Multiscale Complex Genomics

Project Acronym: MuG

Project title: Multi-Scale Complex Genomics (MuG)

Call: H2020-EINFRA-2015-1

Topic: EINFRA-9-2015

Project Number: 676556

Project Coordinator: Institute for Research in Biomedicine (IRB Barcelona)

Project start date: 1/11/2015

Duration: 36 months

Deliverable 4.3: Pipeline Design and Implementation

Lead beneficiary: The European Bioinformatics Institute (EMBL-EBI)

Dissemination level: PUBLIC

Due date: 31/10/2016

Actual submission date: 31/10/2016

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 676556.

Document History

Version	Contributor(s)	Partner	Date	Comments
0.1	Mark McDowall	EMBL-EBI	07/10/2016	First draft
0.2	François Serra	CNAG-CRG	17/10/2016	Added details about DNA-DNA Modelling
0.3	Mark McDowall	EMBL-EBI	17/10/2016	Citations and corrections
0.4	Diana Buitrago	IRB Barcelona	18/10/2016	Added nucleR for analysis of MNase-seq data
0.5	Brian Jimenez, Marco Pasi	BSC/UNOT	24/10/2016	Added details about protein-protein and protein-DNA modelling
0.6	Mark McDowall	EMBL-EBI	25/10/2016	New diagram, file format descriptions
1.0	Mark McDowall	EMBL-EBI	31-10-2016	Final version approved by supervisory board

Table of Contents

1 EXECUTIVE SUMMARY

The following document describes the pipelines that have been developed for the processing of data as part of the MuG Virtual Research Environment (VRE). These pipelines are a mixture of custom code and wrappers around standard tools used for the analysis of RNA-seq, whole genome bisulphate sequencing (WGBS), ChIP-seq, MNase-seq, Hi-C, protein-protein and protein-DNA interactions so that they are able to run on the COMPSs architecture. However, there are still challenges that need to be faced. In relation to microscopy datasets, such as FISH, these will be addressed as part of an on-going priority to manage the storage and analysis. Plus the ongoing task of ensuring that the current and new pipelines are fit for purpose as standards change and knowledge is gathered.

2 INTRODUCTION

There are a large number of experimental tools that are required to resolve the architecture of the chromosomes within the nucleus [vanSteensel2010]. The goal is to be able to integrate a large array of experimental techniques from the work packages and those already in the public domain host in data archives it requires the creation of a number of workflows that are able to process the raw data and generate datasets that are ready for analysis by the community. Being able to do this in an efficient manner will allow for the optimal use of resources while returning the results in reasonable time frame.

Here we present the pipelines that have been created for the processing of sequencing data and interaction modelling. These pipelines have been developed to work within the COMPSs environment and therefore allowing them to be used as part of the MuG VRE. The pipelines are also based on the principles that have been developed by the 6.1 deliverable for defining the interoperability of pipelines within the project. The flow chart from deliverable 6.2 highlights information pathways and the tools that currently exist. As tools are developed to allow for the flow of knowledge within this graph are developed new pipelines will be created.

3 PIPELINE DESIGN

The VRE acts as the major interface between the users and the pipelines. These pipelines run within the COMPSs distributed computing architecture allowing for processes to scale depending on their requirements. The design of the pipelines is based on the recommendations that have been made as part of the 6.1 development plan. This states that they should be modular, easily integrated into other workflows and have defined input and outputs. Each module can be one of 2 types: It can either be a wrapper around a binary function, such as an aligner; or it can be a bespoke function that calls native libraries without calling out the shell environment.

The pipelines are designed to be so that they can be run as part of other larger workflows within the VRE. The current set of pipelines have been designed to run as a single block of functions, but future iterations will allow for the separation of individual functions making each function more granular allowing for greater flexibility when creating workflows within the VRE.

When it comes to implementing each of the pipelines many fall into a common workflow. In the case of sequence based pipelines they require a data retrieval step, an alignment step and then a final transformation step (figure 1). After this point it is then possible to start developing tools manipulate the data and provide further analysis as required by the consortium and the wider community.

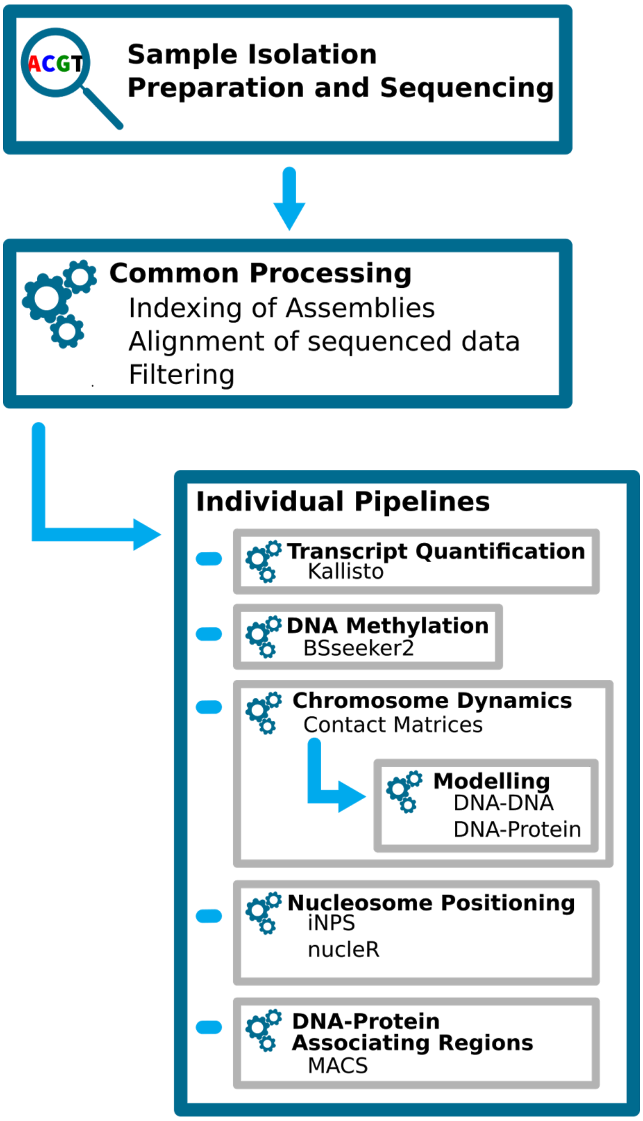

Figure 1: Shows the common flow from experimental data and the generation of raw data to final product. Once sequenced there are a set of conceptually common steps for the filtering and alignment of the reads. The results are then passed to the individual pipelines that for more detailed analysis.

Storage of these processed files is handled via COMPSs, by declaring the files that required as input and those that are defined as output from each of the processes by the use of Python decorators and stored locally within the compute infrastructure. The eventual storage of the data will be handled as part of the 4.4 deliverable in preparation for generation of the data API access (deliverable 4.5).

4 PIPELINE IMPLEMENTATION

Each of the pipelines that have been developed aim to encapsulate the basic functions needed to process the raw data from the sequencing machines into datasets ready for storage and eventual dissemination. Each pipeline leverages the capabilities of the COMPSs infrastructure (developed as part of work package 5) to allow for parallel processing of the data to provide results within a timely manner. All of the pipelines have been developed in Python so that they are compatible with COMPSs via the PyCOMPSs library, but this is also a widely adopted language within the bioinformatics community and therefore lowering the barrier for contributions for community members. Once each pipeline is ready it is tested locally on a virtual machine running COMPSs. This VM has been loaded into the virtual environment in Barcelona as part of the the OpenNebula [Moreno-Vozmediano2012]. From there it is possible to update the code as changes are made or as new pipelines are developed

To increase the openness of the project and allow for the external development of new pipelines all of the pipelines are stored within the MuG GitHub repository as reported in the milestone 13 [Marco-Sola2012]. The repository is open to community users so that they are able to fork and create new pipelines based on the latest code.

4.1 Common Functions

Many of the pipelines are designed around the use of a common set of functions or processes. These have been abstracted out so that if changes are required they can be easily made once, but improve all pipelines that they are involved in. Shared processes include the retrieval of data and assemblies, indexing and alignment of the assemblies and merging of files. These are shared as common functions that are available to the pipelines within the python library.

There are also common sources for data. Sequencing data is retrieved via the European Nucleotide Archive (ENA) [Leinonen2011], which is a partner in the International Nucleotide Sequence Database Collaboration (INSDC) [Cochrane2015]. Sequence data encompasses both the assembly (FASTA) and sequencing read data files (FASTQ). Structural data is retrieved from the Protein Data Bank (PDB) [Berman2000]. Handling these in a single place also means that if URLs change or a more favourable source is identified then the work required to make the change is kept to a minimum.

4.2 Chromosome Conformation Capture (Hi-C)

Hi-C is a method of locating regions of sequence that have come close enough together within the number that it is possible to cross-link the nucleotides. These regions are then extracted by using a selective enzyme to digest the genome and the 2 linked sections of the genome are then sequenced. The generated paired end FASTQ data for the 2 sequences can then be aligned back to the genome and used to infer a network of interactions that occur within the nucleus.

There are practical guidelines for the analysis of Hi-C data [Lajoie2015]. TADbit [Serra2016] has been adopted as the common software library for handling data Hi-C data as there is a close collaboration with the developers as part of the MuG project and it addresses issues highlighted by [Lajoie2015]. The first step in the pipeline is to acquire the sequence reads and the genome assembly, this is done via the common functions defined in section 4.1. The genome assembly is then indexed using the GEM library (http://algorithms.cnag.cat/wiki/The_GEM_library) to which the paired end sequence reads are mapped. The mapped reads are then filtered to remove duplicates and experimental abnormalities do not affect the final results. Each repeat is processed in the same way and the alignments are then merged to generate a final set of mapped paired end reads.

4.2.1 Adjacency Matrix

The adjacency matrices are generated using the TADbit library based on the mapped and merged paired end reads. The matrices are set to predefined resolutions. These different resolutions answer different questions when it comes to determining the modelled 3D structure of the chromosomes within the nucleus. The range of resolutions that the matrices are calculated for are 1kbp, 10kbp, 100kbp, 1Mbp and 10Mbp. The COMPSs architecture, as developed as part of Work Package 5, allows for each of the resolutions to be calculated in parallel. Other resolutions can be added if a finer granularity is required in the future. The output is stored both in the TADbit adjacency files, but as part of the deliverable for 4.4 it also generates a single HDF5 file (https://www.hdfgroup.org/) with all the separate resolutions using the python library h5py (http://www.h5py.org/). The HDF5 file allows for the efficient storage of the sparse adjacency file and allows for on the fly compression and data access, this is ideal for later use in the RESTful API as part of deliverable 4.5.

4.2.2 Hi-C - TAD Analysis

This is performed on a chromosome by chromosome basis, this allows the COMPSs architecture to run each chromosomes in parallel reducing the overall wall clock time for the generation of the data. The TADs are called using the TADbit library to generate the list of TAD regions at different resolutions (see section 4.2.1). The output is a tab separated list of the chromosome and the start and stop regions for each TAD. Work has begun to identify the optimal way to store this information (deliverable 4.4) to allow for a RESTful interface, both MySQL and Cassandra databases are proving to have comparable performance with the current volume of data (deliverable 4.5).

4.3 Whole Genome Bisulphate Sequencing (WGBS)

WBGS is a gold standard for measuring DNA-methylation across the whole genome. There are several common methods that can be used for the analysis of WGBS data [Kunde-Ramamoorthy2014]. Bismark [Krueger2011] has been commonly used within the BLUEPRINTs project, however to make things smoother for use within COMPSs, BSseeker2 [Guo2013] was selected. It has been found to have similar results to Bismark given that it uses the same aligner (Bowtie2 [Langmead2012]). The assembly and sequence reads are downloaded using the common functions defined in 4.1. The sequence reads are first filtered to remove experimental artifacts and the genome assembly is then indexed and the reads are mapped using Bowtie 2. To allow for the process to run quicker the reads are split then submitted to COMPSs for alignment to the assembly. The final alignments are then sorted and merged once all the alignments have completed. BSseeker2 is then able to make the final methylation calls based on the alignments. The results are provided as a mixture of a wig file, a CGMap file and an ATCGMap file.

4.4 RNA-Seq

RNA-seq can be used to infer the level of gene-expression on a genomic scale. This pipeline handles the mapping of the RNA-seq reads to the known transcripts for a species and then performs the quantification steps. The sequencing reads are downloaded using the common functions defined in 4.1 and the cDNA FASTA is downloaded from the Ensembl FTP site [Yates2016]. Kallisto [Bray2016] is used to index the cDNA sequences and then map the RNA-seq reads to the known transcripts. Kallisto is also used to quantify the level of expression for each transcript. The output is a single tab-separated file matching the transcript to the level of expression and a measure of the proportion of transcripts that are in the pool that was sequenced.

4.5 ChIP-Seq

The purpose of this pipeline is to identify the location of DNA associating proteins on the genome. The sequencing reads, background reads and assemblies are downloaded using the common functions. The reads are mapped to the assembly using BWA [Li2010]. The alignments for the repeats or each of the reads and background read sets are merged respectively using SamTools [Li2009] via the python library pysam (https://github.com/pysam-developers/pysam). The reads and background read merged alignments are then filtered using BioBamBam2 (https://github.com/gt1/biobambam2) to remove duplicates. The final peak calls to identify the location of proteins bound to the genome is performed using MACS2 [Zhang2008] based on the reads and background reads bam files.

4.6 Mnase-Seq

The purpose of this pipeline is to identify the location of nucleosomes within the genome and classify them according to their fuzziness. This is done by digesting the free DNA that is exposed to the enzyme, the remaining DNA is then sequenced. The sequencing reads and assembly are downloaded using the common functions. The assembly is then indexed and the reads are aligned using BWA [Li2010]. The calling of peaks can then be performed following two strategies. First the calling of peaks can be performed using iNPS [Chen2014]. iNPS requires the alignments as a bed file so the bam file generated by BWA is converted to a bed file using BedTools [Quinlan2002]. This has been implemented using a python script to run within the COMPSs environment for inclusion within the VRE. The final results are then saved to a wig and bed files. Second, nucleR R package can be used [Flores2011]. nucleR accepts alignments as a BAM file, internally produces nucleosome coverage signal, processes it based on Fourier analysis and applies a peak calling algorithm to find nucleosome dyads. The final result is saved as a gff file of the nucleosome peaks with the corresponding measures of nucleosome fuzziness.

4.7 Protein-Protein and Protein-DNA Structural Modelling

The three dimensional structure of a multi macromolecular complex can provide invaluable insight into the details of the recognition process. This pipeline aims at obtaining structural information on protein-protein and protein-DNA complexes, when such information isn't directly available from experiment (e.g. in the PDB). The starting point for this pipeline is the sequence of the interaction partners, that is strings of either aminoacids or nucleotides in FASTA format. First, the structure of each interaction partner alone is predicted: for protein, structural information on homologs will be used, when available, using the widely accepted homology modelling tool Modeller [Eswar2014]. The structure of DNA partners will be predicted from the sequence using NAFlex [Hospital2013], which uses information derived from long MD simulations of naked DNA [Pasi2014]. The resulting individual protein and DNA structural models are the input for the modeling of the protein-protein and protein-DNA complexes, using the *ab initio* docking tool pyDock [Cheng2007], with multiple runs of the docking procedure required to assemble multimolecular complexes. For example, within the VRE, the two structures in PDB format, one for the receptor protein and one for the DNA ligand, could be selected and then, the option to use "Protein-DNA" tool will be available in the VRE. Once this options is selected, an intermediate web page is displayed where the user will select the desired options of the protein-DNA docking (number of predicted structures, etc.). After the user clicks on the "Run" button, the VRE prepares the execution environment and the calculations are managed by a custom on-demand virtual machine which contains the software needed in order to perform the docking. During the whole calculation process, status and logs are available to the VRE via a REST API. Once the job has been completed, the VRE shows the results gathered from the custom virtual machine. The user has then the option of further refining the resulting structural models by using molecular dynamics simulations.

4.8 DNA-DNA Modelling

This pipeline takes as input normalized interaction matrices. Regions with low number of interactions are masked in the matrix and marked as unrestrained. The normalization applied on the matrix consists in 1 round of ICE [Imakaev2012], which is very similar to the vanilla normalization of [Rao2014]. Once normalized interaction matrices are Z-score transformed and used as a set of restraints to build 3D models. The restraints are applied to simulated particles randomly placed in space. Each particle represents a chromatin locus, or a bin in the interaction matrix. The optimization of the particle positions in space to better satisfy the restraint is archived through a Monte Carlo process implemented in IMP [Russel2012].

Pairs of loci interacting very frequently will be close in space while pairs of loci interacting rarely should be more distant. The definition of what means “distant” and the cutoff pass from repulsion to attraction are parameters that need to be optimized for each region to model. The optimization of modeling parameter is supervised through the correlation between input interaction matrices and simulated contact maps obtained from the 3D models. Finally a new, and more populated set of 3D models are computed with the combination of best parameters. This final set of models (usually 1000) will be used for the analysis.

More details on this pipeline are described in [Baù2012].

4.9 Fluorescence *in-situ* Hybridisation (FISH) Microscopy Data

At the moment it is not possible to get a functional pipeline in place for the handling of image data. Within the EMBL-EBI there are plans for the installation of an image archive based on the software from the Open Microscopy Environment (OME) [Allan2012]. This would all of for the storage of large sets of microscopy images along with providing some initial analysis of the images. The OME also has an API. This API would allow for pipelines to be written to run in the VRE to the retrieval the large sets of images for local processing and then submission of the results back to the OME. This would allow for stacks of FISH images to be downloaded, enhanced and then rendered into a 3D representation of the nucleus. There is a range of software that is able to perform the deconvolution steps and then stacking and rendering of the 3D images ranging from dedicated commercial software like AMIRA [Stalling2005], to open source image analysis libraries, such as ImageJ [Schmid2010].

As on work we will continue to monitor what are the latest analysis techniques and storage for FISH data. As methodologies converge we will actively work to bring our respective pipelines in line with those that are used within the research community as a whole.

5 CONCLUSIONS

Many of the initial pipelines for processing sequencing data and structural data are in place and can be deployed within the COMPSs environment and are therefore accessible via the VRE. The pipelines are designed around the principles proposed within the 6.1 deliverable defining the interoperability of the tools. Further work will be required as new pipelines are needed and the streamlining of the current pipelines to ensure that they remain fit for purpose within the VRE. There might also need to be changes to accommodate the chosen storage methods (deliverable 4.4) and data dissemination (deliverable 4.5).

Initial work has been done to identify the process of moving forward after the generation of the processing pipelines for the storage and eventual dissemination of the data via a RESTful API. Steps have been take to store the Hi-C adjacency data and then build an interface to the data. This has highlighted that there needs to be a level of modularity to the way that the data is eventually store to allow for changes in technology both to the way that data is stored and in the way that the data is served and the questions that biologists want to ask of the resource.

6 REFERENCES

[Allan2012] Allan, C., Burel, J.-M., Moore, J., Blackburn, C., Linkert, M., Loynton, S., *et al* (2012). OMERO: flexible, model-driven data management for experimental biology. Nature Methods, 9(3), 245–253. https://doi.org/10.1038/nmeth.1896

[Baù2012] Baù, D., & Marti-Renom, M. A. (2012). Genome structure determination via 3C-based data integration by the Integrative Modeling Platform. Methods, 58(3), 300–306. https://doi.org/10.1016/j.ymeth.2012.04.004

[Berman2000] Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., *et al* (2000). The Protein Data Bank. Nucleic Acids Research, 28(1), 235–242. https://doi.org/10.1093/nar/28.1.235

[Bray2016] Bray, N. L., Pimentel, H., Melsted, P., & Pachter, L. (2016). Near-optimal probabilistic RNA-seq quantification. Nature Biotechnology, 34(5), 525–527. https://doi.org/10.1038/nbt.3519

[Cachrane2015] Cochrane, G., Karsch-Mizrachi, I., Takagi, T., & Sequence Database Collaboration, I. N. (2015). The International Nucleotide Sequence Database Collaboration. Nucleic Acids Research, gkv1323. https://doi.org/10.1093/nar/gkv1323

[Chen2014] Chen, W., Liu, Y., Zhu, S., Green, C. D., Wei, G., & Han, J.-D. J. (2014). Improved nucleosome-positioning algorithm iNPS for accurate nucleosome positioning from sequencing data. Nature Communications, 5, 4909. https://doi.org/10.1038/ncomms5909

[Cheng2007] Cheng, T. M.-K., Blundell, T. L., & Fernandez-Recio, J. (2007). pyDock: Electrostatics and desolvation for effective scoring of rigid-body protein–protein docking. Proteins: Structure, Function, and Bioinformatics, 68(2), 503–515. https://doi.org/10.1002/prot.21419

[Eswar2014] Eswar, N., Webb, B., Marti-Renom, M. A., Madhusudhan, M. S., Eramian, D., Shen, M.-Y., *et al* (2007). Comparative protein structure modeling using MODELLER. Current Protocols in Protein Science, Chapter 2, Unit 2.9. https://doi.org/10.1002/0471140864.ps0209s50

[Flores2011] Flores, O., and Orozco, M. (2011). nucleR: a package for non-parametric nucleosome positioning. Bioinformatics *27*, 2149–2150. https://doi.org/10.1093/bioinformatics/btr345

[Guo2013] Guo, W., Fiziev, P., Yan, W., Cokus, S., Sun, X., Zhang, M. Q., *et al* (2013). BS-Seeker2: a versatile aligning pipeline for bisulfite sequencing data. BMC Genomics, 14, 774. https://doi.org/10.1186/1471-2164-14-774

[Hospital2013] Hospital, A., Faustino, I., Collepardo-Guevara, R., González, C., Gelpí, J. L., & Orozco, M. (2013). NAFlex: a web server for the study of nucleic acid flexibility. Nucleic Acids Research, 41(W1), W47–W55. https://doi.org/10.1093/nar/gkt378

[Imakaev2012] Imakaev, M., Fudenberg, G., McCord, R. P., Naumova, N., Goloborodko, A., Lajoie, B. R., *et al* (2012). Iterative Correction of Hi-C Data Reveals Hallmarks of Chromosome Organization. Nature Methods, 9(10), 999–1003. https://doi.org/10.1038/nmeth.2148

[Krueger2011] Krueger, F., & Andrews, S. R. (2011). Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications. Bioinformatics, 27(11), 1571–1572. https://doi.org/10.1093/bioinformatics/btr167

[Kunde-Ramamoorthy2014] Kunde-Ramamoorthy, G., Coarfa, C., Laritsky, E., Kessler, N. J., Harris, R. A., Xu, M., *et al* (2014). Comparison and quantitative verification of mapping algorithms for whole-genome bisulfite sequencing. Nucleic Acids Research, gkt1325. https://doi.org/10.1093/nar/gkt1325

[Lajoie2015] Lajoie, B. R., Dekker, J., & Kaplan, N. (2015). The Hitchhiker's guide to Hi-C analysis: Practical guidelines. Methods, 72, 65–75. https://doi.org/10.1016/j.ymeth.2014.10.031

[Langmead2012] Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. Nature Methods, 9(4), 357–359. https://doi.org/10.1038/nmeth.1923

[Leinonen2011] Leinonen, R., Akhtar, R., Birney, E., Bower, L., Cerdeno-Tárraga, A., Cheng, Y., *et al* (2011). The European Nucleotide Archive. Nucleic Acids Research, 39(suppl 1), D28–D31. https://doi.org/10.1093/nar/gkq967

[Li2009] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., *et al* (2009). The Sequence Alignment/Map format and SAMtools. Bioinformatics, 25(16), 2078–2079. https://doi.org/10.1093/bioinformatics/btp352

[Li2010] Li, H., & Durbin, R. (2010). Fast and accurate long-read alignment with Burrows–Wheeler transform. Bioinformatics, 26(5), 589–595. https://doi.org/10.1093/bioinformatics/btp698

[Marco-Sola2012] Marco-Sola, S., Sammeth, M., Guigó, R., & Ribeca, P. (2012). The GEM mapper: fast, accurate and versatile alignment by filtration. Nature Methods, 9(12), 1185–1188. https://doi.org/10.1038/nmeth.2221

[Moreno-Vozmediano2012] Moreno-Vozmediano, R., Rub&#x00E9, Montero, n S., & Llorente, I. M. (2012). IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures. Computer, (12), 65–72. https://doi.org/10.1109/MC.2012.76

[Pasi2014] Pasi, M., Maddocks, J. H., Beveridge, D., Bishop, T. C., Case, D. A., Cheatham, T., *et al* (2014). µABC: a systematic microsecond molecular dynamics study of tetranucleotide sequence effects in B-DNA. Nucleic Acids Research, 42(19), 12272–12283. https://doi.org/10.1093/nar/gku855

[Quinlan2002] Quinlan, A. R. (2002). BEDTools: The Swiss-Army Tool for Genome Feature Analysis. In Current Protocols in Bioinformatics. John Wiley & Sons, Inc. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002/0471250953.bi1112s47/abstract

[Rao2014] Rao, S. S. P., Huntley, M. H., Durand, N. C., Stamenova, E. K., Bochkov, I. D., Robinson, J. T., *et al* (2014). A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping. Cell, 159(7), 1665–1680. https://doi.org/10.1016/j.cell.2014.11.021

[Russel2012] Russel, D., Lasker, K., Webb, B., Velázquez-Muriel, J., Tjioe, E., Schneidman-Duhovny, D., *et al* (2012). Putting the Pieces Together: Integrative Modeling Platform Software for Structure Determination of Macromolecular Assemblies. PLOS Biol, 10(1), e1001244. https://doi.org/10.1371/journal.pbio.1001244

[Serra2016] Serra, F., Baù, D., Filion, G., & Marti-Renom, M. A. (2016). Structural features of the fly chromatin colors revealed by automatic three-dimensional modeling. bioRxiv, 36764. https://doi.org/10.1101/036764

[Schmid2010] Schmid, B., Schindelin, J., Cardona, A., Longair, M., & Heisenberg, M. (2010). A high-level 3D visualization API for Java and ImageJ. BMC Bioinformatics, 11, 274. https://doi.org/10.1186/1471-2105-11-274

[Stalling2005] Stalling, D., Westerhoff, M., & Hege, H.-C. (2005). Amira: a highly interactive system for visual data analysis. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.129.6785

[vanSteensel2010] van Steensel, B., & Dekker, J. (2010). Genomics tools for unraveling chromosome architecture., Genomics tools for the unraveling of chromosome architecture. Nature Biotechnology, Nature Biotechnology, 28, 28(10, 10), 1089, 1089–1095. https://doi.org/10.1038/nbt.1680, 10.1038/nbt.1680

[Yates2016] Yates, A., Akanni, W., Amode, M. R., Barrell, D., Billis, K., Carvalho-Silva, D., *et al* (2016). Ensembl 2016. Nucleic Acids Research, 44(D1), D710–D716. https://doi.org/10.1093/nar/gkv1157

[Zhang2008] Zhang, Y., Liu, T., Meyer, C. A., Eeckhoute, J., Johnson, D. S., Bernstein, B. E., *et al* (2008). Model-based Analysis of ChIP-Seq (MACS). Genome Biology, 9(9), R137. https://doi.org/10.1186/gb-2008-9-9-r137

7 ANNEXES

7.1 Abbreviations

DAC: Data access committee

EGA: European Genome-phenome Archive

ENA: European Nucleotide Archive

FISH: Fluorescence *in-situ* hybridisation

MINSEQE: Minimum Information about a high-throughput Sequencing Experiment

OME: Open Microscopy Environment

PDB: Protein Data Bank

SRA: Sequence read archive

VRE: Virtual Research Environment

WGBS: Whole Genome Bisulphate Sequencing

7.2 File Formats

FASTQ

https://en.wikipedia.org/wiki/FASTQ_format

File format for storing sequences reads (predominantly from high throughput sequencing methods) along with the quality score for each base call. The file consists of 4 lines containing the identifier, the sequence, a description and the quality encoded as a full range of ASCII characters. Derives from the earlier FASTA format.

SAM / BAM

https://samtools.github.io/hts-specs/SAMv1.pdf

SAM is a text based file format for representing a sequence aligned to a reference assembly genome. BAM is a binary compressed and indexed version of a SAM file. Format support is provided by the samtools group who work as part of the Data Working Group of the Global Alliance for Genomics and Health.

GFF3

https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md

File format for describing generic features that exist on a given genetic sequence. These features can consist of a minimum of a single base, but can also be on the forward or reverse strand. The features can also have additional attributes and a matching score.

WIG (Wiggle)

https://genome.ucsc.edu/goldenpath/help/wiggle.html

Text based file format for associating a numeric value to a given region of position on a reference genome. A binary indexed version of this format is available called BigWig. Format support is provided by the UCSC genome browser group.

CGMap and ATCGMap

https://github.com/BSSeeker/BSseeker2/blob/master/README.md

This is a file format for disseminating methylome data [Guo2013]. Both files includes the CpG and CpH sites. The ATCGMap also includes the positions of all features for all bases on both strands.

HDF5

https://support.hdfgroup.org/HDF5/

A file format designed to handle large data arrays. The file can handle the multiple sparse arrays easily in a very compressed format. Manipulation of records stored in this format must be handled by dedicated libraries.

JSON

http://www.json.org/

File format used handle attribute-value pairs in a semi-human readable way. Derives from JavaScript, but is openly accessible by a large number of programming languages. Due to its flexibility in representing any data schema and ubiquitous language support JSON is used as a fast way to deliver data to a client browser (Chrome/Firefox/Edge) or to server processes over RESTful APIs.