Multi Locus View : An Extensible Web Based Tool for the Analysis of Genomic Data

Martin J Sergeant[1], Jim R Hughes[1,2], Lance Hentges[1], Gerton Lunter[1,3], Damien J Downes[2] and Stephen Taylor[1*]

[1]MRC WIMM Centre for Computational Biology, MRC Weatherall Institute of Molecular Medicine, University of Oxford, Oxford, UK and [2]MRC Molecular Haematology Unit, MRC Weatherall Institute of Molecular Medicine, University of Oxford, Oxford, UK [3]University Medical Centre Groningen, Department of Epidemiology, University of Groningen, The Netherlands

*To whom correspondence should be addressed

Abstract

Motivation

Tracking and understanding data quality, analysis and reproducibility are critical concerns in the biological sciences. This is especially true in genomics where Next Generation Sequencing (NGS) based technologies such as ChIP-seq, RNA-seq and ATAC-seq are generating a flood of genome-scale data. These data-types are extremely high level and complex with single experiments capable of mapping ten to hundreds of thousands of biologically meaningful events across the genome. However, such data are usually processed with automated tools and pipelines, generating tabular outputs and static visualizations. These are difficult to interact with and require substantial bioinformatic skills to manipulate and query. Similarly, interpretation is normally made at a high level without the ability to visualise the underlying data in detail and so the complexity and quality of the real underlying biological signal is lost. Also genomics datasets require integration with other genomics datasets to be properly interpreted and this integration with multiple tracks again requires substantial bioinformatics skills and is difficult to visualise across multiple pertinent datasets. Conventional genome browsers do allow for the detailed visualisation of multiple tracks but are limited to browsing single locations and do not allow for interactions with the dataset as a whole. MLV has been developed to allow users to fluidly interact with genomics datasets at multiple scales, from complete metadata labelled and clustered populations to detailed representations of individual elements. It has inbuilt tools to integrate signals across multiple datasets and to perform dimensionality reduction and clustering analysis based on the extracted signal, allowing for the high-level analysis of complex datasets while maintaining visualisation of the fine grain structure of the data. MLV's ability to visualise clustering within the data combined with efficient tools for large-scale tagging of individual elements makes it a unique tool for the generation of annotated datasets for modern machine learning approaches.

Results

Multi Locus View (MLV) is a web based tool for the visualisation, analysis and annotation of Next Generation Sequencing data sets. The user is able to browse the raw data, cluster, and combine the data with other analysis. Intuitive filtering and visualisation then enables the user to quickly locate and annotate regions of interest. User datasets can then be shared with other users or made public for quick assessment from the academic community. MLV is publically available at https://mlv.molbiol.ox.ac.uk and the source code is available at https://github.com/Hughes-Genome-Group/mlv

1 Introduction

Next Generation Sequencing (NGS) technologies such as ChIP-seq, RNA-seq and ATAC-seq generate vast amounts of data, which, once mapped, is analysed with programs such as MACS (Zhang *et al.*, 2008) and DESeq2 (Love *et al.*, 2014) to extract biologically meaningful signals. The final output of these pipelines is usually a list of genomic regions filtered by an enrichment level or fold change and a statistical threshold, such as p-value, q-value or FDR. Selecting thresholds in absence of the ability to effectively see their effect on the final dataset can lead to the loss of biologically meaningful signal or the inclusion of noise and common bioinformatic mapping artefacts depending on the stringency used.

Understanding the effectiveness of the parameters used for a given dataset, data type or analytical tool is extremely challenging and effective quality control of such outputs may require the user to manually go through tens of thousands of regions to validate that the chosen thresholds, which is extremely time consuming in traditional genome browsers. Importantly, with the advent and high impact of machine learning in the genomics field, there is a critical need for a platform to generate curated high quality training sets of genomic regions which match specific criteria. Although many excellent genome browsers exist for looking at genomic locations, such as the UCSC genome browser (Karolchik *et al.*, 2011) , the WashU Epigenome Browser (Zhou and Wang, 2012), IGV (Robinson *et al.*, 2011) and HiGlass (Kerpedjiev *et al.*, 2018) , these are designed for sequential visualization of specific individual loci of interest, rather than looking at an experiment as a whole.

Multi Locus View (MLV) allows rapid filtering of hundreds of thousands of locations based on their metadata, combined with the genome views of regions of interest. MLV additionally allows for interaction with the complete dataset via the use of a highly customisable range of interactive charts fully linked up with the embedded genome browser. Going beyond data interaction MLV also provides the ability to run commonly required procedures, such as intersection between genomic annotation, but also advanced analyses, such as dimensionality reduction. This provides a powerful and easy to use way to discover new insights and quality control large ‘omics data sets. We demonstrate the power of MLV by showing examples of false positive inclusion with ENCODE datasets and novel characterisation of enhancers and promoters in a large published dataset (Kowalczyk *et al.*, 2012). Importantly, MLV only requires BED and BigWig tracks as an input, which unlike BAM files, are lightweight but extremely flexible and information rich summaries of the data that allow for extremely fast and fluid interactions with complete datasets.

2 Materials and methods

MLV takes as input, a tsv or csv file, where the first three columns specify the genomic location and an unlimited number of additional columns containing metadata for that location. Examples include a simple BED file, the output of MACS2 (Zhang *et al.*, 2008) or an Excel file that has been saved in csv or tsv format. The data can then be combined with annotations (e.g. transcription start sites [TSS]) and sequencing data (e.g. BigWig files). Dimension reduction (UMAP, tSNE) can also be carried out to identify clusters. In addition dynamic graphs, genomic tracks and images can be added to further aid visualization/analysis. After sorting and filtering, locations can be annotated (tagged) and then exported and links generated for sharing, such as with reviewers or with a publication. Fig1 shows a general summary of MLV and the functions available to the user

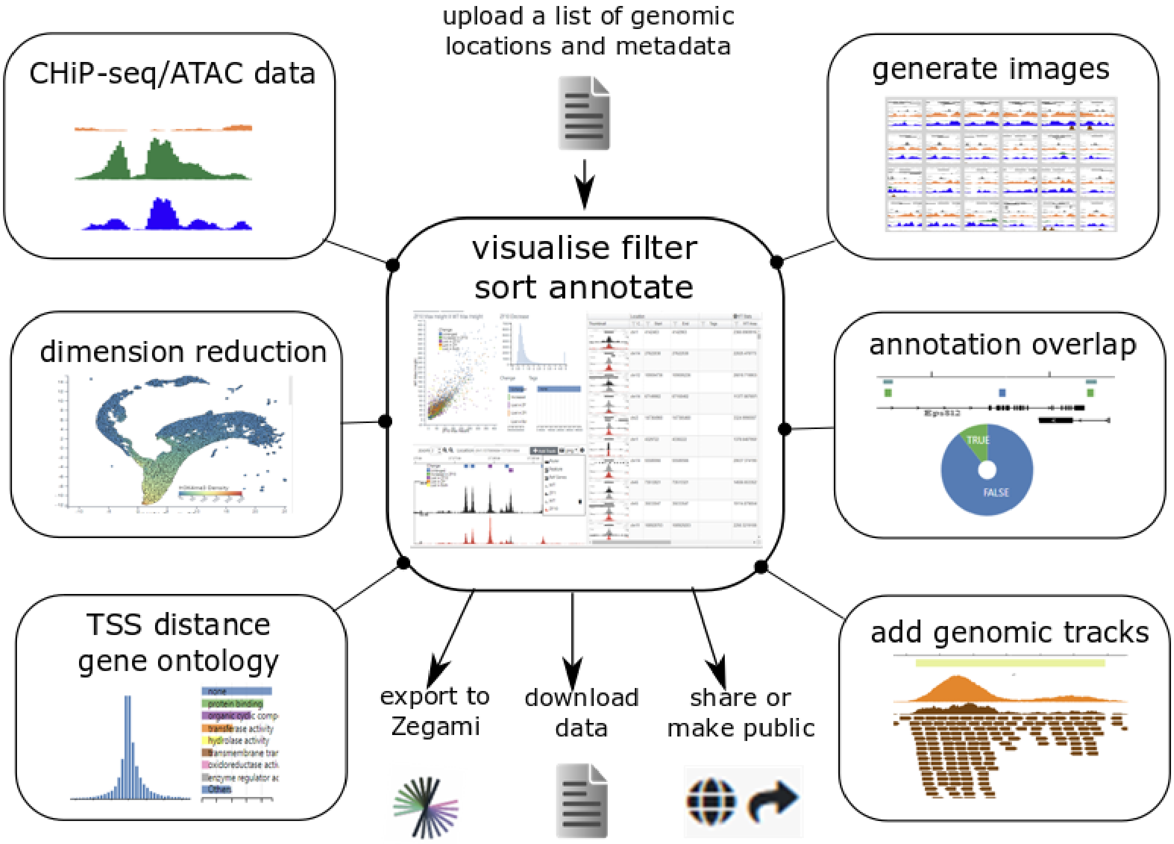

Figure 1. Summary of MLV
Initially a BED or BED-like file of genome locations is uploaded, which can be visualised via a spreadsheet, browser and various interactive graphs (see section 2.1) . Various actions can also be carried out on the uploaded locations in order to explore the data (see section 2.2). These include calculating distances from genomic objects such as TSSs (2.2.1), overlap with genomic features (2.2.2), calculating signal data at each location from BigWig e.g. ChIP or ATAC-seq (2.2.2) and dimension reduction (2.2.4), to produce interactive t-SNE (Laurens van der Maaten, 2008) or UMAP (McInnes *et al.*, 2018) representations of the data. To aid interpretation, genome tracks of other datasets and dynamic graphs can be added and browser images for each location can be generated, which can be viewed in a table format (2.1.4). The analysis can be downloaded as csv/tab delimited text or shared/made public via a URL. A collection can also be created in the analysis software Zegami (see section 2.3) for further interrogation of the data.

2.1 Visualization

The main view consists of three panels, a spreadsheet-like table housing the genomic locations and all the metadata, a genome browser and a panel showing dynamic graphs/charts. All three panels are linked - data can then be filtered by selecting regions/sections on a graph or using the table, which instantly updates the other graphs, table and browser.

2.1.1 Table

The spreadsheet contains the genomic locations and any associated metadata. In addition, if images have been generated (see 2.1.4), these can be displayed either as thumbnails in the spreadsheet or as rows in their own table. This allows instant visualisation of common elements in filtered data sets. When displayed on their own, images are sorted in the same order as the table rows and can have their border coloured according to any field in the data, further aiding the elucidation of patterns. Clicking on a row/image will display the genomic location in the browser and highlight its position on any scatter graphs present. Extra columns can also be generated from applying simple arithmetic to existing columns, For example calculating the ratio between the signal in two BigWig tracks will produce a column that can be sorted by relative enrichment or or depletion between the two datasets. Columns can also be deleted if the data is no longer required.

2.1.2 Internal Browser

The internal browser is a lightweight JavaScript component based on igv.js (https://github.com/igvteam/igv.js). Initially the internal browser displays a gene track (if a genome was specified) and a track showing the uploaded genomic features. To aid visualization, these features can be coloured by any of the metadata fields, positioned along the y-axis proportional to a numeric field and labelled with another field. Only those features that are in the current filter are displayed and clicking on a feature will highlight the relevant image/row in the table and highlight the appropriate point on any scatter plots present. Tracks of common formats. BAM, BigWig, BigBed, tabix indexed bed.gz files etc. and UCSC browser sessions can be added to the browser. In addition, many of the analysis steps will also add tracks. Thumbnails for every location (displayed in the table panel) can be generated based upon the current browser tracks and settings.

2.1.3 Graphs

Graphs including scatter plots, histograms, box plots and pie charts can be added to the view, showing any of the metadata fields that are appropriate to the graph. All graphs respond to filtering, and can also be used to intuitively filter the data by selecting appropriate regions. For example, selecting regions near TSS sites on a histogram will also update a histogram displaying H3K4me3 CHiP-Seq peak data, showing an overall increase in height. Another example would be selecting regions on a UMAP/t-SNE generated scatter plot, which updates graphs displaying fields that were used to generate the clustering, therefore indicating which fields influence different clusters. As well as users being able to add graphs and adjust their settings, appropriate ones are automatically generated in many of the analysis methods.

2.1.4 Images

Although the application contains a browser which will display each selected view, it is often more informative to visualise many locations at once, in order to see if there is commonality in a filtered data set. To this end images based on the browser view can be generated for every location . Once all images are created, they can be viewed as thumbnails as part of a table row or in their own table (see 2.1.1)

2.2 Analysis Methods

A number of methods can be employed to help interpret the data. Most methods will add fields (columns) to the table, graphs and tracks to the browser aiding the interpretation of the data.

2.2.1 Finding TSS's

If a genome was specified, then the nearest Transcription Start Site (TSS) based on the RefSeq annotation will be calculated using BedTools (Quinlan, 2014)**.** The RefSeq id and common name of the gene is also given and there is an option to include molecular function Gene Ontology (GO) annotations (Consortium and The Gene Ontology Consortium, 2012) . These annotations were simplified by first obtaining all GO terms for a RefSeq gene by using gene2go and gene2refseq files from NCBI. Next, using the go-basic.obo file from http://geneontology.org/, multiple terms for each gene were further expanded by traversing up the hierarchy and adding terms at each level. Then at each hierarchical level , terms were collapsed by only keeping the most frequent, resulting in a much simplified scheme, where each refseq gene had a single term at each hierarchical level. Users can choose to include up to five levels of GO annotations in the data returned from a TSS search.

2.2.2 Annotation Overlap

In order to fully interpret some data sets, it is usually useful to combine the existing data with other datasets. Hence MLV enables the user to intersect locations with a list of genomic regions or the locations in other projects. This feature uses bedtools (Quinlan, 2014) behind the scenes and also allows information contained in the intersecting data to be added to the data set, easily allowing other experiments to be incorporated into the project.

2.2.3 Peak Stats

BigWig files from experiments such as ChIP-Seq can be specified and MLV will calculate the area, max height of the track's signal for each genomic location. This data can then be used to plot various graphs or used to generate UMAP/t-SNE scatter plots (see below)

2.2.4 Clustering (Dimension Reduction)

In order to group the genomic locations it may be useful to cluster them based on specified fields. For example, clustering based on ChIP-Seq peaks for histone marks may give an indication of promoters/enhancers. MLV enables this, by using the dimension reduction algorithms, UMAP (McInnes *et al.*, 2018) and t-SNE (Laurens van der Maaten, 2008) implemented with Scikit-learn (Pedregosa *et al.*, 2011). Any number of numeric fields (columns) can be used as input and these are reduced to a specified number (default 2) of output dimensions. Although the initial graphs show the first two dimensions for each algorithm, the user can produce different graphs by mixing and matching dimensions from different algorithms.

2.3 Output Options

In order to communicate your findings with others, annotation of each location is possible. Tasks such as naming clusters, marking outliers/anomalies etc. can be quickly achieved by assigning tags to filtered sets. In addition, individual images/table rows or ranges can be tagged. The data can be downloaded as a tsv or csv file and current settings (graphs and browser layout) can be saved and the view shared with other users, either with or without edit permissions or the project can be made shared via URL such that even non users will be able to view it. The whole data or filtered subsets can be cloned to produce new data sets. Moreover, if images have been generated it can be exported to the visual data exploration software Zegami (https://zegami.com/) for further analysis.

2.4 Implementation and Extensibility

The backend of MLV is implemented using the python framework flask (http://flask.pocoo.org/) and the relational database PostgreSQL (https://www.postgresql.org/). It is composed of two main building blocks: projects (analysis types) and jobs (pipelines) -supplementary figure. It was designed to be modular, with each independent module specifying the analysis types (projects) and jobs (pipelines) required. In addition to MLV, two other modules have been developed, LanceOtron (https://lanceotron.molbiol.ox.ac.uk/) that calls peaks using machine learning and CaptureSee (http://capturesee.molbiol.ox.ac.uk/ (Telenius *et al.*)) for looking at highly multiplexed Capture C data (Davies *et al.*, 2016). The front end is written in JavaScript and is built upon two stand-alone components, MLVPanel (https://github.com/Hughes-Genome-Group/MLVPanel) and CIView (https://github.com/Hughes-Genome-Group/CIView). MLVPanel is a lightweight, extensible genome browser, based on igv.js (https://github.com/igvteam/igv.js), but with the emphasis on displaying multiple genomic locations simultaneously. All tracks are displayed compactly on the same canvas and many panels can be displayed on the same web page, for example as thumbnails in a table. It is highly extensible, making it simple for developers to create their own custom tracks to suit the needs of a project and a node.js version allows images (png,svg or pdf) to be created programmatically. CIView enables users to intuitively look at multivariate data, visualizing the effect that each parameter has on the dataset as a whole. It is based upon dc charts (https://dc-js.github.io/dc.js/), which in turn uses d3 (https://d3js.org/) and crossfilter (https://square.github.io/crossfilter/). However, to address the problem of the slow rendering of large graphs in SVG which can only cope with a few thousand points, the default scatter plots have been replaced with those using webgl technology and thus is able to cope with hundreds of thousands of data points. It also allows users to dynamically create and manipulate graphs in order to tailor the display to their dataset. Graphs can either be linked to a spreadsheet (https://github.com/mleibman/SlickGrid), giving the ability to edit the data or a table displaying images, enabling instant visual feedback at each filtering step.

3 Results

3.1 Identification of ChIP-seq false positives with MLV.

ChIP-seq experiments are performed to identify sites of chromatin modification or protein binding, visualized as peaks. For genome wide analysis, peaks are identified bioinformatically, most commonly with MACS2, though newer methods use digital signal processing (Stanton *et al.*, 2017) and Machine Learning e.g. (Hocking *et al.*, 2017). Often these peak callers use arbitrary statistical thresholds which can lead to inclusion of false positives, affecting downstream analysis. To demonstrate the ability of MLV to filter true- and false-positive peaks, we looked at data from H3K27ac ChIP-seq (a marker of active transcription) in the human prostate cancer cell line 22Rv1 (ENCSR391NPE). We uploaded 46,030 MACS2 peak calls and associated BigWig files to MLV (see supplemental methods) . Sorting by -$\log_{10}$ q-value showed that 12,000 of the identified peaks (26%), those with a value less than 10, were little more than background noise (Fig 2). This demonstrates the ability of MLV to visually interrogate peak calling results and determine an appropriate rather than an arbitrary threshold for filtering high quality peak calls. This allows for easy and intuitive segregation of the basic analysis into strong or weak peaks, producing stringent or more generous annotation, quickly and on the fly.

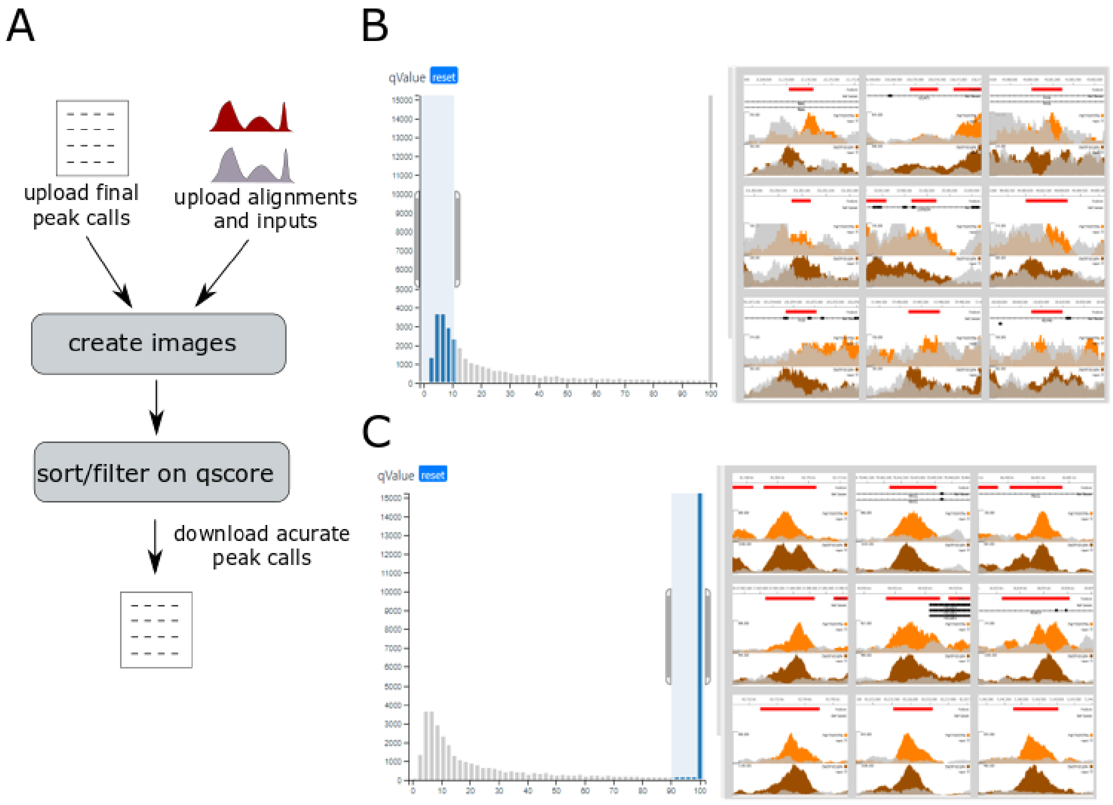

Figure 2. Summary of Peaks in Encode Project ENCSR391NPE
https://mlv.molbiol.ox.ac.uk/projects/multi_locus_view/1434 **A.** Workflow (see supplemental methods for full details). **B** Locations with -$\log_{10}$ q values less than 10, clearly showing regions which are misidentified as peaks compared to **C** Locations with high -$\log_{10}$ q values , showing genuine peaks. The two alignment tracks ENCFF025ZEN and ENCFF421QFK are orange and brown respectively and the corresponding input tracks, ENCFF483ELD and ENCFF769UET are grey.

3.2 Functional annotation of regulatory elements.

The genome contains three main types of regulatory elements (promoters, enhancers and boundaries) whose position can be detected in open chromatin assays such as DNase-seq and ATAC-seq (Song & Crawford 2010; Buenestro 2015). However, open chromatin is common to all these classes and so cannot determine a specific element's identity alone. However epigenetic marks (H3K4me1, H3K4me3, H3K27ac) and transcription factor binding (CTCF) have been used to annotate elements, such as likely promoters and enhancers using the ratio of H3K4me1 to H3K4me3 (Kowalcyzk 2012). To demonstrate how MLV can be used to fluidly explore and classify all the open chromatin elements in a given cell-type, we used chromatin marks to cluster and annotate erythroid open chromatin peaks identified from ATAC-seq based on their relative enrichment of ChIP-seq signals (see supplemental methods and Figure 3). Filtering for peaks with a high H3K4me1 to H3Kme3 ratio (fig3. Ciii), characteristic of enhancers, identifies a distinct cluster with few peaks overlapping TSSs, low levels of CTCF and a range of H3K27ac - expected traits of enhancer regulatory elements (Fig 3Bii,iv,v). Conversely, putative promoter peaks with a low H3K4me1 to H3K4me3 ratio showed a high proportion of TSS overlap (61%), and higher levels of both H3K27ac and CTCF, again confirming the expected traits of promoters. Finally we used the “annotate feature” to append classes to each class of open chromatin region. Using MLV we were able to quickly and efficiently categorize, and annotate peaks, whilst rapidly inspecting specific and random peaks to quality check the annotations - the entire analysis taking less than an hour. These peaks can then be exported for use in downstream processes such as motif discovery or nearest gene analysis.

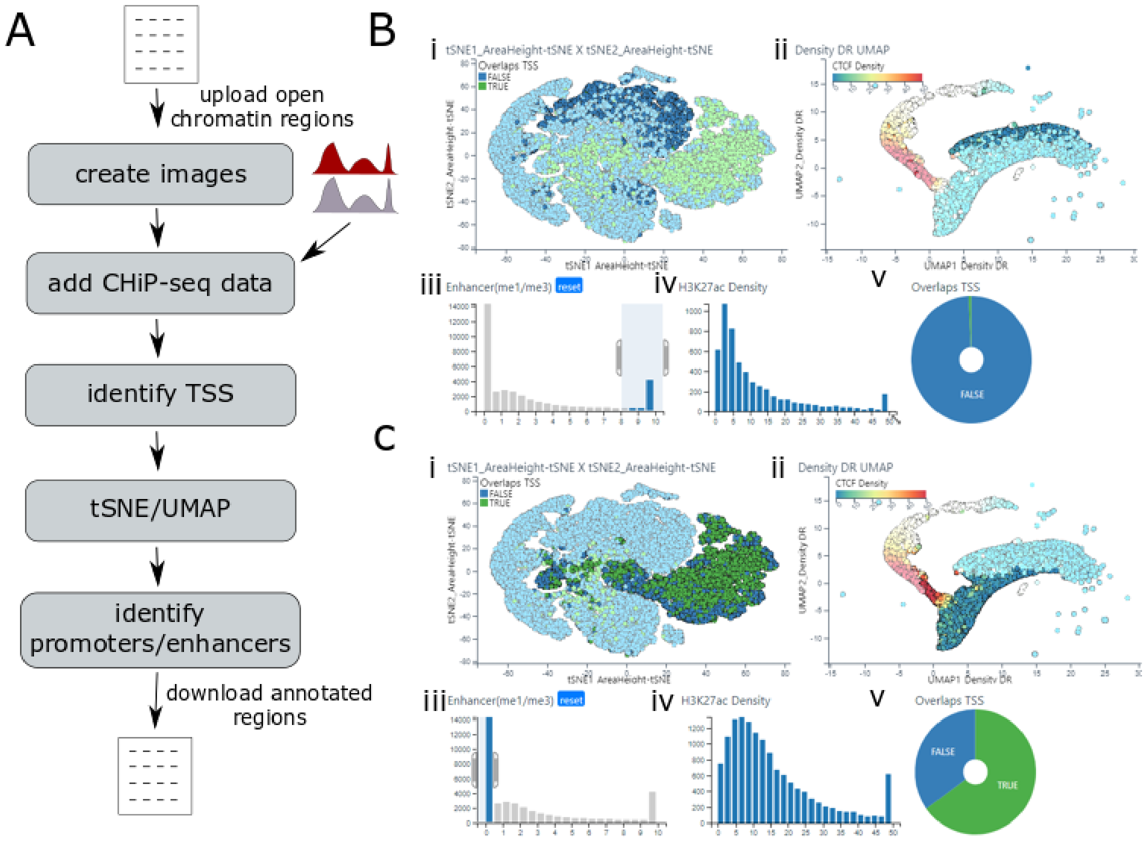

Figure 3. Functional annotation of regulatory elements https://mlv.molbiol.ox.ac.uk/projects/multi_locus_view/1590. **A** Workflow (see supplemental methods for full details) **B** and **C** show the clustering of putative enhancers and promoters respectively based on enrichment for the two chromatin marks and CTCF binding. (**i**) t-SNE plot shows the large enrichment for overlapping annotated TSSs with the H3K4me3 enriched cluster. (**ii**) UMAP plot colored by CTCF peak density from red (most dense) to blue (least dense) identifies clusters of strongly and weakly bound elements. (**iii**) Histogram of the H3K4Me1/H3K4m3 ratio of open chromatin site allows for the interactive selection of differentially enriched elements (**iv**) Histogram of H3K27ac enrichment allows for the interactive selection of elements most enriched for this active chromatin mark. (**v**) Dynamically linked pie charts show the enrichment of a given annotation (e.g TSS overlap, green segment) for the selected or filtered objects.

3.3 Analysis of cohesin/CTCF interactions

The 3D structure of the genome is thought to be mediated via the cohesin complex and CTCF, which bring distal regions of DNA together via a process of loop extrusion (Fudenberg *et al.*, 2016). In a recent paper (Li *et al.*, 2020), the authors mutated CTCF such that it abolished interaction with SCC1, a member of the cohesin complex, in the human HAP1 cell line. They then carried out ChIP-seq of CTCF and SCC1 in both the wild type and the mutant cells. This revealed that in the main CTCF binding to DNA was unaffected in the mutant, whereas cohesin (SCC1) was reduced, especially at locations where CTCF was also bound, hence supporting the hypothesis that CTCF stabilizes cohesin on chromatin. To explore these datasets dynamically, the peak locations and BigWig tracks from the

ChIP-Seq experiments were loaded into MLV along with histone mark data from Hap1 cells (see supplemental methods and fig. 4).

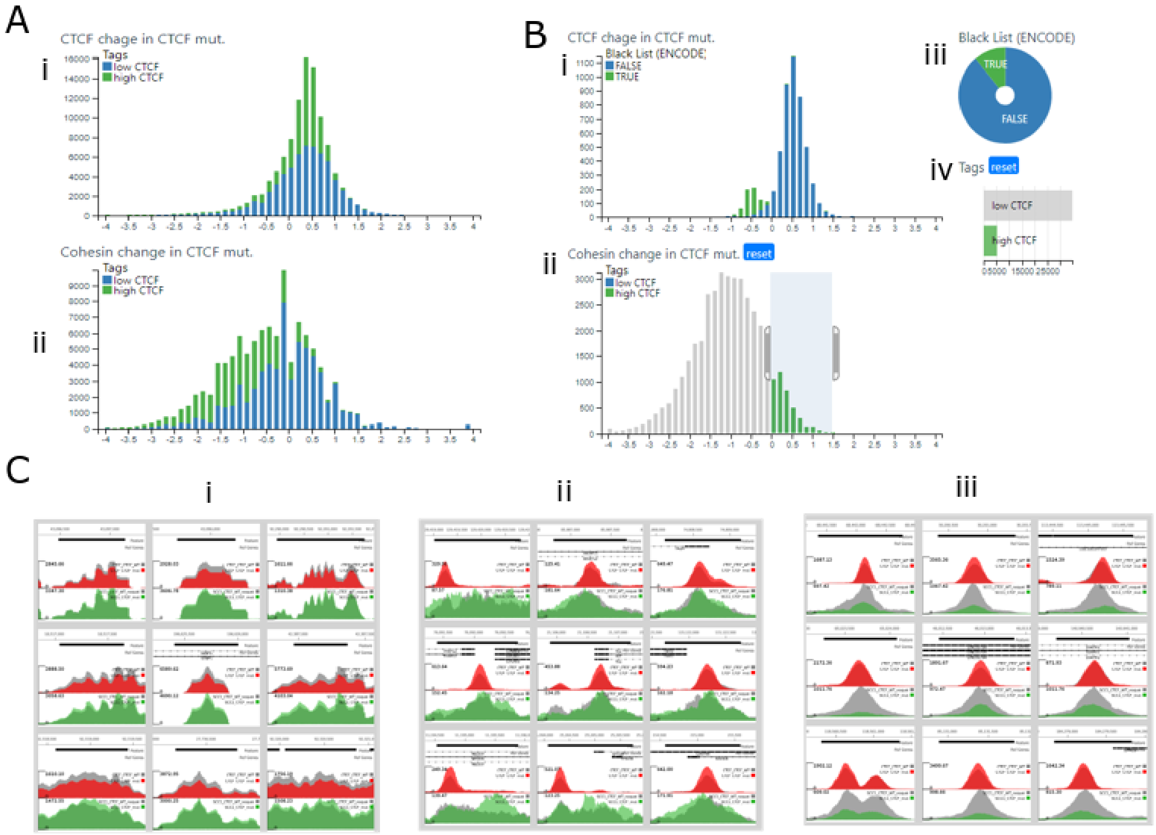

Figure 4. Analysis of the ChIP-seq data from the Li *et al.*
https://mlv.molbiol.ox.ac.uk/projects/multi_locus_view/2057 Data taken from (Li *et al.*, 2020) (**A**) Histograms of CTCF (**i**) and Cohesin(**ii**) log fold changes. The bars are coloured by tags (green -high CTCF, blue - low CTCF). **B** Filtering of strong CTCF binding sites with no decrease in SCC1 binding in the CTCF mutant. (**i**) Histogram showing the bimodal distribution of the CTCF fold changes in the selected regions. The green bars show black listed regions (**ii**) Histogram of cohesin change, the gray box shows the regions (those with a log2 fold change greater than 0) that were selected. (**iii**) Pie chart showing the proportion of black liisted regions (green) in the selected regions (**iv**) Row chart showing tags, which was used to select regions with strong CTCF binding (high CTCF). **C** Representative samples of genome browser images. The upper track shows CTCF ChIP-seq peaks with the gray track being WT and the red track, the CTCF mutant. The lower track shows SCC1 (cohesin) data with the WT gray and the CTCF mutant and green (**i**) the left peak in B(i) consisting of black listed regions, (**ii**) regions with strong CTCF binding, but no reduction of cohesin binding in the CTCF mutant (the right-hand peak in B(ii)), (**iii**) typical peaks for the majority of regions with strong CTCF binding.

A Histogram of CTCF peak height fold change (fig4 Ai) generally supported the paper's observation that the mutation did not reduce CTCF chromatin binding. Indeed, the shape of the histogram indicated an average log2 fold change value of around 0.5, suggesting the mutation may cause a modest increase in CTCF binding, although incomplete normalization of the BigWig tracks may also account for this. The cohesin (SCC1) histogram (fig4 aii) showed a skew to the left indicating reduced binding in the mutant and this region contained mainly locations with high CTCF binding. Again, this was in accordance with the paper's findings that in the CTCF mutant, cohesin binding was generally reduced, especially at locations where CTCF was also bound. Further exploration of data was carried out by selecting those locations with 'high' CTCF (see supplemental methods) but where SCC1 binding in the mutant did not decrease (fig4 Bii) These regions showed a bi-modal distribution in CTCF change (fig4 Bi). The smaller peak mainly contained black listed regions and indeed images from these locations showed these areas with the abnormal peak structure associated with the bioinformatic artifacts found in these regions (Fig4 Ci). However, the larger histogram peak represented regions containing what looked like genuine peak calls (Fig4 Cii). Moreover, the SCC1 peaks in these regions exhibited a different pattern, being broader and flatter than the majority of peaks at other locations (Fig4 Ciii.) and, in many cases, the corresponding CTCF peak was at the edge of the cohesin peak. These peaks also appeared to be enriched for promoters due to their association with TSSs and regions that have greater levels of H3K4em3 (supplemental fig2). This shows the ease and facility with which MLV can be used to explore important published data, to both confirm the basic findings and to add extra insights.

4 Discussion

The massive expansion in NGS data generation and the increasing complexity of datasets and data types has led to the current crisis in both the transparency of interpretation and reproducibility of data in the biomedical sciences. To tackle this challenge requires better ways of analysing and humanly interacting with such large multidimensional datasets. Importantly, such methods should have very low barriers to use, not requiring specialised computational skills and so allowing for their general use in the biological community. Similarly, they need to have quick, fluid, and above all intuitive interfaces to allow researchers to concentrate on asking the pertinent biological questions rather than on the computational tasks required to ask them.

MLV provides a more holistic way of interacting with complex NGS data sets. By combining the use of common lightweight data formats (e.g BED, BigWIG and table delimited text) with a fully featured javascript frontend with powerful server-side Python flask frameworks and PostgreSQL relational databases, MLV provides a powerful and agile web-based interface to complex datasets. The inbuilt and dynamically linked dimensionality reduction functionality, table, graph and image based interfaces within MLV allows a user to simultaneously analyse the dataset as a whole and also quickly drill down to subsets with specific characteristics or behaviors. By allowing for the clustering and subsequent fine grained inspection of multiple genome regions of similar characteristics in a single view, MLV affords a powerful way to look for trends in results data traditionally represented in tables and static figures. Also with MLVs dynamic filtering, graphing and data brushing, it is possible to visually inspect and understand the effects of parameter selection. Importantly, to aid transparency, the data, analysis and visualisations can be shared via online URL to provide a powerful supplement to any publication, which gives the reader or manuscript reviewer immediate access to the datasets and analysis that underpins the findings of the work. Such interfaces will be critical to enable greater transparency and reproducibility in research. Furthermore, we have shown that data files or analyses released with published datasets can be rapidly incorporated into

MLV to allow for the reexploration of existing data and analyses to validate the conclusions of the manuscript, to discover new trends in the data or to ask new biological questions with the inclusion of further datasets or annotations.

Finally, the intrinsic ability to cluster data, or to input clustered data, combined with the fine grained visualisation and ability to tag collections of data points means MLV can quickly generate extremely large high-quality training datasets for machine learning approaches. The generation of such validated training sets is extremely laborious on the operator and so represent the biggest bottleneck to the wide-scale implementation of these powerful methods in genomics research.

This ability is also useful for labelling large data sets to be used as training sets for machine learning, where clusters of related features may be visualised and annotated quickly.

Funding

This work was carried out as part of the WIGWAM Consortium (Wellcome Investigation of Genome Wide Association Mechanisms) funded by a Wellcome Trust Strategic Award (106130/Z/14/Z), Medical Research Council (MRC) Core Funding (MC_UU_00016/14) and with support from the National Institutes of Health (USA) grant number R24DK106766 and from the Alan Turing Institute grant number AfSMBPP1\100037.

.

References

Campagne,A. *et al.* (2019) BAP1 complex promotes transcription by opposing PRC1-mediated H2A ubiquitylation. *Nat. Commun.*, **10**, 348.

Consortium,T.G.O. and The Gene Ontology Consortium (2012) Gene Ontology Annotations and Resources. *Nucleic Acids Research*, **41**, D530–D535.

Davies,J.O.J. *et al.* (2016) Multiplexed analysis of chromosome conformation at vastly improved sensitivity. *Nat. Methods*, **13**, 74–80.

Fudenberg,G. *et al.* (2016) Formation of Chromosomal Domains by Loop Extrusion. *Cell Reports*, **15**, 2038–2049.

Gaspar,J.M. (2017) Improved peak-calling with MACS2. *bioRxiv*.

Hocking,T.D. *et al.* (2017) Optimizing ChIP-seq peak detectors using visual labels and supervised machine learning. *Bioinformatics*, **33**, 491–499.

Karolchik,D. *et al.* (2011) The UCSC Genome Browser. *Current Protocols in Human Genetics*.

Kerpedjiev,P. *et al.* (2018) HiGlass: web-based visual exploration and analysis of genome interaction maps. *Genome Biol.*, **19**, 125.

Kowalczyk,M.S. *et al.* (2012) Intragenic enhancers act as alternative promoters. *Mol. Cell*, **45**, 447–458.

Laurens van der Maaten,G.H. (2008) Visualizing Data using t-SNE. *J. Mach. Learn. Res.*, **9**, 2579–2605.

Li,Y. *et al.* (2020) The structural basis for cohesin-CTCF-anchored loops. *Nature*, **578**, 472–476.

Love,M.I. *et al.* (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.*, **15**, 550.

McInnes,L. *et al.* (2018) UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, **3**, 861.

Pedregosa,F. *et al.* (2011) Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.*, **12**, 2825–2830.

Quinlan,A.R. (2014) BEDTools: The Swiss-Army Tool for Genome Feature Analysis. *Curr. Protoc. Bioinformatics*, **47**, 11.12.1–34.

Ramírez,F. *et al.* (2016) deepTools2: a next generation web server for deep-sequencing data analysis. *Nucleic Acids Res.*, **44**, W160–5.

Robinson,J.T. *et al.* (2011) Integrative genomics viewer. *Nat. Biotechnol.*, **29**, 24–26.

Stanton,K.P. *et al.* (2017) Ritornello: high fidelity control-free chromatin immunoprecipitation peak calling. *Nucleic Acids Res.*, **45**, e173.

Telenius,J.M. *et al.* CaptureCompendium: a comprehensive toolkit for 3C analysis.

Zhang,Y. *et al.* (2008) Model-based analysis of ChIP-Seq (MACS). *Genome Biol.*, **9**, R137.

Zhou,X. and Wang,T. (2012) Using the Wash U Epigenome Browser to examine genome-wide sequencing data. *Curr. Protoc. Bioinformatics*, **Chapter 10**, Unit10.10.

Supplemental

Supplemental Methods

Analysis of data in ENCODE project ENCSR391NPE

Initially the file containing the consensus MACS2 peak calls which include the average -$\log_{10}$ q and p scores (replicated peaks file ENCFF058TAP) was uploaded to MLV. Next, bigwigs of the corresponding alignments, ENCFF025ZEN (orange track) and ENCFF421QFK (brown track) and corresponding inputs, ENCFF483ELD and ENCFF769UET (grey tracks) were added to the browser. Images from each location were then generated and a graph showing the -$\log_{10}$ q values contained in ENCFF058TAP was added. The data was ordered by this value and the view switched to image mode.

Functional annotation of regulatory elements

Bam files from alignments of H3K4me1, H3K4me3, H3K27ac and CTCF ChIP-seq experiments (Kowalczyk *et al.*, 2012), as well as ATAC (ref) were converted to BigWig files using deepTools (Ramírez *et al.*, 2016) *bamCoverage -binSize 50 --normalizeUsingRPKM*. Open chromatin regions were identified by calling peaks from the ATAC BigWig using MACS2 (Gaspar, 2017) *callpeak -f BAM -g hs* . Peak regions were extending 500 bp either side of the peak summit and any peaks with a maximum height of below 10 were removed.

This file containing 38,537 peaks was uploaded to MLV and then the “calculate stat feature” for erythroid H3K4me1, H3K4me3, H3K27ac, CTCF and ATAC-seq (REF) data was used . This feature imports the BigWig files into the genome browser and calculates the maximum height, area, and density for each region. Images from each location were then generated. Then the ‘Find TSS distances’ feature was used to calculate the distance to the nearest annotated transcription start site (TSS) for each peak, annotating if the peak overlapped a TSS. Following the method of Kowalcyzk et al., we calculated the ratio of H3K4me1 to H3K4me3. Using the peak density for all five chromatin datasets we then used both UMAP and t-SNE for dimension reduction of the peak data.

The BigWig tracks H3K4me1, H3K4me3, H3K27ac, CTCF as well as ATAC data were added using the ‘calculate peak stats’ feature. This feature not only adds the tracks to the genome browser, but for each location, calculates the max height, area and density for each track signal. The peak density for all five peaks was used to create UMAP and t-SNE plots (fig3 Bi and Bii). TSS overlaps/distance was calculated (fig3 Bv and Cvi) and an extra column of the H3K4Me1/H3K4me3 peak area ratio was generated (fig3. Bii and Cii)

Analysis of cohesin/CTCF interactions

The four narrowPeak files (GEO GSE126634) from CTCF and SCC1 in the WT and CTCF mutant cell lines were concatenated and merged keeping the maximum -log10 qscore and signal value using *bedtools merge --c 5,6 -o max,max*. The corresponding BigWig files were re-created from the original fastq files (SRA SRP18610) according to the methods outlined in the paper (Li *et al.*, 2020) except the *minMappingQuality* parameter was removed from the *bamCoverage* command. The bed file containing all the 104799 merged peaks was uploaded to MLV and the peak stats function was run on the four BigWig Files. Next two columns were added containing the log2 fold difference of CTCF and SCC1 between the CTCF mutant and WT, and histograms were generated from these columns. Then regions which overlapped with black listed regions, taken from ENCODE (ENCSR636HFF) were calculated using the annotation overlap feature. Finally images of the four BigWigs from each location were generated. Those locations with a max peak height for CTCF in WT cells greater than 100 were tagged as 'high CTCF' and the rest as 'low CTCF'. H3K4me3, H3K27me3 and H2AK119ub1 ChIP-seq data from Hap1 cells (Campagne *et al.*, 2019) was added to the project by using the 'peak stats' feature on the BigWig files associated with GSM2978163, GSM2978165, GSM2978167. The data was visualized by adding a column average bar chart of the H3K4me3, H3K27me3 and H2AK119ub1 peak area (See supplemental fig2)

Supplementary Figures

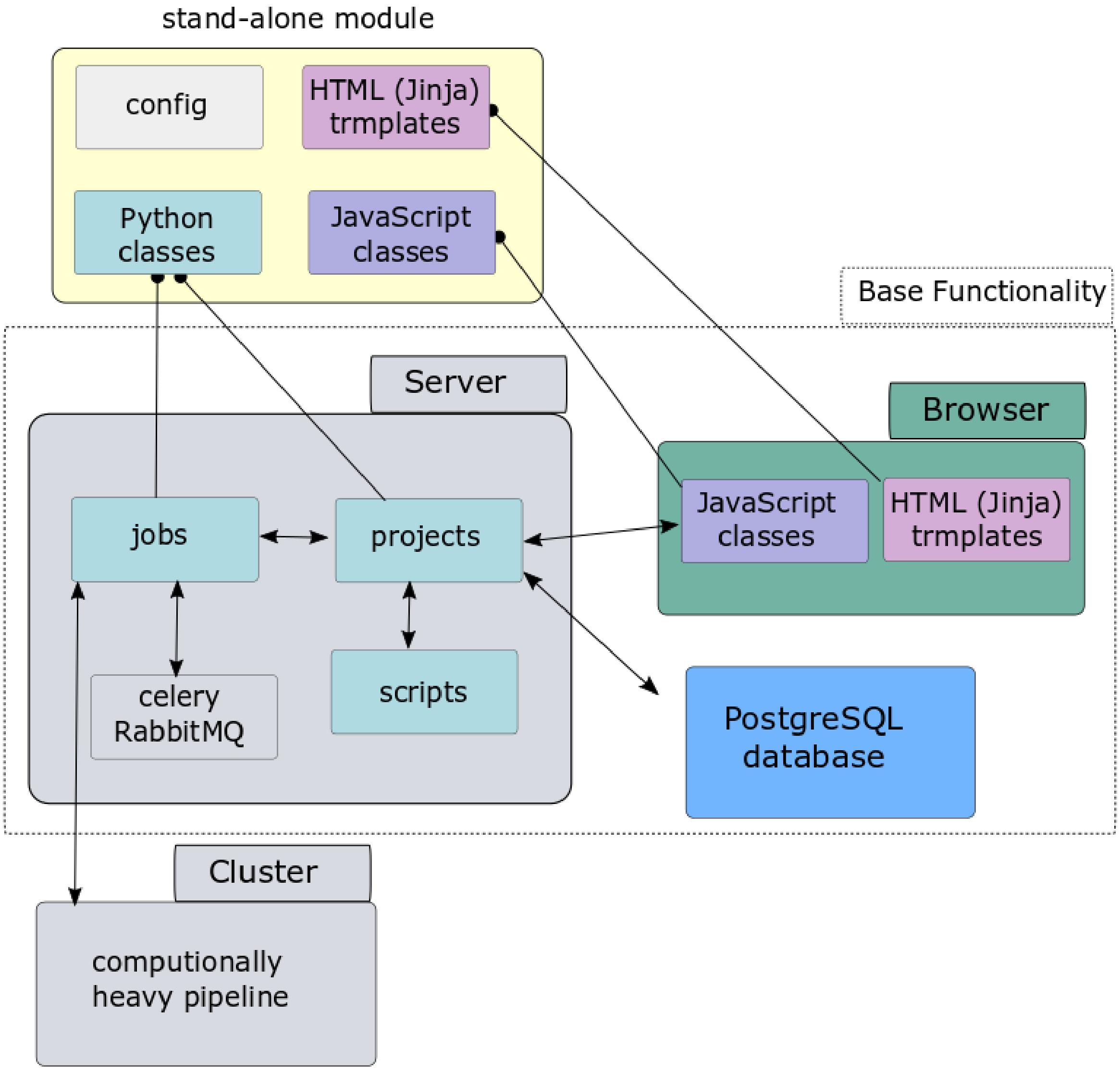

Figure S1. Summary of the architecture of MLV

The backend is written in Python and consists of two main class types, jobs and projects. Projects (analysis types) contain methods for interacting with the application (API) and can be accessed directly via python scripts or through http via a single flask view (method), which checks permissions etc. before calling the appropriate project method. Jobs are responsible for running pipelines and tasks either locally using celery via the rabbitmq message queue or remotely on other servers/clusters and are controlled by the projects. Jobs and Projects store their data in a PostgreSQL database. The frontend consists of HTML (Jinja) templates and Javascript classes, which communicate with the projects via ajax calls. Base Python/JavaScript classes as well as Jinja templates contain all the generic functionality. Modules involve extending these classes and templates to tailor the functionality to a particular analysis and are completely stand alone, in that they can be developed independently and added or removed without affecting other modules.

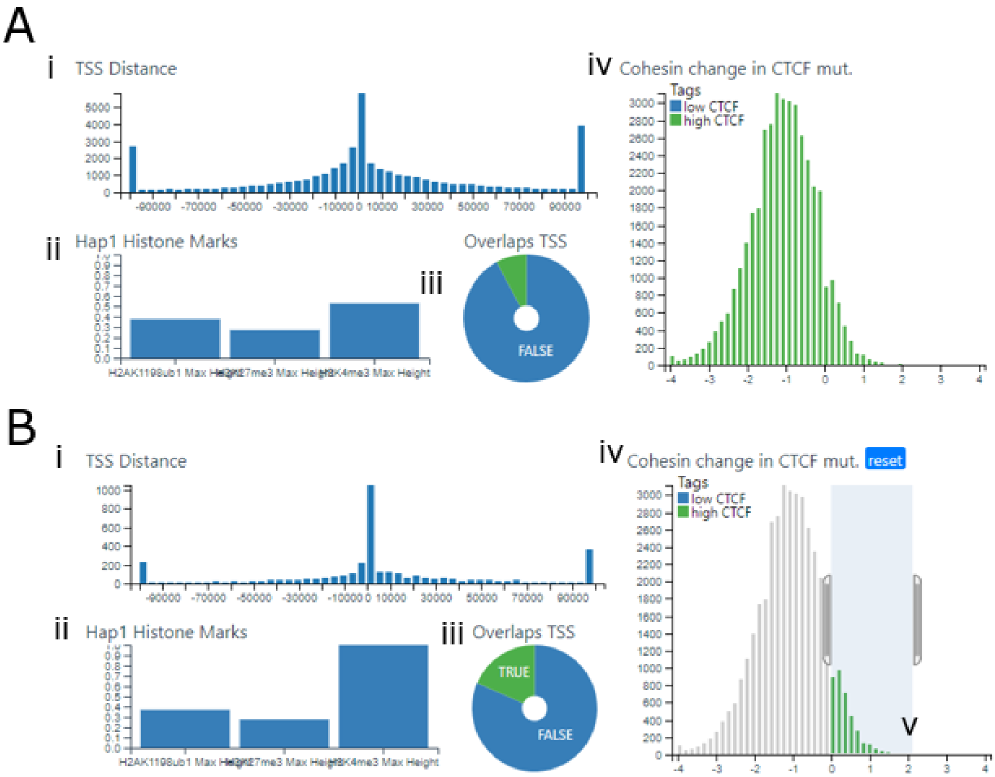

Figure S2. Graphs showing differences in TSS overlap/distance and H3K27me3 binding in regions where SCC1 (cohesin) binding is increased in the CTCF mutant (Li *et al.*, 2020).

(A) All Regions which strongly bind CTCF (excluding black listed regions) **(B)** Further selection of regions where cohesin increases in the CTCF muttant **(i)** Histogram showing the distance from TSSs of the selected regions **(ii)** bar chart showing the relative average of H2AK119ub1, H3K27me3 and H3K4me3 peak area at the selected locations, **(iii)** Pie chart showing number of selected regions which overlap TSS sites **(Iv)** histogram showing log fold change of cohesin binding (SCC1 ChIP-seq peak height) in the CTCF mutant compared to the WT.