Supplemental Materials

Molecular Biology of the Cell

Lawrimore et al.
Supplementary Materials and Methods

Strain Information and Growth Conditions

Prior to timelapse imaging, KBY8088 (MATa CEN(3)(1.1)-GFP[1.7kb] ade1 met14 ura3-52 leu2-3, 112 his3-11,15 lys2Δ::lac-GFP-NLS-NAT 1.1 Kb-CEN3::lacO-KAN Spc29-RFP::Hyg) and KBY8065(MATa CEN(15)(1.8)-GFP[10kb] ade2-1, his3-11, trp1-1, ura3-1, leu2-3,112, can1-100, LacINLSGFP:HIS3, lacO::URA3, Spc29-RFP:Hyg) were grown in YPD at 24°C to log phase.

Timelapse Microscopy

Cells containing a 1.7 kb lacO/LacI-GFP array 1.1 kb from CEN3 (KBY8088) or a 10 kb lacO/LacI-GFP array 1.8 kb from CEN15 (KBY8065) with spindle pole bodies labeled with Spc29-RFP were imaged in in YC-complete media containing 2% glucose for 180 seconds every 3 seconds. Images were acquired using a Nikon Eclipse Ti wide-field inverted microscope with a 100x Apo TIRF 1.49 NA objective (Nikon, Melville, New York, USA) and Andor Clara CCD camera (Andor, South Windsor, Connecticut, USA) using Nikon NIS Elements imaging software (Nikon, Melville, New York, USA) at room temperature (25°C). At each interval a single image plane was taken for GFP, 600 ms exposure, and RFP, 600 ms exposure.

Mean Squared Displacement Analysis

LacO/LacI-GFP and Spc29-RFP foci from timelapse images were tracked using the Speckle Tracker program in MATLAB (The Mathworks; (Wan et al., 2009; Wan et al., 2012). Nuclear/cell motion and microscope drift was removed from the motion of the LacO/LacI-GFP array by subtracting the coordinates of the Spc29-RFP foci from the lacO/LacI-GFP coordinates. The mean squared displacement of the Spc29-RFP subtracted lacO/LacI-GFP coordinates was calculated for timesteps of 3 to 99 seconds from 180 second timelapse images.

Table S1. Diameter of cohesin rings after perturbation by thermal forces.

| Cohesin (n = 3) | 44.6 | 0.1 |
|----------------|------|-----|
| Cohesin encompassing one strand (n = 10) | 44.9 | 0.7 |
| Cohesin encompassing four strands (n = 48) | 47.0 | 0.2 |

The ensemble mean diameter was calculated for individual cohesin rings for the final 2000 timepoints (0.02 seconds of simulation time) of each simulation (see Methods).
Table S2. Comparison of simulated cross-correlation values with published experimental values

| Condition                  | Simulated Values | Published Values | Simulated Values | Published Values |
|----------------------------|------------------|------------------|------------------|------------------|
| With Cohesin, With Condensin | 0.34 ±0.21       | 0.33 ±0.34       | 0.058 ±0.23      | 0.34 ±0.33       |
| No Cohesin, With Condensin  | -0.045 ±0.31     | 0.36 ±0.31       | -0.0028 ±0.32    | 0.35 ±0.36       |
| With Cohesin, No Condensin  | 0.040 ±0.35      | 0.21 ±0.35*      | -0.0043 ±0.37    | 0.31 ±0.31       |

Values are mean cross-correlation values ± standard deviation. Published values are from (Stephens et al., 2013). Simulated values are of cohesin-linked neighbors from Table 3.

Table S3. The average and standard deviation of the axial and radial displacement of the tips of the radial sub-loops.

| Condition            | Average (nm) | Std (nm) | Average (nm) | Std (nm) |
|----------------------|--------------|----------|--------------|----------|
| Tips WT              | 167          | 81       | 193          | 61       |
| Tips No Cohesin      | 158          | 84       | 287          | 81       |
| Tips No Condensin    | 166          | 82       | 227          | 91       |
| Tips No Cohesin No Condensin | 196  | 89       | 288          | 112      |

The first 0.05 seconds of each simulation was discarded to allow time for each centromere model to reach steady state. Each simulation was run for at least 0.1 seconds. Only the tips (most radial bead) of the outermost loops (loops closest to either end) were measured so results could be compared to the simulated 6.8 kb array. SPB = spindle pole body.

Table S4. The average and standard deviation of the axial and radial displacement of the bases of the radial sub-loops.

| Condition               | Average (nm) | Std (nm) | Average (nm) | Std (nm) |
|-------------------------|--------------|----------|--------------|----------|
| Base WT                 | 98.4         | 15       | 395          | 3.2      |
| Base No Cohesin         | 99.5         | 14       | 395          | 3.3      |
| Base No Condensin       | 198          | 88       | 763          | 88       |
| Base No Cohesin No Condensin | 246  | 117      | 833          | 116      |

The first 0.05 seconds of each simulation was discarded to allow time for each centromere model to reach steady state. Each simulation was run for at least 0.1 seconds. Only the bases (least radially displaced bead) of the outermost loops (loops closest to either end) were measured so results could be compared to the simulated 6.8 kb array. SPB = spindle pole body.
Table S5. Radius of gyration and average end to end length.

|                          | Expected | Measured | Expected | Measured | Expected | Measured |
|--------------------------|----------|----------|----------|----------|----------|----------|
| Radius of Gyration (nm)  | 129      | 95 ± 3   | 183      | 148 ± 16 | 224      | 183 ± 18 |
| End-to-end length (nm)   | 316      | 221 ± 6  | 447      | 349 ± 51 | 548      | 454 ± 107 |

Radii of gyration and end to end lengths are \( \langle R_g \rangle = \sqrt{\frac{N\sigma^2}{6}} \) and \( \langle R \rangle = \sqrt{N\sigma^2} \) respectively, where \( b \) is Kuhn length and \( N \) is the number of segments (Rubinstein and Colby, 2003) All values are averages of three independent simulation runs shown with standard deviation between the runs.
**Figure S1.** Mean square displacement curves for 1.1 kb and 10 kb LacO arrays. Curves are the average of individual mean displacement curves. 1.7 kb lacO array (n= 12 cells) or a 10 kb array (n = 14 cells). Error bars are standard deviation.
**Figure S2.** End to end length of a 1 μm chain (A), 2 μm chain (B), and 3 μm chain (C). Dotted black line is the expected end to end length from the equation $\langle R \rangle = \sqrt{N b^2}$, where $b$ is Kuhn length and $N$ is the number of segments (Rubinstein and Colby, 2003). Rouse time is the time for a linear polymer to collapse into a random coil calculated by, $\tau_R = \frac{\eta b^3}{k_B T N}$ where $\eta$ is viscosity, $b$ is Kuhn length, $k_B$ the Boltzmann constant, $T$ temperature, and $N$ the number of Kuhn length segments in the chain (Rouse, 1953; Rubinstein and Colby, 2003).
Figure S3. Illustration of hinge forces. (A) Beam simply supported at both ends. (B) Simply supported beam with center load, $F$. The equations indicate the relationship between the load force and the tangent angle at the beam ends. (C) Beam elastic restoring forces opposing external forces. (D) Translation of continuum model parameters to discretized simulation model with the hinge forces within a simulated chain.
Figure S4. (A) Mean squared displacement curves of diffusing spherical masses. Five separate simulations of a single spherical mass, each with a different random number seed were allowed to diffuse for 0.9 seconds of simulation time. (B) Log_{10}/Log_{10} plots of the mean squared displacement curves in A. The slope of 1 is provided to model normal diffusion (Saxton, 1994; Rubinstein and Colby, 2003). (C) Distribution of mean squared displacement of 1 million beads with a 9 ns time step allowed to step 10 times. Expected displacement is 11.8 nanometers squared based on $\langle \Delta x^2 \rangle = \frac{k_B T}{6 \pi n \eta R} \Delta t$, where $k_B$ is the Boltzmann constant, $T$ is temperature, 298.15 K, $\eta$ is viscosity, 0.001 Pa*s, $R$ is the radius of the spherical bead, 10 nm, and $\Delta t$ is the change in time, 90 ns (time step times step number).
Supplemental References

Rouse, P.E. (1953). A Theory of the Linear Viscoelastic Properties of Dilute Solutions of Coiling Polymers. Journal of Chemical Physics 21, 1272-1280.
Rubinstein, M., and Colby, R.H. (2003). Polymer physics. Oxford University Press: Oxford ; New York.
Saxton, M.J. (1994). Anomalous diffusion due to obstacles: a Monte Carlo study. Biophys J 66, 394-401.
Stephens, A.D., Snider, C.E., Haase, J., Haggerty, R.A., Vasquez, P.A., Forest, M.G., and Bloom, K. (2013). Individual pericentromeres display coordinated motion and stretching in the yeast spindle. J Cell Biol 203, 407-416.
Wan, X., Cimini, D., Cameron, L.A., and Salmon, E.D. (2012). The coupling between sister kinetochore directional instability and oscillations in centromere stretch in metaphase PtK1 cells. Mol Biol Cell 23, 1035-1046.
Wan, X., O’Quinn, R.P., Pierce, H.L., Joglekar, A.P., Gall, W.E., DeLuca, J.G., Carroll, C.W., Liu, S.T., Yen, T.J., McEwen, B.F., Stukenberg, P.T., Desai, A., and Salmon, E.D. (2009). Protein architecture of the human kinetochore microtubule attachment site. Cell 137, 672-684.

ChromoShake Guide

This document will guide the user through installing and running ChromoShake and the rendering and analysis programs. Throughout this guide, we will be using a 1 micron polymer chain as an example. This guide will only cover Windows based systems, although ChromoShake’s source code, which has been provided, has been compiled and run on UNIX based systems.

ChromoShake Installation Using Microsoft Installer

- Download chromoShake_1_2_0.msi installer.
- Run the installer and follow the instructions.

Create a 1 µm chain

- Open the Window’s Command Prompt, cmd.exe, as an administrator. Administrator privileges are needed to write out files.
- Navigate to chromoShake_1_2.0 directory containing the program using cd. By default the location of chromoShake_make_linear_chain.exe is C:\Program Files\CISMM.org\chromoShake_1.2.0\.
  - **Type:** cd “C:\Program Files\CISMM.org\chromoShake_1.2.0”
  - **NOTE:** Include the double quotations for account for the space.
- Run chromoShake_make_linear_chain.exe and save output to a file named default_chain.cfg
  - **Type:** chromoShake_make_linear_chain.exe > default_chain.cfg
- Open the .cfg file using a text editor such as Notepad or WordPad to see the default conditions and locations of the masses.

Run ChromoShake
- Run chromoShake.exe on the default_chain.cfg file, outputting coordinates every 10 microseconds (2 nanosecond timestep per calculation * 5000 calculations between outputs), for 1 millisecond (100 outputs * 10 microsecond per output). Save output to default_chain.out file.
  - Type: chromoShake.exe -openCL_dir openCL -savedefault_chain.out 5000 100 default_chain.cfg
  - The –openCL flag tag directs ChromoShake to the openCL files installed with the MSI installer. The default location is C:\Program Files\CISMM.org\chromoShake_1.2.0\openCL

**NOTE:** ChromoShake is not compatible with all versions of the OpenCL compiler. A common error is:

Constructing simulation
--- Build log ---
C:\cygwin64\tmp\OCLD906.tmp.cl(54): error: identifier "M_PI" is undefined
  CS_REAL angle = fabs( M_PI - acos( dot( left, right ) ) );

^  
C:\cygwin64\tmp\OCLD906.tmp.cl(59): warning: double-precision constant is represented as single-precision constant because double is not enabled
  if ( any( isnan( f ) ) || angle < 0.001 ) {

^  
1 error detected in the compilation of "C:\cygwin64\tmp\OCLD906.tmp.cl".

Internal error: clc compiler invocation failed.
ERROR: clBuildProgram in Force Hinge (Program build failure)

Update drivers to latest version to test for GPU compatibility. Otherwise specify CPU usage by typing

cromoShake.exe –CPU –save default_chain.out 5000 100 default_chain.cfg

View 1 µm chain using ChromoView

- Open chomoView.exe. Unlike ChromoShake, ChromoView has a graphical interface. By default ChromoView is located at C:\Program Files\CISMM.org\chromoShake_1.2.0\
- Click “Load Simulation Output File”
- Select default_chain.out file.
- **NOTE:** chomoView.exe reads in all data before rendering, so larger files (i.e.> 3 Gigabytes) may not load due to lack of memory. You can view simulations that are currently running.
- Rendering with Blender
  - A bash script, a python script, and a blender file are provided to parse .out files in to Blender. Blender is an open source program 3D graphics and animation software. Blender can be downloaded from [https://www.blender.org/download/](https://www.blender.org/download/). The python script has been tested with Blender version 2.68a. The script outputs an untitled.blend file that can be loaded into blender and then rendered.
  - **NOTE:** The bash script is not fully compatible with Cygwin. The bash script will successfully make the files needed, but Blender will fail to locate the files based on differences in directory structure.
  - To create a .blend file using the Windows and the Command Prompt
Copy and rename the vidprecode4_batch.blend file to default_chain.blend in the same directory as default_chain.out
Copy and rename the read_chromoShake_file_into_blender.py to default_chain.py in same directory as default_chain.out
Open default_chain.py with a text editor or IDLE. Change ‘INPUT_FILE.txt’ in line 5 to ‘default_chain.out’
Open Command Prompt
Change directory to directory containing default_chain.blend, default_chain.out, and default_chain.py
Assuming that blender is in this default directory, type: “C:\Program Files (x86)\Blender Foundation\Blender\blender.exe” -b default_chain.blend -P default_chain.py -noaudio
This will output an untitled.blend file. Open this file in Blender to render the file.
NOTE: The default camera angle for linear chains is end on so first rendered image will look like a single bead.

Analysis Programs and Usage

Analysis programs can be run on UNIX based systems or on windows with Cygwin (https://www.cygwin.com/) installed. A C++ compiler is needed. Existing programs require Python (https://www.python.org/), PERL (https://www.perl.org/get.html) and BASH script (.sh) support.

- Mean Squared Displacement Analysis. This program calculates the mean square displacement of a single bead or the mean position of an array of beads. The program requires the .out file have the header removed and the bead coordinates be changed from meters to microns.
  - BASH scripts
    - These scripts link together a set of PERL programs and C++ programs to produce “colorfiles” (known as SRCs) that indicate which beads are measured by mean squared displacement.
    - The bead position files used to generate SRCs that were used in this paper are kept in the folder beadPositionFiles.
    - Each bead position file has a rows of numbers that correspond to bead numbers (bead 0, 1, 2, 3, etc.). The mean 3D position of each of the beads in a row will be calculated by coord_summary.cpp. Each row will then become a single file of the average 3D position of all the beads over time. The MSD is calculated from these average 3D positions.
    - The bead position files can be made by the updated chromoShake_make_spindle.cpp program (not included in MSI installer but in the source code).
    - An example bash script, updated_MSD_example_summary.sh, has been provided.
    - The .out files must be placed in the OUTFILE directory. These data files can also be trimmed to a particular timepoint to allow for equilibration. The OUTFILE directory currently contains truncated (0.05 to 0.055 seconds) centromere models as examples.
      - WT.out = with cohesin and condensin

10
• no_coh.out = no cohesin, with condensin
• no_cond.out = with cohesin, no condensin
• no_coh_no_cond.out = neither cohesin nor condensin

- Compile the coord_summary.cpp to coord_summary.exe (for Windows) or coord_summary (for UNIX).
  - NOTE: if on a UNIX system change line 8 in MSD_summary.sh from coord_summary.exe to coord_summary

- To run scripts: sh updated_MSD_example_summary.sh beadPositionFiles\10kb_loop_default

- Several directories will be created. SRCs contain the SRC files. Coord_summs contain the coordinate summary containing the average 3D positions. MSDs will contain the MSD files plus a summary MSD file with all of the MSDs over time of each file concatenated horizontally.

- ChromoMap.m MATLAB
  - Creates a heatmap of the array position over simulated time. Reads in the coord_summ directory from the MSD analysis script. To only sample “in-focus’ arrays so data can be directly compared to imaging data only the center 200 nm are sampled in the Z direction, so minimum Z is -100 and maximum Z is 100.
  - NOTE – For this program, the axis have been rotated to match our typical imaging axis. Z is orthogonal to the XY plane where the X is parallel to the spindle axis and Y is perpendicular.

- CorrMotion.m MATLAB
  - Calculates the correlated motion of neighboring chromatin loops and sisters.
    Neighboring loops cross-linked by cohesin are “Cohesion Neighbors,” while neighboring loops that are not cross-linked are “Unlinked Neighbors.” The program inputs the coord_summ files of the 10 kb loop (default) spindle model. The axes were not rotated in this program so the Z dimension is parallel to the spindle axis.

- Radius of gyration MATLAB
  - The function radiusOfGyration_color1.m calculates the radius of gyration of all the beads labeled with the color 1 (red in ChromoView). This script was used to measure the radius of gyration of the primary axis of a bottlebrush polymer simulation and the radius of gyration of a ring of chromatin with cohesin. The input is a .out file.
  - The function radiusOfGyration_cohesinRings.m calculates the average radius of gyration over time of cohesin rings labeled with the color 5 (white in ChromoView). The input is a .out file.

The following analysis programs were developed for a prior version of ChromoShake. The .out files need to be converted using the following PERL scripts to be compatible with these analysis programs.

- Header Removal
  - All data before the line “Time 0” must be deleted. This can be done using various text editors, but can be problematic for larger files. We have provided a PERL script to remove the header named ChromoShakeRemoveHeader.pl using standard input and output.
To remove header using ChromoShakeRemoveHeader.pl ensure that PERL is installed on your computer.

- **Type in Command Prompt:** perl ChromoShakeRemoveHeader.pl < default_chain.out > default_chain_noHead.txt

  - **Unit Conversion**
    - We have provided a PERL script to convert coordinates from meters to microns.
    - **Type in Command Prompt:** perl ChromoShakeUnitConvert.pl < default_chain_noHead.txt > default_chain_noHead_um.txt

- **Radial Displacement Analysis**
  - The C++ program, cohesin_summary.cpp, parses the bead coordinate files (lacking header, beads position in microns) based on a color file. The color file contains a list of integers that indicates the color of the bead. This program parses the coordinates of the beads corresponding with the color 4 (white, originally indicated cohesin).
  - Two SRC files have been provided in Radial_analysis directory
    - default_chain_SRC.txt – labels all beads in a default chain
    - someRandomColor.txt – labels cohesin in default centromere model.
  - **To use, type in command prompt:** cohesin_summary.exe default_chain_SRC.txt default_chain_noHead_um.txt

- **brownianMotiontoFluoroSim.py**
  - Python script that converts to the old format of ChromoShake’s output (no header, coordinates in microns) to XML files that Microscope Simulator 2 can parse to generate simulated fluorescent images. A detailed README.html file is included and contains usage instructions.
    - Microscope Simulator 2 can be downloaded from CISMM’s website
      - [http://cismm.cs.unc.edu/downloads/](http://cismm.cs.unc.edu/downloads/)
    - In the README file $point_colors refers to the colorfiles. The same files used in the Radial Displacement Analysis section can also be used to indicate which beads should be fluorescent. The number 4 labels a bead for fluorescence.
    - In the README file $coordinates refers to the .out files after the header has been removed and the coordinate units have been converted from meters to microns.
    - The XML files generated by ParseBrownian.py can be open directly using Microscope Simulator 2. The default view is too large in Microscope Simulator 2 for chromoShake models, so you must zoom into the center of the grid to view the models.