Computational workflow for integrative analyses of DNA replication timing, epigenomic, and transcriptomic data

Temporal profiling of DNA replication timing (RT) in combination with chromatin modifications, chromatin accessibility, and gene expression provides new insights into the causal relationships between chromatin and RT during cell cycle. Here, we describe a protocol for in-depth integrative computational analyses of Repli-seq, ATAC-seq, RNA-seq, and ChIP-seq or CUT&RUN data for multiple marks at various time points across cell cycle and changes in their interrelationships upon an experimental perturbation (e.g., knockdown or overexpression of a regulatory protein).

Publisher’s note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.
Protocol

Computational workflow for integrative analyses of DNA replication timing, epigenomic, and transcriptomic data

Fei Ji,1,7,* Capucine Van Rechem,2,3 Johnathan R. Whetstine,2,4,5 and Ruslan I. Sadreyev1,6,8,*

1Department of Molecular Biology, Massachusetts General Hospital, Boston, MA 02114, USA
2Massachusetts General Hospital Cancer Center and Harvard Medical School Department of Medicine, 13th Street Bldg. 149, Charlestown, MA 02129, USA
3Stanford Medicine Department of Pathology, 269 Campus Drive, Stanford, CA 94305, USA
4Institute for Cancer Research, Fox Chase Cancer Center, 333 Cottman Avenue West 260, Philadelphia, PA 19111, USA
5Cancer Epigenetics Institute, Fox Chase Cancer Center, 333 Cottman Avenue, Philadelphia, PA 19111, USA
6Department of Pathology, Massachusetts General Hospital and Harvard Medical School, Boston, MA 02114, USA
7Technical contact
8Lead contact
*Correspondence: ji@molbio.mgh.harvard.edu (F.J.), sadreyev@molbio.mgh.harvard.edu (R.I.S.)
https://doi.org/10.1016/j.xpro.2022.101827

SUMMARY

Temporal profiling of DNA replication timing (RT) in combination with chromatin modifications, chromatin accessibility, and gene expression provides new insights into the causal relationships between chromatin and RT during cell cycle. Here, we describe a protocol for in-depth integrative computational analyses of Repli-seq, ATAC-seq, RNA-seq, and ChIP-seq or CUT&RUN data for multiple marks at various time points across cell cycle and changes in their interrelationships upon an experimental perturbation (e.g., knockdown or overexpression of a regulatory protein).

For complete details on the use and execution of this protocol, please refer to Van Rechem et al. (2021).

BEFORE YOU BEGIN

This protocol addresses the problem of quantitatively understanding the association between DNA replication, chromatin remodeling, and RNA expression across the genome profiled during the cell cycle at high temporal resolution. The protocol describes processing and analyzing DNA replication data from Repli-seq platform and integrating these data with the data from various epigenomic (ATAC-seq, ChIP-seq, CUT&RUN) and transcriptomic (RNA-seq) platforms. Importantly, the protocol focuses on the experimental design where these datasets include multiple time points across the cell cycle to allow for high-resolution temporal profiling of replication, epigenomic, and transcriptomic signals genome-wide.

For example, this protocol was recently applied to our data (Van Rechem et al., 2021) that evaluated genome-wide effects of overexpressing histone demethylase KDM4A on replication timing (RT), histone modifications, chromatin accessibility, and gene expression in human RPE cells. In brief, to eliminate the impact that drug or metabolic arrest would have on replication and chromatin mark dynamics, we adapted a FACS sort method to isolate four time points during S phase (S1, S2, S3, S4) to profile DNA replication and four time points across the entire cell cycle (G1, early S, late S, and G2/M phase) to profile histone modifications, gene expression, and chromatin accessibility in two independently generated stable sets of control and KDM4A overexpressing cells (Van Rechem et al., 2021).

Before starting, one needs to have access to computational hardware where one can install publicly available software packages, download genome references and other auxiliary files, create genomic...
indices for read mappers, and store raw sequencing data and the results of the analyses. To carry out the described analysis, it is necessary to have a computer with Unix, Linux, or Mac OS X operating systems. RAM requirements are at least 10 times of genome size bytes for sequencing read alignment. For instance, a human genome of ~3 gigabases (Gbp) will require ~30 gigabytes (GB) of RAM. 32 GB is recommended for human genome alignments. It is also necessary to have sufficient storage space (>100 GB) for output files.

**Download and install required software packages**

© Timing: 5 h

1. The packages (see Software and Algorithms in key resources table) should be installed within a Unix/Linux/MacOS X environment.

**Download or generate required auxiliary files**

© Timing: 2–3 h

2. Download reference genome (e.g., human hg38 (GRCh38) reference) as a DNA FASTA file and the gene annotation GTF file from the Ensembl database: [https://useast.ensembl.org/info/data/ftp/index.html](https://useast.ensembl.org/info/data/ftp/index.html).

3. Open a Unix/Linux command line environment (“Terminal” application in a Linux operating system or macOS). Generate the STAR alignment index of the reference genome using the STAR index utility as described at [https://github.com/alexdobin/STAR](https://github.com/alexdobin/STAR) and in Dobin and Gingeras (2015) using the following command:

```
STAR -runMode genomeGenerate -genomeDir <reference directory> -genomeFastaFiles <reference genome fasta file> -sjdbGTFfile <reference genome gtf file> -sjdbOverhang 49
```

For example, for genome reference fasta file “Homo_sapiens.GRCh38.dna.fa” and gene annotation file “Homo_sapiens.GRCh38.92.gtf” downloaded in step 2, use the following command to create the reference STAR index:

```
STAR -runMode genomeGenerate -genomeDir human_GRCh38 -genomeFastaFiles Homo_sapiens.GRCh38.dna.fa -sjdbGTFfile Homo_sapiens.GRCh38.92.gtf -sjdbOverhang 49
```

4. Generate the BWA alignment index of the reference genome - for example:

```
bwa index Homo_sapiens.GRCh38.dna.fa
```

5. Download external files with genomic coordinates and annotations of genomic features of interest to be analyzed for overlaps and associations with the expression, chromatin, and replication data: genomic repeats, Hi-C compartments, lamina associated domains (LADs) etc.

⚠ CRITICAL: Apart from the coordinates of genomic repeats, most other types of genomic features are specific to a given cell type, and these external files should be carefully selected to ensure their compatibility with the project.
KEY RESOURCES TABLE

| REAGENT or RESOURCE | SOURCE | IDENTIFIER |
|---------------------|--------|------------|
| Software and algorithms |        |            |
| BWA 0.7.13          | (Li and Durbin, 2009) | http://bio-bwa.sourceforge.net |
| SAMTools 1.4        | (Li et al., 2009) | http://samtools.sourceforge.net |
| DeepTools 2.4.3     | (Ramirez et al., 2016) | https://deeptools.readthedocs.io/en/develop/content/installation.html |
| chromHMM v1.21      | (Ernst and Kellis, 2017) | http://compbio.mit.edu/ChromHMM/ |
| HOMER v4.10.3       | (Heinz et al., 2010) | http://homer.ucsd.edu/homer/ |
| STAR 2.3.0e,r291    | (Dobin et al., 2013) | https://github.com/alexdobin/STAR |
| Picard 1.47         | https://broadinstitute.github.io/picard | https://broadinstitute.github.io/picard |
| HTSeq 0.5.4         | (Anders et al., 2015) | https://htseq.readthedocs.io/en/release_0.11.1/index.html |
| EdgeR 3.34.0        | (Robinson et al., 2010) | https://www.bioconductor.org/packages/release/bioc/html/edgeR.html |
| Java 1.7.0_71       |        |            |
| R                   | R Core Team, 2020 | https://www.r-project.org |
| preprocessCore 1.54.0 |        | https://bioconductor.org/packages/release/bioc/html/preprocessCore.html |
| Bedtoolsr 2.30.0-1  | http://phanstiel-lab.med.unc.edu/bedtoolsr.html |
| ComplexHeatmap 2.8.0 | https://bioconductor.org/packages/release/bioc/html/ComplexHeatmap.html |
| tidyverse 1.3.1     | Wickham et al. (2019) | https://www.tidyverse.org |
| circlize 0.4.14     | https://cran.r-project.org/web/packages/circlize/index.html |
| runner 0.4.1        | https://cran.r-project.org/web/packages/runner/index.html |
| randomForest 4.6-14 | https://cran.r-project.org/web/packages/randomForest/index.html |
| bedtoolsr 2.30.0    | http://phanstiel-lab.med.unc.edu/bedtoolsr.html |
| Other               |        |            |
| Repli-seq, ChIP-seq, CUT&RUN, ATAC-seq, RNA-seq data | Provided by the user, e.g., (Van Rechem et al., 2021) | Example: GEO GSE175751 https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE175751 |
| Reference genome annotation | Ensembl | https://useast.ensembl.org/info/data/ftp/index.html |

STEP-BY-STEP METHOD DETAILS

Map sequencing reads from Repli-seq, ATAC-seq, ChIP-seq, CUT&RUN platforms

© Timing: 1–3 h per sample

Sequencing reads from FASTQ files are mapped to the reference genome of choice and alignment files in BAM format are generated along with basic data quality metrics.

1. Perform alignment using bwa aln command:

```
bwa aln path_to_reference_genome exp_R1.fq > exp_R1.sai
bwa aln path_to_reference_genome exp_R2.fq > exp_R2.sai
```
where *path_to_reference_genome* is the location of the pre-computed BWA index file, *exp_R1.fq* and *exp_R2.fq* are the names of the input fastq files with read 1 and read 2, respectively, *exp_R1.sai* and *exp_R2.sai* are the output files in the SAI format.

2. With .sai files generated at the previous step, produce genomic alignment in the SAM format using either
   a. bwa sampe command for paired-ended reads:

   ```shell
   bwa sampe path_to_reference_genome -f exp.sam exp_R1.sai exp_R2.sai exp_R1.fq exp_R2.fq
   ```

   or

   b. bwa samse for single-ended reads:

   ```shell
   bwa samse path_to_reference_genome -f exp.sam exp_R1.sai exp_R1.fq
   ```

   where *exp.sam* is the output alignment file in the SAM format.

3. Compress and sort the resulting SAM alignment file into a BAM file *exp.bam* using samtools:

   ```shell
   samtools view -bS -o exp.bam exp.sam
   ```

4. Using alignment BAM files, calculate read mapping rates for each sample:

   ```shell
   samtools flagstat exp.bam
   ```

5. Sort the BAM file:

   ```shell
   samtools sort -o exp.sort.bam exp.bam
   ```

6. Remove duplicate reads and calculate duplication rates for each sample:

   ```shell
   samtools rmdup exp.sort.bam exp.rmdup.bam
   ```

   where *exp.rmdup.bam* is the output BAM file with duplicates removed.

7. Generate bigwig files of genome-wide read density for each sample and input-normalized density for ChIP-seq samples.
   a. Generate bigwig files with read density normalized by total number of reads for each alignment file:

   ```shell
   bamCoverage -b exp.bam -o exp.bw -e -normalizeUsingRPKM
   ```

   where *exp.bw* is the output bigwig file.

   b. Calculate genome-wide bigwig tracks of input-normalized ratio for ChIP-seq experiments:
where ChIP.bam and input.bam are BAM alignment files generated at steps 4–6 for ChIP and input samples, respectively.

**Map RNA-seq sequencing reads**

© Timing: 1–4 h per sample

Sequencing reads from FASTQ files are mapped to the reference genome of choice in a splice-aware fashion.

8. Run STAR aligner to produce alignment BAM files using the following command:

```bash
STAR
```

Note: Consecutive STAR jobs with the same reference genome can be run using shared memory using --genomeLoad LoadAndKeep command option, which saves the time required to load the reference index for each job.

**Calculate Repli-seq, ATAC-seq, ChiP-seq, CUT&RUN, and RNA-seq read density across the genome**

© Timing: 0.5–1 h per sample

Reads mapped to the genome at the previous steps are counted in 50 Kb genomic bins across the genome, at the genomic resolution identical to the resolution of Repli-seq.

9. Calculate coverage over 50 Kb genomic bins across the genome.
   a. Download reference genome chromosome sizes and create 50 kb genomic bin file in BED format:

```
mysql -user=genome -host=genome-mysql.cse.ucsc.edu -A -e "select chrom, size from hg19.chromInfo" > hg19.genome
bedtools makewindows -g hg19.genome -w 50000 -s 50000 > hg19.50kb.bed
```

where reference genome hg19 can be replaced by your reference genome.

b. Calculate coverage over these genomic bins using all previously generated BAM alignment files with removed duplicates from step 6:
Bam_list can be either single bam files ("chip1.bam") or multiple space-delimited BAM file names from a batch of multiple samples (e.g., "chip1.bam chip2.bam"). For example, to simultaneously process Repli-seq samples from four time points S1-S4 in two replicate experiments sample1 and sample2, one can supply the list of eight BAM files:

```
"RT_sample1_S1.bam RT_sample1_S2.bam RT_sample1_S3.bam RT_sample1_S4.bam RT_sample2_S1.bam RT_sample2_S2.bam RT_sample2_S3.bam RT_sample2_S4.bam"
```

output_rawCount.txt is a tab-delimited read count table with columns of all sample from bam_list coverage and rows of genomic bins from hg19.50 kb.bed.

10. Sort output count table by chromosome and starting positions of each genomic bin:

```
cat output_rawCount.txt | sort -k1,1 -k2,2n > output_rawCount.sort.txt
```

⚠️ CRITICAL: This step ensures the consistency of genomic bins in all rows in the integrated count tables of multiple samples that are used in downstream steps.

11. Repeat steps 9 and 10 to calculate similar genomic coverage tables for ChIP-seq, ATAC-seq and RNA-seq samples.

**Note:** To integrate RNA-seq into genomic analysis with ChIP-seq and Repli-seq data, we calculated RNA-seq reads density over genomic bins in addition to a more standard quantification of RNA-seq read density over transcripts.

**Call peaks of focused chromatin marks and chromatin accessibility regions, calculate peak density across the genome**

⊙ Timing: 0.5–1 h per sample

12. Using genome-wide alignments of ChIP-seq reads from ChIP and input DNA samples for focused chromatin marks (H3K27ac, H3K4me1-3 etc), define narrow peaks of mark enrichment across the genome. Run the HOMER tool using commands

```
makeTagDirectory HOMER_tag_directory path_to_bam_file
findPeaks HOMER_tag_directory -style histone
```

13. To quantify large-scale genomic density of these narrow peaks, calculate the number of peaks within +/-250 kb window near each original 50 kb genomic bin.

```
cat peaks.txt | grep -v "^#" | cut -f2-4 | bedtools window -w 250000 -a bin.bed -b -c > peakCount.mark.txt
```

Repeat steps 12 and 13 for bam files of all narrow histone marks and ATAC-seq data from steps 4–6.
Assessment of quality and replicate consistency of Repli-seq, ATAC-seq, ChIP-seq, CUT&RUN data

⏱ Timing: 1 h per replicate set

Using alignments and read counts produced at the previous steps, basic data quality metrics are generated, replicates are assessed for the consistency of read counts over genomic bins and PCA plots were generated showing similarity between samples.

14. Open R and load the ‘preprocessCore’ package using the following command:

```r
> library(preprocessCore)
```

15. Read the file with a table of read counts in genomic bins (produced at step 10), for example from file output_RT_rawCount.txt, and normalize these counts to produce the matrix of counts per million (CPM):

```r
> setwd(''path/to/files/'')
> tab <- read.table(''output_RT_rawCount.txt'', header=TRUE)
> RT.bin.cnt <- as.matrix(tab[, -(1:3)])
> bin.bed = tab[, 1:3]
> rownames(bin.bed) <- paste(''bin'', 1:dim(bin.bed)[1], sep = '')
> bin.bed$binID <- rownames(bin.bed)
> RT.bin.cpm <- t(t(RT.bin.cnt) / (colSums(RT.bin.cnt) / 1e6))
```

16. Use the CPM matrix produced at the previous step to generate pairwise scatter plots comparing read counts over genomic bins between pairs of replicates, and calculate Pearson correlation coefficients between replicates:

```r
> plot(RT.bin.cpm[, 'sample_1_S1'], RT.bin.cpm[, 'sample_2_S1'], log = 'xy')
> cor(log10(RT.bin.cpm[, 'sample_1_S1']) + 0.01, log10(RT.bin.cpm[, 'sample_2_S1']) + 0.01))
```

17. Use the CPM matrix produced at step 15 to perform PCA analysis over all samples

```r
> pca <- prcomp(t(log10(RT.bin.cpm + 0.01)))
> plot(pca$x[, 1], pca$x[, 2], type = 'n', xlab = 'PC1', ylab = 'PC2')
> text(pca$x[, 1], pca$x[, 2], rownames(pca$x))
```

Perform steps 15–17 to import and analyze genomic coverage tables produced at steps 9 and 10 for Repli-seq, ChIP-seq, and ATAC-seq samples.
Perform quantile normalization of Repli-seq, ATAC-seq, ChIP-seq, CUT&RUN signals across the genome

Timing: 0.5–1 h per sample

To allow for the comparison between conditions and mitigate possible differences in the dynamic range of individual samples, we recommend quantile normalization similar to (Marchal et al., 2018). For the analyses at the resolution of 50 Kb similar to the resolution of Repli-seq, this normalization is followed by Loess smoothing of the genomic tracks. Quantile normalization is based on the assumption that signal intensity is not being systematically increased or reduced across the bulk of the genome in one condition compared to the other. Therefore, it is not appropriate to apply quantile normalization when this assumption is violated, for example, when a chromatin mark is expected to be systematically depleted compared to control across the genome.

18. Perform quantile normalization on multiple samples of the same nature (e.g., replicate Repli-seq experiments). We recommend normalizing different RT time points separately:

```r
> RT.bin.S1.norm <- normalize.quantiles(RT.bin.cpm[,c("RPE.ct1.S1","RPE.ct2.S1","RPE.4A1.S1","RPE.4A2.S1")])
> RT.bin.S2.norm <- normalize.quantiles(RT.bin.cpm[,c("RPE.ct1.S2","RPE.ct2.S2","RPE.4A1.S2","RPE.4A2.S2")])
```

19. Repeat step 18 to normalize Repli-seq coverage for each time point, then combine all quantile-normalized Repli-seq counts (for example, counts for four time points S1-S4) into a single matrix:

```r
> RT.bin.norm <- cbind(RT.bin.S1.norm, RT.bin.S2.norm, RT.bin.S3.norm, RT.bin.S4.norm)
```

20. Run Loess smoothing of the resulting normalized tracks and write smoothened values in the output wig files, separate for each sample:

```r
> RT.bin.loess <- RT.bin.norm #initialization
> for(i in 1:dim(RT.bin.norm)[2]){
  for(chr in unique(bin.bed[,1])){
    idx <- which(bin.bed[,1]==chr)
    x <- RT.bin.bed[idx,2]
    y <- RT.bin.norm[idx,i]
    lspan <-500000/(max(bin_crd[idx])-min(bin_crd[idx]))
    Rpla <- loess(y ~ x, span=lspan)
    RT.bin.loess[idx,i] <- Rpla$fitted
  }
} 
> RT.bin.loess [RT.bin.loess <0]=0
```
If necessary, perform quantile normalization steps 18–20 for the data from any other experimental platform (ChIP-seq, ATAC-seq, or RNA-seq), using the corresponding CPM file generated at step 15 as input.

**Annotate chromatin states across the genome**

© Timing: 2–4 h per dataset

Aggregate analyses of multiple chromatin marks across the genome reveal recurring combinations of co-localized marks that often correspond to specific genomic elements and functions, such as active or repressed promoters, enhancers, actively transcribed gene bodies, regions of compacted heterochromatin, etc. These combinatory patterns of chromatin marks are termed “chromatin states”.

Based on the combinations of levels of histone marks, define and annotate chromatin states at a standard 1 Kb resolution and a lower 50 Kb resolution similar to the resolution of Repli-seq. In Linux, use chromHMM command:

21. Binarize genome coverage from multiple ChIP-seq BAM files for histone marks into the presence/absence calls at either 50 kb or 1 kb resolution.

```bash
java -mx4000M -jar ChromHMM.jar BinarizeBam -b bin_size ChromHMM/CHROMSIZES/hg19.txt path_to_BAM_files_folder cell_mark_file_table path_to_binarized_coverage_folder
```

where bin_size could be either 1,000 (1 Kb) or 50,000 (50 Kb) and cell_mark_file_table is a tab-delimited table with columns of cell type, name of the histone mark, the corresponding ChIP BAM file and the ChIP-seq input DNA BAM file, for example:

| RPE | H3K9me1 | H3K9me1.bam | Input.bam |
|-----|---------|-------------|-----------|
| RPE | H3K9me2 | H3K9me2.bam | Input.bam |
| RPE | H3K9me3 | H3K9me3.bam | Input.bam |

22. Learn chromatin states using ChromHMM.

```bash
$ java -mx4000M -jar ChromHMM.jar LearnModel -b bin_size path_to_binarized_coverage_folder path_to_output_folder number_of_state hg19
```

⚠️ CRITICAL: It is important to perform manual inspection of genomic tracks in a genomic viewer, for example, IGV (https://software.broadinstitute.org/software/igv/), to assess regions in various chromatin states (*.dense.bed files generated at step 22) called by chromHMM (Ernst and Kellis, 2017) and annotate these states (e.g., heterochromatin, active state, etc.) based on the observed combinations of chromatin marks (bigWig files generated at step 7) and the overlap with functional genomic elements (active or silent gene bodies, promoters, enhancers etc.). chromHMM requires a predefined number of states as input; therefore, multiple runs with various state numbers may be necessary to arrive at the optimal set of biologically relevant states at a given genomic resolution. For example, at the standard higher resolution, n=18 states was optimal, whereas at
the Repli-seq (50 Kb) resolution, the optimal number of states was n=4 (Van Rechem et al., 2021).

Sub-clustering genomic regions with the same chromatin state by their replication timing

★ Timing: 1 h per group of datasets

To understand the general replication behavior of regions under specific chromatin states, it is useful to analyze these chromatin states separately. Within each chromatin state, it is informative to define and analyze subgroups of regions sharing similar replication timing.

23. Cluster genomic bins in a given chromatin state by the patterns of Repli-seq signal across profiled time points.

In R, load chromHMM state annotation and cluster genomic bins within each state by their replication timing patterns.

```r
> chromHMM.bed <- read.table("path_to_chromHMM_state_bed",sep="\t",skip = 1)[,1:4]
> colnames(chromHMM.bed) <- c("chr","start","end","state")
> target.state <- "1"
> target.state.bins <- bedtoolsr::bt.intersect(bin.bed,chromHMM.bed[chromHMM.bed$state==target.state,],wa=T)
> state.RT <- log10(RT.bin.norm[target.state.bins,c("RPE.ct1.S1","RPE.ct1.S2","RPE.ct1.S3","RPE.ct1.S4")]+1e-3)
> km <- kmeans(state.RT,4)
> ComplexHeatmap::Heatmap(state.RT,
  use_raster=TRUE,
  show_row_names = FALSE,
  show_column_names = FALSE,
  show_row_dend = FALSE,
  cluster_columns=FALSE,
  split = km$cluster,
  cluster_row_slices=T,
  name='RPE')
```

⚠ CRITICAL: K-means clustering requires a pre-defined number of clusters. It is important to inspect the resulting clustered heatmap of Repli-seq intensities across time points to determine the optimal number of clusters for each chromatin state. In the example above, 4 clusters produced an optimal partitioning of replication patterns among all regions in the given chromatin state.

Assessing the average temporal dynamics of replication, chromatin marks, chromatin accessibility, and transcription in individual subclusters

★ Timing: 2 h per group of datasets
Based on the subgrouping of genomic regions by their RT produced at the previous step, it is useful to focus on each subgroup and graphically compare the average patterns of the temporal progression of replication signal against the progression of ChIP-seq enrichment of individual chromatin marks or ATAC-seq chromatin accessibility across the cell cycle. Based on our observations (Van Rechem et al., 2021), many of these patterns are consistent with a dilution of chromatin marks as the new DNA copy is synthesized during replication (Alabert et al., 2014; Reverón-Gómez et al., 2018; Stewart-Morgan et al., 2020), however, the patterns in some regions deviate from these associations, suggesting a more complex and active relationship between histone modifications and RT.

24. To investigate these temporal dynamics throughout cell cycle, normalized Repli-seq tag density, input-normalized ChIP-seq tag density of broad histone marks, lower-resolution ATAC-seq tag density, and RNA-seq signal are calculated in each 50 Kb bin at each profiled time point (G1, ES, LS, G2/M), and then normalized as a log2 ratio to the average of all time points for the given bin.

a. The calculation of these average-normalized values is performed using these commands (with H3K36me3 ChIP-seq coverage as an example):

```r
> H3K36me3.bin.enrich <- log2(H3K36me3.bin.cpm+pseudo) - log2(Input.bin.cpm+pseudo)
> rownames(H3K36me3.bin.enrich) <- rownames(bin.bed)
> H3K36me3.ctrl.enrich <- (H3K36me3.bin.enrich[,c("ct1-G1-K36me3","ct1-ES-K36me3","ct1-LS-K36me3","ct1-G2-K36me3")]+
H3K36me3.bin.enrich[,c("ct2-G1-K36me3","ct2-ES-K36me3","ct2-LS-K36me3","ct2-G2-K36me3")])/2
> H3K36me3.cycle <- H3K36me3.ctrl.enrich-rowMeans(H3K36me3.ctrl.enrich)
```

b. These normalized values across cell cycle can be visualized as a heatmap using this command:

```r
> ComplexHeatmap::Heatmap(H3K36me3.cycle[target.state.bins,],
  use_raster=TRUE,
  show_row_names = FALSE,
  show_column_names = FALSE,
  show_row_dend = FALSE,
  cluster_columns=FALSE,
  cluster_rows=FALSE,
  split = km$cluster)
```

Calculate replication timing index (RTI) as a measure of RT based on Repli-seq data at multiple time points across cell cycle

© Timing: 0.5 h per set of Repli-seq profiles

Using counts of Repli-seq reads at 50 Kb genomic bins for multiple profiling S-phase time points (n ≥ 2) produced at step 20, calculate the Replication Timing Index (RTI, (Van Rechem et al., 2021)) as the estimate of replication timing for each genomic locus across the genome. RTI calculation is
based on a weighted sum of normalized replication signals (Repli-seq read densities $D_n$) from each time point $n$:

$$ RTI = \frac{\sum_{n=1}^{N} n D_n}{\sum_{n=1}^{N} D_n} $$

where $n$ is the time point of the S-phase (for example 1–4 corresponding to time points S1 to S4 in (Van Rechem et al., 2021)) and $D_n$ is the density of BrdU reads (per bp) within the given region at this time point.

25. From the table of normalized and smoothened Repli-seq coverage across the genome (produced at step 20), calculate RTI at all genomic bins and average RTI across biological replicates:

a. Calculate RTI at all genomic bins across the genome using the following R function:

```r
> calc_rt_index <- function(dat) {
  dat.norm <- dat/rowSums(dat)
  for(i in 1:dim(dat.norm)[2]){
    sum = sum+i*dat.norm[,i]
  }
  RTI = (sum-1)/(dim(dat.norm)[2]-1) # scale from 0 to 1 instead of 1 to 4
  return(RTI)
}
```

```r
> RTI.ctrl1 <- calc_rt_index(RT.bin.loess[,c('ctrl1_S1', 'ctrl1_S2', 'ctrl1_S3', 'ctrl1_S4')])
> RTI.ctrl2 <- calc_rt_index(RT.bin.loess[,c('ctrl1_S1', 'ctrl1_S2', 'ctrl1_S3', 'ctrl1_S4')])
```

where `calc_rt_index` function calculated RTI using columns S1, S2 ... S4 in LOESS smoothen RT coverage.

b. Calculate average RTI values among biological replicates:

```r
> RTI.ctrl <- rowMeans(cbind(RTI.oe.1, RTI.oe.2))
```

Quantify the magnitude and statistical significance of RTI differences between biological conditions

© Timing: 0.5 h

Use genome-wide RTI values in two conditions, e.g., in control vs perturbation (gene overexpression, knockdown etc), to estimate the magnitude of changes in RT and their statistical significance at each genomic locus. Call statistically significant RT differences.

26. Calculate RTI differences between replicate RTI samples and between two conditions using this R command:

```r
> RTI.diff <- rowMeans(cbind(RTI.oe.1, RTI.oe.2)) - rowMeans(cbind(RTI.ctrl.1, RTI.ctrl.2))
```
27. Based on RTI differences between biological Repli-seq replicates, build a genome-wide distribution of random RTI differences. Estimate mean and standard deviation and then define the difference cutoff corresponding to two standard deviations away from the mean. This approach is conceptually similar to the traditionally used approach for the differences in Early-to-Late ratios (Marchal et al., 2018; Rivera-Mulia et al., 2015; Sarni et al., 2020), but applied to RTI differences:

```r
> RTI.rep.sd <- sd(c(RTI.oe.1-RTI.oe.2, RTI.ctrl.1-RTI.ctrl.1), na.rm=T)
```

28. Based on these cutoffs, identify genomic bins with significant RTI changes and merge adjacent bins into larger regions of RT change:

```r
> RTI.diff.pval <- 2*pnorm(-abs(RTI.diff/RTI.rep.sd))
> RTI.diff.bins <- RTI.bin.bed[abs(RTI.diff)>0.05 & RTI.diff.pval<0.01,]
> RTI.diff.region <- bedtoolsr::bt.merge(RTI.diff.bins)
```

**Train and evaluate computational models for the prediction of RT from the combined dynamics of ATAC-seq, ChIP-seq/CUT&RUN, and RNA-seq signals across the genome**

© Timing: 5 h per group of datasets

One way to quantitatively assess the association between RT and chromatin modifications, chromatin accessibility, and gene expression is to train and evaluate genome-wide computational models of RT as a function of the combination of ChIP-seq, ATAC-seq, and RNA-seq signals at a given genomic region. Assessment of the predictive power of these models may help quantify the functional link between RT and chromatin remodeling, chromatin accessibility, transcription, and compare the contributions of each individual component to this link. Either basic linear regression or more advanced machine learning approaches, e.g., random forest techniques, can be used to build these quantitative models.

Here we perform training and validation of linear regression models using the leave-one-out cross-validation approach, where in multiple repetitions, a randomly sampled quarter of all 50 Kb genomic bins was used as the testing set after the other three quarters were used as the training set.

29. Load and combine multiple predictive features for each genomic bin (ChIP-seq, ATAC-seq, RNA-seq read density from step 15, number of peaks in the vicinity of genomic bins from step 13) and RTI for each bin from step 25:

```r
> peak.num.ATAC <- read.table("peakCount.ATAC.txt", header=TRUE)
> peak.num.H3K27ac <- read.table("peakCount.H3K27ac.txt", header=TRUE)
...
> dat <- data.frame(rt=RTI.ctrl, ChIP.bin.CPM, ATAC.bin.CPM, peak.num.ATAC, peak.num.H3K27ac,...)
```

30. Select a random subset of chromosomes whose total length amounts to ~25% of the genome. Leave genomic bins within these chromosomes for the later testing, and use the rest of the genomic bins as a training set:
31. Use the training set, generate a linear regression model of RTI as a function of multiple ChIP-seq, ATAC-seq, and RNA-seq values:

```r
> fit <- lm(rt ~., data = dat_trn)
```

32. Assess the accuracy of the resulting model using the testing set, with Pearson R between predicted and observed RTI values as the accuracy measure:

```r
> pred <- predict(fit, newdata = dat_tst)
> cor(dat_tst$rt, pred)
```

33. Generate a scatter plot of observed vs predicted RTI:

```r
> plot(pred, dat_tst$rt, xlab = "Predicted", ylab = "Actual")
```

To assess the contributions of various combinations of predictive variables (various combinations of enrichment of ChIP-seq marks, ATAC-seq, RNA-seq) as well as each individual value by itself, one can train and evaluate the accuracy of partial models based on these variables.

34. Perform training and validation of random forest models using leave-one-out cross-validation approach identical to steps 29–33, with the exception of step 31 where the modeling is changed to the random forest technique:

```r
> library(randomForest)
> fit <- randomForest(rt ~., data = dat_trn, mtry = round(sqrt(dim(dat_trn)[2]-1)), importance = TRUE, ntree = 500)
```

**Note:** As additional approaches to assess the importance of individual features (individual chromatin marks, ATAC-seq, RNA-seq) in predicting RT, one can use various metrics of feature importance developed for random forest models: Gini importance, permutation feature importance, Drop Column feature importance etc.
Train and evaluate computational models for the prediction of RTI changes from the combined ATAC-seq, ChIP-seq/CUT&RUN, and RNA-seq signals across the genome

© Timing: 5 h per group of datasets

A direct approach to assess the causal effect of a specific protein or cellular function on RT is to experimentally perturb this protein or function and analyze the resulting changes in RT across the genome. A genome-wide approach to this analysis is to train and evaluate computational models of quantitative RTI changes as a function of the combination of ChIP-seq, ATAC-seq, and RNA-seq signals and their changes at each genomic region. As in steps 29–34, these models can be based on simpler and tractable linear regression or more advanced machine learning approaches (e.g., random forest techniques).

Perform training and validation of linear regression models using the leave-one-out cross-validation approach, where in multiple repetitions, a randomly sampled quarter of all 50 Kb genomic bins was used as the testing set after the other three quarters were used as the training set.

35. Load and combine multiple predictive features for each genomic bin (ChIP-seq, ATAC-seq, RNA-seq read density from step 15, number of peaks in the vicinity of genomic bins from step 13) and RTI for each bin from step 25. Similar to step 29.

```r
> pseudo <- 0.01
> log2.ATAC.bin <- log2(ATAC.bin.CPM[,''OE'']+pseudo)- log2(ATAC.bin.CPM[,''ctrl'']
+pseudo)
```

Or:

```r
> log2.ATAC.bin <- log2(ATAC.bin.CPM[,c(''OE-S1'',''OE-S2'',''OE-S3'',''OE-S4'')]
+pseudo)- log2(ATAC.bin.CPM[,c(''control-S1'',''control-S2'',''control-S3'',''control-S4''
) +pseudo)
```

...  
> dat <- data.frame(drt=diff.RTI, log2.ChIP.bin, log2.ATAC.bin)

36. Select a random subset of chromosomes whose total length amounts to ~25% of the genome. Leave genomic bins within these chromosomes for the later testing, and use the rest of the genomic bins as a training set:

```r
> rand.chr.list <- sample(unique(bin.bed[,1]))
> inp_idx <- c()
> for(chr in rand.chr.list){
>   inp_idx <- c(inp_idx,which(bin.bed[,1]==chr))
>   if(length(inp_idx) > floor(nrow(bin.bed)/4)){
>     break
>   }
> }
> dat_trn = dat[-inp_idx,]
> dat_tst = dat[inp_idx,]
```
37. Using the training set, generate a linear regression model of RTI change as a function of chromatin state annotation, multiple ChIP-seq, ATAC-seq, and RNA-seq values and their changes:

```r
> fit <- lm(drt ~ ., data = dat_trn)
```

38. Perform steps 32 and 33 to assess the accuracy of the resulting model using the testing set, with Pearson R between predicted and observed RTI values as the accuracy measure, and generate a scatter plot of observed vs predicted differential RTI.

To assess the contributions of various combinations of predictive variables (combinations of enrichment of ChIP-seq marks, ATAC-seq, RNA-seq) as well as each value by itself, one can train and evaluate the accuracy of partial models based on these variables.

39. Perform step 34 to train and validate random forest models using the leave-one-out cross-validation approach.

**Note:** As additional approaches to assess the importance of individual features (individual chromatin marks, ATAC-seq, RNA-seq) in predicting RT, one can use various metrics of feature importance developed for random forest models: Gini importance, permutation feature importance, Drop Column feature importance etc.

**Define genomic regions with specific types of local replication patterns**

© Timing: 2 h per RTI dataset

Using genome-wide RTI values calculated at the previous step, define genomic locations of initiation zones, termination sites, initiation constant timing regions, and termination constant timing regions (Van Rechem et al., 2021; Zhao et al., 2020).

40. Define initiation zones (IZ) as local minima, termination sites (TS) as local maxima in RTI profile and constant timing regions (CTRs) as wide regions of > 500 Kb with almost constant RTI profile using these commands:

```r
> library(runner)
> library(tidyverse)
> w <- 10
> var.cutoff <- 1e-3
> bs <- head(bin.bed[,3]-bin.bed[,2],1)
> dist <- floor(w/2)*bs
> rti = RTI.ct
> win.stat <- data.frame(max=runner(rti,k=w,f=max),
                        min=runner(rti,k=w,f=min),
                        var=runner(rti,k=w,f=var),
                        cnt.noneNA=runner(rti,k=w,f=function(x){sum(!is.na(x))}),
                        cnt.chr=runner(bin.bed[,1],k=w,f=function(x){length(unique(x))}))
```
Note: Specific parameter values in these definitions may be adjusted depending on the number of time points, genomic resolution, quality, structure of Repli-seq data, as well as the specific biological goals of the study.

EXPECTED OUTCOMES

This protocol produces a few major outcomes. First, it calculates RTI values for all genomic bins across the genome for each sample and condition, as shown in an example of RTI genomic tracks in Figure 1. Second, it identifies genomic regions with differential RT between conditions (Figure 1). Third, it computationally defines genomic regions with specific types of local replication patterns: initiation zones, termination sites, constant timing regions, and timing transition regions (Figure 2). Finally, it develops, trains, and tests linear and machine learning models for genome-wide quantitative prediction of RT and its changes from the combination of ChIP-seq, ATAC-seq, and RNA-seq signals and their changes at each genomic region. As an example of the quality of such predictions, Figure 3 shows a scatter plot of predicted vs observed changes of RTI between control RPE cells and cells overexpressing KDM4A protein at each individual 50 Kb bin across the genome, where RT prediction was made based on the densities of seven broad chromatin marks in the two conditions (see also (Van Rechem et al., 2021)).

LIMITATIONS

The genomic resolution of the protocol is limited by the current experimental Repli-seq resolution of ~50 Kb. To accommodate this resolution, the protocol uses 50 Kb genomic bins and calculates the density of broad chromatin marks with diffuse distribution patterns (H3K36me2-3, H3K9me3, H3K27me3, etc) within each bin, whereas for focused marks with narrow peaks (H3K27ac, H3K4me3, etc) and ATAC-seq peaks, the counts or densities of individual peaks are used as a measure of the mark’s level within a bin.

RTI as a measure of RT can accommodate any number of time points profiled by Repli-seq during the cell cycle. However, the actual temporal resolution of RTI is always limited by the number of time points profiled by Repli-seq in a given study. The examples described in this protocol are based on four profiled time points, but a larger number of time points would increase the temporal resolution of RTI and the sensitivity of detecting RTI changes between experimental conditions.
Although the protocol applies standard input normalization techniques to ChIP-seq data, one cannot completely exclude the possibility that this normalization does not fully eliminate the confounding correlation of input DNA, which may be affected, for example, by DNA copy number and might additionally contribute to the relationships between ChIP-seq signal with RT. However, there are a few arguments in favor of the biological role of chromatin marks as opposed to DNA copy number as the main driver of these correlations. For example, different individual chromatin marks have different and sometimes opposite correlation patterns with RT despite being normalized by the same input (Van Rechem et al., 2021). In addition, these data reproduce previously reported correlations with RT and structural chromatin compartments (Van Rechem et al., 2021).

**TROUBLESHOOTING**

**Problem 1**
Basic QC parameters (Q30 read fraction, mapping rate etc) of ChIP-seq, CUT&RUN, RNA-seq, ATAC-seq, or Repli-seq samples do not meet recommended standards.

**Potential solution**
Assess the experiments in question and consider repeating these experiments to produce new sequencing libraries.

**Problem 2**
The number of reads in fastq files used in step 1 is below recommended amounts.

---

Figure 1. Example of genomic tracks of replication timing index (RTI) at a genomic region (5 Mb neighborhood of human chromosome 5), based on Repli-seq data at four time points within S-phase of cell cycle. Black, two replicates in control RPE cells. Red, two replicates in RPE cells with KDM4A overexpression. A region with a change of RT in KDM4A overexpressing cells to a later timing is shown in the center.

Figure 2. Example of RTI track with identified local replication patterns: Initiation zone (IZ, red), termination site (TS, blue), and constant timing region (CTR, gray).
Problem 3
The number of mapped reads produced at steps 1–6 is within normal range but genomic tracks produced at step 7 do not show strong peaks or regions of enrichment and/or suggest a low signal-to-noise ratio.

Potential solution
Repeat the experiments in question.

Problem 4
Replicate consistency of ChIP-, ATAC-, RNA-, or Repli-seq is low according to pairwise scatter plots produced at step 16 (Pearson R<0.98–0.99) or to a PCA plot produced at step 17 (replicates do not clearly group together).

Potential solution
Repeat the experiments corresponding to the outlier replicates.

Problem 5
Manual inspection and genome-wide analyses of chromatin states called by chromHMM (step 21) do not show clear correspondence to specific biological functions.

Potential solution
Consider changing the number of states submitted to chromHMM as a parameter. If possible, consider expanding the set of profiled histone marks to include a wider range of chromatin functions (active chromatin, heterochromatin, H3K27me3-enriched regions, intergenic H3K36me2 regions, etc.).
Problem 6
Regions of statistically significant RTI difference (steps 26–28) are too few (fewer than several thousand 50 Kb bins).

Potential solution
Carefully inspect the distribution of RTI differences between replicates constructed at step 27. This distribution should be symmetrical and near-normal, with pronounced left and right shoulders. If these regions do not show a relatively random localization across the genome and are concentrated to only a few genomic neighborhoods, carefully inspect RTI tracks for technical artefacts but also consider biologically plausible explanations of this non-random localization.

Problem 7
Quantitative predictions of RTI (steps 29–34) or RTI changes (steps 35–39) are not showing a substantive correlation with observed data.

Potential solution
Confirm sufficient size of statistical samples used in training and testing sets and consider including additional predictive parameters (ChIP-seq data for additional histone marks, chromatin states of each region, etc) or changing the type of predictive model.

Problem 8
The assignment of local RT patterns to genomic regions (step 40) does not look biologically relevant according to manual inspection of RTI tracks.

Potential solution
Consider adjusting quantitative parameters used to define these local RT patterns (step 40). The optimal parameter values may depend on the number of time points profiled by Repli-seq, quality of Repli-seq tracks, genomic resolution (bin size), etc.

RESOURCE AVAILABILITY
Lead contact
Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Ruslan I. Sadreyev (sadreyev@molbio.mgh.harvard.edu).

Materials availability
This study did not generate new unique reagents.

Data and code availability
All original code has been deposited and is publicly available at GitHub: https://github.com/MolBioBioinformatics/RTI_Epigenome, Zenodo version of record: https://doi.org/10.5281/zenodo.7135154. All sequencing data that were analyzed using this protocol and published in (Van Rechem et al., 2021) have been deposited in GEO: GSE175751.

ACKNOWLEDGMENTS
R.I.S. was supported by NIH NIDDK P30 DK040561. J.R.W. was supported by NIH R01GM097360 and the NIH/NCI P30 CA006927. J.R.W. is a recipient of an American Lung Association Lung Cancer Discovery Award.

AUTHOR CONTRIBUTIONS
F.J. and R.S.: conceptualized and designed quantitative analyses. F.J.: performed computational analyses. F.J., R.S., C.V.R., and J.W.: methodology, investigation, writing – original and revised draft.
DELACATION OF INTERESTS

J.R.W. has or is serving as a consultant or advisor to the following companies or organizations: Daiichi Sankyo, Inc., Vyne Therapeutics, Inc., and Salarius Pharmaceuticals (consulted and has sponsored research).

REFERENCES

Alabert, C., Bukowski-Wills, J.-C., Lee, S.-B., Kustatscher, G., Nakamura, K., de Lima Alves, F., Menard, P., Meijer, J., Reppilber, J., and Groth, A. (2014). Nascent chromatin capture proteomics determines chromatin dynamics during DNA replication and identifies unknown fork components. Nat. Cell Biol. 16, 281–293. https://doi.org/10.1038/ncb2918.

Anders, S., Pyl, P.T., and Huber, W. (2015). HTSeq—a Python framework to work with high-throughput sequencing data. Bioinformatics 31, 166–169. https://doi.org/10.1093/bioinformatics/btu638.

Dobin, A., Davis, C.A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., and Gingeras, T.R. (2013). STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29, 15–21. https://doi.org/10.1093/bioinformatics/bts635.

Dobin, A., and Gingeras, T.R. (2015). Mapping RNA-seq Reads with STAR. Curr Protoc Bioinformatics 51. https://doi.org/10.1002/0471118455.bi1114s51.

Ernst, J., and Kellis, M. (2017). Chromatin-state discovery and genome annotation with ChromHMM. Nat. Protoc. 12, 2478–2492. https://doi.org/10.1038/nprot.2017.124.

Heinz, S., Benner, C., Spann, N., Bertolino, E., Lin, Y.C., Laslo, P., Cheng, J.X., Murre, C., Singh, H., and Glass, C.K. (2010). Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. Mol. Cell 38, 576–589. https://doi.org/10.1016/j.molcel.2010.05.004.

Li, H., and Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25, 1754–1760. https://doi.org/10.1093/bioinformatics/btp324.

Li, H., Handaker, B., Wysock, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and Durbin, R. (2010). 1000 Genome Project Data Processing Subgroup (2009). The sequence alignment/map format and SAMtools. Bioinformatics 25, 2078–2079. https://doi.org/10.1093/bioinformatics/btp352.

Marchal, C., Sasaki, T., Vera, D., Wilson, K., Sima, J., Rivera-Mulia, J.C., Trevilla-García, C., Nogues, C., Nafie, E., and Gilbert, D.M. (2018). Genome-wide analysis of replication timing by next-generation sequencing with E/L Repli-seq. Nat. Protoc. 13, 819–839. https://doi.org/10.1038/nprot.2017.148.

R Core Team. (2020). R: A language and environment for statistical computing (Vienna, Austria: R Foundation for Statistical Computing).

Ramirez, F., Ryan, D.P., Grünig, B., Bhardwaj, V., Kilpert, F., Richter, A.S., Heyne, S., Dundar, F., and Manke, T. (2016). deepTools2: a next generation web server for deep-sequencing data analysis. Nucleic Acids Res. 44, W160–W165. https://doi.org/10.1093/nar/gkw257.

Van Rechem, C., Ji, F., Chakraborty, D., Black, J.C., Sadreyev, R.I., and Whetstine, J.R. (2021). Collective regulation of chromatin modifications predicts replication timing during cell cycle. Cell Rep. 37, 109799. https://doi.org/10.1016/j.celrep.2021.109799.

Reverón-Gómez, N., González-Aguilera, C., Stewart-Morgan, K.R., Petryk, N., Flury, V., Graziano, S., Johansen, J.V., Jakobsen, J.S., Alabert, C., and Groth, A. (2018). Accurate recycling of parental histones reproduces the histone modification landscape during DNA replication. Mol. Cell 72, 239–249.e5. https://doi.org/10.1016/j.molcel.2018.08.010.

Rivera-Mulia, J.C., Buckley, Q., Sasaki, T., Zimmerman, J., Didier, R.A., Naxor, K., Loring, J.F., Liu, Z., Weissman, S., Robins, A.J., et al. (2015). Dynamic changes in replication timing and gene expression during lineage specification of human pluripotent stem cells. Genome Res. 25, 1091–1103. https://doi.org/10.1101/gr.187989.114.

Robinson, M.D., McCarthy, D.J., and Smyth, G.K. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26, 139–140. https://doi.org/10.1093/bioinformatics/btp616.

Sarni, D., Sasaki, T., Iony Tur-Sinaï, M., Miron, K., Rivera-Mulia, J.C., Magnuson, B., Ljungman, M., Gilbert, D.M., and Kerem, B. (2020). 3D genome organization contributes to genome instability at fragile sites. Nat. Commun. 11, 3613. https://doi.org/10.1038/s41467-020-17448-2.

Stewart-Morgan, K.R., Petryk, N., and Groth, A. (2020). Chromatin replication and epigenetic cell memory. Nat. Cell Biol. 22, 361–371. https://doi.org/10.1038/s41467-020-0487-y.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L.D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., et al. (2019). Welcome to the tidyverse. J. Open Source Softw. 4, 1686. https://doi.org/10.21105/joss.01686.

Zhao, P.A., Sasaki, T., and Gilbert, D.M. (2020). High-resolution Repli-Seq defines the temporal choreography of initiation, elongation and termination of replication in mammalian cells. Genome Biol. 21, 76. https://doi.org/10.1186/s13059-020-01983-8.