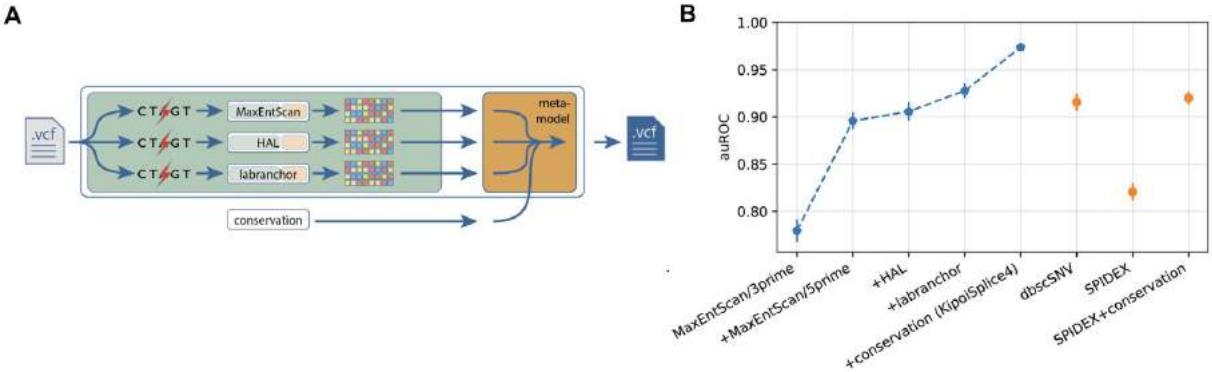


Figure 3 Composite models for improved pathogenic splice variant scoring. (**A**) A model trained to distinguish pathogenic from benign splicing region variants is easily constructed by combining Kipoi models for complementary aspects of splicing regulation. (**B**) Different versions of the ensemble model were trained and evaluated in 10-fold cross-validation for the ClinVar datasets. The four leftmost models are incrementally added to the composite model in chronological order of their publication. These performances were compared to a logistic regression model using state-of-the-art splicing variant effect predictors (SPIDEX[7], SPIDEX+conservation, dbscSNV[8]).

References

1. Avsec et al., bioRxiv. (2018) doi:10.1101/375345
2. Kulakovskiy et al. Nucleic Acids Res. 44, D116–25 (2016).
3. Alipanahi et al.. Nat. Biotechnol. 33, 831–838 (2015).
4. Zhou et al. Nat. Methods 12, 931–934 (2015).
5. Ghandi et al. PLoS Comput. Biol. 10, e1003711
6. Quang et al. bioRxiv (2017). doi:10.1101/151274
7. Xiong et al. *Science* (2015). 347(6218): 1254806.
8. Dong, C. *et al. Hum. Mol. Genet.* **24,** 2125–2137 (2014).

A method for systematically generating explorable visualization design spaces

Background. Stakeholders within public health can use the results of genomic analyses to establish practice guidelines and enact policies. Yet, these stakeholders vary in their abilities to interpret genomic findings and contextualize the results with other sources of data. Data visualization is an emergent solution to address interpretability challenges, but absent is a systematic and robust method to help identify the appropriate visualization to use in different contexts.

Methods. We have developed a systematic method for generating an explorable visualization design space, which catalogues visualizations existing within the infectious disease genomic epidemiology literature. Our method uses an automated literature analysis phase to establish *why* data were visualized, followed by a manual visualization analysis phase to establish *what* data were visualized and *how*. The literature analysis phase queried PubMed and used an unsupervised cluster analysis on article titles and abstracts to discover topic clusters that suggested why data were visualized. In order to ensure that we had a variety of data visualizations for further analysis, we sampled articles from across topic clusters and then extracted their figures. We then applied open and axial coding techniques, from qualitative research methods, to the sampled figures in order to iteratively derive taxonomic codes that described elements of each data visualization, thus enabling us to compare visualizations.

Results. We applied our method to a document corpus of approximately 18,000 articles, from which we sampled 204 articles for analysis. We added 17 articles manually for a final 221 articles that yielded 801 figures and 49 missed opportunity tables. These figures served as inputs to the visualization analysis phase and resulted in taxonomic codes along three descriptive axes of visualization design: chart types within the visualization, chart combinations, and chart enhancements. We refer to the collective complement of derived taxonomic codes as GEViT (Genomic Epidemiology Visualization Typology). To operationalize GEViT and the results of the literature analysis we have created a browsable image gallery (http://gevit.net), that allows an individual to explore the myriad of complex types of data visualizations (i.e. the visualization design space). Our analysis of the visualization design space through GEViT also revealed a number of data visualization challenges within infectious disease genomic epidemiology that future bioinformatics work should address.

Conclusions. Data visualization is powerful medium to help stakeholders better understand complex heterogenous data. By enumerating a visualization design space and empowering others to explore it, we enable a richer dialogue around the processes and practices for designing and evaluating contextually appropriate data visualizations.

Gallery: http://gevit.net/
Analysis Source Code: https://github.com/amcrisan/GEViTAnalysisRelease
Gallery Source Code: https://github.com/amcrisan/gevit_gallery_v2
Publication (Open Source): https://doi.org/10.1093/bioinformatics/bty832

Day2 14:00-14:20 (P-02)

snakePipes enable flexible, scalable and integrative epigenomic analysis

Epigenomics is a fast growing field, and due to the consistent fall in the price of sequencing,increase in multiplexing abilities, and multiple innovations in laboratory protocols, it has become increasingly convenient to perform multiple epigenomic assays within a project. However, a major bottleneck on the way to process and analyze this data in a reproducible way, particularly for novice analysts, is the availability of analysis pipelines. Next-generation sequencing (NGS) analysis pipelines are composed of a series of data processing steps, employ standardized processing parameters, and are usually scalable to large number of samples. Due to such properties, most pipelines are currently developed and deployed for settings where standardized, large scale analysis is required. Examples are RNA-seq variant-calling pipelines deployed in clinical settings, or processing pipelines developed for large-scale consortia.

However, in a typical basic science research setting, researchers also seek to modify parameters, update tool versions or extend the workflows, while maintaining their scalability and ease-of-use. Conventional NGS pipelines, although scalable, do not allow for this flexibility. Options for exploratory and downstream analysis have been limited, resulting in various expert users developing their own custom pipelines suited to their needs. Computational frameworks such as Galaxy and Nextflow exist, but they still demand novice users to be trained and implement their workflows themselves from scratch. This leads to a conundrum, how can we provide a set of workflows following best-practices that are easy to install and run for the novice users, while still providing the flexibility of extending and upgrading the workflows to the expert users?

We developed snakePipes to address such requirements. snakePipes provide flexible processing as well as downstream analysis of data from the most common assays used in epigenomic studies: ChIP-seq, RNA-seq, whole-genome bisulfite-seq (WGBS), ATAC-seq, Hi-C and single-cell RNA-seq in a single package. It employs snakeMake as a workflow language, which benefits from easy readability of the code,widespread adoption, and offers scalability using most cluster and cloud computing platforms. snakePipes also makes use of Conda environments and the Bioconda platform, which allow hassle-free installation and upgrade of tools. Conda environments allow execution of tools avoiding dependency conflicts, and do not require root permissions to run. Due to a modular architecture, various tools are shared between workflows,which simplifies data integration since data from multiple technologies are processed using identical tool versions and genome annotations. The genome annotations and indices are shared by all workflows, and can also be generated directly via snakePipes, facilitating easy setup as well as integrative analysis. Finally, workflows in snakePipes employ extensive quality-checks and also produce reports using multiQC and R, that inform the user of processing and analysis results.

Apart from conventional processing steps such as mapping, counting and peak calling, workflows in snakePipes also include various downstream analysis. All workflows (except scRNA-seq workflow) optionally accept a sample information (tab-separated) file that can be used to define groups of sample. This allows comparative analysis such as differential gene or transcript expression analysis for the RNA-seq workflow, differential peak calling for ChIP-Seq workflow, differential open chromatin detection for ATAC-seq workflow, and detection of differentially methylated regions (either de-novo or on user-specified regions)

for WGBS workflow. The HiC workflow uses sample information to merge groups and can perform TAD calling with parameters adapted to the resolution of produced matrix (using HiCExplorer). This preliminary analysis, combined with visualization-ready bed and bigWig files, allows users to quickly interpret their data.

We also utilize a public dataset using multiple sequencing modalities to demonstrate how a multi-assay epigenomic analysis toolkit like snakePipes can simplify data processing, reproduce previously published results, and allow new biological interpretations with minimal effort.

Day2 14:20-14:40

nf-core: Community built bioinformatics pipelines

Philip A Ewels[1], Alexander Peltzer[2], Sven Fillinger[2], Johannes Alneberg[1], Harshil Patel[3], Andreas Wilm[4], Maxime Ulysse Garcia[5], Paolo Di Tommaso[6], Sven Nahnsen[2]

1. Science for Life Laboratory (SciLifeLab), Department of Biochemistry and Biophysics, Stockholm University, Stockholm, Sweden
2. Quantitative Biology Center (QBiC), University of Tübingen, Tübingen, Germany
3. Bioinformatics and Biostatistics, The Francis Crick Institute, London, United Kingdom
4. A*STAR Genome Institute of Singapore, Bioinformatics Core Unit, Singapore, Singapore
5. Department of Oncology, Karolinska Institute, Stockholm, Sweden
6. Centre for Genomic Regulation (CRG), The Barcelona Institute for Science and Technology, Barcelona, Spain.

The standardization, portability, and reproducibility of analysis pipelines is a renowned problem within the bioinformatics community. In the past, bioinformatic analysis pipelines have often been designed to work on-premise, deeply integrated into the local infrastructure and did show a customized architecture style. Because of this tight coupling of software to its surrounding environment, the resulting pipelines provided poor portability and reproducibility. Nextflow is a system that is able to provide functionality to make analysis reusability, portability and reproducibility complete, with built-in support for most computational infrastructures and container technologies such as Docker, Conda and Singularity. nf-core is a community effort to implement and collect Nextflow pipelines based on community best practices and tools. The guidelines and templates provided by the nf-core community along with a detailed documentation enable users to add new workflows and get started with Nextflow seamlessly. The outcome is a set of high-quality bioinformatics pipelines that researchers can apply broadly across various institutions and research facilities, as all workflows share common usage patterns and robust community support. Our primary goal is to provide a community-driven

platform for a high-quality set of reproducible bioinformatics pipelines that researchers can utilize across various Institutions and research facilities.

Day2 14:40-15:00 (P-14)

NGLess: a domain-specific language for NGS analysis (NG-meta-profiler as a case study)

Luis Pedro Coelho (coelho@fudan.edu.cn)[1,2,3], Renato Alves[1,4], Paulo Monteiro[5], Jaime Huerta-Cepas[1,6], Ana Teresa Freitas[5], Peer Bork[1,7,8,9,1]

Project Website: https://ngless.embl.de
Source Code: https://github.com/ngless-toolkit/ngless (License: MIT)

Linking different programs is an integral part of bioinformatics, which is most often performed using either traditional programming scripting languages or, increasingly, workflow-engines that coordinate calling multiple programs. Our hypothesis was that a domain-specific language could form the basis of a better tool for the specific problem of processing next-generation sequencing (NGS) data. The resulting language, NGLess, enables the user to work with abstractions that are closer to the problem domain and we show how this enhances usability and correctness, while implementing scientific best practices. Results computed with NGLess are independent of the environment in which they are generated and, thus, perfectly reproducible.

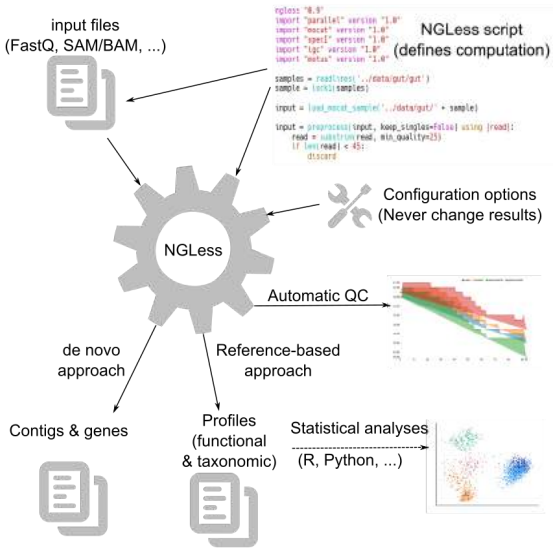


Figure 1. Cartoon depiction of NGLess' approach: A script in the NGLess language defines the computational pipeline, which defines how the outputs relate to the input data, while configuration options can provide accessory information (e.g., temporary file storage) which do not change the results.

[1] European Molecular Biology Laboratory, Heidelberg, Germany; [2] Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, Shanghai, China; [3] Key Laboratory of Computational Neuroscience and Brain-Inspired Intelligence (Fudan University), Ministry of Education, China; [4] Collaboration for joint PhD degree between EMBL and Heidelberg University, Faculty of Biosciences; [5] INESC-ID, Instituto Superior Técnico, University of Lisbon, Portugal; [6] Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid (UPM), Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria (INIA), Madrid, Spain; [7] Max Delbrück Centre for Molecular Medicine, Berlin, Germany; [8] Molecular Medicine Partnership Unit, University of Heidelberg and European Molecular Biology Laboratory, Heidelberg, Germany; [9] Department of Bioinformatics, Biocenter, University of Würzburg, Würzburg, Germany.

NGLess is a Domain Specific Language (DSL) for scientific pipelines analysing short-read data. With NGLess, users write a short script (for an example, see Fig. 2) describing the computation.

As correcting errors (debugging) takes a significant fraction of total development time, NGLess aims to reduce the time involved and performs several checks on the inputs *prior* to the start of interpretation, so that errors are found faster. Errors that can be detected include includes traditional elements of statically typed programming languages, but also factors such as file permissions, and the existence of all necessary dependencies.

NGLess' builtin functionality can be easily extended with text-based descriptions of external tools. The file formats can be described precisely, but the user needs only to match at a semantic level. For example, a tool that produces a BAM file (which is mapped to the MappedShortReadSet type in NGLess) can have its outputs passed to another tool which consumes a SAM file (as this is also of type MappedShortReadSet). In this case, NGLess will automatically insert the conversion step between these two tools without extra effort for the user.

```
ngless "1.0"
import "mocat" version "1.0"
import "igc" version "1.0"

input = load_mocat_sample(ARGV[1])
RESULTS = ARGV[2]

qc_reads = preprocess(input, keep_singles=False) using |read|:
    read = substrim(read, min_quality=25)
    if len(read) < 45:
        discard

human_mapped = map(qc_reads, reference='hg19')

non_human = select(human_mapped) using |mr|:
    mr = mr.filter(min_match_size=45, min_identity_pc=90, action={unmatch})
    if mr.flag({mapped}):
        discard

non_human_reads = as_reads(non_human)

igc_mapped = map(non_human_reads, reference='igc', mode_all=True)
igc_mapped_post = select(igc_mapped) using |mr|:
    mr = mr.filter(min_match_size=45, min_identity_pc=95, action={drop})
    if not mr.flag({mapped}):
        discard

igc_counts = count(igc_mapped_post,
                   features=['OGs'],
                   multiple={dist1},
                   normalization={scaled})
write(igc_counts,
        ofile=RESULTS </> 'eggNOG.traditional.counts.txt',
        auto_comments=[{hash}, {script}])
```

Figure 2. Abridged version of NG-meta-profiler as an example of the NGLess language.

Finally, unlike tools based on traditional programming languages or targeting a broader set of uses, NGLess, as a scientific tool, makes reproducibility one of its design goals. Given the same inputs and the same NGLess script, the output will be identical in every situation. As a result of this design constraint, configuration options that do not affect the result (e.g., where to store temporary files) are kept separate from the specification of the computational task to be performed.

Using NGLess 1.0, we implemented NG-meta-profiler, a tool for producing taxonomic and functional profiles from metagenomes based on pre-annotated gene catalogs. In particular, NG-meta-profiler supports (1) data preprocessing, (2) mapping (short read alignment), (3) filtering mapping results, and (4) profiling (computation of summary statistics). Through the use of NGLess, NG-meta-profiler is significantly faster than MOCAT2 and htseq-count.

Day2 15:00-15:05

Benten: An experimental language server for the Common Workflow Language

Kaushik Ghose

Repository : https://github.com/rabix/benten
License : Apache 2.0

Many experienced Common Workflow Language (CWL) users are comfortable creating tools and workflows "by hand" using a plain text editor. When creating complex enough workflows, navigating and editing the resultant document and subdocuments can get tedious. Keeping track of the bigger picture (what components have been added, what connections have been set) can also get hard.

Benten is a language server component that assists CWL development in a code editor by providing auto-complete suggestions and document outlines. It has been built and tested with VS Code but can be used with any editor/IDE that implements the language server protocol.

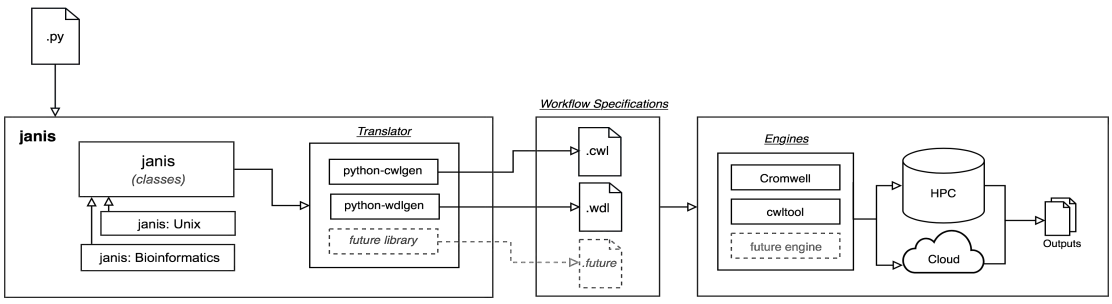
Day2 15:05-15:10 (P-22)

"Janis: An open source tool to machine generate type-safe CWL and WDL workflows"

The rapid development of Next Generation Sequencing (NGS) in recent years has enabled the generation of large volumes of data, which will typically be analysed through a series of bioinformatics tools, often referred to as a pipeline or workflow. With the increasing number of bioinformatics tools and workflow systems available to analyse these data, we face a problem in sharing and reproducing our work. Addressing this issue, there are ongoing global efforts to improve the reproducibility and portability of bioinformatics pipelines. Projects such as Common Workflow Language (CWL) aim to standardise how workflows are being specified, while workflow execution engines such as Toil and Cromwell provide many useful features to run pipelines across high performance computing (HPC) and cloud infrastructures. However, there is often debate on whether to adopt CWL, Workflow Definition Language (WDL) or other competing standards. CWL provides rigid, easy-to-parse specifications and is supported by variety of engines, but is considered more difficult to write, while other standards such as WDL offer more features and an easier learning curve but are tightly coupled to a specific engine such as Cromwell.

To address this issue, we have created Janis[1], an open source tool to machine generate CWL and WDL workflows. It is designed to assist in building standardised workflows via a translation mechanism that generates validated workflow specifications (CWL, WDL or both) as output. These translated workflow acts as a 'transport layer' that can be shared and executed using any workflow execution engine that supports the selected specifications. Janis also offers input and output type checking during workflow construction to connect steps in pipelines and enforce the input requirements of executed tools. This is especially important when dealing with those that generate or expect secondary files. For example, reference genome file in fasta format is often associated with various index files.

The diagram below illustrates Janis' architecture and how the translated workflows are used to run bioinformatics analysis across HPC and cloud infrastructure.



Bioinformatics tools are defined as python classes that import methods from the main Janis module (for example: janis-bioinformatics[2]). A Janis workflow will define a series of connections (edges) between the inputs and outputs from bioinformatics tools (object of the above python classes) specified as workflow steps (for example: janis-examplepipelines[3]). These python in-memory representation of the bioinformatics tools and workflows can then be exported to CWL and WDL via translator libraries (python-cwlgen[4] and python-wdlgen[5]). The resulting workflow specifications from Janis are valid CWL and WDL files that can be executed locally, on HPC or cloud environment, using workflow engines like Cromwell, cwltool or Toil.

Through Janis, we have developed germline and somatic variant-calling pipelines that are functional across the HPC environment at three different Australian research institutes as well as cloud environment. These pipelines are written in python (Janis), translated to both CWL and WDL, and executed with Cromwell.

In future work, we may extend Janis to support additional output formats, should the need arise. This output formats may include other workflow specifications, translation to generate shell scripts or pipeline description in plain human readable languages. We believe that the extra abstraction provided by Janis provides a powerful way to write bioinformatics pipeline specifications that are portable across many different computing platforms.

[1]https://github.com/PMCC-BioinformaticsCore/janis
[2]https://github.com/PMCC-BioinformaticsCore/janis-bioinformatics
[3]https://github.com/PMCC-BioinformaticsCore/janis-examplepipelines
[4]https://github.com/common-workflow-language/python-cwlgen
[5]https://github.com/illusional/python-wdlgen

Day2 15:10-15:15 (P-04)

Collecting runtime metrics of genome analysis workflows by CWL-metrics

Tazro Ohta[1], Tomoya Tanjo[2], Osamu Ogasawara[3]

1. Database Center for Life Science, Joint Support-Center for Data Science Research, Research Organization of Information and Systems, Shizuoka 411-8540, Japan
2. National Institute of Informatics, Research Organization of Information and Systems, Tokyo 101–8430, Japan
3. DNA Data Bank of Japan, National Institute of Genetics, Research Organization of Information and Systems, Shizuoka 411-8540, Japan

Project Website: https://inutano.github.io/cwl-metrics/
Source Code: https://github.com/inutano/cwl-metrics
License: MIT

Portability of computational data analysis environment is largely improved by container virtualization technologies such as Docker [1] and workflow description frameworks represented by Common Workflow Language (CWL) [2]. To deploy an environment for data analysis workflows, researchers have to select an appropriate computational platform for the given application. To provide information to estimate the computational resource such as CPU or memory required by workflow execution, we developed CWL-metrics, a utility system of cwltool (the reference implementation of CWL) to collect runtime metrics of CWL workflows. The system summarizes resource usage of each step of workflow with the input parameters and the information of the host machine.

We demonstrated the performance comparison of RNA-Seq quantification workflows by using metrics captured by CWL-metrics [3,4]. The comparison analysis results recorded in Jupyter Notebook are published on GitHub [5]. The system is utilized in the new pipeline system being deployed on the high-performance computing platform of the DNA Data Bank of Japan (DDBJ) to collect the metrics to help administrators of the platform. The metrics information captured by CWL-metrics is also being used by the development of resource prediction algorithms. We also would like to present the progress of the new version of CWL-metrics currently under development, which will increase the coverage of supported workflow runners and container technologies including Singularity container [6].

1. Docker, https://www.docker.com
2. Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., ... & Scales, M. (2016). Common Workflow Language, v1. 0.
3. Ohta T, Tanjo T, Ogasawara O. Accumulating computational resource usage of genomic data analysis workflow to optimize cloud computing instance selection. bioRxiv; 2018. DOI: 10.1101/456756
4. Ohta T, Tanjo T, Ogasawara O. Accumulating computational resource usage of genomic data analysis workflow to optimize cloud computing instance selection. GigaScience; 2019. In press.
5. Code for CWL-metrics manuscript https://github.com/inutano/cwl-metrics-manuscript
6. Kurtzer GM, Sochat V, Bauer MW. Singularity: Scientific containers for mobility of compute. 2017. PloS one, 12(5), e0177459. DOI: 10.1371/journal.pone.0177459

Day2 15:30-15:50

Inclusiveness in Open Science Communities: examples from the EMBL Bio-IT project

Malvika Sharan[1,2,*], Toby Hodges[1]

[1]*European Molecular Biology Laboratory (EMBL)*, Heidelberg, Germany
[2]*Heidelberg Center for Human Bioinformatics (HD-HuB)* within the *German Network for Bioinformatics Infrastructure (de.NBI)- ELIXIR Germany*

[*]**Presenting author's email**: malvika.sharan@embl.de

Project Website:
EMBL Bio-IT: https://bio-it.embl.de/

Abstract:

Bio-IT (https://bio-it.embl.de) at the European Molecular Biology Laboratory (EMBL) (https://embl.org) is a community-driven initiative established in 2010 to support the development and technical capacity of its diverse bio-computational community. There has been a consistent increase in the number of researchers who use computational techniques in their work in the last decade. As of now, ~50% of researchers at EMBL (out of ~600) devote ≥50% of their time to computational work. Several of these members are service staff who dedicate their time to building or maintaining computational infrastructure and providing computational support to others. The Bio-IT community at EMBL has grown organically, aiming to address the various computational needs in research on campus.

As community coordinators of Bio-IT, we provide support to our members by conducting training events on computing skills, developing/maintaining resources for reproducible science, adopting best practices in our workflow, and creating diverse opportunities for open discussions, participation, and networking. EMBL is a member of de.NBI, the German Network for Bioinformatics Infrastructure, which constitutes ELIXIR Germany. This allows Bio-IT to disseminate its resources to different ELIXIR states. Additionally, we collaborate with other Open Science communities such as The Carpentries, Software Sustainability Institute, and Mozilla to exchange expertise, share resources and bring valuable aspects of the larger and more diverse communities into Bio-IT.

In all these efforts, I work at the intersection of community building, bio-computation, and inclusion of underrepresented groups in STEM. In my talk, I will highlight the importance of inclusiveness in open science communities and share some of the lessons learned while adopting them in my work. This talk will benefit other community managers and researchers interested in creating local or virtual sustainable communities and will stimulate discussions around inclusion and representation in technical fields such as computational biology.

We invite others to connect with us through global Open Science communities or by any of the following possibilities:

1. External participants as speakers, trainers or learners

We invite experts, trainers, and speakers to our events to teach computing or bioinformatics skills useful for our learners. Many of our courses are open for external researchers to attend. These opportunities allow our participants to exchange ideas and collaborate with each other.

2. Networking with other community leaders

In the past, Bio-IT has hosted a few community calls and an in-person meeting of the community managers to swap practical notes for supporting community of practices. We would be interested in connecting with others in the conference to create similar opportunities in the future.

3. Re-use our learning resources

Bio-IT hosts learning resources developed by the EMBL staff on its homepage (https://bio-it.embl.de/course-materials/). These resources are available under open licenses and can be freely used.

Day2 15:50-16:10 (P-34)

ECRcentral: An open-source platform to bring early-career researchers and funding opportunities together

Aziz Khan[1,*], Juan F. Quintana[2], Charlotte M de Winde[3] and Cristiana Cruceanu[4]

1. Centre for Molecular Medicine Norway (NCMM), University of Oslo, Norway
2. Wellcome Trust Centre for Anti-Infective Research (WCAIR), University of Dundee, United Kingdom
3. Stromal Immunology Group, MRC Laboratory for Molecular Cell Biology, University College London, United Kingdom
4. Max Planck Institute of Psychiatry, Munich, Germany

* Correspondence: aziz.khan@ncmm.uio.no

Abstract

For early-career researchers (ECRs), getting funding for their research ideas is becoming more and more competitive, and there is growing pressure in all disciplines to obtain grants. Although there is a plethora of funding opportunities for postdoctoral scientists and other ECRs, there has been no central platform to systematically search for such funding opportunities and/or to get professional feedback on the proposal. With a group of eLife Ambassadors, we developed ECRcentral (ecrcentral.org), a funding database and an open forum for the ECR community. The platform is open to everyone and currently contains 700 funding schemes in a wide range of scientific disciplines, 100 travel grants, and a diverse range of useful resources. In the first two months since its release approximately 500 ECRs already joined this community. The platform is developed using open-source technology, with all the source code and related content made openly available through our GitHub repository (github.com/ecrcentral). ECRcentral aims to bring ECRs and resources for funding together, to facilitate discussions about those opportunities, share experiences, and create impact through community engagement. We strongly believe that this resource will be highly valuable for ECRs and the scientific community at large.

Detailed abstract

For early career researchers (ECRs), getting funding for their research ideas is becoming more and more competitive, and there is growing pressure in all disciplines to obtain grants. Although there are a good number of fellowships and travel funding opportunities for postdoctoral scientists and other ECRs, until now there has been no central platform to search for such funding opportunities and/or to get professional feedback on one's proposal. A group of eLife Ambassadors, as ECRs ourselves, we were aware of this problem and developed ECRcentral (ecrcentral.github.io), an open platform offering the largest available (and continually growing) database of funding opportunities. ECRcentral is a community-driven initiative, which aims to bring ECRs together to discuss funding opportunities, share experiences, and create impact through community engagement. ECRcentral is a searchable database of fellowship opportunities, travel grants and a diverse range of useful resources. The list is curated by the community, and anyone can add new fellowships and travel grants. It also enables scientists to easily offer and seek advice from each other. Similarly, the platform gives access to a number of useful resources related to funding and career development, key skills for better reproducibility, and science communication. Additionally, there is also a discussion forum to facilitate topical conversations. The ECRcentral platform comes with several modules and with options to add, edit and/or delete funding schemes, funders, travel grant, and resources. The platform has an administrative interface to create and manage the content at different roles (admin, moderator, contributor, and user). ECRcentral.org contains almost 700 funding schemes in several STEM disciplines, 100 travel grants and resources, freely available to all and without a need to register. In just two months since its release, 500 ECRs already joined the community and it received 169,000 page views by 90,000 users from almost all countries of the world. The platform is developed using the open-source PHP web framework Laravel and Bootstrap used for the user interface. All the source code and documentation openly available through our GitHub repository github.com/ecrcentral, so the community can contribute and/or reuse the code.

The next goal of ECRcentral is to provide an open forum to obtain feedback with regards to the specific application processes, and to help prospective candidates connect with former recipients of the funding scheme in order to get tips for their applications as well as to seek mentoring opportunities. We aim to bring ECRs and funding together, to facilitate discussions about those opportunities, share experiences, and create impact through community engagement. We strongly believe that this resource will be highly valuable for ECRs and the scientific community at large.

Acknowledgments

ECRcentral is community-driven initiative supported by eLife and the eLife Community Ambassador program. The initial idea of this database was pitched by a former eLife ECAG member Sonia Sen. Naomi Penfold and Gary Mcdowell were instrumental in discussing ideas at the initial stages. The initial fellowship funding list was curated/edited by Dieter Lukas (Max Planck Institute for Evolutionary Anthropology) and eLife Community Ambassador Juan F Quintana (Wellcome Centre for Anti-Infectives Research, University of Dundee). The travel grant list is curated by Charlotte M de Winde (MRC Laboratory for Molecular Cell Biology, University College London, UK), eLife Community Ambassador and member of the eLife Early Career Advisory Group. The idea to integrate a community forum was initiated by Cristiana Cruceanu (Max Planck Institute of Psychiatry, Germany). The ECRcentral platform is designed and developed by Aziz Khan (University of Oslo, Norway).

Day2 16:10-16:15 (P-30)

The Carpentries builds global capacity for conducting efficient, open, and reproducible research. We train and foster an active, inclusive, diverse community of learners and instructors that promotes and models the importance of software and data in research. We collaboratively develop openly-available lessons and deliver these lessons using evidence-based teaching practices.

Within The Carpentries, Data Carpentry is a lesson program that focuses on novices and teaches data skills through domain-specific lessons centered around a dataset, teaching two-day hands-on workshops. Our Data Carpentry Genomics lessons focus on the core skills throughout the genomics data analysis lifecycle, from data and project organization to analysis and visualization. Using open data from a published analysis of the evolution of bacterial genomes over 50,000 generations (Tenaillon et al 2016), we cover the following material:

- Project organization and management - How to structure metadata, organize and document genomics data and bioinformatics workflow, and access data on the NCBI sequence read archive (SRA) database.
- Introduction to the command line - How to navigate file system, create, copy, move, and remove files and directories, and automate repetitive tasks using scripts and wildcards.
- Data wrangling and processing - How to use command-line tools to perform quality control, align reads to a reference genome, and identify and visualize between-sample variation.
- Introduction to cloud computing for genomics - How to work with Amazon AWS cloud computing and how to transfer data between local computer and cloud resources.
- Intro to R and RStudio for Genomics - How to use R to analyze and visualize between-sample variation.

The goal of the workshop is to introduce learners to the concepts and tools they need to get started with the analysis of genomics data and learn best practices for reproducibility. More than 20 people have contributed to the development of this curriculum, and the workshop has been taught 12+ times in the last year. A recent update to the curriculum to reflect current sequencing technologies and tools, has projections for even more workshops this year.

As part of our workshop activities, we ask participants to complete pre- and post-workshop surveys. The results of these surveys allow us to ensure that the workshop took place in the positive learning environment we aim to create, and to evaluate how the workshop impacts learners' skills and confidence.

Initial feedback shows that workshops are well-received, with a median recommendation of 96% and surveys show that learners report significant confidence gains in using these approaches and applying them to their work. While not specific to Genomics workshops, Data Carpentry workshops generally show ~20% increase in confidence after just the two day workshop (Fig. 1).

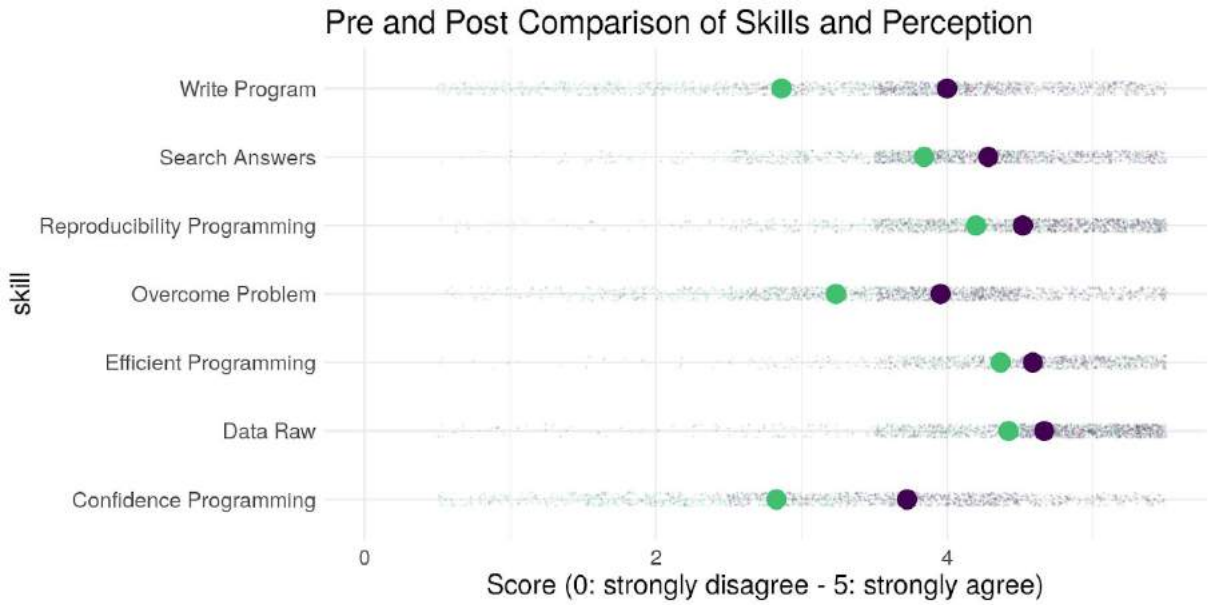


Figure 1: Comparison of the pre- and post-workshop scores for skills evaluated by The Carpentries survey

Our lessons are developed and maintained by volunteer members of the community on GitHub where contributions are welcomed, and are released under a Creative Commons Attribution Licence.

We are looking to share lessons learned in collaborative curriculum development and maintenance and expand the community involved in teaching and maintaining this curriculum.e hope conference attendees will be interested in taking, teaching or sharing this workshop and curriculum and getting involved. Bringing this content to more people can help scale the number of people with the skills and perspectives to work effectively and reproducibly with genomic data and begin to develop and collaborate around software development in bioinformatics.

Tenaillon O, Barrick JE, Ribeck N, Deatherage DE, Blanchard JL, Dasgupta A, Wu GC, Wielgoss S, Cruveiller S, Médigue C, Schneider D, and Lenski RE, 2016. Tempo and mode of genome evolution in a 50,000-generation experiment. Nature. (doi: 10.1038/nature18959)

Day2 16:15-16:20 (P-35)

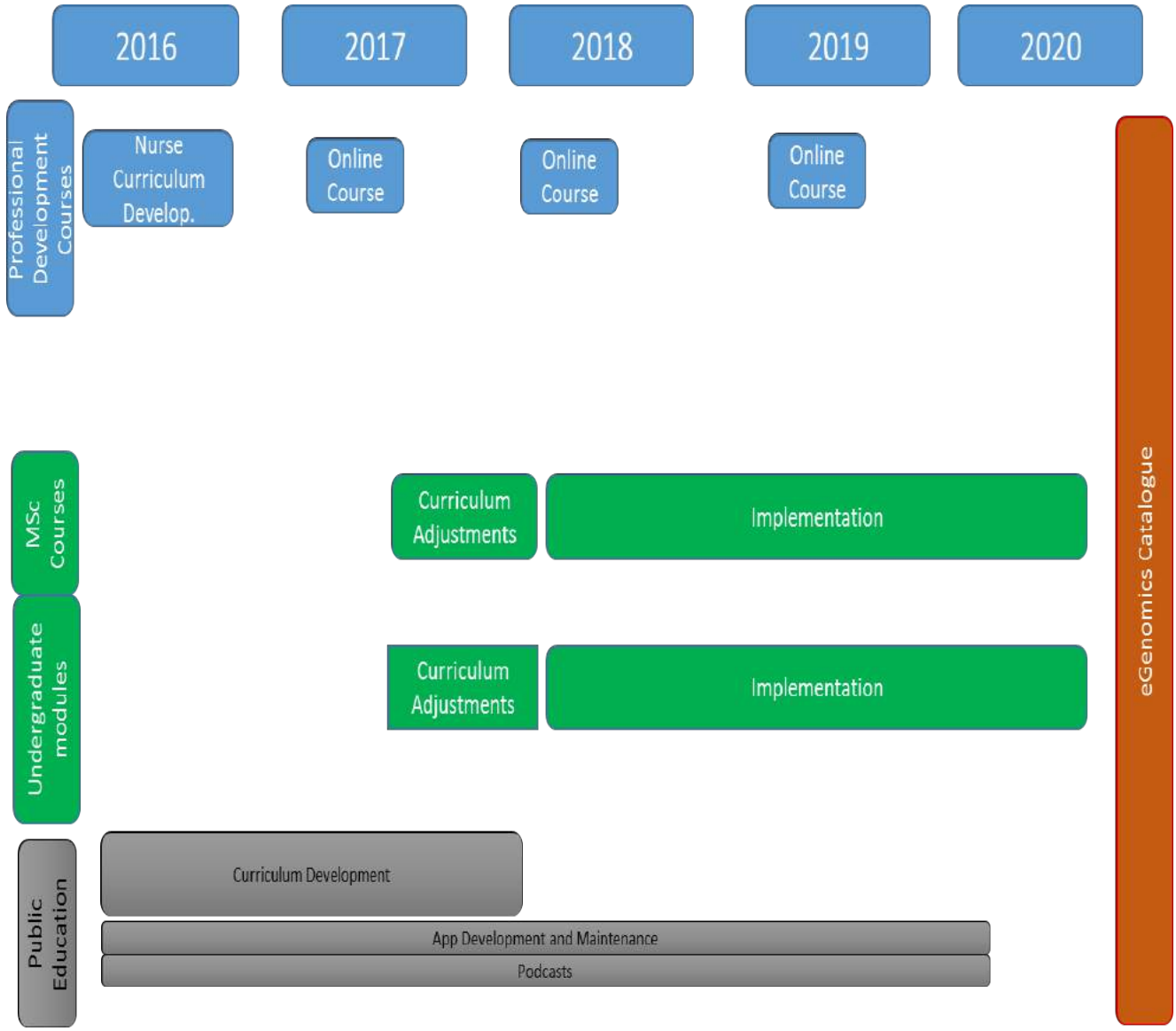
The African Genomic Medicine Training Initiative: Showcasing A Community-Driven Genomic Medicine Competency-Based Training Model for Nurses in Africa

Victoria Nembaware, Paballo Chauke, Nicola Mulder and Planning team

The potential of Genomic Medicine to improve the quality of healthcare both at population and individual-level is well-established, however adoption of available genetic and genomics evidence into clinical practice is limited. Widespread uptake largely depends on the task-shifting of Genomic Medicine to key healthcare professionals such as nurses, who could be promoted through professional development courses. Globally, trainers, and training initiatives in Genomic Medicine are limited, and in resource limited settings such as Africa, logistical and institutional challenges threaten to thwart large-scale training programmes. The African Genomic Medicine Training (AGMT) Initiative was created in response to such needs. It aims to establish sustainable Genomic Medicine training initiatives for healthcare professionals and the public in Africa. This work describes the AGMT and reports on a strategy recently piloted by this group to design and implement an accredited, competency and community-based distance learning course for nurses across 11 African countries. This model takes advantage of existing consortia to create a pool of trainers and adapts evidence-based approaches to guide curriculum and content development. Existing curricula were reviewed and adapted to suit the African context. Accreditation was obtained from university and health professional bodies. Both the acceptability of this model, the feasibility of replication in similar settings, and training a wide-range of healthcare professionals, is supported by data from an implementation evaluation that was informed by class mini-projects tailored to African diseases submitted for peer-reviewed publication, reflections and surveys from the working group members, advisors, course coordinator, facilitators, trainers and students. A toolkit is proposed to help guide adoption of the AGMT distance-learning model.

Figure 1: **African Genomic Medicine Training Initiative – Official Launch 12 May – 2016: Dakar Senegal**

AGMT Strategic Planning:



Impact of the course:

- 368 Applicants, 225 Registered, ~90 completed in 2017
- There were 1300 applicants in 2019 for the second iteration
- Continuing Professional Development Points (CPD) are given to participants
- Data on motivation, stigmas, resources
- Quantitative assessment of change in knowledge, attitudes, perceptions and practises.
- Class projects – published in a special collection in a journal Global Health, Epidemiology and Genomics (GHEG)
- **Egypt class** using the course to adjust their nurse training

Day2 16:20-16:25

Biopython Project Update 2019

Peter Cock[a],
and the Biopython Contributors[b]

20th Bioinformatics Open Source Conference (BOSC) 2019, Basel, Switzerland

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]. This talk will look ahead to the year to come, and give a summary of the project news since the 1.72 release in June 2018, and the talk at GCCBOSC 2018.

While there were no major new modules introduced in Biopython 1.73 (December 2018) or Biopython 1.74 (expected May/June 2019), there have been lots of incremental improvements. In terms of lines of code changed, a substantial proportion has been in-line documentation (Python docstrings), used to generate human-readable API documentation. While we are still using `epydoc` for this, our continuous integration system has been generating more modern HTML output using `sphinx`, which we hope to host on our domain, or at Read The Docs, making this work much more visible to the world. We have been using the tool `flake8` with various plugins for this (as well as checking coding style), showing a steady improvement in best practice compliance - every public API should be documented this year.

In 2017 we started a 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, to gradually dual license the entire project. All new contributions are dual licensed, and currently, half the Python files in the main library have been dual licensed.

Another important going effort is improving the unit test coverage. Sadly this is currently fairly static at about 85% (excluding online tests), but can be viewed online at CodeCov.io.

We are using GitHub-integrated continuous integration testing on Linux (using TravisCI) and Windows (using AppVeyor), including enforcing the Python PEP8 and PEP257 coding style guidelines. We hope to be able to recommend a simple `git pre-commit` hook for our contributors shortly, and have discussed the idea of adopting the new yet popular Python code formatting style tool `black` to reduce the human time costs in writing compliant code.

Looking further ahead, in 2020, in line with most major scientific Python libraries, we will be dropping support for Python 2. See https://python3statement.org/

Finally, since our last update talk in June 2018, Biopython has had 32 named contributors including 14 newcomers. This reflects our policy of trying to encourage even small contributions. Our total named contributor count is now at 248 since the project began, and looks likely to break 250 by our 20th Birthday in August 2019.

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]Information and Computational Sciences, James Hutton Institute, Invergowrie, Dundee, UK
[b]See contributor listing on GitHub.

pysradb: A Python package to query next-generation sequencing metadata and data from NCBI Sequence Read Archive

Saket Choudhary*

Website : http://saketkc.github.io/pysradb/
Repository : https://github.com/saketkc/pysradb
Example : https://github.com/saketkc/pysradb/blob/master/docs/usage_scenarios.rst
License : BSD 3-Clause

NCBI's Sequence Read Archive (SRA) is the primary archive of next-generation sequencing datasets. SRA makes metadata and raw sequencing data available to the research community to encourage reproducibility, and to provide avenues for testing novel hypotheses on publicly available data. However, methods to programmatically access this data are limited. NCBI's SRA toolkit [1] provides utility methods to download raw sequencing data, while the metadata can be obtained by querying the website or through the Entrez `efetch` command line utility [2]. Most workflows analyzing public data rely on first searching for relevant keywords in the metadata either through the command line utility or the website and then downloading these. A more streamlined workflow can enable doing both these steps at once.

We introduce a Python package `pysradb` that provides a collection of command line methods to query and download metadata and data from SRA utilizing the curated metadata database available through the SRAdb [3] project.

`pysradb` package builds upon the principles of `SRAdb` providing a simple and user-friendly command-line interface for querying metadata and downloading datasets from SRA. It obviates the need for the user to be familiar with any programming language for querying and downloading datasets from SRA. Additionally, it provides utility functions that will further help a user perform more granular queries, that are often required when dealing with multiple datasets at large scale. By enabling both metadata search and download operations at the command-line, `pysradb` aims to bridge the gap in seamlessly retrieving public sequencing datasets and the associated metadata.

`pysradb` is written in Python and is currently developed on Github under the open-source BSD 3-Clause License. Each sub-command of `pysradb` contains a self-contained help string, that describes its purpose and usage example with additional documentation available on the project's website. In order to simplify the installation procedure for the end-user, it is also available for download through both PyPI and bioconda [4].

References

[1] SRA Toolkit Development Team, "Sra toolkit." https://ncbi.github.io/sra-tools/, Dec 2018. [Online; accessed 10-December-2018].

[2] J. Kans, "Entrez direct: E-utilities on the unix command line," 2018.

[3] Y. Zhu, R. M. Stephens, P. S. Meltzer, and S. R. Davis, "Sradb: query and use public next-generation sequencing data from within r," *BMC bioinformatics*, vol. 14, no. 1, p. 19, 2013.

[4] B. Grüning, R. Dale, A. Sjödin, B. A. Chapman, J. Rowe, C. H. Tomkins-Tinch, R. Valieris, J. Köster, and T. Bioconda, "Bioconda: sustainable and comprehensive software distribution for the life sciences.," *Nature methods*, vol. 15, no. 7, p. 475, 2018.

*Computational Biology and Bioinformatics, University of Southern California, Los Angeles, USA. Email: skchoudh@usc.edu

P-02

snakePipes enable flexible, scalable and integrative epigenomic analysis

Epigenomics is a fast growing field, and due to the consistent fall in the price of sequencing,increase in multiplexing abilities, and multiple innovations in laboratory protocols, it has become increasingly convenient to perform multiple epigenomic assays within a project. However, a major bottleneck on the way to process and analyze this data in a reproducible way, particularly for novice analysts, is the availability of analysis pipelines. Next-generation sequencing (NGS) analysis pipelines are composed of a series of data processing steps, employ standardized processing parameters, and are usually scalable to large number of samples. Due to such properties, most pipelines are currently developed and deployed for settings where standardized, large scale analysis is required. Examples are RNA-seq variant-calling pipelines deployed in clinical settings, or processing pipelines developed for large-scale consortia.

However, in a typical basic science research setting, researchers also seek to modify parameters, update tool versions or extend the workflows, while maintaining their scalability and ease-of-use. Conventional NGS pipelines, although scalable, do not allow for this flexibility. Options for exploratory and downstream analysis have been limited, resulting in various expert users developing their own custom pipelines suited to their needs. Computational frameworks such as Galaxy and Nextflow exist, but they still demand novice users to be trained and implement their workflows themselves from scratch. This leads to a conundrum, how can we provide a set of workflows following best-practices that are easy to install and run for the novice users, while still providing the flexibility of extending and upgrading the workflows to the expert users?

We developed snakePipes to address such requirements. snakePipes provide flexible processing as well as downstream analysis of data from the most common assays used in epigenomic studies: ChIP-seq, RNA-seq, whole-genome bisulfite-seq (WGBS), ATAC-seq, Hi-C and single-cell RNA-seq in a single package. It employs snakeMake as a workflow language, which benefits from easy readability of the code,widespread adoption, and offers scalability using most cluster and cloud computing platforms. snakePipes also makes use of Conda environments and the Bioconda platform, which allow hassle-free installation and upgrade of tools. Conda environments allow execution of tools avoiding dependency conflicts, and do not require root permissions to run. Due to a modular architecture, various tools are shared between workflows,which simplifies data integration since data from multiple technologies are processed using identical tool versions and genome annotations. The genome annotations and indices are shared by all workflows, and can also be generated directly via snakePipes, facilitating easy setup as well as integrative analysis. Finally, workflows in snakePipes employ extensive quality-checks and also produce reports using multiQC and R, that inform the user of processing and analysis results.

Apart from conventional processing steps such as mapping, counting and peak calling, workflows in snakePipes also include various downstream analysis. All workflows (except scRNA-seq workflow) optionally accept a sample information (tab-separated) file that can be used to define groups of sample. This allows comparative analysis such as differential gene or transcript expression analysis for the RNA-seq workflow, differential peak calling for ChIP-Seq workflow, differential open chromatin detection for ATAC-seq workflow, and detection of differentially methylated regions (either de-novo or on user-specified regions)

P-03

The FAIR data principles and their practical implementation in InterMine

D. Butano[1], J. Clark-Casey[1], S. Contrino[1], J. Heimbach[1], R. Lyne[1], J. Sullivan[1], Y. Yehudi[1] and G. Micklem[1]

[1]*Department of Genetics, University of Cambridge, Cambridge, United Kingdom*

Abstract

The FAIR Data Principles [1] are a set of guidelines which aim to make data findable, accessible, interoperable and reusable. The principles are gaining traction, especially in the life sciences. We will present our experience of the practical implementation of the FAIR principles in InterMine [2], a platform to integrate and access life sciences data. We will cover topics such as the design of persistent URLs, standards for embedding data descriptions into web pages, describing data with ontologies, and data licences.

Introduction

Science is generating ever more data, faster than ever before. Reliably storing and retrieving this data isn't enough; integration between different datasets, from different sources is becoming equally important. Wider adoption of the FAIR principles will make this process easier and facilitate data use by machine and humans.

InterMine is a platform to integrate and access life sciences data, providing flexible querying through a user-friendly web interface as well as RESTful web services [3] . Whilst InterMine comes with a core data model for common biological entities, and loaders for popular data sources and file types, different deployments can extend these components to publish any type of data.

InterMine is an established platform first released in 2006, and already includes some FAIR principles such as search and structured query functionalities, web services, and cross-references to other InterMine instances and resources. We will describe here how we are improving InterMine adherence to FAIR principles.

Generating persistent URLs for web pages

InterMine already has unique URLs to identify the report pages for biological entities, but these are based on internal InterMine IDs that change at every database build and are therefore not persistent.

To achieve data **findability** and **accessibility**, we have generated new URLs based on the InterMine class names combined with local IDs provided by the data resource providers. For example, in HumanMine, the URL of the report page for the protein MYH7_HUMAN, with UniProt accession P12883, will be *www.humanmine.org/protein:P12883*.

Describing data with ontologies

The InterMine system is based on a core data model, described in an XML file, which defines classes (the entities in the model) and the relationships between them. The core model can be

P-04

Collecting runtime metrics of genome analysis workflows by CWL-metrics

Tazro Ohta[1], Tomoya Tanjo[2], Osamu Ogasawara[3]

1. Database Center for Life Science, Joint Support-Center for Data Science Research, Research Organization of Information and Systems, Shizuoka 411-8540, Japan
2. National Institute of Informatics, Research Organization of Information and Systems, Tokyo 101–8430, Japan
3. DNA Data Bank of Japan, National Institute of Genetics, Research Organization of Information and Systems, Shizuoka 411-8540, Japan

Project Website: https://inutano.github.io/cwl-metrics/
Source Code: https://github.com/inutano/cwl-metrics
License: MIT

Portability of computational data analysis environment is largely improved by container virtualization technologies such as Docker [1] and workflow description frameworks represented by Common Workflow Language (CWL) [2]. To deploy an environment for data analysis workflows, researchers have to select an appropriate computational platform for the given application. To provide information to estimate the computational resource such as CPU or memory required by workflow execution, we developed CWL-metrics, a utility system of cwltool (the reference implementation of CWL) to collect runtime metrics of CWL workflows. The system summarizes resource usage of each step of workflow with the input parameters and the information of the host machine.

We demonstrated the performance comparison of RNA-Seq quantification workflows by using metrics captured by CWL-metrics [3,4]. The comparison analysis results recorded in Jupyter Notebook are published on GitHub [5]. The system is utilized in the new pipeline system being deployed on the high-performance computing platform of the DNA Data Bank of Japan (DDBJ) to collect the metrics to help administrators of the platform. The metrics information captured by CWL-metrics is also being used by the development of resource prediction algorithms. We also would like to present the progress of the new version of CWL-metrics currently under development, which will increase the coverage of supported workflow runners and container technologies including Singularity container [6].

1. Docker, https://www.docker.com
2. Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., ... & Scales, M. (2016). Common Workflow Language, v1. 0.
3. Ohta T, Tanjo T, Ogasawara O. Accumulating computational resource usage of genomic data analysis workflow to optimize cloud computing instance selection. bioRxiv; 2018. DOI: 10.1101/456756
4. Ohta T, Tanjo T, Ogasawara O. Accumulating computational resource usage of genomic data analysis workflow to optimize cloud computing instance selection. GigaScience; 2019. In press.
5. Code for CWL-metrics manuscript https://github.com/inutano/cwl-metrics-manuscript
6. Kurtzer GM, Sochat V, Bauer MW. Singularity: Scientific containers for mobility of compute. 2017. PloS one, 12(5), e0177459. DOI: 10.1371/journal.pone.0177459

Global Alliance for Genomics & Health

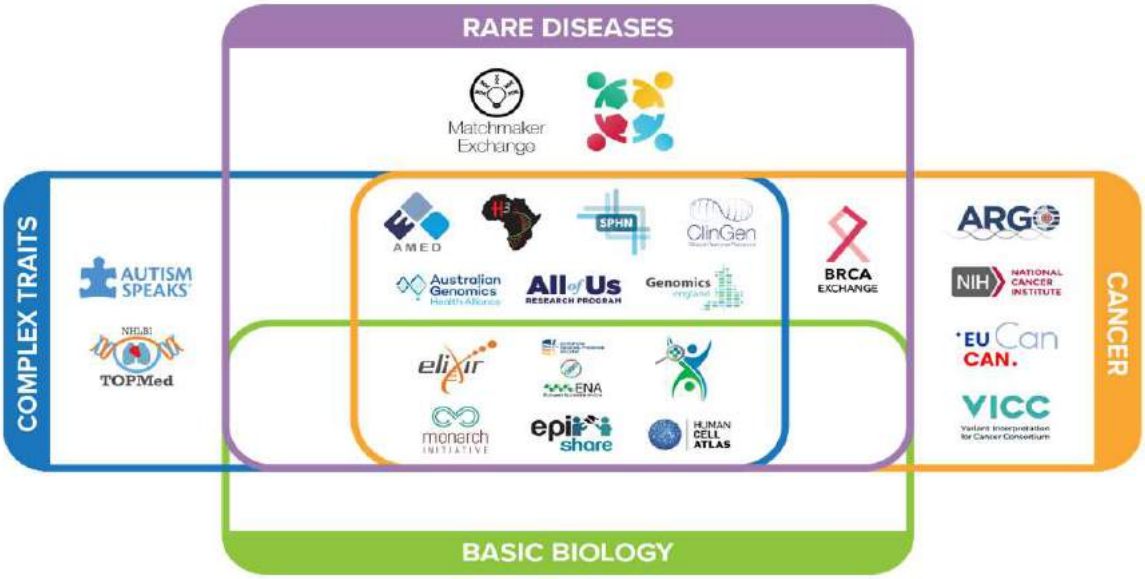
Collaborate. Innovate. Accelerate.

GA4GH: Developing Open Standards for Responsible Data Sharing

The Global Alliance for Genomics and Health (GA4GH) is creating frameworks and standards to enable the responsible, voluntary, and secure sharing of genomic and health-related data. This talk will introduce these specifications, the community based process in which they are developed, and show how you can contribute to this process.

GA4GH Development Process

There are benefits in being able to process large cohorts that range from improved understanding of basic biology to the areas of diagnosis and treatment of rare genetic disease and cancer. As genomic data collections are built outside of the research community technical and regulatory challenges are presented to researchers. By bringing together those building large data collections such as AllOfUS, the European Genome-Phenome Archive and GEnome Medical Alliance (GEM) Japan, and those working in specialist areas such as Matchmaker Exchange and ICGC-ARGO, GA4GH identifies and builds the open standards that are needed to cross bridges and boundaries in efforts to harness large cohorts of human genetic data. The development process is open and participation from members outside these projects are part of this cycle.



Creating a pluggable visualisation toolsuite with BlueGenes Tool API

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

[1]Department of Genetics, University of Cambridge, Cambridge, United Kingdom

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

BlueGenes is a browser-based open source analysis and visualisation tool for biological data, powered by InterMine web services. We present the BlueGenes Tool API, a tool specification that enables easy creation, distribution, and embedding of JavaScript-based biological data visualisations.

Background: BlueGenes is a ClojureScript-based user interface for InterMine databases, currently in pre-release testing phase. Clojure and Clojurescript offer advantages while developing web tools, as the same code can be compiled for the JVM or the browser depending on the intended uses. It also presents a challenge: most browser-based biological visualisation tools are written in JavaScript, and existing tool authors are unlikely to re-write their tools in another language. Fortunately we can use ClojureScript's ability to run JavaScript packages to easily embed or "wrap" JavaScript tools. By providing a clear specification, tool authors do not need to know any ClojureScript in order to embed their work in the BlueGenes environment.

Distribution: Javascript's existing package managers are popular and effective. BlueGenes tools use npm (node package manager, https://www.npmjs.com/) for distribution and installation, so installing and enabling a new compatible visualisation in a local install of BlueGenes is generally as simple as running `npm install my-tool-name`. BlueGenes tools are automatically tagged with the tag bluegenes-intermine-tool, facilitating findability.

Specification: We designed the specification to be flexible, with few minimum requirements - tool initialisation methods must have a fixed method signature that matches the package name, so it can be initiated automatically without additional configuration from a system administrator. All tools must include a config.json in the package root which specifies the type of data expected, e.g. gene, protein, etc., and whether they deal with a single entity (e.g. a protein feature viewer) or a list of entities (e.g. an interaction viewer). The complete specification is available from https://github.com/intermine/bluegenes/blob/d9a896964b80820ace9de885090ffd03fbc1673d/tools/docs/tool-api.md

Implementation: In BlueGenes, entity report pages (e.g. a report page for a single gene or a list of proteins) will automatically initialise and display data vis tools based on the config specified in a tool's config.json file.

In order to ensure the required files and properties are present in each new tool, we have created a Yeoman Generator - a command-line based tool which asks users a series of questions and creates the correct folder structure, initialising the files and folder structure based

ELIXIR Europe on the Road to Sustainable Research Software

Mateusz Kuzak
Dutch Techcentre for Life Sciences
ELIXIR Netherlands
the Netherlands
mateusz.kuzak@dtls.nl

Jen Harrow
ELIXIR Hub Wellcome Genome Campus
Hinxton, UK

Rafael C. Jimenez
ELIXIR Hub Wellcome Genome Campus
Hinxton, UK

Paula Andrea Martinez
ELIXIR Belgium

Fotis E. Psomopoulos
Institute of Applied Biosciences, Centre for Research and Technology Hellas
Thessaloniki, Greece

Allegra Via
ELIXIR Italy, National Research Council of Italy (CNR)
Institute of Molecular Biology and Pathology (IBPM)
Italy
allegra.via@uniroma1.it

***Index Terms*—training, Open Source, software guidelines, best practices, recommendations, Open Science, Reproducible Research, sustainability, FAIR**

I. Introduction

ELIXIR [1] is an intergovernmental organization that brings together life science resources across Europe. These resources include databases, software tools, training materials, cloud storage, and supercomputers. One of the goals of ELIXIR is to coordinate these resources so that they form a single infrastructure. This infrastructure makes it easier for scientists to find and share data, exchange expertise, and agree on best practices. ELIXIR's activities are divided into the following five areas Data, Tools, Interoperability, Compute and Training known as "platforms". The ELIXIR Tools Platform works to improve the discovery, quality and sustainability of software resources. Software Best Practices task of the Tools Platform aims to raise the quality and sustainability of research software by producing, adopting, promoting and measuring information standards and best practices applied to the software development life cycle. We have published four (4OSS) simple recommendations to encourage best practices in research software [2] and the Top 10 metrics for life science software good practices [3].

II. Four Simple recommendations

The 4OSS simple recommendations are as follows:

1) Develop publicly accessible open source code from day one. Start a project as open source from the very first day, in a publicly accessible, version controlled repository (e.g. github.com, gitlab.com and bitbucket.org). The longer a project is run in a closed manner, the harder it is to open source it later.
2) Make software easy to discover by providing software metadata via a popular community registry. Facilitate the discoverability of the open source software projects by registering metadata related to the software in a popular community registry (e.g. bio.tools [4]), making your source code more discoverable. Metadata might include information such as source code location, contributors, license, references and how to cite the software.
3) Adopt a license and comply with the licence of third-party dependencies. Provide instructions and guidelines for other projects and software to use, modify and redistribute the software and the source code. Adopt a suitable Open Source license, include it in a publicly accessible source code repository, and ensure the software complies with the licenses of all third party dependencies.
4) Have a clear and transparent contribution, governance and communication processes. Open sourcing your software does not mean the software has to be developed in a publicly collaborative manner. Although it is desirable, the OSS recommendations do not mandate a strategy for collaborating with the community. However projects should be clear and transparent about how to contribute to them as well as, their governance model, and their communication channels.

III. Building Sustainabilty

IIn order to encourage researchers and developers to adopt the 4OSS recommendations and build FAIR (Findable, Accessible, Interoperable and Reusable) software, best practices

P-08

Cellular Genetics Informatics support group: Nextflow and Jupyter on Kubernetes, Nextflow web interface

Anton Khodak, Stijn van Dongen, Vladimir Kiselev, Keiran Raine and Luca Barbon

Repository : https://github.com/cellgeni/nf-web, https://github.com/cellgeni/kubespray
License : GPL-3.0, Apache 2.0

We are presenting the results of the work of Cellular Genetics Informatics team from Wellcome Sanger Institute, UK. Our team provides efficient access to cutting-edge analysis methods for the Cellular Genetics programme. Our focus is on the development and operation of pipelines, tools and infrastructure for data analysis that support the programme's research goals.
For these purposes, we developed a reproducible and battle-tested Kubernetes-on-OpenStack setup using Kubespray with the primary use-cases of running Nextflow pipelines (custom RNA-seq and nf-core pipelines) and hosting a multiuser JupyterHub server with a custom image together with other web applications. It integrates data management platform iRODS, Lustre and GlusterFS filesystems. Nextflow pipeline performance has been benchmarked on Kubernetes on Openstack versus LSF scheduler on a high-performance cluster.
Another part of the work we are presenting is an open-source web application for running Nextflow pipelines on Openstack. It allows users to start pipelines by uploading data and filling in input parameters without any Nextflow knowledge, be notified once an analysis is finished and get the results uploaded to S3. Although it has been written for Openstack use-case, it allows plugging in a different backend for other cloud providers, such as GCE or AWS.

P-09

A method for systematically generating explorable visualization design spaces

Background. Stakeholders within public health can use the results of genomic analyses to establish practice guidelines and enact policies. Yet, these stakeholders vary in their abilities to interpret genomic findings and contextualize the results with other sources of data. Data visualization is an emergent solution to address interpretability challenges, but absent is a systematic and robust method to help identify the appropriate visualization to use in different contexts.

Methods. We have developed a systematic method for generating an explorable visualization design space, which catalogues visualizations existing within the infectious disease genomic epidemiology literature. Our method uses an automated literature analysis phase to establish *why* data were visualized, followed by a manual visualization analysis phase to establish *what* data were visualized and *how*. The literature analysis phase queried PubMed and used an unsupervised cluster analysis on article titles and abstracts to discover topic clusters that suggested why data were visualized. In order to ensure that we had a variety of data visualizations for further analysis, we sampled articles from across topic clusters and then extracted their figures. We then applied open and axial coding techniques, from qualitative research methods, to the sampled figures in order to iteratively derive taxonomic codes that described elements of each data visualization, thus enabling us to compare visualizations.

Results. We applied our method to a document corpus of approximately 18,000 articles, from which we sampled 204 articles for analysis. We added 17 articles manually for a final 221 articles that yielded 801 figures and 49 missed opportunity tables. These figures served as inputs to the visualization analysis phase and resulted in taxonomic codes along three descriptive axes of visualization design: chart types within the visualization, chart combinations, and chart enhancements. We refer to the collective complement of derived taxonomic codes as GEViT (Genomic Epidemiology Visualization Typology). To operationalize GEViT and the results of the literature analysis we have created a browsable image gallery (http://gevit.net), that allows an individual to explore the myriad of complex types of data visualizations (i.e. the visualization design space). Our analysis of the visualization design space through GEViT also revealed a number of data visualization challenges within infectious disease genomic epidemiology that future bioinformatics work should address.

Conclusions. Data visualization is powerful medium to help stakeholders better understand complex heterogenous data. By enumerating a visualization design space and empowering others to explore it, we enable a richer dialogue around the processes and practices for designing and evaluating contextually appropriate data visualizations.

Gallery: http://gevit.net/
Analysis Source Code: https://github.com/amcrisan/GEViTAnalysisRelease
Gallery Source Code: https://github.com/amcrisan/gevit_gallery_v2
Publication (Open Source): https://doi.org/10.1093/bioinformatics/bty832

Biotite: A comprehensive and efficient computational molecular biology library in Python

Patrick Kunzmann, Kay Hamacher

April 11, 2019

1 Introduction

A typical computational molecular biology workflow consists of combining different programs in order to reach the desired goal. Each software is usually made for a very specific purpose, like sequence alignment or secondary structure annotation. Manually converting between the required file formats and adjusting the input data and parameters for these programs can be unhandy for the user. Furthermore, such a workflow can be inefficient due to an overhead of file read/write operations. These problems can be overcome by shifting the workflow to comprehensive computational biology library in a easy-to-learn scripting language like Python.

Biopython [1] is such a comprehensive library, however, its foundation was almost 20 years ago. Hence, it largely lacks modern scientific programming standards in Python. *NumPy* [2], for example, is only sparsely used.

We would like to present the open source Python package *Biotite*. It is a modern and comprehensive computational molecular biology library in the spirit of *Biopython*. Through extensive usage of *NumPy*, most operations in *Biotite* are C-accelerated. Originally published in BMC Bioinformatics [3] in October 2018, the package is in continuous development. With the presentation at the BOSC 2019 we hope to extend the userbase of the package and attract more developers, who like to contribute new functionalities.

2 Library

Biotite stores sequence and structure data internally as *NumPy* `ndarray` objects. These `ndarray` objects are directly accessible to the user. Hence, *Biotite* can be used as flexible library to build software upon, removing the need to implement basic functionality like file parsers. Furthermore, the extensive application of *NumPy* renders most operations efficient via vectorization. Additionally, *Cython* [4] code is used in places where vectorization is not feasible.

The package is divided into four subpackages: `sequence`, `structure`, `application` and `database`.

The `sequence` subpackage contains utilities to handle biological sequences. The base type for all sequences is the `Sequence` class, that stores a sequence encoded as a `ndarray` of integers [3]. These objects can be used to perform DNA translation, subsequence searches, alignments, etc. Additionally, this subpackage provides functions for visualization of sequence related objects, e.g. alignments (Fig. 1A).

The `structure` subpackage revolves around handling macromolecular structures, ranging from single models (`AtomArray` class) to entire trajectories (`AtomArrayStack` class). Both, the `AtomArray` and the `AtomArrayStack`, internally store the atom coordinates and the atom annotations (chain ID, residue name, etc.) as separate `ndarray` objects. In addition to a high performance, this has the advantage that `AtomArray` and `AtomArrayStack` objects can be indexed like an `ndarray`: The index is simply propagated to the coordinates and annotations. This subpackage offers a large variety of analysis functions ranging from simple geometric measurements to the calculation of the solvent accessible surface area.

OpenBio-C: An Online Social Workflow Management System and Research Object Repository

Alexandros Kanterakis, Galateia Iatraki, Leuteris Koumakis, Konstantina Pitianou, Manolis Koutoulakis, Nikos Kanakaris, Nikos Karacapilidis, George Potamias

Repository : https://github.com/kantale/arkalos

Today there is a plethora of Open Source Workflow Management Systems (WMS) aiming at organizing research, automating analysis, discovering valuable Research Objects (ROs) and, ultimately, battling the reproducibility crisis. Despite the technical maturity of these tools and the efforts of communities like OBF to spread their use, we can roughly estimate that less than 1% of the published analysis that combine multiple ROs use any WMS. Besides, there are still important features that are in the best case partially supported by existing WMSs, thus limiting their applicability. In brief, these features are: (i) steep learning curve from non-IT experts, (ii) custom Domain Specific Languages, (iii) requirement for local installation, (iv) inability to cooperate with other WMS, (v) lack of rewarding to scientists that add content, (vi) lack of a single, browsable repository with easily downloadable and executable ROs that also contains usage statistics and resource requirements, and (vii) inability to rate and comment existing ROs. To remedy these issues, we present the first version of OpenBio-C, which is an online WMS, workflow composer, RO repository, and Q&A site, targeting all science enthusiasts. It requires no IT knowledge and supports RO import and export from a variety of existing WMS.

CWLab: an open-source, platform-agnostic, and cloud-ready framework for simplified deployment of the Common Workflow Language using a graphical web interface

Kersten Henrik Breuer, Yoann Pageaud, Yassen Assenov, Reka Toth, Christoph Plass, Pavlo Lutsik

Repository : https://github.com/CompEpigen/CWLab

The Common Workflow Language (CWL) allows to wrap and link up bioinformatic software in a standardized and portable way. However, setting up and operating a CWL-based workflow management system can be a labor-intensive challenge for data-driven laboratories. To this end, we developed CWLab: a framework for simplified, graphical deployment of CWL.

P-13

pdb-tools: a dependency-free cross-platform swiss army knife for PDB files.

João Rodrigues, João Teixeira, Mikael Trellet, Alexandre Bonvin

Repository : https://github.com/haddocking/pdb-tools
License : Apache License

The pdb-tools are a collection of Python scripts for working with molecular structure data in the Protein Data Bank (PDB) format. The tools allow users to easily and efficiently edit and validate PDB files as well as convert coordinate data to and from the now-standard mmCIF format. Moreover, their simple and consistent command-line interface makes them particularly adequate for non-expert users. All tools are implemented in Python, without external dependencies, and are freely available under the open-source Apache License at https://github.com/haddocking/pdb-tools and on PyPI.

NGLess: a domain-specific language for NGS analysis (NG-meta-profiler as a case study)

Luis Pedro Coelho (coelho@fudan.edu.cn)[123], Renato Alves[14], Paulo Monteiro[5], Jaime Huerta-Cepas[16], Ana Teresa Freitas[5], Peer Bork[1789]

Project Website: https://ngless.embl.de
Source Code: https://github.com/ngless-toolkit/ngless (License: MIT)

Linking different programs is an integral part of bioinformatics, which is most often performed using either traditional programming scripting languages or, increasingly, workflow-engines that coordinate calling multiple programs. Our hypothesis was that a domain-specific language could form the basis of a better tool for the specific problem of processing next-generation sequencing (NGS) data. The resulting language, NGLess, enables the user to work with abstractions that are closer to the problem domain and we show how this enhances usability and correctness, while implementing scientific best practices. Results computed with NGLess are independent of the environment in which they are generated and, thus, perfectly reproducible.

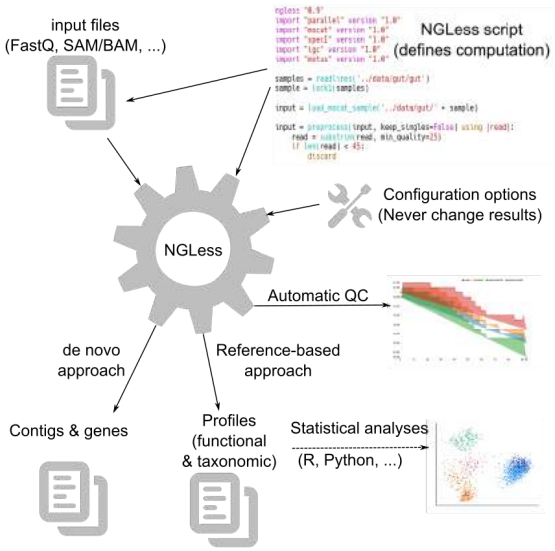


Figure 1. Cartoon depiction of NGLess' approach: A script in the NGLess language defines the computational pipeline, which defines how the outputs relate to the input data, while configuration options can provide accessory information (e.g., temporary file storage) which do not change the results.

[1] European Molecular Biology Laboratory, Heidelberg, Germany; [2] Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, Shanghai, China; [3] Key Laboratory of Computational Neuroscience and Brain-Inspired Intelligence (Fudan University), Ministry of Education, China; [4] Collaboration for joint PhD degree between EMBL and Heidelberg University, Faculty of Biosciences; [5] INESC-ID, Instituto Superior Técnico, University of Lisbon, Portugal; [6] Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid (UPM), Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria (INIA), Madrid, Spain; [7] Max Delbrück Centre for Molecular Medicine, Berlin, Germany; [8] Molecular Medicine Partnership Unit, University of Heidelberg and European Molecular Biology Laboratory, Heidelberg, Germany; [9] Department of Bioinformatics, Biocenter, University of Würzburg, Würzburg, Germany.

P-15

OmicsSIMLA: A multi-omics data simulation tool for complex disease studies

Ren-Hua Chung, Chen-Yu Kang

Repository : https://omicssimla.sourceforge.io/
License : GPL 2.0

Multi-omics analysis incorporating multiple types of omics data has become important in complex disease studies. Many multi-omics analysis methods have been developed. However, only a few tools for simulating multi-omics data are available. We developed OmicsSIMLA, which simulates genetic data including SNPs and CNVs, methylation data based on bisulphite sequencing, RNA-seq and normalized protein expression data. OmicsSIMLA also simulates meQTLs (SNPs influencing methylation), eQTLs (SNPs influencing gene expression), and eQTM (methylation influencing gene expression). Furthermore, a disease model can be specified to model the relationships between the multi-omics data and the disease status. We used OmicsSIMLA to simulate a multi-omics dataset with a scale similar to the ovarian cancer data from the TCGA project. The simulated data included 2,884 focal CNVs, 2,753 CpGs on chromosome 1, gene expression levels for 12,004 genes, and protein expression levels for 200 genes in 500 samples with short-term and long-term survival. A neural network-based multi-omics analysis method was applied to the real and simulated ovarian cancer data, and similar results such as the classification rate were observed. The results demonstrated that OmicsSIMLA can simulate realistic multi-omics data and will be useful to generate benchmark datasets for comparisons among multi-omics analysis methods.

P-16

Forome Anfisa – an Open Source Variant Interpretation Tool

Bouzinier M[1,2], Trifonov Sl[2], Krier J[1,2], Etin D[2], Olchanyi D[2], Kargalov A[2],Ghazani AA[1], Sunyaev SR[1,3]

Forome Anfisa is a highly customizable suite of software for downstream genetic analysis, clinical variant interpretation, curation, and collaboration. It supports 3 real-life scenarios for effective WES/WGS and panel of genes variants analysis:

- the traditional clinical workflow for variant curation based on predefined guidelines;
- a workflow for design and development new guidelines for variant interpretation;
- a collaboration in variant interpretations.

Anfisa is developed under the Apache 2.0 license and is available on GitHub in the Forome Association repository: https://github.com/ForomePlatform/anfisa.

Anfisa is a modular system with three main components and a number of support modules. Main components are:

- annotation pipeline,
- backend database,
- frontend user interface.

Annotation pipeline starts with Ensembl VEP [1] and then adds annotations based on functional analysis, population genetics, clinical knowledge, epigenetics, etc. This is done by traversing databases such as gnomAD, ClinVar [2], HGMD [3], including results from spliceAI [4] and other sources. The backend stores the data in Druid Open Source OLAP [5] and metadata, such as user environment, curation notes and preferences are stored in MongoDB. The frontend is implemented using Vue JavaScript Framework [6] and Bootstrap toolkit [7]. Annotation Pipeline and Backend provide public REST API and technically can be used in standalone mode, integrated with other genomics tools and EMRs.

Anfisa is designed to forge collaboration between people with different goals and skills, and with different organizational roles, from treating physicians to clinical geneticists to researchers and bioinformaticians. The system is designed to efficiently operate with small and large genomic datasets. It is transparent for users.

A patient case consisting of the panel of genes loads several thousands variants directly into the main UI and offers filtering capabilities to quickly narrow the list down to a few dozens. It is then a workable amount of data to review manually.

A case with WES/WGS data loads into the advanced filtering tool which would help users creating a custom workspace. The workspace is a combination of the various available filters and more complex rules (clinical guidelines), used by the user to reach a reasonable and meaningful amount of the variants to work in the manual mode. When the workspace is defined and applied to the case data, a list of variants is loaded into the main UI.

We have implemented two distinct scenarios to address the variety of users goals of the complexity of clinical research tasks:

[1] Division of Genetics, Brigham & Women's Hospital
[2] Forome Association
[3] Department of Biomedical Informatics, Harvard Medical School

Introduction

The rapid growth in the amount of biomedical literature makes it impossible for humans alone to extract and exhaust all of the useful information it contains. Automated extraction of disease names enables, for example, the integration with other data types and the generation of new hypotheses by combining facts that have been extracted from several sources.[1]

Here, we will use the open source KNIME Analytics Platform to create a model that learns disease names in a set of documents from the biomedical literature. The model has two inputs: an initial list of disease names and the documents. Our goal is to tag disease names that are part of our input as well as novel disease names. Hence, one important aspect of this project is that our model should be able to autonomously detect disease names that were not part of the training.

To do this, we create a workflow (see Fig 1.) that automatically extracts abstracts from PubMed and uses these documents (the corpus) to train a machine learning model starting with an initial list of disease names (the dictionary).

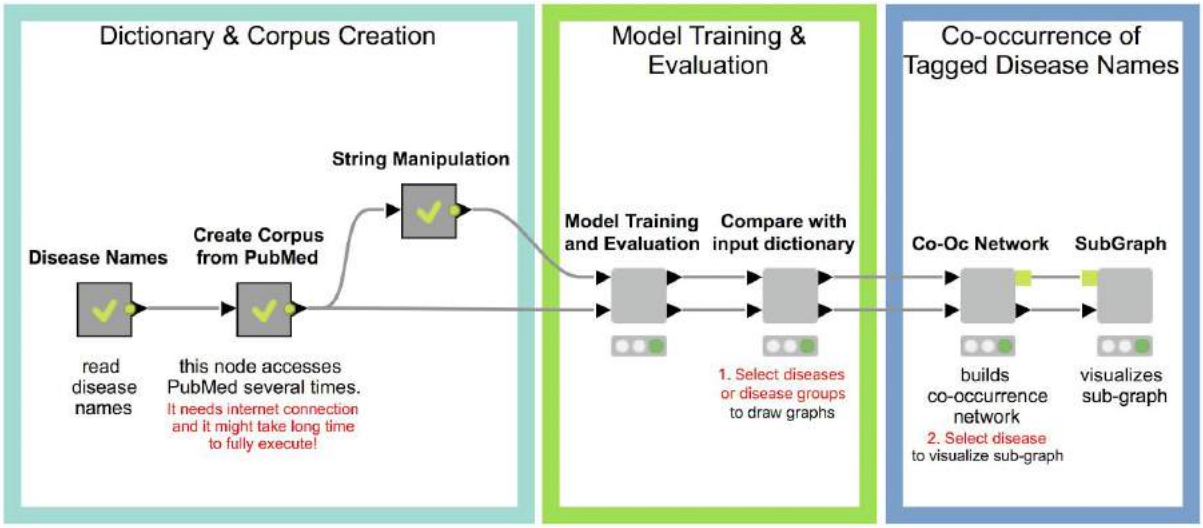


Figure 1: Overview of the workflow to automatically extract disease related information from biomedical literature. First, the literature corpus as well as the dictionary of known disease names are gathered. Next, the model is trained and evaluated. Last, the results are investigated in a network approach.

For the initial dictionary we download phenotypes (diseases and traits) that are associated to genes or variants from Ensembl Biomart. We first split our collected documents into a training (10%) and a test set (90%) and train a StanfordNLP NE Learner in KNIME using the training data. Stanford NLP is a Natural Language Processing software[2], licensed under the GNU General Public License. The StanfordNLP NE Learner creates a Conditional Random Field (CRF) model based on documents and entities in the dictionary that occur in the documents.

We evaluate the resulting model using documents in the test set that were not part of the training. We achieve a Precision of 0.966, Recall of 0.917, F1 of 0.941. Additionally, we test whether the model can extract new information by comparing the detected disease names to our initial dictionary.

---

[1] (2006, February 1). Literature mining for the biologist: from information retrieval to ... - Nature. Retrieved July 12, 2018, from https://www.nature.com/articles/nrg1768

[2] "The Stanford CoreNLP Natural Language Processing Toolkit." https://nlp.stanford.edu/pubs/StanfordCoreNlp2014.pdf. Accessed 10 Apr. 2019.

WhatsHap: fast and accurate read-based phasing

Marcel Martin, Murray Patterson, Jana Ebler, Peter Ebert, Sven Schrinner, Rebecca Serra Mari, Shilpa Garg, Marco Dell'Acqua, Sarah O. Fischer, Nadia Pisanti, Gunnar W. Klau, Alexander Schönhuth, Tobias Marschall

Repository : bitbucket.org/whatshap
License : MIT

The relationship between parents and their offspring is fundamentally anchored in the genome: we inherit one copy of each chromosome from our mother and another copy from our father. These two versions of the genome are similar but not identical. Reconstructing these two individual copies (called haplotypes) can offer important insights in areas as diverse as population genetics, evolutionary genetics, or personalized medicine. We present WhatsHap, a production-ready bioinformatics software suite for highly accurate haplotyping based on the latest sequencing technologies. WhatsHap has a large user base (>250 downloads/week), is under active development, comes with a multitude of features including the use of pedigree information, the ability to determine genotypes, and the application in de novo assembly settings. WhatsHap has been designed as an easy to use application, sporting comprehensive documentation and a continuously tested code base to ensure reliability. WhatsHap is freely available under the terms of the MIT license at bitbucket.org/whatshap

Recommendations and guidelines for tumor heterogeneity quantification using deconvolution of methylation data: data challenges as a tool for benchmarking studies

.

Clémentine Decamps, Florian Privé, Michael Blum, Magali Richard*

Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMC-IMAG, F-38000 Grenoble
* Corresponding authors: magali.richard@univ-grenoble-alpes.fr

Consortium authors: Blum M., Broseus, L., Amblard E., Houseman A., Kaoma T., Nazarov P., Achard S., Bergmann F., Permiakova O., Scherer M., Blum Y., Durif G., Jonchere V., Nguyen, N., Jedynak P., Rolland M., de Jong E., Bottaz-Bosson G., Markowski J., Melnykova A., Jumentier B., Lurie E., Spill Y., Lutsik P., Merlevede J., Chuffart F., Devijver E., Feofanov V.,Gallopin M., Bacher R., Decamps C,. Privé F., Richard M.

Introduction.

Since the recent development of high-throughput sequencing technologies, cancer research has focused on characterizing the genetic and epigenetic changes that contribute to the disease. However, these studies well often neglect the fact that tumours are constituted of cells with different identities and origins (cell heterogeneity). Quantification of tumor heterogeneity is of utmost interest as multiples components of a tumor are key factors in tumor progression and response to chemotherapy.

Advanced microdissection techniques to isolate a population of interest from heterogeneous clinical tissue samples are not feasible in daily practice. An alternative is to relate on computational deconvolution methods that infer cell-type composition [1-3]. However, state-of-the-art reference free computational methods, based on global DNA methylation of surgical specimens, do not account for confounding factors, such as age and sex, which are a major source of variability. Better bioinformatic tools to assess the different cell populations from bulk samples are therefore urgently needed. Their development and efficacy assessment have been impaired by the lack of dedicated benchmarking studies.

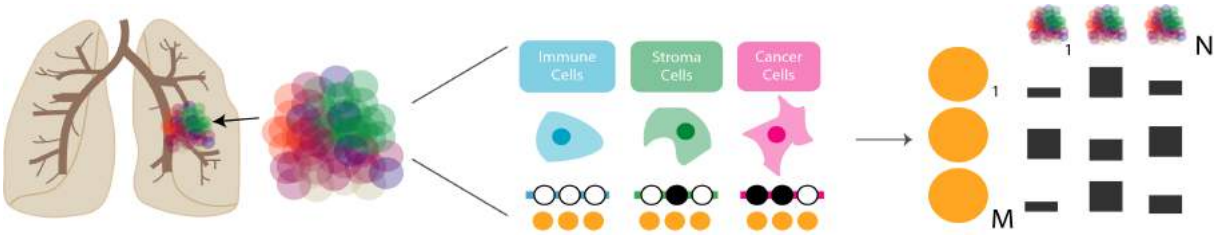


Fig.1 Illustration of the data challenge. The D matrix was simulated as following: $D = AT$ with : $D_{ij} = \sum_{k=1}^{K} A_{kj} T_{ki} D_{ij}$ the methylation value associated to the site j, $j \in \{1, \dots, M\}$ of the tumour i , $i \in \{1, \dots, N\}$, T_{kj} the methylation at site j in each cell type k and A_{ki} to the fraction of each cell type k in the tumour i. We used real cell types methylation public data to initiate the T matrix, and we subsequently added confounders effect (inspired from real lung cancer patient history). The A matrix was simulated using Dirichlet distribution.

Data challenge.

To address this benchmarking issues, we organized a data challenge dedicated to the quantification of intra-tumor heterogeneity on lung cancer methylation data. The aim of the challenge was to estimate cell types and proportion in simulated biological samples, based on averaged DNA methylation and full patient history. We invited participant to explore various statistical methods for source separation/deconvolution analysis (Non-negative Matrix Factorization, Surrogate Variable Analysis, Principal component Analysis, Latent Factor

P-20

Sustainability of legacy software - Making the antiSMASH genome mining tool ready for the future

Kai Blin, Sang Yup Lee, Marnix Medema, Tilmann Weber

Repository : https://github.com/antismash/antismash
License : GNU AGPL 3.9

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. 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.
Genome mining can be used to assist life scientists in the discovery of new drug leads. One such tool is the Open Source software antiSMASH. Since its initial release in 2011, it has become one of the most popular tool in the area of antibiotics discovery, combining comprehensive analyses with an easy to use web UI. We have recently released version 5 of antiSMASH, which next to adding new featuers updated the whole code base to Python 3. This poster will present some challenges encountered when porting a large legacy code base to a new language and how we solved those challenges in our case.

Methrix: An R package for efficient processing of bedGraph files from large-scale methylome cohorts

Anand Mayakonda, Maximilian Schönung, Joschka Hey, Yoann Pageaud, Christoph Plass, Pavlo Lutsik, Reka Toth

Repository : https://github.com/CompEpigen/methrix
License : MIT

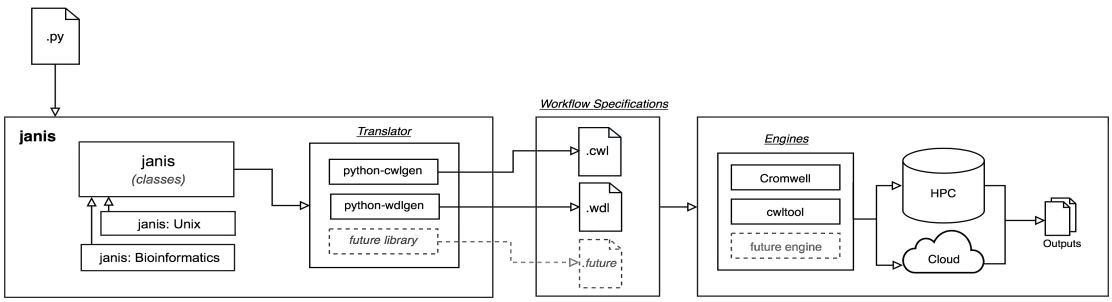
DNA methylation is an epigenetic modification associated with transcriptional regulation and establishment of cellular identity. Whole genome bisulfite sequencing (WGBS) has become the gold standard for measuring DNA methylation. WGBS data processing often results in bedGraph files containing methylation and coverage statistics. Downstream analysis requires summarization of these files into methylation/coverage matrices whereby the dimensions rapidly increase along with the number of samples. However, currently available tools are limited by file format specifications, speed/memory requirements. To overcome these limitations, we have developed an R package called methrix which provides a fast and efficient solution for processing WGBS data. Core functionality of methrix includes a universal bedGraph reader which handles missing reference CpG sites, annotates and collapses strands - while being fast and memory efficient. The methrix-object is an extension to the Bioconductor SummarizedExperiment class thereby inheriting its core modules. Additionally, several methods are offered for downstream processing, including functions for data visualization, and an interactive html report generation. Methrix interacts with the popular bsseq-package thereby providing a faster pre-processing of data for existing DNA methylation analysis pipelines. In conclusion, methrix addresses the existing limitations by offering a resource efficient way of analysing WGBS data.

"Janis: An open source tool to machine generate type-safe CWL and WDL workflows"

The rapid development of Next Generation Sequencing (NGS) in recent years has enabled the generation of large volumes of data, which will typically be analysed through a series of bioinformatics tools, often referred to as a pipeline or workflow. With the increasing number of bioinformatics tools and workflow systems available to analyse these data, we face a problem in sharing and reproducing our work. Addressing this issue, there are ongoing global efforts to improve the reproducibility and portability of bioinformatics pipelines. Projects such as Common Workflow Language (CWL) aim to standardise how workflows are being specified, while workflow execution engines such as Toil and Cromwell provide many useful features to run pipelines across high performance computing (HPC) and cloud infrastructures. However, there is often debate on whether to adopt CWL, Workflow Definition Language (WDL) or other competing standards. CWL provides rigid, easy-to-parse specifications and is supported by variety of engines, but is considered more difficult to write, while other standards such as WDL offer more features and an easier learning curve but are tightly coupled to a specific engine such as Cromwell.

To address this issue, we have created Janis[1], an open source tool to machine generate CWL and WDL workflows. It is designed to assist in building standardised workflows via a translation mechanism that generates validated workflow specifications (CWL, WDL or both) as output. These translated workflow acts as a 'transport layer' that can be shared and executed using any workflow execution engine that supports the selected specifications. Janis also offers input and output type checking during workflow construction to connect steps in pipelines and enforce the input requirements of executed tools. This is especially important when dealing with those that generate or expect secondary files. For example, reference genome file in fasta format is often associated with various index files.

The diagram below illustrates Janis' architecture and how the translated workflows are used to run bioinformatics analysis across HPC and cloud infrastructure.



Run Scanner: a tool for monitoring sequencer runs and accessing run information

Heather Armstrong, Dillan Cooke, Andre Masella, Alexis Varsava, Morgan Taschuk

Repository : https://github.com/miso-lims/runscanner
License : GPL v3

As sequencing runs become faster and instruments can be run more frequently, data analysis can similarly be made more efficient by automating run monitoring. We have developed an application called Run Scanner to monitor sequencer run output directories and process run metadata (information about the run that excludes sequence data) from Illumina, PacBio, and Oxford Nanopore instruments. The run metadata is presented on a web server in both user-readable and machine-readable ways. Basic run metadata is presented in a standardized way for all modern sequencing platforms. Additional per-cycle metrics, which complement Illumina's BaseSpace tools, are added to Illumina runs for all instruments from the HiSeq 2000 to the NovaSeq 6000. In addition to being accessible to users, Run Scanner data can also be queried by a variety of software consumers. The Run Scanner's data can enhance lab workflows by updating information in a LIMS; can be sent to an ETL data integration to be queried for reports; and can provide valuable information to automated bioinformatics pipelines. Run Scanner has decreased our lab's workload, increased our reporting capabilities, and dramatically decreased our time from run completion to analysis initiation. It is open source and freely available online: https://github.com/miso-lims/runscanner

Ada Discovery Analytics: All-in-One Data Platform for Clinical and Translational Medicine with Scalable Machine Learning

Peter Banda, Sascha Herzinger, Venkata Pardhasaradhi , Reinhard

Repository : https://ada-discovery.org
License : Apache 2.0

Ada is a performant and highly configurable system for secured integration, visualization, and collaborative analysis of heterogeneous data sets, primarily targeting clinical and experimental sources. Ada's main features include a convenient web UI for an interactive data set exploration and filtering, and configurable views with widgets presenting various statistical results, such as, distributions, scatters, correlations, independence tests, and box plots. The platform offers several types of data set imports and transformations as well as an industry-level machine learning module powered by the scalable Spark ML library, which provides many classification, regression, clustering, and time-series processing routines at a fingertip. Furthermore, Ada facilitates robust access control through LDAP authentication and an in-house user management with fine-grained permissions. The main instance of Ada has served as a key infrastructural backbone of NCER-PD project (https://ada.parkinson.lu), which focuses on improving the diagnosis and stratification of Parkinson's disease by combining detailed clinical and molecular data of patients to develop novel disease biomarker signatures, mainly within Luxembourg. Ada is an open-source project with a web site available at https://ada-discovery.org.

P-25

Fake it 'til You Make It: Open Source Tool for Synthetic Data Generation to Support Reproducible Genomic Analyses

Abstract:

The lack of readily accessible large scale public genomic data sets currently limits the reproducibility of published biomedical research to a subset of authorized users. Tool developers, educators, journal editors and researchers alike are affected by the lack of open access genomic datasets appropriate for reproducing biologically meaningful analysis at scale. We will present a prototype pipeline that promotes reproducible analysis by making it easy to generate publicly shareable custom synthetic datasets. The prototype workflow links existing tools into a consolidated community resource for generating synthetic data cheaply and efficiently. We will demonstrate how to use this workflow on Broad Institute's open access Terra platform, to reproduce someone else's analysis and make your own work reproducible. The workflow, as written, is portable to any cloud platform that runs the Cromwell Engine, an open source scientific Workflow Management System.

Case Study:

We presented the first prototype of this workflow at American Society of Human Genetics(ASHG) 2018 as part of a template for reproducible research. The workflow was used to generate the data needed to reproduce work by Matthieu Miossec and collaborators described in a biorXiv preprint titled "Deleterious genetic variants in *NOTCH1* are a major contributor to the incidence of non-syndromic Tetralogy of Fallot" (ToF). In the original study, the authors analyzed high-throughput exome sequence data from 829 cases and 1252 controls, identifying 49 rare deleterious variants within the *NOTCH1* gene that appeared associated with this congenital heart disease. Other researchers had previously identified *NOTCH1* in families with congenital heart defects, including ToF; however Miossec *et al*. were the first to scale variant analysis of ToF to nearly a thousand case samples and show that *NOTCH1* is a significant contributor to ToF risk. Preprint: https://www.biorxiv.org/content/10.1101/300905v1. Final publication: https://doi.org/10.1161/CIRCRESAHA.118.313250

Overall Approach:

We used information from the preprint and its Supplemental Materials to reconstruct the main phases of the work, including Data Input, Processing and Analysis (Figure 1). For the Data Input phase, we created a synthetic dataset to get around the lack of appropriate public data at the time of publication. For the Processing Phase, we applied a variant discovery workflow that we judged equivalent to the original study. For the Analysis phase, we collaborated with Dr. Miossec to reimplement them in two parts: the prediction of variant effects as a workflow in WDL (Workflow Description Language) and the clustering analysis as R code in a Jupyter notebook. We did all the work in the Broad Institute's Terra platform - a cloud-native platform for biomedical researchers to access data, run analysis tools, and collaborate.

The Kipoy repository: accelerating the community exchange and reuse of predictive models for genomics

Machine learning models trained on large-scale genomics datasets hold the promise to be major drivers for genome science. However, lack of standards and limited centralized access to trained models have hampered their practical impact. To address this, we present Kipoi, an initiative to define standards and to foster reuse of trained models in genomics[1]. The Kipoi repository currently hosts over 2,000 trained models from 21 model groups that cover canonical prediction tasks in transcriptional and post-transcriptional gene regulation (Fig. 1). The Kipoi model standard enables automated software installation and provides unified interfaces to apply models and interpret their outputs. Use cases include model benchmarking, variant effect prediction (Fig. 2), transfer learning and building new models from existing ones (Fig. 3). By providing a unified framework to archive, share, access, use, and extend models developed by the community, Kipoi will foster the dissemination and use of ML models in genomics.

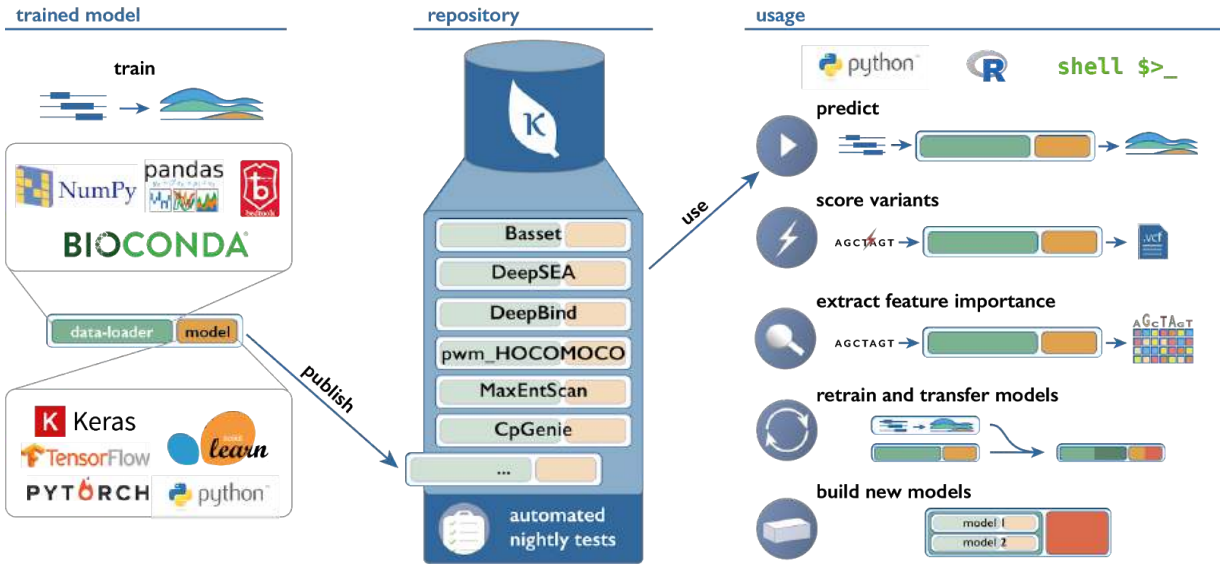


Figure 1 Overview. Kipoi (https://kipoi.org) defines an API for data-loaders and predictive models. Data-loaders translate genomics data types into numeric representation and enforce that all models can be applied to standard file format (fasta, bed, vcf, etc.). Kipoi models can be implemented using a broad range of ML frameworks. The models are automatically versioned, nightly tested and systematically documented with examples for their use. They can be accessed through unified interfaces from python, R, and command line. All models and their software dependencies get installed in a fully automatic manner. Kipoi streamlines the application of trained models to make predictions on new data, to score variants stored in the standard genetic variant file format, and to assess the effect of variation in the input to model predictions (feature importance score). Moreover, Kipoi models can be adapted to new tasks by either retraining them, or by combining existing ones.

CViTjs: Dynamic Whole Genome Visualisation

Andrew Wilkey, Ethalinda Cannon, Anne Brown

Repository : https://github.com/LegumeFederation/cvitjs
License : MIT

A whole genome view of genomic or genetic features that can can lead to new insights. For example, gene density compared with specific repeat families, regions of higher heterozygosity in mapping populations, and distribution of BLAST hits across a genome assembly are all cases where the researcher could benefit from a genome-wide view. CViTjs is an interactive JavaScript implementation of the original Chromosome Visualization Tool (CViT), which was written in Perl. Expanded from the original tool, CViTjs has been generalized to display any sort of feature than can be located on one or more backbones (typically chromosomes or linkage groups), in any linear coordinate system. Features displayed are interactive and customizable from a variety of configuration options. In order to increase portability, the tool is written to take advantage of HTML5 canvas, making it usable in across most web-browsers in a stand-alone or embedded context.

P-28

Dockstore: Enhancing a community platform for sharing cloud-agnostic research tools

Denis Yuen[1], Louise Cabansay[2], Charles Overbeck[2], Andrew Duncan[1], Gary Luu[1], Walt Shands[2], Natalie Perez[2], David Steinberg[2], Cricket Sloan[2], Brian O'Connor[2], Lincoln Stein[1]

[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

Project Website : https://dockstore.org/

Source Code : https://github.com/ga4gh/dockstore/

Background

Dockstore was created in response to the many challenges faced during the PCAWG (Pan-Cancer Analysis of Whole Genomes) study—an international collaboration to identify common patterns of mutation in more than 2,800 cancer whole genomes from the International Cancer Genome Consortium. The study involved fourteen highly heterogeneous computing environments that were not only geographically distributed, but spanned different cloud and HPC machines that encompassed both academic and commercial varieties. The scale and complexity of modern biomedical science efforts like these have driven a rethink of bioinformatics infrastructure to leverage big data, containerization, and cloud technologies that increase the mobility, interoperability, and reproducibility of research. To address these goals, we continued extending and developing Dockstore for use by the wider research community. This report highlights the key updates to the platform since its 1.2.5 release last presented at BOSC in 2017, as well as the future work going forward.

Platform

Dockstore is a platform for sharing Docker-based resources that allow bioinformaticians to bring together tools and workflows into a centralized location.

By packaging software into portable containers and utilizing popular descriptor languages, Dockstore standardizes computational analysis, making workflows precisely reproducible and runnable in any environment that supports Docker. Supported descriptors now include Nextflow in addition to the Common Workflow Language (CWL) and Workflow Description Language (WDL). These descriptor documents and test parameter files can now be stored directly on Dockstore.org or through external registries like GitHub, Bitbucket, Quay.io, and Docker Hub.

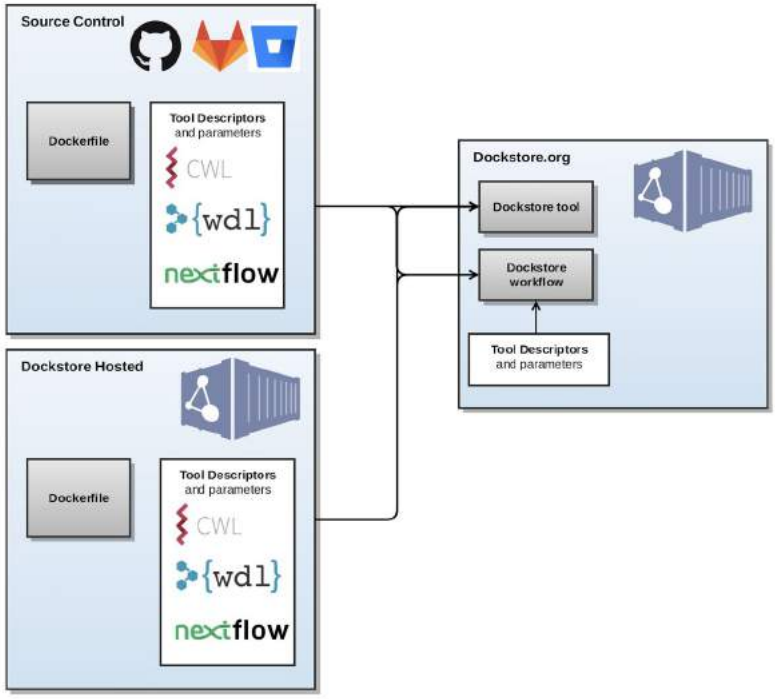


Figure: Overview of hosting for the Dockstore platform.

Integrations

We have combined hosting features with integrated deployment to cloud platforms and analysis environments. By registering workflows on Dockstore, developers can now provide the ability for users to run their workflows directly through a variety of launch-with partners like FireCloud/Terra, DNAnexus, and DNAstack. They can also share and launch workflows programmatically on compatible GA4GH WES-compatible platforms. When developing workflows locally, a handy Dockstore CLI also includes implementations for file-provisioning that support a variety of protocols including HTTP(S), S3, GS, FTP, ICGC Score, and a plugin that resolves the location of files using the GA4GH Data Registry Service (DRS) standard.

Disq, a library for manipulating bioinformatics sequencing formats in Apache Spark.

Code repository: https://github.com/disq-bio/disq
Software license: The MIT License (MIT)
Software license in code repository: https://github.com/disq-bio/disq/blob/master/LICENSE.txt

ADAM and GATK have independently developed parallel and distributed genomic applications on Apache Spark.

To access flat file formats such as BAM, CRAM, SAM, and VCF, both depend on the htsjdk library, which provides low-level codecs, and the Hadoop-BAM library, which extends these for parallel and distributed access.

Hadoop-BAM was found to have correctness (invalid BAM file splits, leading to corrupt read data) and performance (sequential implementation of some parallelizable tasks) issues. The Spark-BAM project demonstrated these issues could be addressed, and developed a comprehensive benchmark.

Thus members of the ADAM, Hadoop-BAM, htsjdk, GATK, Spark-BAM, and ViraPipe projects identified an opportunity to collaborate on a replacement library. Discussion between collaborators began virtually, then in-person at OpenBio Winter Codefest 2018 in Boston, and continued at GCCBOSC Collaboration Fest 2018 in Portland. A new project Disq was started in 2018, and has since made at least three releases (most recently version 0.3.0, released 19 March 2019).

Benchmarks show that Disq is faster and more accurate than Hadoop-BAM, and at least as fast as Spark-BAM.

Disq also adds significant new features, such as support for writing sharded files for efficiency, for taking advantage of index files while reading (e.g. .sbi index files to find splits between BAM records, .crai index files to find record boundaries in CRAM files), and for writing index files where appropriate.

In addition to unit tests, Disq includes integration tests that run against real-world files (multi-GB in size). SAMtools and BCFtools are used to verify files written with Disq can be read successfully.

Disq has been incorporated into ADAM and GATK, and will provide a convenient venue for further collaboration between those project teams. We also welcome new collaborators seeking correct and performant access to flat file formats on Apache Spark.

The Carpentries builds global capacity for conducting efficient, open, and reproducible research. We train and foster an active, inclusive, diverse community of learners and instructors that promotes and models the importance of software and data in research. We collaboratively develop openly-available lessons and deliver these lessons using evidence-based teaching practices.

Within The Carpentries, Data Carpentry is a lesson program that focuses on novices and teaches data skills through domain-specific lessons centered around a dataset, teaching two-day hands-on workshops. Our Data Carpentry Genomics lessons focus on the core skills throughout the genomics data analysis lifecycle, from data and project organization to analysis and visualization. Using open data from a published analysis of the evolution of bacterial genomes over 50,000 generations (Tenaillon et al 2016), we cover the following material:

- Project organization and management - How to structure metadata, organize and document genomics data and bioinformatics workflow, and access data on the NCBI sequence read archive (SRA) database.
- Introduction to the command line - How to navigate file system, create, copy, move, and remove files and directories, and automate repetitive tasks using scripts and wildcards.
- Data wrangling and processing - How to use command-line tools to perform quality control, align reads to a reference genome, and identify and visualize between-sample variation.
- Introduction to cloud computing for genomics - How to work with Amazon AWS cloud computing and how to transfer data between local computer and cloud resources.
- Intro to R and RStudio for Genomics - How to use R to analyze and visualize between-sample variation.

The goal of the workshop is to introduce learners to the concepts and tools they need to get started with the analysis of genomics data and learn best practices for reproducibility. More than 20 people have contributed to the development of this curriculum, and the workshop has been taught 12+ times in the last year. A recent update to the curriculum to reflect current sequencing technologies and tools, has projections for even more workshops this year.

As part of our workshop activities, we ask participants to complete pre- and post-workshop surveys. The results of these surveys allow us to ensure that the workshop took place in the positive learning environment we aim to create, and to evaluate how the workshop impacts learners' skills and confidence.

Initial feedback shows that workshops are well-received, with a median recommendation of 96% and surveys show that learners report significant confidence gains in using these approaches and applying them to their work. While not specific to Genomics workshops, Data Carpentry workshops generally show ~20% increase in confidence after just the two day workshop (Fig. 1).

Epiviz File Server - Query, Compute and Interactive Exploration of data from Indexed Genomic Files

The feasibility and reducing costs of running sequencing experiments has led to the generation of large amounts of genomic data. Genomic data repositories like The Cancer Genome Atlas (TCGA), Encyclopedia of DNA Elements (ENCODE), Bioconductor AnnotationHub and ExperimentHub etc., provide public access to large amounts of genomic data as files. Researchers often download a subset of data from these repositories and perform their data analysis. As these data repositories become larger, researchers often face bottlenecks in their data analysis and exploration. Increasing data size requires longer time to download, pre-process and load files into a database to run queries efficiently. Currently available genome browsers fall into two broad categories. One that uses a database management system to load genomic data from files into tables, create indexes/partitions for faster query of data by genomic intervals. The other category of genome browsers query data directly from indexed genomic file formats like bigbed, bigwig or tabix. Interactive visualization of data can be a powerful tool to enable visual exploration and generate insights. As users get familiar with the data and gain insights, it would be even more efficient to test, validate, visualize and compute the intermediate results of the analysis.

Based on the concepts of a NoDB paradigm, we developed Epiviz file server Python library, an in-situ data query system on indexed genomic files, not only for visualization but also for transformation. The library provides various modules to perform various tasks - Import, Query, Compute, Server API and Visualization. Using the file server, users will be able to explore data from publicly hosted files. We currently support various genomic file formats with indexing - BigBed, BigWig, HDF5 and any format that can be indexed using tabix. Once the data files are defined, users can also define summarizations and transformations on these data files using numpy functions. We use dask to manage, distribute and schedule various query and compute requests on files. Our cache implementation also makes sure we only access bytes not already cached locally. To make it easy for developers, we implemented a server module using the Python Sanic library to be able to make REST queries and access data. Once, the server is in place, We can use our Epiviz genome browser to visualize these results. The browser supports various types of visualizations, heatmap and scatter plots for gene expression, blocks (linear and stacked) tracks for visualizing peaks and line tracks (stacked, multi stacked) for visualizing signal (ChIP-seq, methylation etc). Hovering over a region in one visualization highlights this region in other tracks providing instant visual feedback to the user. These visualizations are developed using web component architecture, are highly customizable, reusable and can be integrated with most framework that support HTML. We also require the server hosting the data files to support HTTP range requests so that the file server's parser module can only request the necessary byte-ranges needed to process the query. A higher level architecture of how Epiviz browser and the file server library is shown in Figure 1.

A lightweight approach to Research Object data packaging

http://www.researchobject.org/ro-crate/
http://github.com/researchobject/ro-crate
Apache License, version 2.0

Eoghan Ó Carragáin, Carole Goble, Peter Sefton, Stian Soiland-Reyes

A **Research Object** (RO) provides a machine-readable mechanism to communicate the diverse set of digital and real-world resources that contribute to an item of research. The aim of an RO is to replace traditional academic publications of static PDFs, to rather provide a complete and structured archive of the items (such as people, organisations, funding, equipment, software etc) that contributed to the research outcome, including their identifiers, provenance, relations and annotations. This is increasingly important as researchers now rely heavily on computational analysis, yet we are facing a *reproducibility crisis* [1] as key components are often not sufficiently tracked, archived or reported.

We propose **Research Object Crate** (or **RO-Crate** for short), an emerging lightweight approach to package research data with their structured metadata, based on schema.org annotations in a formalized JSON-LD format that can be used independent of infrastructure to encourage FAIR sharing of reproducible datasets and analytical methods.

Background

Earlier work introduced the notion of *Research Objects* [2]. Their formalization combines existing *Linked Data* standards: W3C RDF, JSON-LD, OAI-ORE, W3C Web Annotations, PROV, Dublin Core Terms, ORCID. The RO ontologies [3] combined these to describe ROs, but do not themselves formalize how ROs are saved or transmitted. Multiple formats have since been realized: the portal RO Hub [4] use RDF REST resources; while workflow provenance make RO Bundle ZIP files [5] or Big Data BagIt archives [6, 7]. Each of these require RO support in the packaging infrastructure.

Multiple *data packaging* initiatives have recently emerged, within Research Data Alliance, Force11, DataOne and elsewhere; like Frictionless data [8] for table-like files, BioCompute Objects for regulatory science [9], CodeMeta for software, Psych-DS for psychology studies, and DataCrate [10] for datasets. RDA has surveyed a large variety of data packaging formats across different domains.

Common among these is *structured metadata*, e.g. with a single JSON file that refer to neighbouring data files and scripts maintained and published together, e.g. in GitHub. Many of these initiatives use schema.org [11] as basis for common metadata. With JSON-LD this offers a developer-friendly experience and interoperability with web conventions outside of the research domain.

Data packaging principles

At a RDA meeting on data packaging we concluded that many initiatives arrive at similar principles: simple folder structure; JSON-LD manifest; schema.org for core metadata; BagIt for fixity; OAI-ORE for aggregation. This points to: a) appetite for general package/folder-oriented approach in different contexts; b) a generic solution won't work for all and needs to be domain-extensible; c) a tendency to re-invent the wheel, leading to sub-optimal interoperability and duplication of effort.

Cite as: https://doi.org/10.5281/zenodo.3250687

Developing Python and Rust libraries to improve the ontology ecosystem

Ontologies are structures used to describe knowledge in a formal way, by defining entities and the relationships between them. Originally developed by logicians, they became widespread in the bioinformatics community thanks to projects such as the Gene Ontology or the OBO Foundry. They can be used as controlled vocabularies for data repositories or data formats, but the possibility to use *reasoners* – programs that can infer consequences from ontology assertions – opens the way for more in-depth analyses.

For historical and technical reasons, there are two languages used to describe most biomedical ontologies:

- The OWL language, developed as a normative specification by the W3C OWL Working Group as a component of the W3C Semantic Web. It is using RDF triples to model the knowledge graph, consequently supporting several serialization (OWL/XML, OWL/Turtle, OWL/Manchester). Modern reasoners expect OWL ontologies as an input, or intermediate representations like the object model of the Java OWL API.
- The OBO language, an attempt made by the OBO consortium to address the integration issues that emerged with the growing number of ontologies in the biomedical field. For integration and backward compatibility purposes it has never really been a normative format, and the de-facto standard comes from the OWL API which can has a provisional support for the OBO format.

The version 1.4 of the OBO format and language, which is still under development, highly restricts the syntax of OBO files in exchange for an improved compatibility with the OWL language. With this edition, OBO ontologies can be mapped exactly to an equivalent OWL2 document.

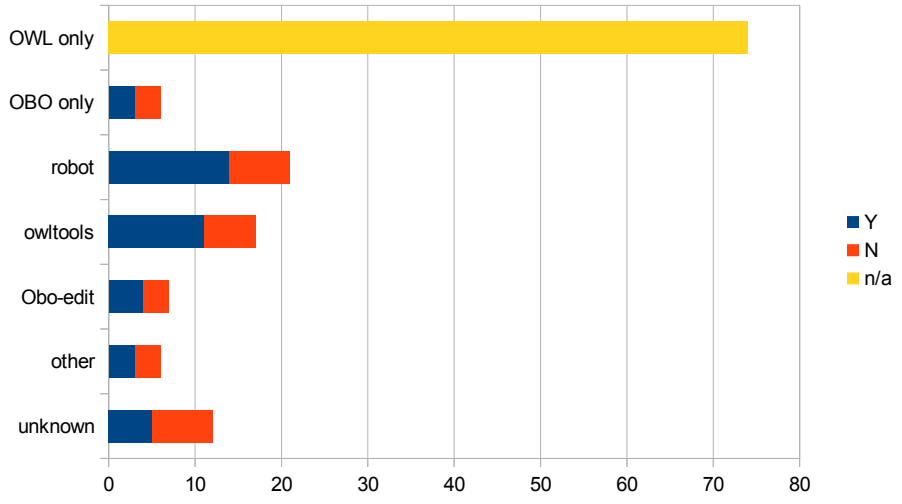


Figure 1: *Distribution of OBO 1.4 compliance for ontologies of the OBO foundry, grouped by tool used for conversion to OBO format (if applicable).*

OWL-only ontologies included as a reference.

ECRcentral: An open-source platform to bring early-career researchers and funding opportunities together

Aziz Khan[1,*], Juan F. Quintana[2], Charlotte M de Winde[3] and Cristiana Cruceanu[4]

1. Centre for Molecular Medicine Norway (NCMM), University of Oslo, Norway
2. Wellcome Trust Centre for Anti-Infective Research (WCAIR), University of Dundee, United Kingdom
3. Stromal Immunology Group, MRC Laboratory for Molecular Cell Biology, University College London, United Kingdom
4. Max Planck Institute of Psychiatry, Munich, Germany

* Correspondence: aziz.khan@ncmm.uio.no

Abstract

For early-career researchers (ECRs), getting funding for their research ideas is becoming more and more competitive, and there is growing pressure in all disciplines to obtain grants. Although there is a plethora of funding opportunities for postdoctoral scientists and other ECRs, there has been no central platform to systematically search for such funding opportunities and/or to get professional feedback on the proposal. With a group of eLife Ambassadors, we developed ECRcentral (ecrcentral.org), a funding database and an open forum for the ECR community. The platform is open to everyone and currently contains 700 funding schemes in a wide range of scientific disciplines, 100 travel grants, and a diverse range of useful resources. In the first two months since its release approximately 500 ECRs already joined this community. The platform is developed using open-source technology, with all the source code and related content made openly available through our GitHub repository (github.com/ecrcentral). ECRcentral aims to bring ECRs and resources for funding together, to facilitate discussions about those opportunities, share experiences, and create impact through community engagement. We strongly believe that this resource will be highly valuable for ECRs and the scientific community at large.

The African Genomic Medicine Training Initiative: Showcasing A Community-Driven Genomic Medicine Competency-Based Training Model for Nurses in Africa

Victoria Nembaware, Paballo Chauke, Nicola Mulder and Planning team

The potential of Genomic Medicine to improve the quality of healthcare both at population and individual-level is well-established, however adoption of available genetic and genomics evidence into clinical practice is limited. Widespread uptake largely depends on the task-shifting of Genomic Medicine to key healthcare professionals such as nurses, who could be promoted through professional development courses. Globally, trainers, and training initiatives in Genomic Medicine are limited, and in resource limited settings such as Africa, logistical and institutional challenges threaten to thwart large-scale training programmes. The African Genomic Medicine Training (AGMT) Initiative was created in response to such needs. It aims to establish sustainable Genomic Medicine training initiatives for healthcare professionals and the public in Africa. This work describes the AGMT and reports on a strategy recently piloted by this group to design and implement an accredited, competency and community-based distance learning course for nurses across 11 African countries. This model takes advantage of existing consortia to create a pool of trainers and adapts evidence-based approaches to guide curriculum and content development. Existing curricula were reviewed and adapted to suit the African context. Accreditation was obtained from university and health professional bodies. Both the acceptability of this model, the feasibility of replication in similar settings, and training a wide-range of healthcare professionals, is supported by data from an implementation evaluation that was informed by class mini-projects tailored to African diseases submitted for peer-reviewed publication, reflections and surveys from the working group members, advisors, course coordinator, facilitators, trainers and students. A toolkit is proposed to help guide adoption of the AGMT distance-learning model.

Figure 1: **African Genomic Medicine Training Initiative – Official Launch 12 May – 2016: Dakar Senegal**

P-36

Parallel, Scalable Single-cell Data Analysis

Ryan Williams, Tom White, Uri Laserson

Repository : https://github.com/lasersonlab/ndarray.scala
License : Apache 2

Single-cell sequencing generates a new kind of genomic data, promising to revolutionize understanding of the fundamental units of life. The Human Cell Atlas is a multi-year, multi-institution effort to develop and standardize methods for generating and processing this data, which poses interesting storage and compute challenges.
I'll talk about recent work parallelizing analysis of single-cell data using a variety of distributed backends (Apache Spark, Dask, Pywren, Apache Beam). I'll also discuss the Zarr format for storing and working with N-dimensional arrays, which several scientific domains have recently gravitated toward in response to challenges using HDF5 in parallel and in the cloud.

Another point of view for the fast and accurate large MSA, the regressive approach

Edgar Garriga Nogales

Repository : https://github.com/cbcrg/tcoffee
License : GPL license

Inferences derived from large multiple alignments of biological sequences are critical to many areas of biology, including evolution, genomics, biochemistry, and structural biology. However, the complexity of the alignment problem imposes the use of approximate solutions. The most common is the progressive algorithm, which starts by aligning the most similar sequences, incorporating the remaining ones following the order imposed by a guide-tree. We developed and validated on protein sequences a regressive algorithm that works the other way around, aligning first the most dissimilar sequences. Our algorithm produces more accurate alignments than non-regressive methods, especially on datasets larger than 10,000 sequences. This computation is also more efficient, as it uses a divide-and-conquer strategy to run third-party alignment methods in linear time, regardless of their original complexity. As a consequence, the regressive algorithm puts an end to a recurrent dilemma between the use of slow/accurate or fast/approximate methods. It will enable the full exploitation of extremely large genomic datasets.

Analyzing protein structure and evolution using Julia with MIToS.jl

Diego Javier Zea[1]
LCQB UMR 7238 CNRS, IBPS, Sorbonne Université, 7 Quai Saint-Bernard, 75005, Paris, France

Corresponding author: diegozea@gmail.com

1 Introduction

MIToS is a *Julia* package for analyzing protein sequence and structure, with the main focus on coevolutionary analysis [1]. However, its utilities go beyond the calculation of covariation scores in multiple sequence alignments. *MIToS* is a flexible suite that has been used to measure residue conservation, to deal with protein structures in homology modelling and molecular dynamics pipelines, to perform structural alignment of tertiary and quaternary structures, etc. *MIToS* allows to access the power of *Julia*, a high-level programming language for scientific computing with a close to *C* performance [2].

MIToS defines functions and types for dealing with multiple sequence alignments, parsing protein structures, determine inter-residue contacts and interactions, mapping information between sequence and structure using *SIFTS* [3] and many other tasks. Their modules allow to write and run an entire protein sequence and structure analysis pipeline in a single programming language. Julia performance and easy to use parallelism allow us to run these analyses on large datasets and to test multiple hypotheses, parameter combinations, etc. As a result, it was used to create new knowledge about the relation between the evolutionary signals and the change of protein structures through evolution [4].

It is a common task in bioinformatic pipelines to link structural information coming from *PDB* [5] and evolutionary information calculated from multiple sequence alignments. *MIToS* makes this task easier by keeping the mapping information in the multiple sequence alignment annotations. In this way, it is possible to track residue positions, even after deleting or selecting alignment columns. Also, the ability of *MIToS* to parse *SIFTS* files allows to access their residue level mapping between PDB and other databases, e.g. *UniProt* [6]. Both things together, allow the correct mapping of sequence and structure without performing error-prone pairwise alignments.

The software is totally implemented in Julia and supports Linux, OS X and Windows. It is open source and freely available on GitHub under MIT license: *https://github.com/diegozea/MIToS.jl*

Acknowledgements

That work was supported by Sorbonne Université, CONICET, FIL and PICT 2014-1787.

References

[1] Diego J Zea, Diego Anfossi, Morten Nielsen, and Cristina Marino-Buslje. Mitos. jl: mutual information tools for protein sequence analysis in the julia language. *Bioinformatics*, 33(4):564–565, 2016.

[2] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[3] Jose M Dana, Aleksandras Gutmanas, Nidhi Tyagi, Guoying Qi, Claire O'Donovan, Maria Martin, and Sameer Velankar. Sifts: updated structure integration with function, taxonomy and sequences resource allows 40-fold increase in coverage of structure-based annotations for proteins. *Nucleic acids research*, 47(D1):D482–D489, 2018.

[4] Diego Javier Zea, Alexander Miguel Monzon, Gustavo Parisi, and Cristina Marino-Buslje. How is structural divergence related to evolutionary information? *Molecular phylogenetics and evolution*, 127:859–866, 2018.

[5] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 01 2000.

[6] The UniProt Consortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Research*, 47(D1):D506–D515, 11 2018.

Pedigree-based analysis pipeline version 2 (PBAP v.2): new features added

Alejandro Nato, Nicola Chapman, Mohamad Saad, Harkirat Sohi, Charles Cheung, Andrea Horimoto, Rafael Nafikov, Khalid Kunji, Ehsan Ullah, Hiep Nguyen, Ellen Wijsman

Repository : https://faculty.washington.edu/wijsman/progdists/pbap/pbap_v1.00.tar.gz
License : GNU General Public License

Dense genetic data are now common due to accessibility of next generation sequencing technologies. Genetic analysis software has been developed mostly for population-based studies. However, recognition of how important rare variants are has made pedigree-based studies common again. We previously developed a pedigree-based analysis pipeline (PBAP) v.1, which allows users to perform several procedures for pedigree-based genetic analysis including file manipulation, selection of subset of markers from a dense panel, pedigree structure validation, and sampling of inheritance vectors (IVs), i.e., the flow of founder alleles in a pedigree. Here, we describe a second version of PBAP with new features that include setting up of files to use IVs for downstream analyses. PBAP v.2 accesses programs to implement the following analyses: a) parametric linkage analysis allowing modification of marker allele frequencies for admixed populations, b) variance components linkage analysis, c) family-based genotype imputation, and d) genotype-based kinship estimation with option to use external allele frequencies. PBAP v.2 users may also calculate pairwise kinship coefficients based on the sampled IVs and visualize spacing of the sub-selected panel of markers. All these features extend the capabilities of PBAP v.2 and give users more options to maximize use of their data for family-based analyses.

P-40

RAWG: RNA-Seq Analysis Workflow Generator

Alessandro Pio Greco, Patrick Hedley-Miller, Filipe Jesus, Zeyu Yang

May 15, 2019

The rapid pace of innovation in the field of sequencing has meant an explosion in the number of tools available for analysis. This creates problems when interpreting differences of downstream analyses between different RNA-Seq pipelines because there are multiple junctures at which discrepancies can occur. This issue is compounded since there are numerous parameters within each step of the pipeline that a user can manually adjust. The result is that inter-pipeline comparisons of RNA-seq analysis are difficult to interpret and users need to ensure a consistent set of parameters are used for all samples.

We developed a data analysis framework, RNA-Seq Analysis Workflow Generator (RAWG), which can act as a one stop shop for anyone wanting to preform RNA-Seq differential expression analysis. RAWG consists of a webportal, a set of server-side scripts, and a collection of command-line tool wrappers in Common Workflow Language (CWL). The webportal is the end-user's primary interface and is used to upload RNA-Seq reads and define analysis pipelines. The server-side scripts, written in Python, dynamically generate CWL workflows base on user-selected tools from the webportal and execute the workflows. Together, we achieved an user-friendly and easy to use data analysis framework which is also extensible so that developers can integrate tools into RAWG easily.

The main advantage of RAWG is that users are liberated from writing workflows manually as all the connections between tools are handled automatically. RAWG is also capable of performing multiple pipelines in one workflow, which means common steps in different pipelines are merged into a single step, hence saves computational resources. Leveraging container technology, researchers are freed from setting up complex software environments and the analysis workflow is more reproducible and portable. A demo server and the user guide is available here: https://github.com/rawgene/rawg/blob/master/doc/userguide.md

Application

To showcase the ability of using RAWG to make top quality scientific discoveries, we present two application examples. A differential expression analysis on neuroblastoma data and a comparison between RNA-Seq analysis pipelines.

Neuroblastoma Data Analysis

Previous studies have linked neuroblastoma progression and development to p53, NGF and TrkA expression. This study aims to find features which are differentially expressed solely due to NGF-independent TrkA activation via interactions with exogenous TrkA and mutant p53. Note that Fig. 1(d) is plotted by the webportal's visualisation section.

Crowdsourcing towards Antimicrobial Resistance & Open Source Drug Discovery

Anshu Bhardwaj

Repository : https://github.com/AnshuBhardwajCRI,
http://ab-openlab.csir.res.in/gitlab/openlab/reptb/tree/master

Advancement in global health over the last half-century has been significant and can mostly be attributed to discovery of new drugs by the pharmaceutical industry. However, the patent paradigm that rewards the investment in research and development, and drives this effort are not sufficient for discovery of new antimicrobials. Given the increasing issue of antimicrobial resistance, alternate models are needed in drug discovery. OSDD (Open Source Drug Discovery) is an alternative innovation model where distributed community of researchers work together towards common goals sharing data and resources. As an example of this is the Connect to Decode project which utilized the potential of crowdsourcing to generate the most comprehensive systems level models of Mycobacterium tuberculosis (Mtb) (PMID: 22808064), repository of anti-TB compounds (PMID: 29785561), etc. Published estimates suggest that this innovative approach packed nearly 300 man-years into 4 months. More recently, the community also generated a repository of potential anti-Nipah compounds demonstrating the power of collective open efforts towards outbreaks (http://bioinfo.imtech.res.in/anshu/nipah). I would like to apply similar principles of open data sharing on tools/methods that are developed in the field. Attending BOSC will provide me the right platform to leverage from the interactions to move forward in this direction.

MISO LIMS : managing information for sequencing operations

Morgan Taschuk[1], Heather Armstrong[1], Dillan Cooke[1], Andre Masella[1], Alexis Varsava[1], Lars Jorgensen[1]
1. Ontario Institute for Cancer Research, Toronto, Ontario, Canada

MISO is a laboratory information management system designed for eukaryotic sequencing operations. It supports genomic, exomic, transcriptomic, methyl-omic, and CHiP-seq protocols; long reads and short reads; and microarrays. MISO incorporates a wide feature set useful for both large and small facilities to track their lab workflows in great detail. MISO has two primary goals: 1) to allow laboratory technicians to record their work accurately, without having to adapt their protocols to match the system's model, with a minimum of data entry overhead and 2) to keep the associated metadata valid and structured enough to use for automation and other downstream applications.

History
The software was developed by Robert Davey's lab at the Earlham Institute in Norwich, UK, with the first release in 2011. By 2015, development had slowed to primarily maintenance activities. Our group was looking for a new LIMS and so approached the MISO development team with the intent of developing it to meet specific goals.

Goal 1: Record laboratory activities
MISO is flexible enough to keep up with rapidly evolving research protocols and methods while simultaneously providing input validation and reducing the pain of data entry. Our detailed sample hierarchy mimics actual laboratory processes for receiving tissues, creating stocks and aliquots, propagating to sequencing libraries, pooling and loading onto a sequencer. Quality control measures, volume, and concentrations can be recorded for each entity. Everything corresponding to a physical specimen is also barcoded, located by freezer and shelf, and tracked by changelogs. MISO supports new instruments like the Illumina NovaSeq, 10X Chromium, and Oxford Nanopore PromethION, added more extensive location tracking, and has improved overall performance.

We improved UI interfaces to simplify data entry by providing a spreadsheet-like bulk entry interface for every entity in MISO with functions like fill-down and auto-increment. If more power is necessary, MISO allows exporting and importing Excel spreadsheets. MISO also provides automatic name generators based on project names and templates to automatically fill in values for standard laboratory protocols.

Goal 2: Facilitate analysis automation
A major goal of MISO was to encourage laboratory technicians to enter information in enough detail and with sufficient rigor to automate downstream analysis, particularly base calling and alignment. MISO provides in-browser validation to ensure that controlled vocabulary stays

P-43

10 recommendations to make your research software FAIRer

Christopher Erdmann, Leyla Jael García Castro, Mateusz Kuzak, Anna-Lena Lamprecht, Carlos Martinez Ortiz, Paula Martinez

The Findable, Accessible, Interoperable and Reusable (FAIR) principles provide a set of minimum elements required for an effective management of digital resources. Although the FAIR principles are meant to work on any digital resource, datasets were mostly in mind when the principles were initially published in 2016. Some of the main aspects, particularly within the Findability and Accessibility scope, are indeed directly applicable across digital resources, for instance persistent and global identification, licensing and longevity commitment policies. However, when used to model research software, i.e., scripts, packages, and applications, principles related to Interoperability and Reusability require further discussion, understanding and agreement. Here we present ten recommendations to get FAIRer research software. We take into account elements that will make it easier for other to (re)use the software such as functionality, input, output, citation and documentation. We also include a mention on best practices on software development as they cover aspects such as versioning, dependency management, and so on. With this effort we at ELIXIR Europe aim to contribute and promote the discussion around FAIRness for research software.

SAPPORO: workflow management system that supports continuous testing of workflows

Hirotaka Suetake[1], Tazro Ohta[2]

1. The University of Tokyo, Department of Creative Informatics, Graduate School of Information Science and Technology, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
2. Database Center for Life Science, Joint Support-Center for Data Science Research, Research Organization of Information and Systems, Shizuoka 411-8540, Japan

Project Website: https://suecharo.github.io/SAPPORO/
Source Code: https://github.com/suecharo/SAPPORO/
License: Apache2.0

Sharing personal genome data is critical to advance medical research. However, sharing data including personally identifiable information requires ethical reviews which usually takes time and often has limitations of computational resources that researchers can use. To allow researchers to analyze such data in controlled access efficiently, DNA Data Bank of Japan (DDBJ) developed a new workflow execution system called SAPPORO (Figure1). We designed the system to allow users to execute workflows with controlled access data without touching them. Users select a workflow on the SAPPORO's web interface to run it on a node for personal genome data analysis in the DDBJ's high-performance computing (HPC) platform. The system supports the Common Workflow Language (CWL) as its primary format to describe workflows; thus it can import the workflows developed by different institutes as long as they are described in CWL [1]. We implemented the workflow run service component by following the Workflow Execution Service API standard developed by the Cloud working group of Global Alliance for Genomics and Health (GA4GH) [2]. This highly flexible and portable system can be an essential module on data and workflow sharing in biomedical research.

1. Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., ... & Scales, M. (2016). Common Workflow Language, v1. 0.
2. GA4GH Cloud Work Stream https://github.com/ga4gh/wiki/wiki

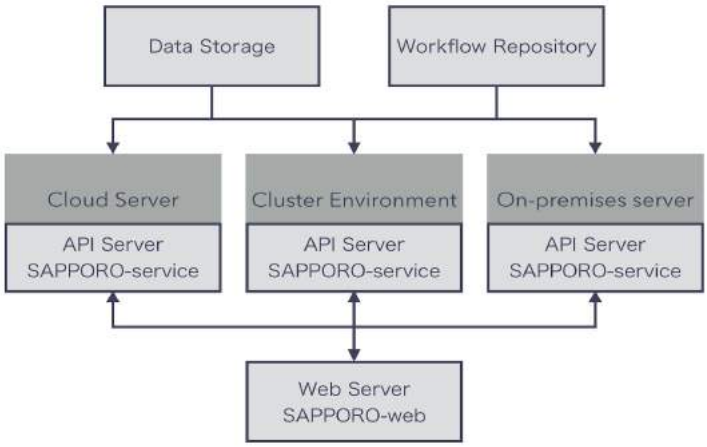


Figure 1. The system overview of SAPPORO

Response to reviewers

With increasing challenges in understanding very large and complex cancer genomic data, this abstract presents robust, scalable R/Bioconductor software data representations and statistical methods to help tackle significant problems in cancer biology. Bioconductor is a widely used, highly respected open-source environment for statistical analysis and comprehension of high-throughput genomic data, so these developments are useful to many researchers.

With large archives of publicly available genomic data implementing FAIR (findable, accessible, interoperable, reusable) principles, urgent demands are present for computational and bioinformatics tools to efficiently translate the data into clinical important insights. In order to reduce memory usage and optimize performance, Bioconductor has developed different data structures and interfaces to lazily represent big data sets either in “array” format (e.g., count data from single cell RNA sequencing (scRNA-seq) experiment), or in “data.frame” format (e.g., the feature or sample annotation information with clinical characteristics and relevance). Lightweight, lazy containers provide easy data manipulation within familiar R/Bioconductor paradigms, and support scalability and interoperability with existing bioinformatics tools available in R/Bioconductor.

The DelayedArray has been developed to represent very big genomic data sets, such as count data from scRNA-seq and variant data from high-throughput DNA-seq experiments. DelayedArray allows users to perform common array operations on it without loading the object in memory. Operations on the DelayedArray objects are either recorded but delayed, or executed using a block processing mechanism. Bioconductor has made DelayedArray easily extendable to different 'back end' representations of data, such as Hierarchical Data Format (HDF) (available in Bioconductor as HDF5Array), the Genomic Data Structure (GDS) (available in Bioconductor as GDSArray) and the Variant Call Format (VCF) (available in Bioconductor as VCFArray). The scalability offered by DelayedArray has enabled computational biologists and bioinformaticians to take advantage of rich programming semantics and diverse big data solutions. DelayedArray builds on familiar R / Bioconductor programming paradigms, and requires little effort on the part of the bioinformatician to learn new technologies.

In addition to the lazy representation of assay data obtained from biological experiments, Bioconductor has developed a data structure called DelayedDataFrame to represent the metadata for features (e.g., gene symbols, tests of gene-wise statistical significance) or samples (e.g., clinical characteristics) with a DataFrame-like metaphor. DelayedDataFrame accommodates DelayedArray (and direct extension) objects in the columns. Operations on DelayedDataFrame are recorded but delayed until a specific realization call is invoked. Lazy sample and feature metadata can be combined with lazy assay data to be analyzed in R with common and familiar methods to bioinformaticians. These operations do not require loading the whole data into memory.

SQLDataFrame is another Bioconductor package that has been developed to lazily represent and efficiently analyze SQL-based tables in R. SQLDataFrame supports common and familiar 'DataFrame' operations such as “[” subsetting, rbind, cbind, etc.. The internal implementation is based on the widely adopted dplyr grammar and SQL commands. This provides advanced

Terra Open Science Contest

Win a trip to attend BOSC in Switzerland

Your mission: Create an open workspace that showcases a fully reproducible analysis!

Do you have a pet workflow or favorite notebook? Have you thought about sharing them with the world, but keep pushing it off? Here's your opportunity to get it done, feel good about it AND **win a trip to Switzerland in July**!

We've all been in the situation of needing to **share our tools and methods with a collaborator**, or wanting to **build on someone else's work for our own research**. We've made a lot of progress toward making it easier to share, collaborate, and build on previous computational biology work by adopting best practices like version control, containers and so on. But there can still be a big gap between handing over the code and enabling others in the field to actually run it and reproduce the work. Let's show **how much easier it can be with a showcase of working examples** in an environment like Terra **that integrates code, data and execution**.

Through this friendly contest, we want to challenge the Terra community to **raise the bar in how we share computational methods with each other**. The best workspace wins (we're crowdsourcing evaluations based on overall ease of reproducibility, including clarity and completeness of workspace documentation... and science chops!). By showcasing a variety of reproducible workspaces, we hope to demonstrate the high value benefits to the entire community of Open Science approaches that **boost computational reproducibility**.

For some background reading on computational reproducibility, see this blog post.

Prizes

The **Grand Prize is a roundtrip to Basel, Switzerland** to attend the joint ISMB/ECCB* and BOSC** conference, July 21-25, as well as **$5,000 in Google credits**. Flights, hotel accommodation, per diem to cover meals, and conference registration are all included in this prize. The winner will be expected to attend the conference and may be asked to give a short oral presentation about the workspace.

**ISMB: International Society for Computational Biology*
***BOSC: Bioinformatics Open Source Conference*

Additional prizes:

- The first runner-up will win $5,000 in Google credits.
- Second runner-up will win $2,500 in Google credits.

Entry submissions will open May 17 and close June 17. Judging will open May 20 and close June 20.

P-47

Sequenceserver: a modern graphical user interface for custom BLAST databases

Anurag Priyam, Yannick Wurm

Repository : https://github.com/wurmlab/sequenceserver
License : AGPL

The advances in DNA sequencing technologies have created many opportunities for novel research that require comparing newly obtained and previously known sequences. This is commonly done with BLAST, either as part of an automated pipeline, or by visually inspecting the alignments and associated meta-data. We previously reported Sequenceserver to facilitate the latter. Our software enables a user to rapidly setup a BLAST server on custom datasets and presents an interface that is modern looking and intuitive to use. However, interpretation of BLAST results can be further simplified using visualisations.
We have integrated three existing visualisations into Sequenceserver with the aim to facilitate comparative analysis of sequences. First, we provide a circos plot to rapidly check for conserved synteny, identify duplications and translocation events, or to visualise transposon activity. Second, we provide a histogram of length of all hits of a query to quickly reveal if the length of a predicted protein sequence matches that of its homologs. Finally, for each query-hit pair, the relative length and position of matching regions are shown. This is helpful to identify large insertion or deletion events between two genomic sequences, can reveal putative exon shuffling, and help confirm a priori knowledge of intron lengths.

P-48

EDAM: the ontology of bioinformatics operations, types of data, topics, and data formats (2019 update)

Matúš Kalaš, Hervé Ménager, Veit Schwämmle, Jon Ison, Edam Contributors

Repository : https://github.com/edamontology/edamontology

EDAM is an ontology of well established, familiar concepts that are prevalent within bioinformatics, and life science data analysis in general. The scope of EDAM includes types of data and data identifiers, data formats, operations, and topics. EDAM has a relatively simple structure, and comprises a set of concepts with terms, synonyms, definitions, relations, links, and some additional information (especially for data formats).

P-49

Abstract

Sequencing large molecules of DNA has drastically improved the contiguity of genome sequence assemblies. Long read sequencing has reduced sequence fidelity compared to short read sequencing and is currently more expensive. Linked read sequencing from 10x Genomics Chromium combines the benefits of large DNA molecules with the sequence fidelity and cost of short read sequencing. Our tool, Physlr, constructs a physical map of large DNA molecules from linked reads without first assembling those reads. A barcode-overlap graph is constructed, where each edge represents two barcodes sharing minimizer k-mers. The underlying molecule-overlap graph is reconstructed from the barcode-overlap graph by identifying k-clique communities, where each community is born from a DNA molecule. The physical map is a set of contigs, where each contig is an ordered list of barcodes. The scaffolds of an existing assembly may be ordered and oriented using the physical map. We constructed a physical map of the 1.34 Gbp zebrafish (*Danio rerio*) genome. A Supernova assembly was scaffolded by mapping it to this physical map, improving the NG50 from 4.8 Mbp to 9.1 Mbp. Physlr can employ multiple libraries of linked reads, necessary for genomes larger than mammals such as conifer genomes, which can exceed 20 Gbp.

Select Unique Features of Archaeopteryx.js

Available at: https://www.npmjs.com/package/archaeopteryx

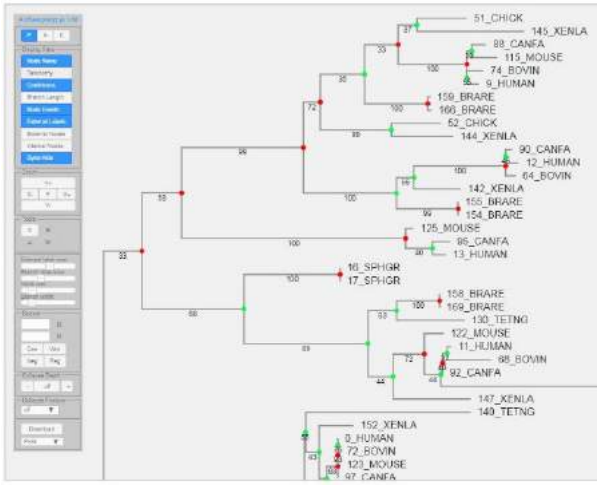


Figure 1: Archaeopteryx.js displaying gene duplications.

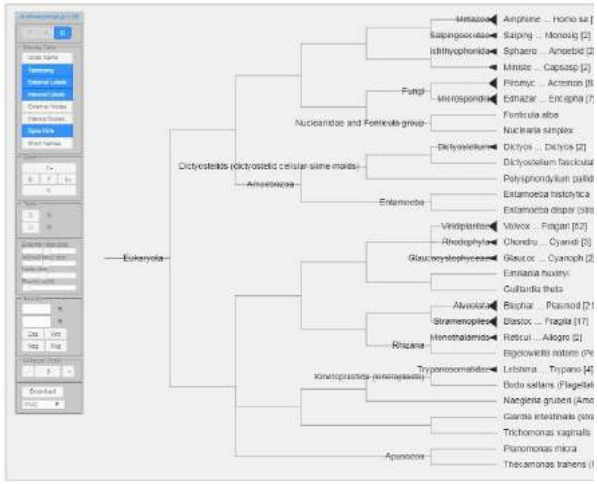


Figure 2: Demonstration of auto-collapse feature.

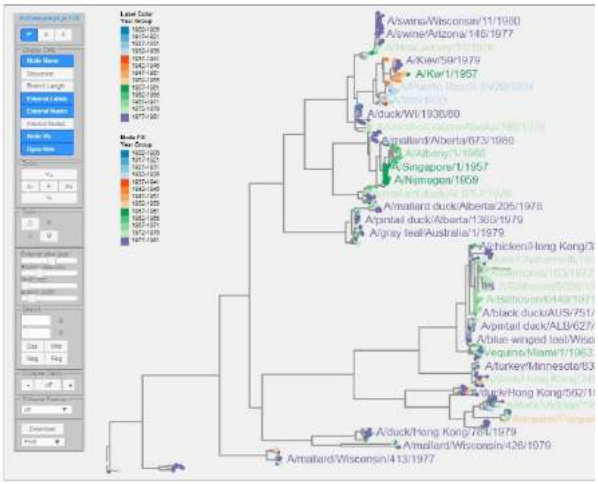


Figure 3: Example of visualization of meta-data: year as label color and node fill.

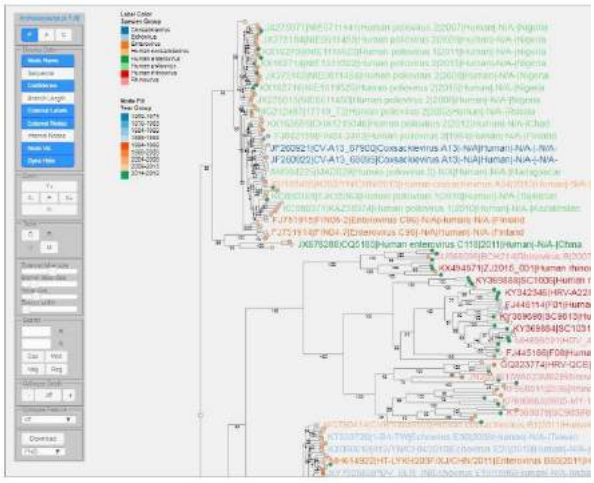


Figure 4: Example of visualization of meta-data: virus species as label color and year as node fill.

The Monarch Initiative: Closing the knowledge gap with semantics-based tools

Monica Munoz-Torres[1], Melissa Haendel[1,2], Chris Mungall[3], Peter N. Robinson[4], David Osumi-Sutherland[5], Damian Smedley[6], Julius Jacobsen[7], Sebastian Köhler[8], Julie McMurry[1,2], and the members of The Monarch Initiative.

[1]Oregon State University, Corvallis, OR, USA. [2]Oregon Health & Science University, Portland, OR, USA. [3]Lawrence Berkeley National Laboratory, Berkeley, CA, USA. [4]The Jackson Laboratory for Genomic Medicine, Farmington, CT, USA. [5]European Bioinformatics Institute, Hinxton, UK. [6]Genomics England, Cambridge, UK. [7]Queen Mary University of London, London, UK. [8]Charité Universitätsmedizin, Berlin, Germany.

The Monarch Initiative is a consortium that seeks to bridge the space between basic and applied research, developing tools that facilitate connecting data across these fields using semantics-based analysis. The mission of the Monarch Initiative is to create methods and tools that allow exploration of the relationships between genotype, environment, and phenotype across the tree of life, deeply leveraging semantic relationships between biological concepts using ontologies. These tools include Exomiser, which evaluates variants based on the predicted pathogenicity, amongst many others. The goal is to enable complex queries over diverse data and reveal the unknown. With the semantic tools available at www.monarchinitiative.org, researchers, clinicians, and the general public can gather, collate, and unify disease information across human, model organisms, non-model organisms, and veterinary species into a single platform. Monarch defines phenotypic profiles, or sets of phenotypic terms, which are associated with a disease or genotype recorded using a suite of phenotype vocabularies (such as the Human Phenotype Ontology and the Mondo Ontology). Our niche is computational reasoning to enable phenotype comparison both within and across species. Such explorations aim to improve mechanistic discovery and disease diagnosis. We deeply integrate biological information using semantics, leveraging phenotypes to bridge the knowledge gap.

The Monarch Initiative GitHub Organization
https://github.com/monarch-initiative

Human Phenotype Ontology
Website: https://hpo.jax.org
License: https://hpo.jax.org/app/license

Exomiser
Website: https://github.com/exomiser/Exomiser
License: https://hpo.jax.org/app/license

Mondo
https://github.com/monarch-initiative/mondo
License: CC-BY

P-52

DAISY: a tool for the accountability of Biomedical Research Data under the GDPR

Regina Becker, Pinar Alper, Valentin Grouès, Sandrine Munoz, Yohan Jarosz, Jacek Lebioda, Kavita Rege, Christophe Trefois, Venkata Pardhasaradhi Satagopam, Reinhard, Schneider

Repository : https://github.com/elixir-luxembourg/daisy

GDPR requires the documentation of any processing of personal data, including data used for research and to be prepared for information provision to the data subjects. For institutions this requires a data mapping exercise to be performed and to keep meticulously track of all data processings. While there is no formal guidance on how data mapping should be done, we're seeing the emergence of some commercial "GDPR data mapping" tools and academic institutions creating registers with those tools. When it comes to mapping data in biomedical research, we observe that commercial tools may fall short as they do not capture the complex project-based, collaborative nature of research that leads to many different scenarios.

In this poster we describe a Data Information System (DAISY), our data mapping tool, which is specifically tailored for biomedical research institutions and meets the record keeping and accountability obligations of the GDPR. DAISY is open-source and is actively being used at the Luxembourg Centre for Systems Biomedicine and the ELIXIR-Luxembourg data hub.

OpenEBench. The ELIXIR platform for benchmarking.

Benchmarking is intrinsically referred to in many aspects of everyday life from assessing the quality of stock market predictions to weather forecasting to predictions in the life sciences, such as 3D protein structure predictions or functional annotations. On an abstract level, benchmarking is comparing the performance of software under controlled conditions. Benchmarking encompasses the technical performance of individual tools, servers and workflows, including software quality metrics, as well as their scientific performance in predefined challenges. Scientific communities are responsible for defining reference datasets and metrics, reflecting those scientific challenges (Capella-Gutierrez et al. bioRxiv, 2017). In the context of ELIXIR, we have developed the OpenEBench platform aiming at transparent performance comparisons across life sciences. OpenEBench supports scientific communities by assisting in setting up emerging benchmarking efforts, foster exchange between communities and ultimately aims at making benchmarking not only more transparent, but also more efficient.

We will present the current OpenEBench and a preview on the upcoming implementations, which will be strongly focused on assisting communities to join the platform. OpenEBench is composed of two major sections. On one side, the OpenEBench tools monitoring section aims to provide a comprehensive observatory of bioinformatics software in terms of quality, FAIRness, and performance. At present OpenEBench collects data from more than 15,000 bioinformatics tools. The main target users would be researchers seeking to choose the most appropriate tool for a given analysis, considering availability, hardware requirements, software quality including documentation and/or deployment options, and comparative performance. We collect data from the ELIXIR Tools and Services registry, bio.tools; BioConda, Galaxy, BioContainers, software repositories e.g. Github, and perform text mining analysis on web sites and documentation to collect assessment items. In addition, several widgets as sites availability (including response time), and bibliographic citation history for each tool are provided.

On the other side, OpenEBench is dedicated to scientific benchmarking. The aim is to provide an infrastructure to support community-led scientific benchmarking initiatives at different levels of maturity. Target users are i) researchers seeking to choose the most appropriate tool for a specific scientific case; ii) developers aiming to test new software in the context of accepted benchmarking efforts; iii) communities aiming either to disseminate their existing results, and/or needing a technical infrastructure to perform the challenges, and iv) funders aiming to gather a collective view of a specific field. At OpenEBench we are working in three levels of operation (see Figure 1): level 1 (available) aims to collect and distribute data from established benchmarking communities; level 2 (beta state) is based on providing a technical infrastructure for computing benchmarking metrics, and to design benchmarking challenges; level 3 will extend the existing OpenEBench platform to execute benchmarkable workflows (provided as software containers) using controlled conditions to ensure an unbiased technical and scientific assessment. Level 1 complements the activities already done by benchmarking communities e.g. Quest for Orthologs (Altenhoff et al, Nat Meth 2016, CAMEO (Haas et al. Database 2013), TCGA (Bailey et al. Cell 2018); providing alternative views of benchmarking results adequate for the non-experts, and connected to the tools monitoring section. At level 2 OpenEBench provides a Virtual Research Environment (Codó et al. bioRxiv 2019) (https://openebench.bsc.es/submission) where two types of users are defined. On one hand, community managers may use the workspace to design new challenges, distribute reference datasets, and test and execute metrics on participant's providers data. On the other hand, developers can access the workspace to test their own tools using the available

ImmPort: Ensuring FAIR Data through a Trustworthy Biomedical Data Repository

Dawei Lin

Repository : https://www.immport.org/shared/home
https://www.immport.org/resources/documentation
https://bioconductor.org/packages/release/bioc/html/RImmPort.html
License : https://www.immport.org/agreement

Data-driven science is facilitated by datasets that comply with the FAIR (Findable, Accessible Interoperable and Reusable) Data Principles to maximize the value of the data. While the social sciences and geosciences have long recognized that Trustworthy Data Repositories (TDRs) are critical components to enable data to be FAIR on a sustainable basis, the concepts of trustworthiness and the means to assess TDRs are new to some biomedical fields.

java2script/SwingJS for bioinformatics: Reintroducing Jalview on the Web as JalviewJS

Robert Hanson, Geoff Barton, Jim Procter, Mungo Carstairs, Benedict Soares

Repository : https://github.com/BobHanson/java2script
License : GPL, LGPL

We have successfully developed the technology used to produce JSmol with Jmol to produce JalviewJS with Jalview for the integration and visualisation of biological sequence and three-dimensional structures and other data within an evolutionary alignment framework. java2script/SwingJS has been used to produce well over 500 JavaScript apps from Java code bases (including Java applets and full Java applications). This is not a "total rewrite". This is a minor tweaking of a Java program to be "JavaScript compatible" with automated co-production of Java class files and their JavaScript equivalents simultaneously. java2script/SwingJS is breakthrough technology that allows open-source developers to produce multiple flavors of their programs (Java desktop power application, fully contextualized JavaScript web page-based embedded app, and self-contained progressive web app) from the same Java code base. Little or no actual programming is done in JavaScript. The java2script transpiler rides along with the Eclipse Java compiler to produce the JavaScript "class" files, which then "run" on a web page using the SwingJS equivalent of the Java Virtual Machine. We will discuss how this technology allows the Jalview Desktop (3000+ Java classes) to be used within a rich web-based environment, and how easy it is for any developer to do this themselves.