PEGR: a flexible management platform for reproducible epigenomic and genomic research

Danying Shao¹, Greta D. Kellogg², Ali Nematbakhsh², Prashant Kuntala³, Shaun Mahony³, B. Franklin Pugh⁴ & William KM Lai^{4*}

¹ Institute for Computational and Data Sciences, Pennsylvania State University, University Park, PA 16802, USA.

² Cornell Institute of Biotechnology, Cornell University, Ithaca, New York 14850, USA

³ Department of Molecular Biology and Genetics, Cornell University, Ithaca, NY 14850, USA.

⁴ Department of Biochemistry & Molecular Biology, Pennsylvania State University, University Park, PA 16802, USA.

* Corresponding author

Abstract

Reproducibility is a significant challenge in (epi)genomic research due to the complexity of experiments composed of traditional biochemistry and informatics. Recent advances have exacerbated this challenge as high-throughput sequencing data is generated at an unprecedented pace. Here we report on our development of a Platform for Epi-Genomic Research (PEGR), a web-based project management platform that tracks and quality controls experiments from conception to publication-ready figures, compatible with multiple assays and bioinformatic pipelines. It supports rigor and reproducibility for biochemists working at the wet bench, while continuing to fully support reproducibility and reliability for bioinformaticians through integration with the Galaxy platform.

Keywords

genomics, Galaxy, high-throughput sequencing, data management system, reproducibility, science gateway.

Background

Reproducibility is one of the cornerstones of scientific research. However, reproducibility has been a longtime challenge across many scientific fields [1-3]. These difficulties arise from complexities in experimental and bioinformatic workflows that diverge over time, across different operators, and often with limited versioning [4-6]. In the field of genomics, collections of massive datasets that can be parsed in many ways has added to the reproducibility challenge [7-11]. What is needed is systematic metadata capture and management software that is tailored to (epi)genomic data collection.

In general, a genomic project is composed of two distinct but interrelated components: ‘wet-bench’ biochemistry experiments and ‘dry-bench’ bioinformatic analysis. In wet-bench experiments: sample type (human tissue biopsy, yeast, etc.), reagents (catalogue number, wash buffer recipe, etc.), growth environment (log growth, % confluence, etc.), and experimental protocols (ChIP-seq, Western blot, etc.) are examples of critical metadata that need to be captured. Minor variations in these experimental components can result in distinct experimental outcomes [12, 13]. Confounding these issues is the traditional reliance on storing experiment metadata in hand-written notebooks, which are not searchable and often incomprehensible to a third party [14]. Consequently, it can be difficult to follow and accurately reproduce an experimental protocol from start to finish.

Similarly, in bioinformatics analysis, different analytical tools, software versions, and tool parameters may generate different analytical outcomes. While progress has been made in tracking and reproducing informatic workflows (e.g., Pegasus, Galaxy), these platforms are generally limited to reproducing software workflows [15, 16]. To our knowledge, there are no free open-source platforms that manage entire experimental pipelines, from wet-bench experiments to bioinformatic analyses. Laboratory information management systems (LIMS) typically focus on inventory management and sample tracking and have limited capability to record experimental metadata, data analysis parameters, and for interfacing with project team members [17-19]. Although there have been several commercial efforts in this direction, they can be limited in scope (e.g., only tracking sequencing reagents) and/or rather expensive to small academic laboratories [20, 21]. These platforms typically have limited integration between data production and a wide range of experimental metadata.

We developed the Platform for Epigenomic and Genomic Research (PEGR), a web-based project management platform that integrates wet-bench sample tracking with downstream bioinformatic workflows (managed by Galaxy.org workflows) to address the challenge of experimental reproducibility. PEGR provides end-to-end management of (epi)genomics projects from initial reagent usage through to the reproducible generation of publication-quality figures. PEGR logs sample information and experimental details. It manages metadata produced by Galaxy or any other workflow system and provides sample reporting and visualization. It supports Findable, Accessible, Interoperable and Reusable (FAIR) best practices by tracking the metadata throughout the entire sequencing pipeline to enable experimental reproducibility [22]. Previously, we presented a proof-of-concept vision of PEGR software [23]. We now present a fully functional and open-source version of PEGR software, including streamlined experiment

tracking with reagent barcoding, flexibility for multiple heterogeneous bioinformatics workflows, and a project management approach. PEGR is freely available and open source at <https://github.com/seqcode/pegr>.

Results

Overview

PEGR is a project management platform designed to organize, track, and disseminate the workflow of (epi)genomic projects from the start of an experiment through DNA sequencing, bioinformatic analyses, and figure generation (**Figure 1**). It tracks sample information, sequencing metadata, manages the bioinformatics workflow, and provides QC-reporting and visualization. PEGR is intended to enable a more complete scientific workflow from hypothesis generation through publication-quality figures.

PEGR supports user submission of detailed sample and experiment information through several methods, including a web-interface, real-time QR reagent barcode tracking with an Android app, and an Excel-based sample submission form. After experimental metadata has been recorded in PEGR (e.g., cell line, species, assay), a sequencing run can then be relationally linked to this information. PEGR tracks Illumina sequencing runs in real-time by periodically probing the sequencer's output data repository. When PEGR detects the completion of a sequencing run (i.e., RunCompletionStatus.xml), it will automatically initiate an external bioinformatics workflow platform. Presently PEGR natively supports the Galaxy platform and is extensible to other workflow engine platforms such as Pegasus and bash shell scripting [16]. PEGR collects the metadata information of bioinformatic outputs in real-time and displays them in an online workflow monitoring dashboard. Users are also able to query PEGR programmatically using a RESTful API for metadata related to specific samples or workflow runs. Critically, PEGR maintains and tracks the relational links between the figures and analyses generated by the bioinformatic software back through the starting reagents for any given sample.

Inventory management

PEGR provides an inventory management section in support of a key aspect of experimental reproducibility, creating the ability to simply and easily track the exact chemicals, enzymes, reagents, antibodies, equipment, and sample material that form the basis of experimental assays. While this form of reagent tracking is accomplished by most laboratories at a basic level, in practice, the quality of record-keeping can be variable depending on laboratory organizational structure and personnel training [14]. PEGR was architected to include an integrated inventory management system that seamlessly tracks all aspects of an experiment's metadata. To reduce incorrect and faulty information from being uploaded into PEGR, approved inventory entries (ItemType) can be pre-defined by administrative users (**Figure 2A**). Metadata fields such as name, vendor, catalog number, and lot number can be added to each ItemType to help guide the initial deployment and match LIMS fields available for import.

PEGR was designed to provide maximum flexibility for an assay workflow. Custom fields can be defined for a specific item type in the admin console. Given the wide range of possible reagents and variables a lab may choose to track, PEGR provides a simple CSV upload form which allows an administrative user to upload a list of all inventory item types and the related aspects of inventory metadata that a laboratory desires to track (**Figure 2A**). This provides compatibility with common laboratory management systems such as Agilent iLab and Quartzzy which possess CSV export functionality of their tracked inventory [18, 19]. To prevent disorganization resulting from tracking all possible inventory items in a laboratory, ItemType's are organized into an ItemTypeCategory (**Figure 2B**). This structure allows for the intuitive organization of the full spectrum of a laboratory's inventory. The item types in the ItemTypeCategory list are defined through the web interface and provides the ability to dynamically assign and re-assign ItemTypes as the needs of the laboratory change.

The '**Inventory**' tab on the main PEGR navigation bar is the primary interface for tracking all instances of ItemTypes in a laboratory. New instances of ItemTypes can be created and old instances can be marked 'inactive' by regular users (e.g., technicians, graduate students) as reagent stockpiles are finished (**Figure 2C**). To reduce the 'activation energy' required for adopting and maintaining in-depth tracking of reagent catalog numbers, specific lot numbers, aliquot dates, etc., PEGR leverages an easy-to-use QR barcode scanning application (ZXing) on Android devices that updates the PEGR backend database system in real-time [24]. The barcode scanner can be activated directly from a webpage in PEGR, and the result is returned to PEGR via a callback URL. Materials received by the lab that already have an attached barcode can be scanned from the Android devices and the appropriate metadata is recorded along with a time stamp. Purchased reagents and client samples with no existing barcode can be assigned a new barcode generated by PEGR. The barcode is shown in both text and 2D QR image in PEGR. PEGR's barcode system also integrates with existing label printers which allows for the 2D image to be printed in different sizes to accommodate the physical dimensions of the inventory.

Figure 2. Inventory management system. (A) Browser shot of PEGR Admin console showing an example ItemType list. CSV files can be generated from existing lab management platforms such as iLab and Quatzzy. (B) Browser shot of PEGR Admin console showing an example ItemTypeCategory list. (C) Example of instantiating a new ItemType in PEGR under the 'Inventory' interface. Boxes and arrows from red to blue to green, highlight the steps taken in order, showing the links and windows that appear for generating a new barcode. QR barcode sizes are set in the web interface and are compatible with most label printers.

Experimental protocol versioning and integration with inventory management

PEGR provides a protocol assembly and management module. While the traditional method of protocol management for most wet laboratories is a physical paper binder containing common buffer recipes and basic experimental procedures, there are cloud-based approaches for experimental protocol management such as OpenWetware (<https://openwetware.org>) and Protocol-Online (<http://www.protocol-online.org/>). In contrast to these approaches, PEGR's protocol management system directly links its laboratory inventory metainformation with tracked and version-controlled experimental protocols (**Figure 3A**).

Defining the exact ItemType input and output for each protocol is crucial for PEGR to properly track laboratory metadata across an experiment. When new experiments are initialized, the user is required to follow the predefined protocol and record the items used during an experimental setup. It is required that all the item types defined in the protocol be linked to an item instance before the user can move on to the next step. The process of defining a new protocol in PEGR is possible through two distinct options under PEGR's '**Protocol**' tab. The first method of generating a new protocol is directly through the PEGR user interface. A simple webform links to PEGR's ItemType database and allows for the creation of protocols ranging from simple buffer recipes to highly complex multi-stage assays such as ChIP-seq using a controlled reagent vocabulary (**Figure 3B**). The other method of protocol initialization uses a CSV file upload similar to the one used by the ItemType tab (**Figure 3C**). This allows for bulk upload of multiple protocols, a convenient feature for adding in a large number of novel assays.

The minimum requirements for creating a new protocol include a protocol name, version number, and a protocol description. Users are encouraged to also upload a protocol file in PDF format that is stored by PEGR for users to download and print. When creating a new protocol, a user selects starting and ending materials for the protocol. These fields are within an enforced list of all ItemTypes currently defined in PEGR. It uses autocomplete to assist in creation. In the case of a simple buffer protocol, the individual components of the protocol are the input ItemTypes (i.e., 5M NaCl, 500 mM EDTA) and the end product is the final buffer (e.g., NaCl 250 Wash Buffer). In the case of a protocol such as PCR, the 'Traced Sample' field is also used to track a sample's state entering and exiting a protocol stage. The concept of a 'Traced Sample' allows PEGR to link a final sample across multiple protocols that the sample may participate in (**Figure 3D**).

Similar to how ItemTypeCategory is used to organize the wide variety of ItemTypes in the '**Inventory**', Protocol Groups are used to consolidate and organize the variety of protocols that often come sequentially in a pipeline (**Figure 3E**). Protocol Groups contain any number of protocols (e.g., ChIP reaction, DNA end repair, etc.) in an ordered set. This enforces both experimental organization as well as provides flexibility for a user to generate a new Protocol Group (e.g., ChIP-seq v2) by re-using previously defined protocols in combination with new protocols.

A

Lab Protocols

Name	Version	Description	User	Approved
2.5M Glycine Recipe	1	Makes 2.5M Glycine for quenching formaldehyde	labadmin	Y
1M LiCl Recipe	1	Makes 1M LiCl stock solution	labadmin	Y
5M NaCl Recipe	1	Makes 5M NaCl stock solution	labadmin	Y
20% NP-40 Recipe	1	Makes 20% NP-40 stock solution	labadmin	Y
20% SDS Recipe	1	Makes 20% SDS stock solution	labadmin	Y
25% Triton X-100 Recipe	1	Makes 25% Triton X-100 stock solution	labadmin	Y

B

Create Protocol

Name: Wash Buffer Recipe Version: 1.0

Assay: ChIP-seq

Description: Makes 500 mL NaCl based Wash Buffer

Shared Item Types (multi-select):

- ☒ 10% NaDeoxycholate
- ☒ 1M HEPES-KOH pH 7.5
- ☒ 25% Triton X-100
- ☒ 500mM EDTA
- ☒ 5M NaCl

End Product Type (multi-select):

- ☒ NaCl 250 Buffer

Required Images: (multiple fields should be separated by comma)

Traced Samples:

Start State: Select...

End State: Select...

Add-on: ☐ Antibody ☐ Index

Sample Pool:

Import Pool: Select... Create Pool: Select...

URL:

Protocol approved? Y

Choose File: NaCl250_Wash_Recipe.pdf (only pdf files)

Save Cancel

C

Import CSV file

Choose File No file chosen

Upload

CSV file template

Name	Version	Assay	Description	Shared Item Types	End Product Types
2.5M Glycine Recipe	1	ChIP-seq	Makes 2.5M Glycine for quenching formaldehyde	Chemical:Glycine	Buffer:2.5M Glycine

Notes:

- The first row lists the headers.
- The sequence of the columns should be exactly the same as follows.
- An item type (columns E, F, H, I, L and M) should follow the format 'CATEGORY:NAME'.
- Multiple item types or images should be delimited by comma ",".

D

Protocol: ChIP DNA PCR 1

Assay: ChIP-seq

Description: Amplify Illumina libraries

Shared Item Types: Enzyme - Phusion Hot Start II DNA Polymerase Buffer - 5X Phusion HF Buffer

End Product Types:

Required Images:

Traced Sample:

Start State: Biosample - DNA Library

End State: Biosample - DNA Library

Add-on: ☐ Antibody ☒ Index

Sample Pool:

Import Pool: Create Pool:

Protocol File:

Choose File No file chosen

Upload (only pdf files)

Protocol approved? Y

E

ProtocolGroup List

Name: ChIP-seq Protocol

User: labadmin

Protocol Group List

Protocol Group: ChIP-seq Protocol

Date Created: 2021-06-21 18:09:24 EDT

Protocols:

- ChIP Reaction 1
- ChIP DNA End Repair 1
- ChIP DNA A-tailing Reaction 1
- ChIP DNA TruSeq Adapter Ligation 1
- ChIP DNA Elution and Purification 1
- ChIP DNA PCR 1

Figure 3. Protocol management system. (A) Browser shot of PEGR ‘**Protocols**’ interface. Default view displays protocols that have been approved by an administrator. Personal (nonapproved) protocols are available under ‘My Protocols’. (B) Browser shot of PEGR webform to create a new protocol. PEGR inventory database autocompletes fields where possible in Shared Item Types (input) and End Product Type (output) fields. (C) A CSV webform to upload protocols in bulk to the PEGR ‘**Protocol**’ interface. A template guide appears to assist upload. (D) Traced samples are defined in a ‘Lab Protocol’ and represent a sample that undergoes changes to its state (e.g., cell pellet into purified DNA). (E) ‘ProtocolGroup’ is accessible through the Admin console and provides a webform and CSV option for defining new ProtocolGroups based on approved protocols.

Tracking experimental metadata as it is generated

PEGR provides a section to record experiment metadata under the ‘**Experiment**’ tab. In designing a section devoted to capturing experiment metadata, we considered that an experiment involves a) inventory, b) a protocol, c) an input sample along with controls, and d) a resulting product. The PEGR ‘**Experiment**’ interface is designed to track and maintain the relational links between reagents (i.e., ‘**Inventory**’), protocols (i.e., ‘**Protocol**’), and the resulting end products. A new experiment can be initiated directly from the web interface (**Figure 4A**). A new experiment can be assembled by combining any number of previously defined protocols into any desired organizational structure (**Figure 4B**). Alternatively, a user can initialize a new experiment based on a Protocol Group (**Figure 4C**). This provides an easy mechanism for quickly assembling common laboratory protocols and assays in a structured and well-defined manner.

Once an experiment is initialized in PEGR, the experimental metadata and status can be updated directly through the web interface. Starting a simple experiment (e.g., creating a wash buffer) allows users to add relevant inventory metadata to PEGR using a wizard-style guide that walks users through adding reagents and all their associated metadata that have previously been stored in PEGR inventory (**Figure 4D**). While this can all be performed directly through the web interface, PEGR leverages a QR barcoding system, similar to the inventory system, to allow users to progress through experimental stages and collect their metadata in real-time using a hand-held Android device. This information includes but is not limited to Protocol ID, Reagent ID, Equipment ID, Tech ID, date, etc. Thus, each scanned item is linked to each experiment as associated metadata. Although we display the functionality of the webform for visualization purposes, the QR barcode scanner is the recommended method for linking experimental metadata in PEGR. In cases where the inventory item has never been previously instantiated within PEGR, a web-interface allows the user to define a new QR barcode and instance of the inventory item.

A typical lab process is to generate common laboratory reagent stocks (e.g., wash buffers) that are used multiple times across many different downstream experiments. However, more complicated experimental setups like ChIP-seq, involve a ‘traced’ sample which moves through multiple sequential experiments and combines with different reagents as it transitions through product states (e.g., sonicated chromatin converts to DNA library). Importantly in the case of traced samples, PEGR can display all of the states that a sample has transitioned through allowing for full experimental history tracking. A traced sample can be added to an experiment using either the web-interface or the QR barcode system (**Figure 4E**). Importantly, PEGR allows multiple samples to be attached to a single protocol. This enables the operator to process multiple samples in a batch while only needing to enter the related information once (e.g., when performing ChIP-seq on 8 samples in parallel).

Figure. 4 Experiment management system. (A) Browser shot of PEGR's 'Experiment' interface showing all tracked experiment records. The red box and arrow display the new webform that appears when a user initializes a new experiment (B) The black circle indicates the effect of selection custom

protocol construction which pulls from PEGR's approved protocol database with autocomplete. (C) The black circle indicates the effect of electing a Protocol Group which automatically loads and manages the requirements for the entire experiment. (D) Starting the experiment (red boxes and arrow) produces a webform for users to add relevant inventory metadata to PEGR. The transition of colored boxes (orange, yellow, green, and blue) displays the workflow for adding inventory metainformation to an experiment. (E) For experiments tracking traced samples, an additional interface (red box and red arrow) appears on the webform which allows users to search PEGR's database for existing compatible reagents (i.e., cell pellet) to add to the experiment.

Sequencing and automated bioinformatic workflows

PEGR provides a section to record metadata for DNA sequencing and bioinformatic data processing. Samples processed in parallel through PEGR's 'Experiment' module are natively grouped as "cohorts". Cohorts are typically generated by a researcher when addressing specific questions within a scientific project. The biochemical end for these (epi)genomic cohorts is typically high-throughput sequencing (or other detection systems) and downstream data analysis. As the throughput of DNA sequencers continues to expand, the available sequencing bandwidth for any given sequencing run will often exceed the needs of a cohort of samples. As a result, multiple cohorts are often sequenced together in a single sequencing run. These multiplexed samples may originate from different scientific projects (**Figure 5A**). Reciprocally, one or more cohorts comprise a Project, to which cohorts may be added over time. Therefore, we define a 'sequencing cohort' as the group of samples that belong to the sample project and a specific sequencing run.

PEGR provides substantial integration with common Illumina platforms. It implements a real time workflow tracking and quality control dashboard through integration with external bioinformatics systems, such as the Galaxy platform (**Figure 5B**) [15]. Galaxy workflows designed to communicate with PEGR contain simple XML-wrapper python scripts which send a JSON file to PEGR in a standard POST request. The JSON file contains a variety of information tracked by PEGR, and one critical element is the History ID from Galaxy. This allows PEGR to connect reproducible bioinformatic Galaxy workflows with the biochemical records stored and managed by PEGR [25].

As the output data from each analysis step returns to PEGR at the completion of each step, the status of the workflow is updated in real time (**Figure 5C**). The status of each analysis step being tracked is represented by a square. If the script completes successfully and passes the preliminary validation, the square will be colored in green. A script that results in one or more error messages has its square colored red. Clicking on the square renders the error messages in detail. API calls with 'permission denied' have their square colored orange, and analysis steps with missing datasets have their square colored blue. For analysis steps that have not communicate back to PEGR, the square remains in gray. If all squares become green, it indicates that the entire workflow has completed successfully. Note that bioinformatic workflows may vary for different sample types, and they may include different sets of analysis steps. To accommodate different workflows, PEGR defines a configuration for each workflow that lists all the analysis steps to be tracked. The workflow tracking panel is dynamically rendered according to this configuration.

In addition to tracking the status workflow-specific metadata (e.g., peak-calling completes successfully, MEME failed), PEGR also tracks assay-independent quality control metrics such as total reads per sample, adapter dimers, mapped reads, uniquely mapped reads and PCR-duplication level (**Figure 5C**). Through the web interface, the administrator may define the acceptable range for each field indicated at the header and fields that have values outside the acceptable ranges are colored in red. This combination of statistics gives users an overview of the quality of the sequencing experiment. The thresholds for what constitute an acceptable result are user-specified through the web-frontend of PEGR. After reviewing the statistics, Admins can indicate if the sample has passed the quality control check and been "verified". If the statistics indicates that errors may exist in the sequencing result (e.g., incorrect adapter index assignment), the authorized user can "delete" the sample directly on this page.

The workflow tracking and quality control dashboard can become quite wide as there is no upper limit to the number of scripts (i.e., columns) that may be tracked in PEGR. Users can hide columns by clicking the "-" sign on the header. The columns can be restored by clicking the "+" sign at the top. Multiple bioinformatics workflows can be applied to the samples in a single sequencing run or even a single sample. In this case, PEGR will display separate tabs for each workflow.

Reporting, visualization, and data dissemination

The Reporting module of PEGR is an interface to report, visualize, and disseminate the data it stores. PEGR provides a ‘Project’ interface to organize sequencing cohorts and individual

samples in a reporting and visualization dashboard (**Figure 6A**). The project dashboard provides links to the dynamically generated reports of entire cohorts of samples or individual samples (**Figure 6B**). This interface also provides a mechanism for granting project permissions to the various users of PEGR. This allows PEGR to provide its stored sample metainformation to external collaborators while limiting their ability to access all data within PEGR that may not apply to a specific shared project.

When selecting an individual sample in PEGR, users are presented with a custom report that contains all affiliated metadata with that sample (**Figure 6C**). Understanding that compliance can be difficult to achieve in certain settings, PEGR does not require a complete metadata trail for data visualization and will only display the data that it has. In addition to providing the biochemical sample metadata, PEGR will also display the results of the bioinformatic Galaxy workflows live-streamed directly from Galaxy (**Figure 6D**). As a key feature, PEGR does not duplicate raw or processed data files in Galaxy. PEGR only stores the relevant metadata needed to point to sequencing datasets and downstream analysis stored on Galaxy (i.e., Galaxy HistoryID). This design choice allows PEGR to track millions of sample details with a single CPU server. The current version of PEGR used by the Cornell EpiGenomics Core tracks ~70 Tb of Galaxy-generated analyses using less than 10 Gb of hard disk space.

project. The red box and arrow show the transition to the individual project page. **(B)** Browser shot of example PEGR project displaying affiliated users, associated experiments, sequencing cohorts, and samples. The red box and arrow show the transition to the individual sample page from the affiliated project. **(C)** Browser shot of example sample in PEGR displaying all known metainformation and details of experiments performed with this sample. The black arrow represents the live communication with the Galaxy webserver hosting the bioinformatics analyses of the sample. **(D)** Data hosted on Galaxy is dynamically displayed in the same PEGR web frame that reports on sample metadata to link the bioinformatics back to the experimental metadata.

Conclusions and Discussion

PEGR is a management platform for epigenomic and genomic research pipelines. It supports reproducibility by tracking samples from the very first step of sample preparation to the end of bioinformatic analysis and data reporting in support of the FAIR principle and the data reproducibility goals of the Galaxy platform proposed to guide scientific data management [22, 26]. PEGR links people, samples, protocols, sequencers and bioinformatics computation together, and facilitates research on genome regulation. The ability to visualize the downstream analyses of samples of interest within the same frame as the biochemical information helps users to better interpret and understand the results of their experiments. One of the primary challenges of genomic research is how to make different experimental assays and subsequent bioinformatic analysis reproducible and available through different organizations.

To tackle this challenge, we developed PEGR to provide a flexible system that maintains scientific rigor. PEGR provides a continuously tracked link from the original reagent preparation all the way through to the final downstream bioinformatic processing and figure generation. To date, PEGR has been used to support publication of over a dozen papers comprising thousands of unique experiments across a range of distinct assays (e.g., ChIP-exo, RNA-seq, PIP-seq) [27-31]. PEGR is adaptable to manage and organize common genomic assays with support for tracking any metainformation deemed of value by an administrator of PEGR. Through PEGR's RESTful API, analysis results can be accepted from any client (e.g., Galaxy and Pegasus) and the workflow tracking dashboard is configurable to track any number of distinct informatic workflows. The RESTful API also provides a mechanism by which PEGR can be queried remotely to output its stored metadata to any other desired system.

Future PEGR development will focus on supporting additional bioinformatic workflows and genomic assays. The current supplied bioinformatic analysis processing workflow is hard-coded to the Illumina sequencing platform. Future upgrades that can be made to PEGR include providing compatibility with non-Illumina sequencing pipelines (e.g., PacBio, Oxford Nanopore), enhancing the sample submission process using the native web interface, and enhancing role security to provide compliance with the EU GDPR, NY SHIELD, and California CCPA privacy laws for storing de-identified patient meta-information. These upgrades and more provide a clear path forward for providing rigorous and reproducible research across the biochemical and biomedical fields.

Methods

Architecture

The software stack used in the development of PEGR includes OpenJDK 1.8.0_212, Groovy 2.4.17, Grails 3.3.10 and MariaDB 10.4.6. PEGR is built on Grails, a high productivity web application framework for JVM [32]. Grails follows the “coding by convention” paradigm, and provides mechanisms such as injection, templating, and scaffolding, which makes the development much more efficient. The architecture of PEGR follows the Model-View-Controller (MVC) pattern [14]. The model layer contains 75 domain classes that are mapped to the database. In support of open-source software, we chose MariaDB to host PEGR's database. The database is a relational database that contains normalized tables for various information, including users (**Supplemental Figure 1A**), projects (**Supplemental Figure 1B**), sample details (e.g., strain, antibody, target, and growth media) (**Supplemental Figure 1C**), sequencing run details (**Supplemental Figure 1D**), and bioinformatics analysis (**Supplemental Figure 1E**). Data in the database are queried and persisted through Grails' object relational mapping (GORM). During development, the database schema often needs to be updated to meet stakeholder's requirements, e.g., adding or removing a table or a column. In this situation, we follow the code-first approach, that is, we first change the involved domain classes, run Grails database migration plugin, and the database will be automatically updated. This will guarantee synchronization between the codes and the database schema.

The controllers generate responses to clients' requests. For HTML requests, the controllers will delegate them to the views where data will be presented. Inside views, PEGR heavily utilizes JavaScript, Bootstrap library, and AJAX to improve user experience. It adopts the responsive web design and provides consistent data presentation across desktops, tablets and mobile devices. In addition, REST-compliant (RESTful) APIs are made available that allow external applications to query data from and send data to PEGR. In response to an API request, the controllers will render the data in JSON format. Note that the business logic in the controller can become complicated. For example, it may involve nontrivial data manipulation and decision making. In such cases, it will push the business logic to a separate layer, called "service", leaving the controllers relatively light weighted. The separation of models, controllers, views, and services confines different logic concerns to their own layer and makes it possible to reuse shared components. This enables the application to be easily developed, tested, and maintained.

Supplemental Figure 1. MariaDB table schematic underlying PEGR. (A) Relational structure for User roles in PEGR (B) Relational project structure (C) Relational structure for sample details (D) Relational structure for tracking sequencing runs relative to samples (E) Relational structure for tracking bioinformatic analyses and linking them to samples.

System requirements

PEGR is an extremely lightweight software platform. A production PEGR system tracking ~30,000 unique experiments across >500 sequencing runs serves multiple concurrent users with no detectable delays on a CentOS 7 server with 1 CPU, 8 Gb of RAM, and 200 Gb of hard disk space. By keeping all data analysis in a remote Galaxy webserver, data duplication and the file system footprint is kept to a minimum.

PEGR Guide

A quick-start on installing PEGR is provided on GitHub <https://github.com/seqcode/pegr>, and additional tutorials on how to configure PEGR and deploy it to production are provided on GitHub Wiki <https://github.com/seqcode/pegr/wiki>. Once PEGR is running, users can find the tutorial under the ‘Guide’ tab of the PEGR interface which details the various extended capabilities of PEGR (**Supplemental Figure 2A**).

RESTful API

For advanced users, PEGR provides a RESTful API for users to query and download the sequencing data and analysis results. Similar to the web interface, users can query samples based on sample ID, strain, antibody, target or run ID, and PEGR will return the qualified samples along with their major statistics and the links to the sequencing datasets such as the BAM files. The multiple ways that PEGR provides for reporting and visualization make it easy to share and further utilize the sequencing data. Guidance on how to interact with PEGR and what API commands are available are detailed at ‘Bioinformatics API’ under the PEGR Guide tab (**Supplemental Figure 2B**). This guide provides sample code and examples on how to connect to PEGR’s APIs in several common programming languages (Java, Python).

Supplemental Figure 2. Guide to advanced PEGR operations. **(A)** The ‘Guide’ tab on the PEGR navbar provides a comprehensive breakdown on PEGR’s structure and usage. **(B)** Details and sample code for accessing PEGR’s RESTful API are provided here as well in two common programming languages (i.e., Java and Python)

Orchestrating bioinformatics

Since an Illumina sequencer deposits data to a pre-defined designated repository, a cron job can be setup on the PEGR-hosting server to probe the sequencer to check on the status of new sequencing run. Once PEGR detects the completion of the sequencing run, it will match the sequencing output data with the information stored in its database and initiate a bioinformatics workflow. PEGR creates a set of files that contain the sequencing run information, e.g., the sequencing run ID, the path to the raw sequencing data repository, sample ID, the library index attached to a sample (a way to biologically identify each sample in a sequencing run), and the reference genomes to align the samples to. PEGR supports alignment of sample to more than one reference genomes (e.g., hg19 and hg38) and tracks the results of multiple alignments.

External clients, e.g., bioinformatics scripts or a workflow management tool such as Galaxy [15], can then sync with the relevant sample metainformation (i.e., genome build) and process the samples through all selected workflows. A typical processing workflow for ChIP-seq are comprised of a series of analysis steps, including raw data transformation, sequence alignment against reference genomes, peak-calling, and motif discovery. Since PEGR is charged with hosting all the metadata and final reporting, the output data needs to be communicated back to PEGR. We developed a RESTful API in PEGR to accept POST requests that contain the output data generated from the bioinformatics workflow (**Figure 5B**). Each API call corresponds to a single analysis step in the workflow. When an analysis step finishes, its output data will be posted to PEGR immediately. The benefits of sending results immediately instead of gathering all the results at the end of the workflow in a batch include that (1) we can track the status of the workflow run in real-time and (2) in the event that an analysis step failed (e.g., cluster error, networking issues), we can resume the workflow from the break point.

The data sent through the API needs to be constructed in a JSON format and the fields accepted are listed in **Table 1**. The field "userEmail" identifies the author of the analysis in workflow management tool and is also used to authenticate the user in combination with the API key. The fields "run" and "sample" are used to match the analysis to the sequencing run and the sample already submitted to and stored in PEGR, and "genome" is the reference genome that the sample is aligned to in this analysis. The analysis workflow and step are labeled by the corresponding "workflowId" and "workflowStepId". There could be multiple workflow runs on the same sample and each analysis step sends its results separately, therefore, we need the field "historyId" to uniquely map analysis steps to each workflow run. The software and its version used in the step is recorded in the field "toolId". In bioinformatics, many of the tools achieve the same functionality. Therefore, we also record "toolCategory" to facilitate reporting and future comparison. The "parameter" field is formatted as a dictionary with each input parameter name of tool is the key in the dictionary and value of the parameter is the value in the dictionary. The storage of all the parameters, along with the software and version used, helps with reproducibility and potential extension in the future. The output of the steps ranges from simple statistics (e.g., % read alignment to reference genome) to more complicated datasets (e.g., MEME motif results). The former is sent in the "statistics" field. As for the large datasets, the path to the dataset file and the file type are sent in the "datasets" field. Both "statistics" and "datasets" are formatted as a list of dictionaries.

Table 1 PEGR API fields that accept analysis results

Field	Format	Example
userEmail	String	"xxx@xxx.xxx"
run	Long	123
sample	Long	10023
genome	String	"sacCer3"
historyId	String	99afe35e5e550d2c
history url	String	https://xxx.xxx/view?id=99afe35e5e550d2c
alignmentId	Long	12345
workflowId	String	"b7p9d3k629e985d6"
workflowStepId	String	"d42c9e9785b96e77"
statsToolId	String	'bam to scidx output stats'
toolId	String	"toolshed.g2.bx.psu.edu/repos/iuc/bam_to_scidx/bam_to_scidx/1.0.1"
toolCategory	String	"output bamToScidx"
parameters	{ "parameterName": "parameterValue" }	{ "dbkey": "sacCer3", "require_proper_mate_pairing": 0 }
statistics	[{ "statisticsName": "statisticsValue" }]	[{ "genomeCoverage": 0.0206 }]
datasets	[{ "type": "xxx", "id": "xxx", "uri": "xxx" }]	[{ "type": "scidx", "id": "a39ec9e14951b012", "uri": "https://xxx.xxx/datasets/9e14951_a39ec_b012" }]
toolStderr	String	'some error'

There can be various downstream analysis following the core bioinformatics workflow. And results from those downstream analysis can be posted to PEGR through the same API and

linked to its upstream alignment using the field "alignmentId". The client may also send a "note" to PEGR, such as warning or error messages.

In order to maximize flexibility and interoperability with other software platforms. PEGR is also capable of hosting links to external platforms. These tables are accessible under the Guide tab in PEGR (**Supplemental Figure 2B**). The genome build table in the pegrDB links to the external source of the genome (e.g., UCSC, SGD). In order to further support bioinformatic reproducibility outside of the Galaxy platform, PEGR also tracks common reference features (gene coordinates, protein binding locations, origins of replication) which are hosted on GitHub.

Security and privacy

PEGR leverages Spring Security to control the access to the application [33]. Spring Security is a framework that provides authentication, authorization and other security features for enterprise applications. For authentication, PEGR uses four mechanisms: "daoAuthenticationProvider", "preAuthenticatedAuthenticationProvider", "anonymousAuthenticationProvider" and "rememberMeAuthenticationProvider". Through the "daoAuthenticationProvider", users can login to PEGR using their registered username and password. The "preAuthenticatedAuthenticationProvider" works with external identity hosts so that users don't have to create additional password with PEGR. It supports both header-based and attribute-based authentication, e.g., CoSign (<http://weblogin.org>) and Shibboleth (<https://www.shibboleth.net>) Single Sign-on. The "anonymousAuthenticationProvider" denies access to the application from unauthorized users except for the login page; and the "rememberMeAuthenticationProvider" stores the login information in a cookie and allows for automatic login for an extended period.

PEGR authorizes data access through a number of security layers. Primarily, PEGR implements Role Based Access Control which is mostly implemented through URL intercept. It assigns a role group to each user, and users in a specific role group can only access the URL's authorized for that role group. For example, a user who is assigned to the "Admin" role group will have access to all the URL's, including those for the Admin console, and be able to update all the samples and projects. In contrast, users in the "Member" role group cannot access certain URL's, e.g., those for the Admin console. "Members" do have "read" access to all projects where they have been assigned, including the 'Inventory', 'Lab Protocols', 'Experiments', 'Sequencing Runs', and 'Samples'. Members do not have "write" access to those objects unless they have a specific "project role" (see below) or ownership to an inventory item or protocol. A user cannot access any part of PEGR until their ID is assigned to a role group. By default, users are initially added to the "Guest" role group with the most restricted access in the PEGR platform, only able to view unprotected information.

Additionally, more granular access controls are defined for each project and sample through controller interceptors. The controller interceptors apply certain logics across a group of controller actions, and they are executed before the related controller actions are invoked. For example, users can be added to a project with different roles, (e.g., "owner", "participant", or "guest"). While all users linked to the project can view the project summary and all the samples

in the project, only the owners and participants of the project have the ability to add (or remove) samples to (or from) the project and edit the samples that belong to the sample. And only the owners of the project can edit the project's information, such as project name, project description and funding source.

Acknowledgements

The authors thank Belinda Giardine, Abeer Almutairy, Hedgie Jo, and Pierce Chafflin for their user testing and feedback, code base tests, development code contributions and helpful discussions and feedback to this project. Computations for this research were performed on the Pennsylvania State University's Institute for Computational and Sciences' ROAR supercomputer.

Funding

This work was supported by the US National Institutes of Health (NIH) grants 5R01-ES013768 and 3R01-GM125722-03S1 for funding the development and dissemination of PEGR. The Authors acknowledge the support of the Institute for Computational and Data Sciences at the Pennsylvania State University through ICDS Seed Grants to Dr. B. Franklin Pugh and Dr. William KM Lai.

Authors' Information

Institute for Computational and Data Sciences, Pennsylvania State University, University Park, PA 16802, USA

Danying Shao

Department of Biochemistry & Molecular Biology, Pennsylvania State University, University Park, PA 16802, USA

Prashant K Kuntala & Shaun Mahony

Department of Molecular Biology and Genetics, Cornell University, Ithaca, NY 14850, USA.

Gretta D. Kellogg, Ali Nematbakhsh, B. Franklin Pugh & William KM Lai

Corresponding author

Correspondence to William KM Lai (wk129@cornell.edu)

Ethics declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

The PEGR software is publicly available at <https://github.com/seqcode/pegr>. The PEGR-Galaxy communication scripts are available at https://github.com/CEGRcode/pegr-galaxy_tools and the PEGR NGS pipeline scripts are available at https://github.com/CEGRcode/pegr-ngs_pipeline under the MIT license.

References

1. Baker M: **Reproducibility crisis: Blame it on the antibodies.** *Nature* 2015, **521**:274-276.
2. Hunter P: **The reproducibility "crisis": Reaction to replication crisis should not stifle innovation.** *EMBO Rep* 2017, **18**:1493-1496.
3. Stupp A, Singerman D, Celi LA: **The reproducibility crisis in the age of digital medicine.** *NPJ Digit Med* 2019, **2**:2.
4. Kukurba KR, Montgomery SB: **RNA Sequencing and Analysis.** *Cold Spring Harb Protoc* 2015, **2015**:951-969.
5. Buenrostro JD, Wu B, Chang HY, Greenleaf WJ: **ATAC-seq: A Method for Assaying Chromatin Accessibility Genome-Wide.** *Curr Protoc Mol Biol* 2015, **109**:21 29 21-21 29 29.
6. Rossi MJ, Lai WKM, Pugh BF: **Simplified ChIP-exo assays.** *Nat Commun* 2018, **9**:2842.
7. ENCODE: **An integrated encyclopedia of DNA elements in the human genome.** *Nature* 2012, **489**:57-74.
8. GTEx: **The Genotype-Tissue Expression (GTEx) project.** *Nat Genet* 2013, **45**:580-585.
9. Roadmap Epigenomics C, Kundaje A, Meuleman W, Ernst J, Bilenky M, Yen A, Heravi-Moussavi A, Kheradpour P, Zhang Z, Wang J, et al: **Integrative analysis of 111 reference human epigenomes.** *Nature* 2015, **518**:317-330.
10. Papageorgiou L, Eleni P, Raftopoulou S, Mantaïou M, Megalooikonomou V, Vlachakis D: **Genomic big data hitting the storage bottleneck.** *EMBnet J* 2018, **24**.
11. Reuter JA, Spacek DV, Snyder MP: **High-throughput sequencing technologies.** *Mol Cell* 2015, **58**:586-597.
12. Howard BR: **Control of variability.** *ILAR J* 2002, **43**:194-201.
13. Bakay M, Chen YW, Borup R, Zhao P, Nagaraju K, Hoffman EP: **Sources of variability and effect of experimental approach on expression profiling data interpretation.** *BMC Bioinformatics* 2002, **3**:4.
14. Pain E: **How to keep a lab notebook.** Science; 2019.
15. Goecks J, Nekrutenko A, Taylor J, Galaxy T: **Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences.** *Genome Biol* 2010, **11**:R86.
16. Deelman E, Vahi K, Juve G, Rynge M, Callaghan S, Maechling P, Mayani R, Chen W, Ferreira da Silva R, Livny M, K. W: **Pegasus, a workflow management system for science automation.** *Future Generation Computer Systems* 2015, **46**:17-35.
17. Skobelev DO, Zaytseva TM, Kozlov AD, Perepelitsa VL, Makarova AS: **Laboratory information management systems in the work of the analytic laboratory.** *Measurement Techniques* 2011, **53**:1182-1189.
18. Quartzzy: <https://www.quartzzy.com/>. 2021.
19. Agilent iLab: <https://www.agilent.com/en/service/laboratory-services/lab-operations-management>. 2021.

20. BCPlatforms: <https://www.bcplatforms.com/>. 2021.
21. Illumina Basespace: <https://basespace.illumina.com/>. 2021.
22. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten JW, da Silva Santos LB, Bourne PE, et al: **The FAIR Guiding Principles for scientific data management and stewardship**. *Sci Data* 2016, **3**:160018.
23. Shao D, Kellogg G, Mahony S, Lai W, Pugh B: **PEGR: A management platform for ChIP-based next generation sequencing pipelines**. *ACM International Conference Proceeding Series* 2020:285-292.
24. Owen S, Switkin D, Team Zx: **Barcode Scanner**. 2019.
25. Gruning B, Chilton J, Koster J, Dale R, Soranzo N, van den Beek M, Goecks J, Backofen R, Nekrutenko A, Taylor J: **Practical Computational Reproducibility in the Life Sciences**. *Cell Syst* 2018, **6**:631-635.
26. Blankenberg D, Von Kuster G, Bouvier E, Baker D, Afgan E, Stoler N, Galaxy T, Taylor J, Nekrutenko A: **Dissemination of scientific software with Galaxy ToolShed**. *Genome Biol* 2014, **15**:403.
27. Krietenstein N, Wal M, Watanabe S, Park B, Peterson CL, Pugh BF, Korber P: **Genomic Nucleosome Organization Reconstituted with Pure Proteins**. *Cell* 2016, **167**:709-721 e712.
28. Lai WK, Pugh BF: **Genome-wide uniformity of human 'open' pre-initiation complexes**. *Genome Res* 2017, **27**:15-26.
29. Badjatia N, Rossi MJ, Bataille AR, Mittal C, Lai WKM, Pugh BF: **Acute stress drives global repression through two independent RNA polymerase II stalling events in Saccharomyces**. *Cell Rep* 2021, **34**:108640.
30. Rossi MJ, Kuntala PK, Lai WKM, Yamada N, Badjatia N, Mittal C, Kuzu G, Bocklund K, Farrell NP, Blanda TR, et al: **A high-resolution protein architecture of the budding yeast genome**. *Nature* 2021, **592**:309-314.
31. Lai WKM, Mariani L, Rothschild G, Smith ER, Venters BJ, Blanda TR, Kuntala PK, Bocklund K, Mairose J, Dweikat SN, et al: **A ChIP-exo screen of 887 PCRP transcription factor antibodies in human cells**. *bioRxiv* 2021:2020.2006.2008.140046.
32. Smith G, Ledbrook P: *Grails in Action*. 2 edn: Manning; 2014.
33. Scarioni C: **Pro Spring Security**. APress; 2013.