Single-molecule mapping of replisome progression.

Clémence Claussin, Jacob Vazquez and *Iestyn Whitehouse

Supplemental Information

Script S1: Tornado plot script. To generate all the tornado plots in Figure 1, 3, 4 and 5. Also related to the Figure S2, S3, S4, and S7.

Script S2: rDNA heatmap script. To generate all the rDNA heatmap in Figure 2.

Supplementary figure legends

Figure S1: Mcm4-Mnase tag does not affect cell growth. **A.** Drop test showing no growth defect of MCM4-MNase fusions. **B.** Flow cytometry data from one of the replicon-seq sequencing from WT and *rrm3*Δ mutant. BrdU is added at 2h15 during α-factor arrest for 30min and cells are released in S phase in presence of BrdU **C.** The Mcm4-Mnase tag cleaves origin DNA in presence of calcium in G1. DNA was digested with *EcoRI* and probed with radiolabeled DNA binding to ARS305. In absence of cleavage by Mcm4-MNase 6kb band is observed, upon calcium addition Mcm4-MNase specifically cleave DNA at ARS releasing a 3kb band. **D.** Frequency plot of WT reads length, excluding rDNA. Blue bars show all BrdU containing reads, n=$1.59.10^5$ (Methods). Orange bars show reads selected to overlap origins of replication (n=$8.9.10^4$). **E.** Venn diagram comparing Replicon-seq early S-phase replication origin calls (Methods) to known origins of replication. Total origins called: replicon-seq n=176, total OriDB (Confirmed) origins n= 401, late OriDB n=76, early OriDB n=72, likely n=214. Related to Figure 1.

Figure S2: Tornado plots of WT Replicates. Related to Figure 1.

Figure S3: Tornado plots of the budding yeast chromosomes 1 to 8 for WT cells. Related to Figure 1 and to compare to Figure S6, S7 and Figure 3.

Figure S4: Tornado plots of the budding yeast chromosomes 9 to 16 for WT cells. Related to Figure 1 and to compare to Figure S6, S7 and Figure 3.

Figure S5: Schematic representation of MNase cleaving ssDNA at replication fork and library preparation for Replicon-seq. Cleavage of ssDNA at the replication fork should result in preferential digestion of the unwound parental DNA. Related to Figure 1A.

Figure S6: Tornado plots of the budding yeast chromosomes 1 to 8 for *rrm3*Δ mutants. Black arrows indicate prominent sites of termination failure. Related to Figure 4. Compare with Figure S3 and S4.

Figure S7: Tornado plots of the budding yeast chromosomes 9 to 16 for *rrm3*Δ mutants. Black arrows indicate prominent sites of termination failure. Related to Figure 4. Compare with Figure S3 and S4.

Figure S8: Replication fork pausing sites in *rrm3*Δ mutant. **A.** Median position of read ends in 250bp intervals was calculated for different length replicons emanating from the replication origin (from Figure 3A). The relative positions of the left and right replisomes are plotted at increasing distance from the origin. Green track represents the tRNA position. Red line shows the expected

trajectory if both replicons are moving at the same rate. **B.** Same as A, but depicting data from Figure 3B. Red track represents the centromere position. Red line shows the expected trajectory if both replicons are moving at the same rate **C.** Diagram representing all the pausing site called from replicon-seq sequencing in *rrm3*Δ mutant in early S-phase. **D.** Meta-plot showing the distance of the 3' of the rightward moving fork leading strand (black) or the leftward moving fork of the leading strand 5' end (red) near the transcription site of 253 tRNA genes in *rrm3*Δ. Green dotted line shows the upstream edge of TFIIIB. **E.** Same as C, and Figure 4E for pausing at centromeres. Related to Figure 4.

Figure S9: Converging replisomes do not significantly slow at termination site. Relative progression of sister replisomes is plotted for Left origin (ARS510**)** and Right origin (ARS511). The Right fork of the ARS510 and the Left fork of ARS511 converge to terminate approximately 13kb from each origin. The median position of replisomes in 250bp intervals from the replication origin is calculated. Related to Figure 5.

Figure S10: Replication of the rDNA locus in *rrm3*Δ cells. **A.** 5' and 3' DNA ends are plotted for of either Bottom or Top-strand reads at the rDNA locus. Upper tracks show the genome coverage of the 3' or 5' ends of reads. **B.** Replication fork pausing at RFB site is not altered in *rrm3*Δ mutant**.** Counts of 5' and 3'ends of the lagging or leading strands of replisomes approaching the RFB from the rightward moving fork. Upper graphs show 2-4kb reads, lower show 6-8kb reads. Prominent rDNA features are displayed in the lower track. Compare to Figure **2 C, D, E** and **F.** Related to Figure 4 and 5.

Figure S11: Gene block sequence for N-terminal MNase tag of MCM4. The MNase-sequence is in Bold letters and the linker in purple. Related to Start Method section and Figure S1 for generation of the replicon-seq strain.

Figure S12: Nascent RNA sequencing. Correlation of two biological replicates of nascent RNA seq data in early S-phase. Related to Figure 3.

Script S1

```
import pandas as pd
from matplotlib import patches
from matplotlib import pyplot as plt
from matplotlib.collections import BrokenBarHCollection
import sys



# input files
gff_track = "input.gff"
bed_file = "input.bed"


# GFF Track Format:
# chromosome        source  feature    start    stop    score
orientation    frame   description
# chr4                        EMBL  CDS          1858    1890    .       +
               0      Parent=chr4.chr4.final.6

# Bed Format:
# chromosome    start    stop    readID    brdU    orientation
# chr4          1920     3933    >read101  0.2     +


# input variables
chromosome = input("Enter the chromosome: ")
bed_column = "strand"
track1_feature_type = "gene"
track2_feature_type = "nascent_gene"
# To view all: Comment out line 44 or line that says this: track =
track[track["feature"] == feature_type]

# All features have to be written and assigned a color.
color_lookup = {
    '+': "blue",
    '-': "red"
}



# read bed file with read fragment information
genome = pd.read_table(bed_file, header=None, comment='#')
genome = genome[[0, 1, 2]]
genome = genome.rename(columns={0: "chr", 1: "start", 2: "stop"})
genome = genome[genome["chr"] == chromosome]
genome["length"] = genome["stop"] - genome["start"]
x = genome["stop"].max()
y = genome["length"].max()
lst = genome.values.tolist()

# read bed file with genome track information
track1 = pd.read_table(gff_track, header=None, comment='#')
```

```
track1 = track1.dropna(subset=[8])
track1 = track1[[0, 2, 3, 4, 6, 8]]
track1 = track1.rename(columns={0: "chr", 2: "feature", 3: "start", 4:
"stop", 6: "strand", 8: "description"})
track1 = track1[track1["chr"].str.contains(chromosome)]
track1["colors"] = track1[bed_column].apply(lambda x: color_lookup[x])
track2 = track1
track1 = track1[track1["feature"] == track1_feature_type]

track2 = track2[track2["feature"] == track2_feature_type]

# initiate plots
fig, axs = plt.subplots(3, gridspec_kw={'height_ratios': [30, 1, 1]},
sharex=True, sharey=False)

def features(df):
    df['width'] = df['stop'] - df['start']
    for chrom, group in df.groupby('chr'):
        yrange = (0, 1)
        xranges = group[['start', 'width']].values
        yield BrokenBarHCollection(xranges, yrange,
facecolors=group["colors"])

for i in range(len(lst)):
        box_size = 300
        #axs[0].add_patch(patches.Rectangle((lst[i][1], lst[i][3]-1/2),
lst[i][3], 1, color="blue", lw = 3, alpha = 0.1))
        #axs[0].add_patch(patches.Ellipse((lst[i][1], lst[i][3]),
box_size, box_size, angle = 225, color='blue', lw = 0.1, alpha = 0.4))
        #axs[0].add_patch(patches.Ellipse((lst[i][2], lst[i][3]),
box_size, box_size, angle = 225, color='purple', lw = 0.1, alpha = 0.4))
        axs[0].add_patch(patches.Rectangle((lst[i][1], lst[i][3]-1/2),
lst[i][3], 1, color='darkblue', lw = 2, alpha = 0.1))
axs[0].axis('tight')
axs[0].set_title(chromosome)
axs[0].set_ylim(0, y)
axs[0].set_ylabel("Length (bp)")

for collection in features(track1):
    axs[1].add_collection(collection)
axs[1].axis('tight')
axs[1].set_xlim(0, x)
axs[1].get_yaxis().set_ticks([])

for collection in features(track2):
    axs[2].add_collection(collection)
axs[2].axis('tight')
axs[2].set_xlim(0, x)
axs[2].set_xlabel("Position (bp)")
axs[2].get_yaxis().set_ticks([])

for key, value in color_lookup.items():
    plt.plot([0], label=key, color=value)
```

```
plt.legend(bbox_to_anchor=(1, 34.5), frameon=False, ncol=2)

plt.subplots_adjust(wspace=0, hspace=0)
plt.show()
```

Script S2

```
#################################################################
# Creates a heatmap of length of read vs position
# Also creates a genome track
#################################################################
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors
from matplotlib.collections import BrokenBarHCollection

bed_file = 'input.bed'
gff_track = 'features.gff'


# GFF Track Format:
# chromosome        source  feature    start    stop    score
orientation    frame   description
# chr4                        EMBL  CDS          1858    1890    .       +
               0       Parent=chr4.chr4.final.6

# Bed Format:
# chromosome    start    stop    readID    brdU    orientation
# chr4          1920     3933    >read101  0.2     +

chr = 'rDNA'  # Chromosome
start = 500  # Enter the start position of the region window.
stop = 8000  # Enter the end position of the region window.

chromosome = "rDNA"
bed_column = "strand"
track1_feature_type = "rDNA"

color_lookup = {
    '+': "orange",
    '-': "red"
}

#################################################################

## read table into pandas
df = pd.read_csv(bed_file, delimiter="\t", names=["chr", "start", "end",
"readID", "brdU", "strand"])

## only select rows in which the start is greater than 500 and the stop
is less than 8kb and the chr is rDNA
df = df[(df['start'] >= start) & (df['end'] <= stop) & (df['chr'] ==
chr)]

## create length column
length = df["end"] - df["start"]
```

```
## Norm start values by subtracting 500 by each row in start column
rel_start = df['start'] #- start  # Normailze Start Values.

## Norm end values by subtracting the start values by each row in end
column
rel_stop = df['end'] #- start  # Normailze End Values.

## create length column
length = df["end"] - df["start"]

## insert Norm start column, norm end column, and length column
df.insert(0, 'Normalized Start', rel_start)
df.insert(1, 'Normalized End', rel_stop)
df.insert(2, 'length', length)

## sort the values by the length
df.sort_values(by=['length'])

## convert the dataframe to a list
lst = df.values.tolist()

## create the matrix length of being 8kb minus 500 (stop-start)
matrix_length = (stop - start) + 500

## create empty matrix with zeros has the shape 7500x7500
matrix = np.zeros((matrix_length, matrix_length))

## for each group of elements in the list, assign variables
for y in range(len(lst)):

    ## the normalized start row
    norm_start = lst[y][0]

    ## the normalized end row
    norm_end = lst[y][1]

    ## convert normalized start to int and normalized end to int
    norm_start = int(norm_start)
    norm_end = int(norm_end)

    ## the chromosome length column
    chr_length = lst[y][2]

    ## create matrix lengths
    for x in range(norm_start, norm_end):
        matrix[chr_length][x] += 1

############################################################

def parula_cmap():
    cmap = plt.cm.viridis  # Set cmap to viridis
    cmaplist = [cmap(i) for i in range(cmap.N)]  # Convert the cmap
values into a list, to be able to edit.
    cmaplist[0] = (0, 0, 0, 0)  # Set the value of 0 to white.
```

```
    cmap = matplotlib.colors.LinearSegmentedColormap.from_list('mcm',
cmaplist, cmap.N)
    return cmap

#############################################################

def features(df):
    df['width'] = df['stop'] - df['start']
    for chrom, group in df.groupby('chr'):
        yrange = (0, 1)
        xranges = group[['start', 'width']].values
        yield BrokenBarHCollection(xranges, yrange,
facecolors=group["colors"])

#############################################################

# read bed file with genome track information
track1 = pd.read_table(gff_track, names=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
10, 11, 12, 13], comment='#')
track1 = track1.dropna(subset=[8])
track1 = track1[[0, 2, 3, 4, 6, 8]]
track1 = track1.rename(columns={0: "chr", 2: "feature", 3: "start", 4:
"stop", 6: "strand", 8: "description"})
track1 = track1[track1["chr"].str.contains(chromosome)]
# track1["start"] = track1["start"] - start
# track1["stop"] = track1["stop"] - start
track1 = track1[track1["start"] >= start]
track1 = track1[track1["stop"] <= stop]
track1["colors"] = track1[bed_column].apply(lambda x: color_lookup[x])

#############################################################

fig, axes = plt.subplots(2, gridspec_kw={'height_ratios': [30, 1]},
sharex=True, sharey=False)
im1 = axes[0].imshow(matrix, cmap=parula_cmap())
axes[0].axis('tight')
axes[0].invert_yaxis()
axes[0].set_xlim(start - 500, stop - 500)
# axes[0].set_xlim(start, stop)
fig.subplots_adjust(right=0.85)
cbar_ax = fig.add_axes([0.88, 0.15, 0.04, 0.7])
fig.colorbar(im1, cax=cbar_ax)
for collection in features(track1):
    axes[1].add_collection(collection)
axes[1].axis('tight')
axes[1].get_yaxis().set_ticks([])
# axes[1].set_xlim(start - 500 , stop - 500)
axes[1].set_xlim(start, stop)
im1.set_clim(0,4)
plt.subplots_adjust(wspace=0, hspace=0)
plt.show()
```

Figure S1

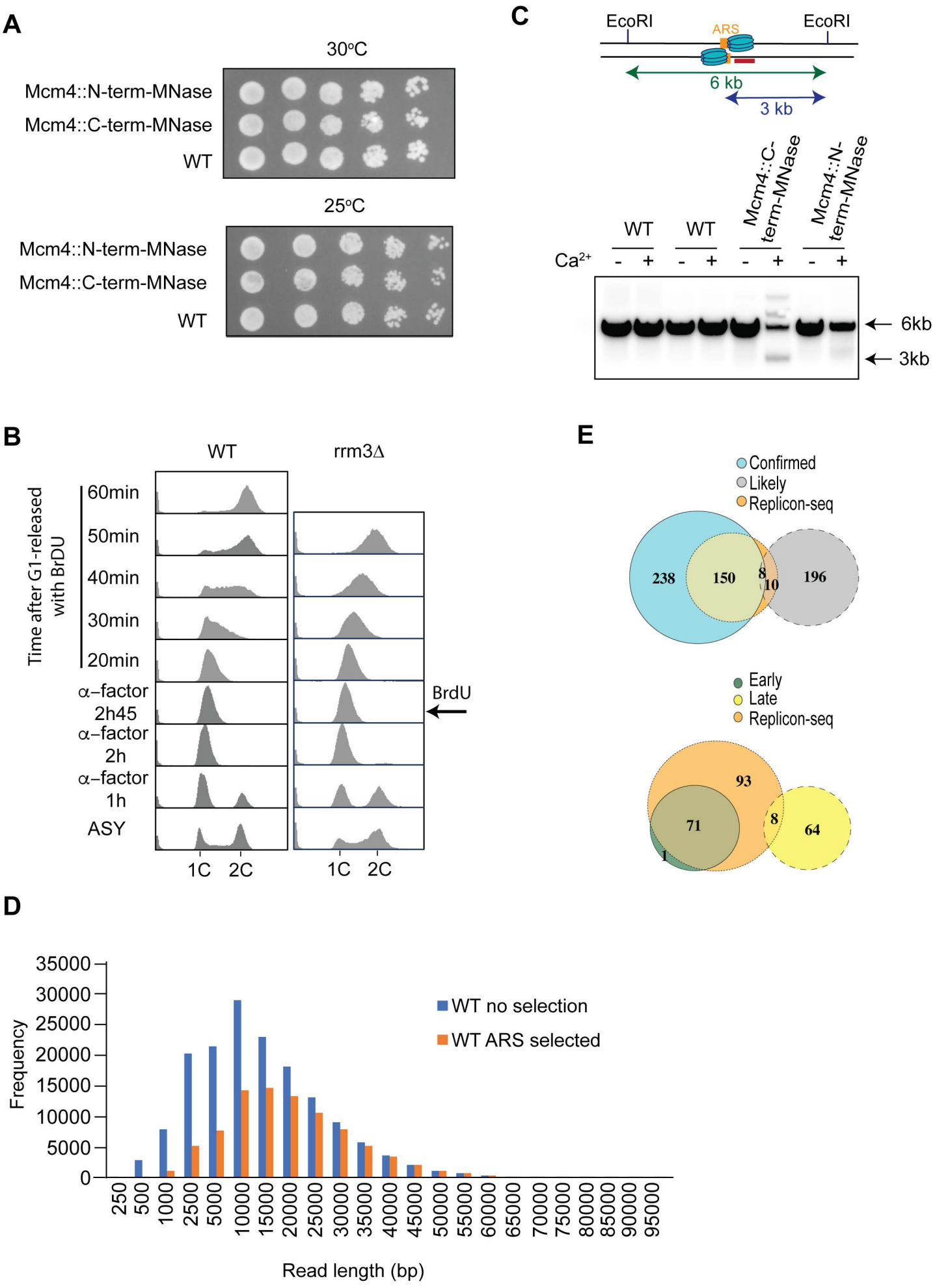
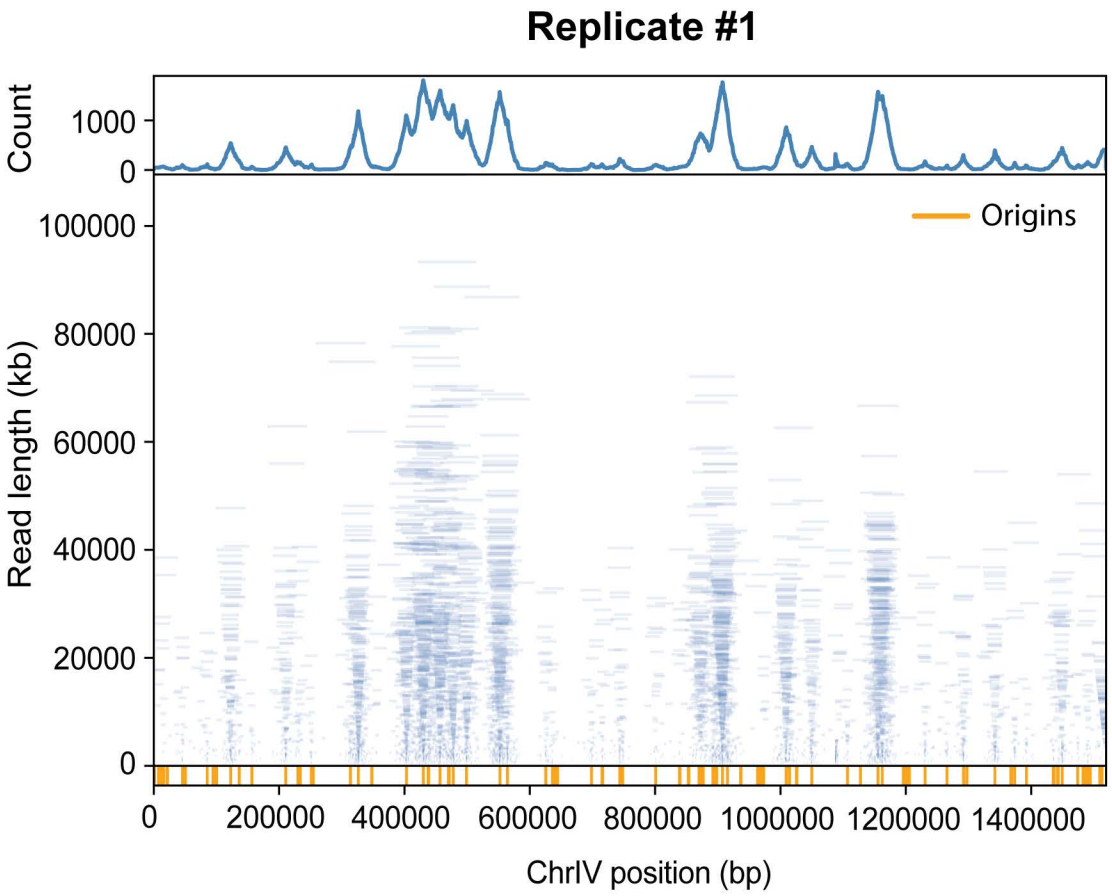

A
30°C
Mcm4::N-term-MNase
Mcm4::C-term-MNase
WT
25°C
Mcm4::N-term-MNase
Mcm4::C-term-MNase
WT
C
EcoRI
ARS
EcoRI
6 kb
3 kb
WT
WT
Mcm4::C-term-MNase
Mcm4::N-term-MNase
Ca2+
- + - + - + - +
6kb
3kb
B
WT
rrm3Δ
Time after G1-released with BrDU
60min
50min
40min
30min
20min
α–factor 2h45
α–factor 2h
α–factor 1h
ASY
BrdU
1C 2C
1C 2C
E
Confirmed
Likely
Replicon-seq
238
150
8
10
196
Early
Late
Replicon-seq
93
71
8
64
1
D
Frequency
35000
30000
25000
20000
15000
10000
5000
0
WT no selection
WT ARS selected
250 500 1000 2500 5000 10000 15000 20000 25000 30000 35000 40000 45000 50000 55000 60000 65000 70000 75000 80000 85000 90000 95000
Read length (bp)

Figure S2


Replicate #1
Count
1000
0
Origins
100000
80000
60000
40000
20000
0
Read length (kb)
0
200000
400000
600000
800000
1000000
1200000
1400000
ChrIV position (bp)

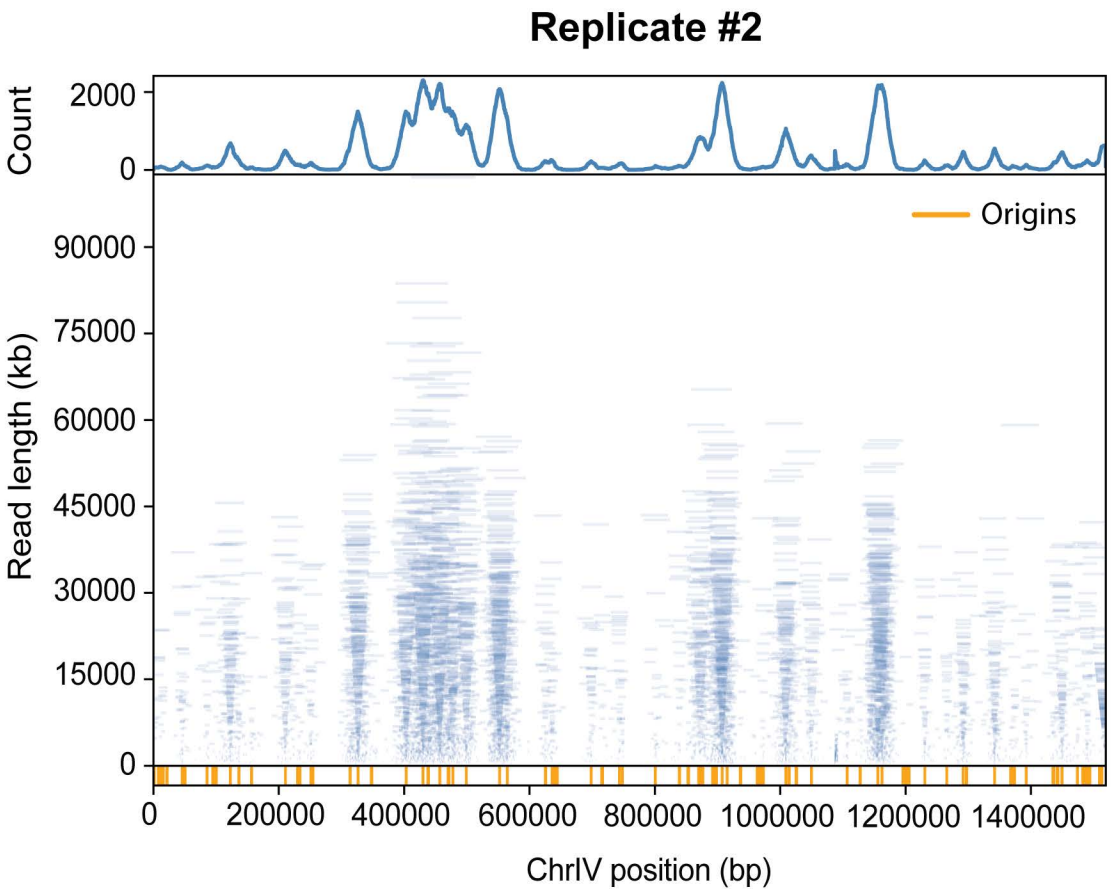
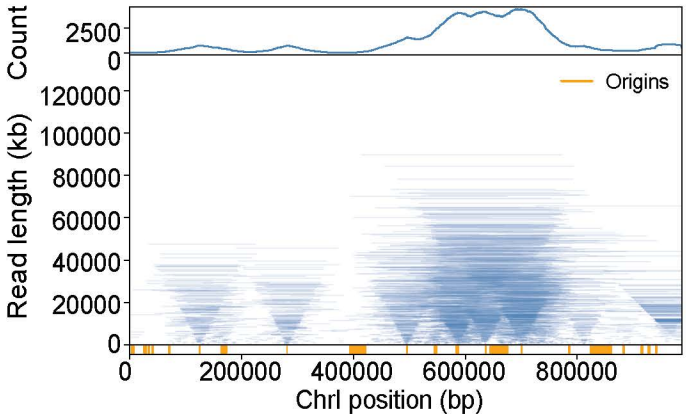
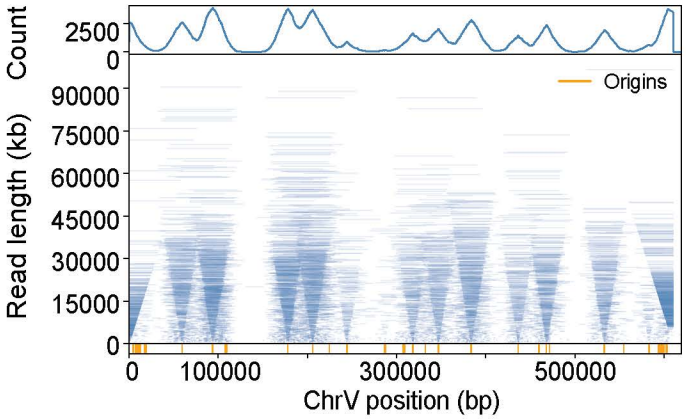
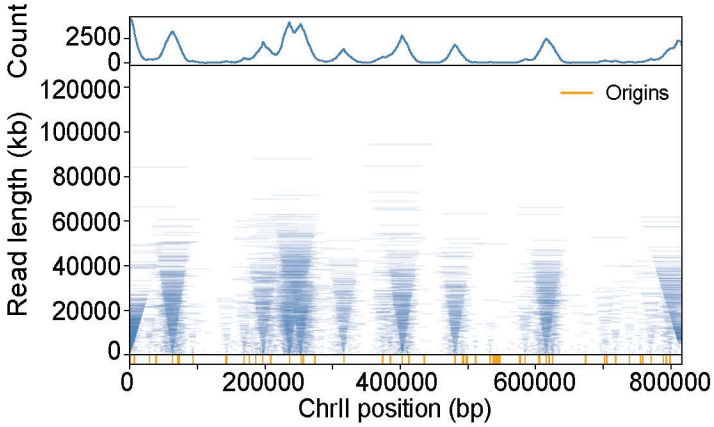
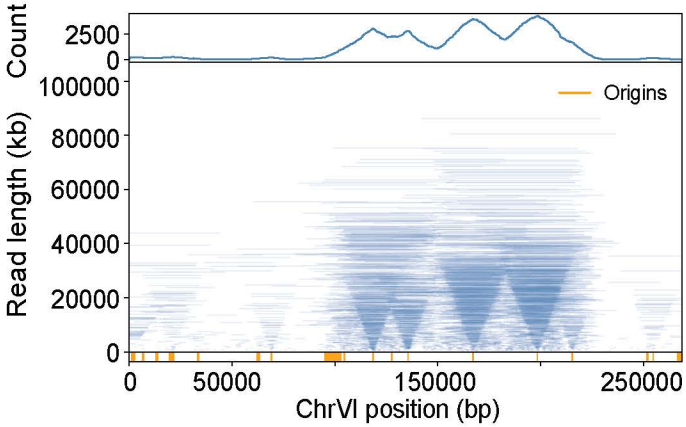

Replicate #2
Count
2000
0
Origins
90000
75000
60000
45000
30000
15000
0
Read length (kb)
0
200000
400000
600000
800000
1000000
1200000
1400000
ChrIV position (bp)

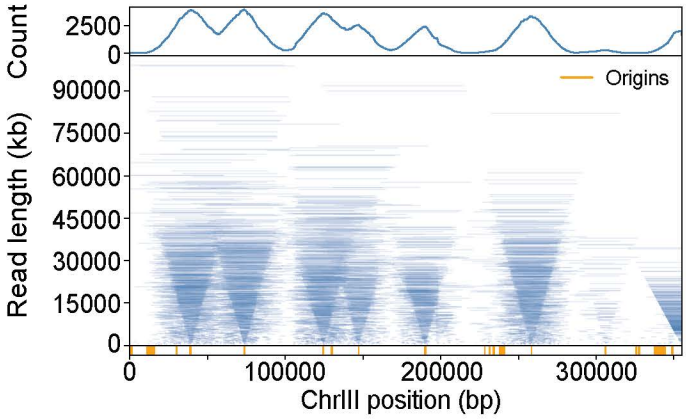
Figure S3

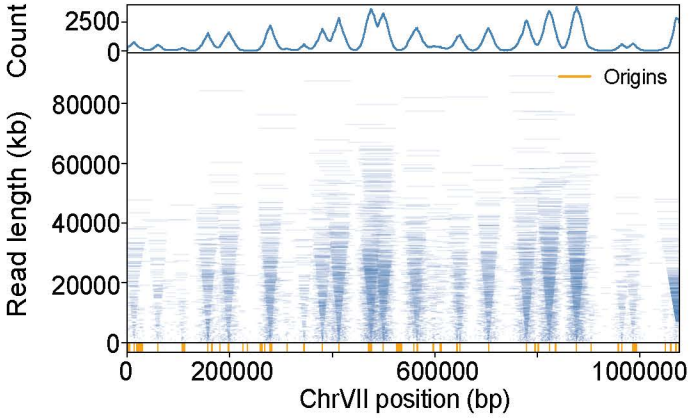

Count
2500
0
Origins
Read length (kb)
120000
100000
80000
60000
40000
20000
0
0
200000
400000
600000
800000
ChrI position (bp)

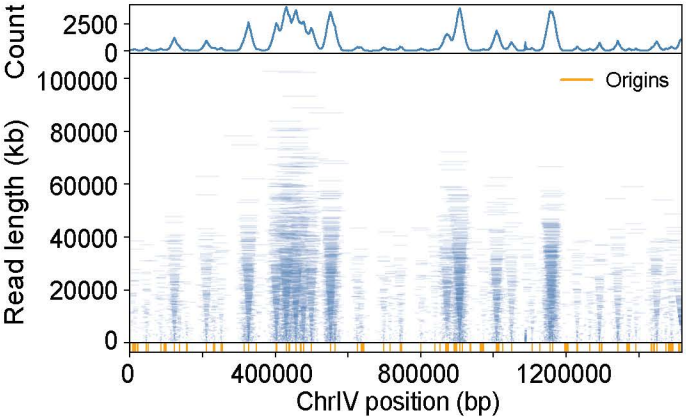

Count
2500
0
Origins
Read length (kb)
90000
75000
60000
45000
30000
15000
0
0
100000
300000
500000
ChrV position (bp)


Count
2500
0
Origins
Read length (kb)
120000
100000
80000
60000
40000
20000
0
0
200000
400000
600000
800000
ChrII position (bp)


Count
2500
0
Origins
Read length (kb)
100000
80000
60000
40000
20000
0
0
50000
150000
250000
ChrVI position (bp)


Count
2500
0
Origins
Read length (kb)
90000
75000
60000
45000
30000
15000
0
0
100000
200000
300000
ChrIII position (bp)


Count
2500
0
Origins
Read length (kb)
80000
60000
40000
20000
0
0
200000
600000
1000000
ChrVII position (bp)


Count
2500
0
Origins
Read length (kb)
100000
80000
60000
40000
20000
0
0
400000
800000
1200000
ChrIV position (bp)

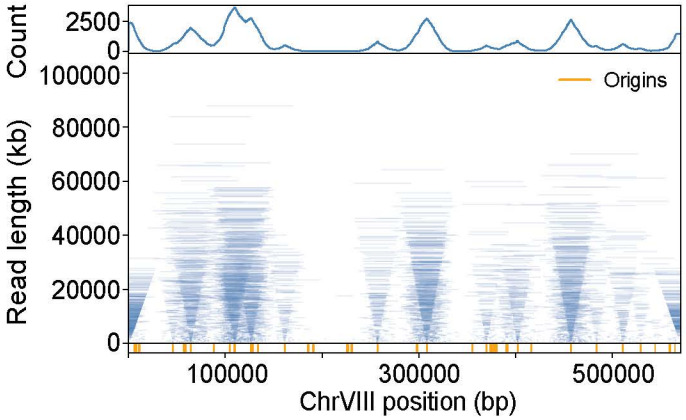

Count
2500
0
Origins
Read length (kb)
100000
80000
60000
40000
20000
0
100000
300000
500000
ChrVIII position (bp)

Figure S4

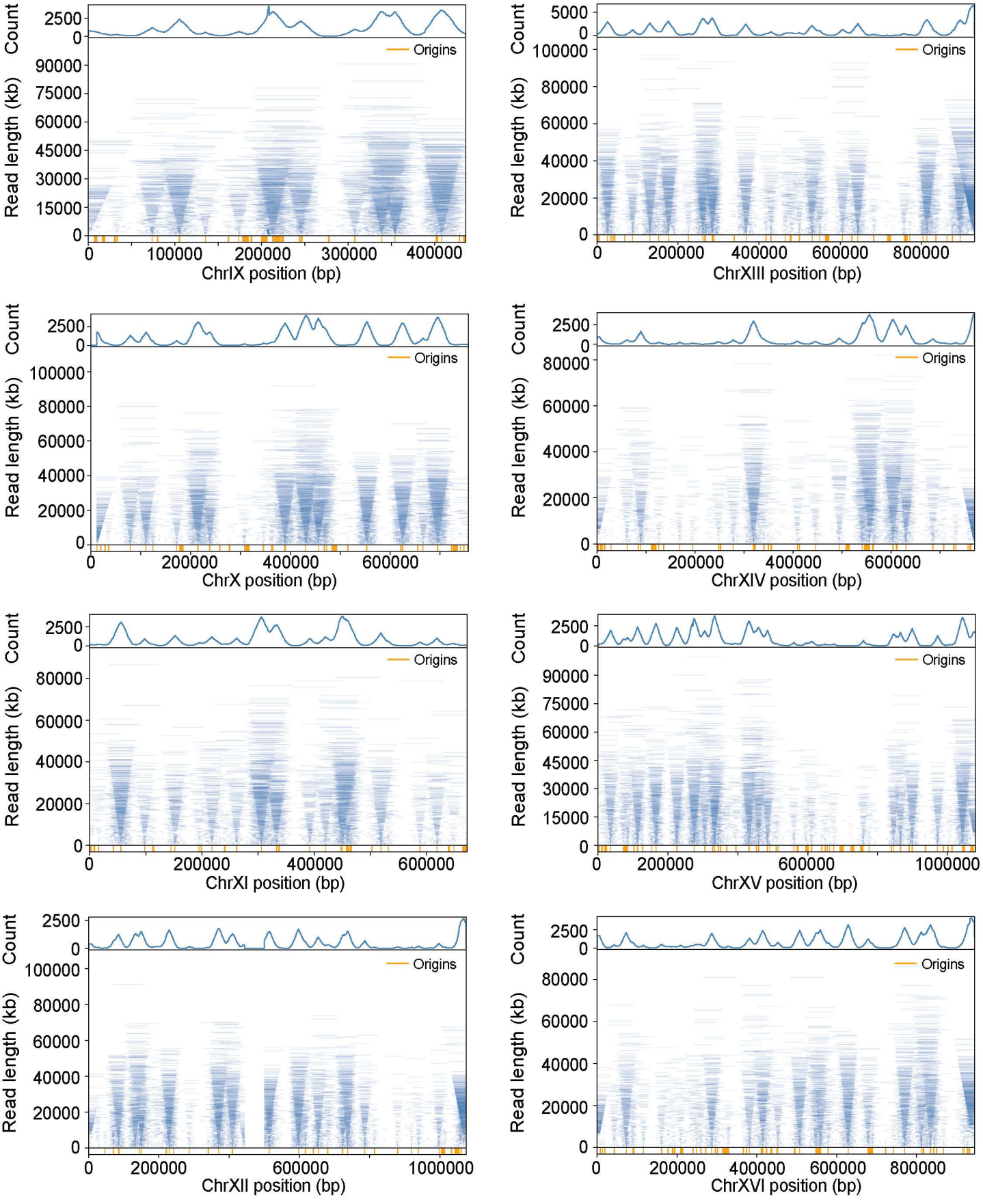

Count
Read length (kb)
Origins
ChrIX position (bp)
ChrXIII position (bp)
ChrX position (bp)
ChrXIV position (bp)
ChrXI position (bp)
ChrXV position (bp)
ChrXII position (bp)
ChrXVI position (bp)

Figure S5

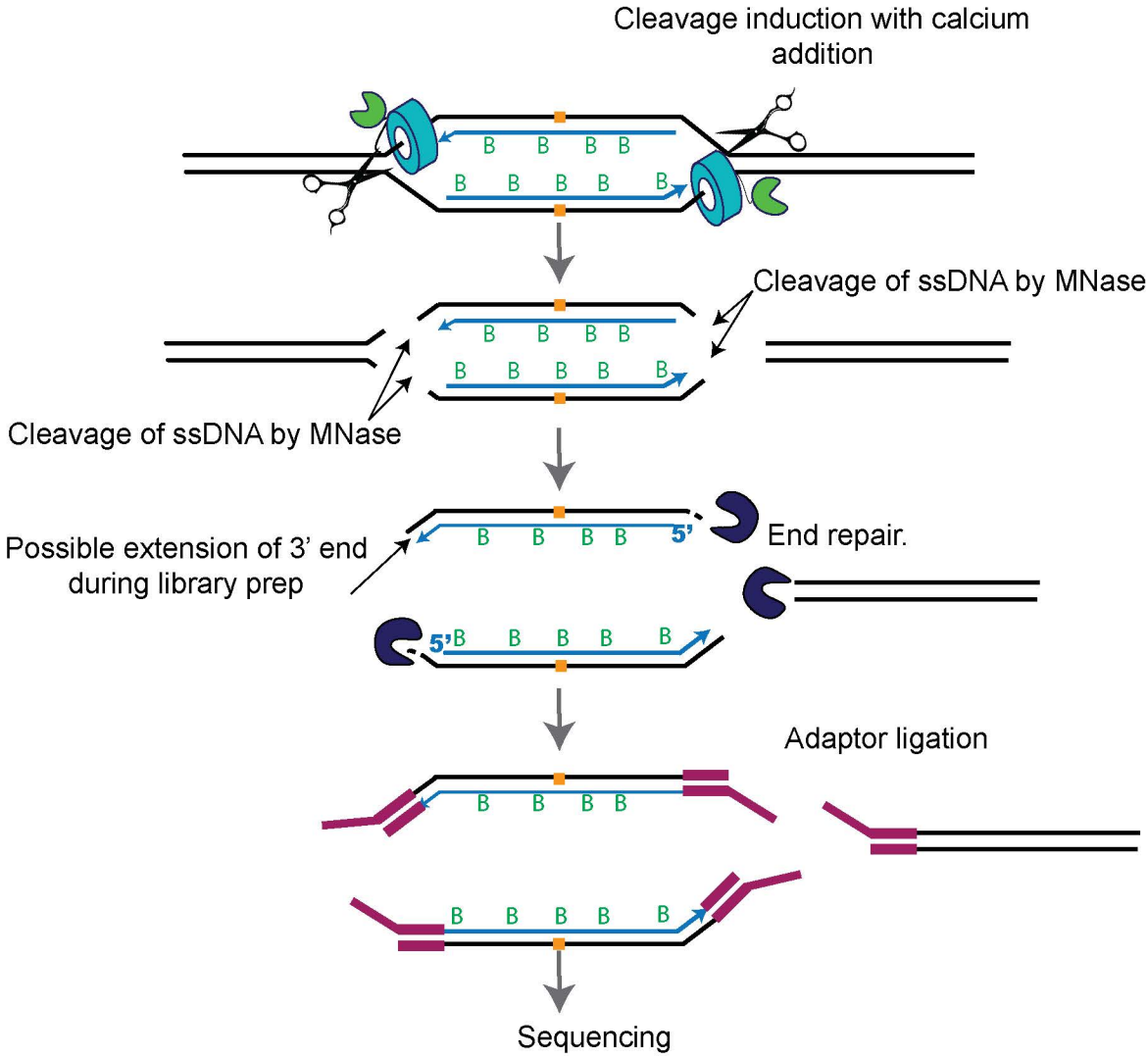

Cleavage induction with calcium addition
Cleavage of ssDNA by MNase
Cleavage of ssDNA by MNase
Possible extension of 3' end during library prep
End repair.
Adaptor ligation
Sequencing

Figure S6

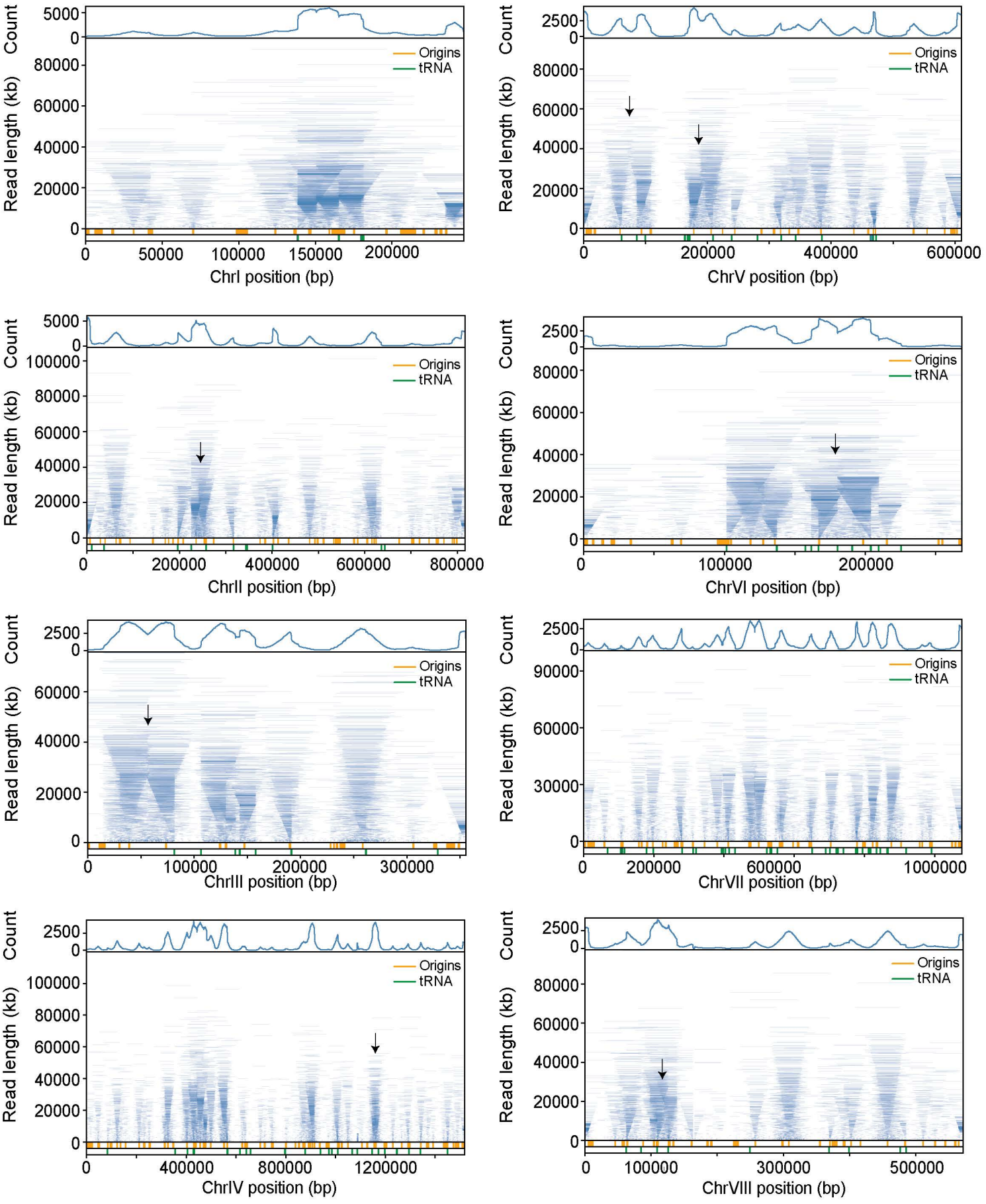

Count
Read length (kb)
Origins
tRNA
ChrI position (bp)
ChrV position (bp)
ChrII position (bp)
ChrVI position (bp)
ChrIII position (bp)
ChrVII position (bp)
ChrIV position (bp)
ChrVIII position (bp)

Figure S7

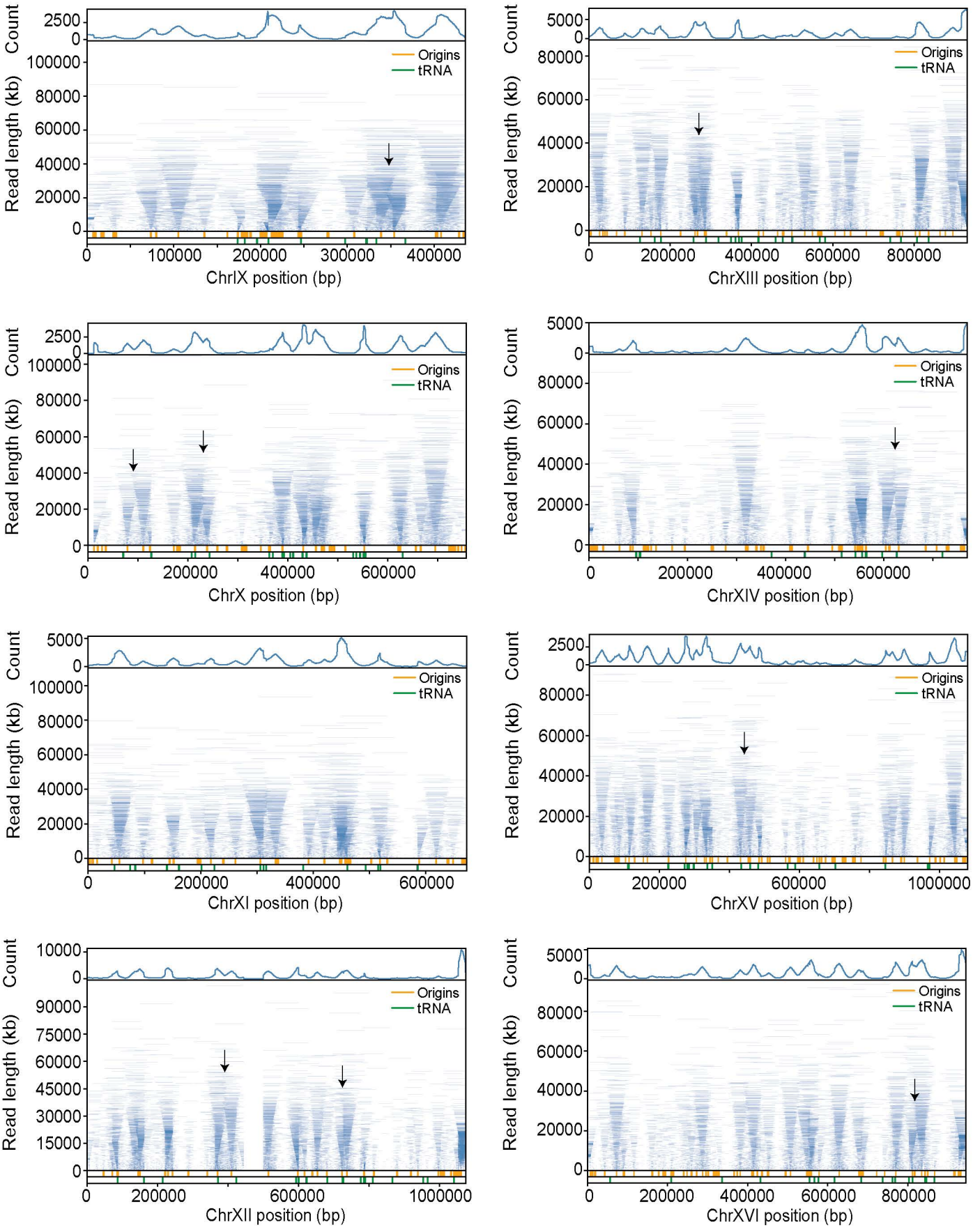

Count
Read length (kb)
Origins
tRNA
ChrIX position (bp)
ChrXIII position (bp)
ChrX position (bp)
ChrXIV position (bp)
ChrXI position (bp)
ChrXV position (bp)
ChrXII position (bp)
ChrXVI position (bp)

Figure S8

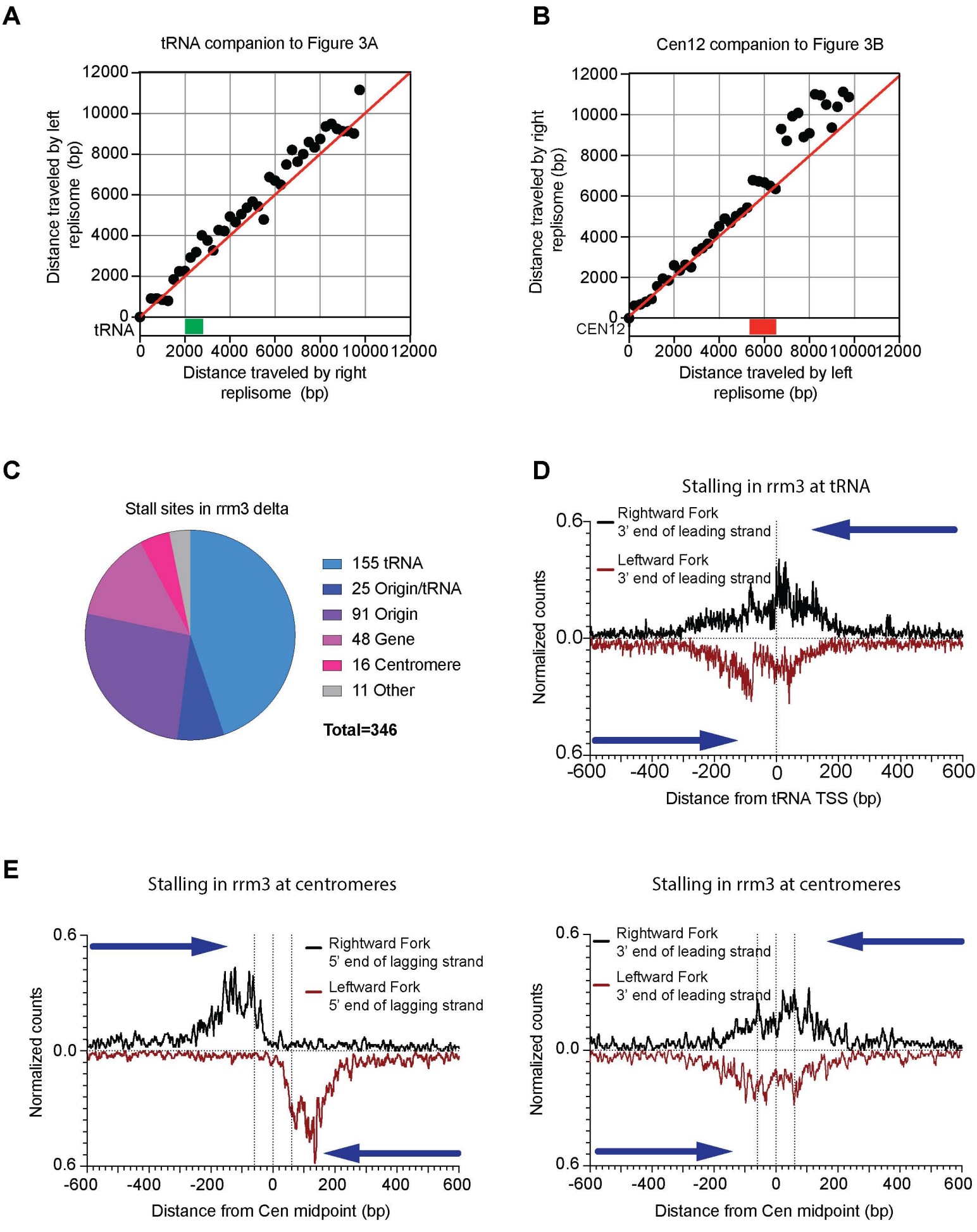
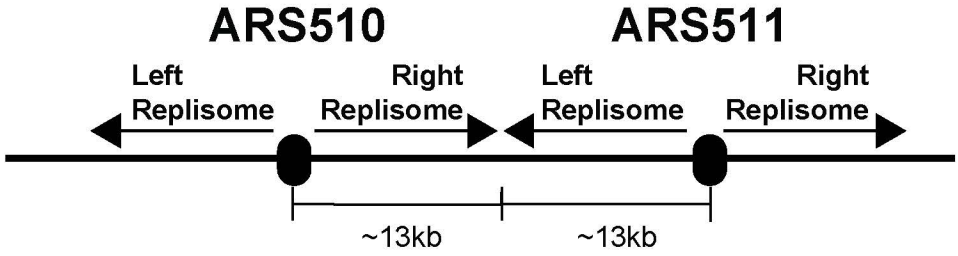
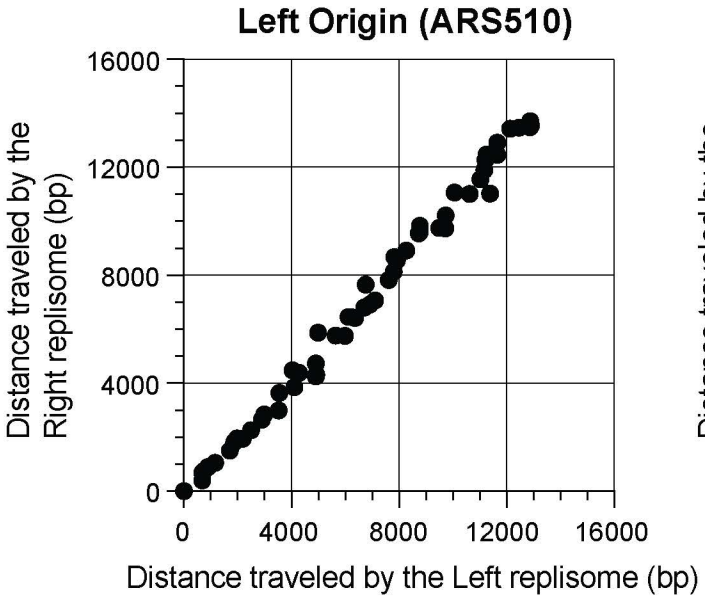

A
tRNA companion to Figure 3A
Distance traveled by left replisome (bp)
Distance traveled by right replisome (bp)
tRNA
B
Cen12 companion to Figure 3B
Distance traveled by right replisome (bp)
Distance traveled by left replisome (bp)
CEN12
C
Stall sites in rrm3 delta
155 tRNA
25 Origin/tRNA
91 Origin
48 Gene
16 Centromere
11 Other
Total=346
D
Stalling in rrm3 at tRNA
Rightward Fork
3' end of leading strand
Leftward Fork
3' end of leading strand
Normalized counts
Distance from tRNA TSS (bp)
E
Stalling in rrm3 at centromeres
Rightward Fork
5' end of lagging strand
Leftward Fork
5' end of lagging strand
Normalized counts
Distance from Cen midpoint (bp)
Stalling in rrm3 at centromeres
Rightward Fork
3' end of leading strand
Leftward Fork
3' end of leading strand
Normalized counts
Distance from Cen midpoint (bp)

Figure S9


ARS510
ARS511
Left Replisome
Right Replisome
Left Replisome
Right Replisome
~13kb
~13kb


Left Origin (ARS510)
Distance traveled by the Right replisome (bp)
16000
12000
8000
4000
0
0
4000
8000
12000
16000
Distance traveled by the Left replisome (bp)

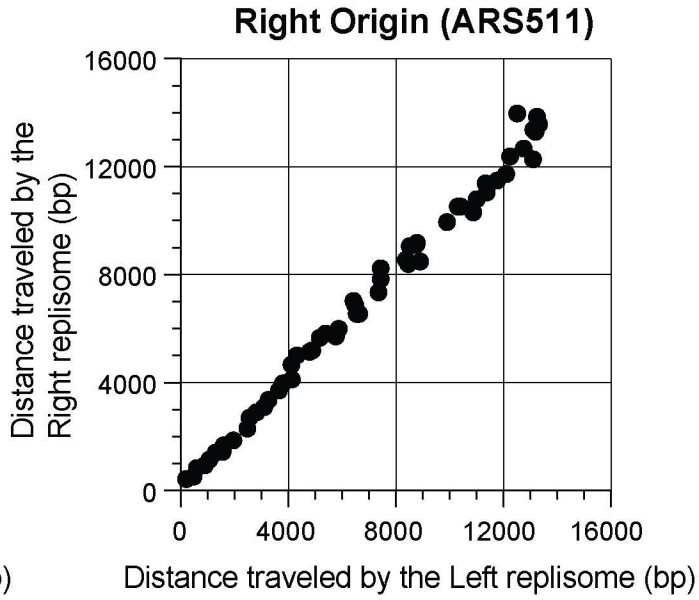

Right Origin (ARS511)
Distance traveled by the Right replisome (bp)
16000
12000
8000
4000
0
0
4000
8000
12000
16000
Distance traveled by the Left replisome (bp)

Figure S10

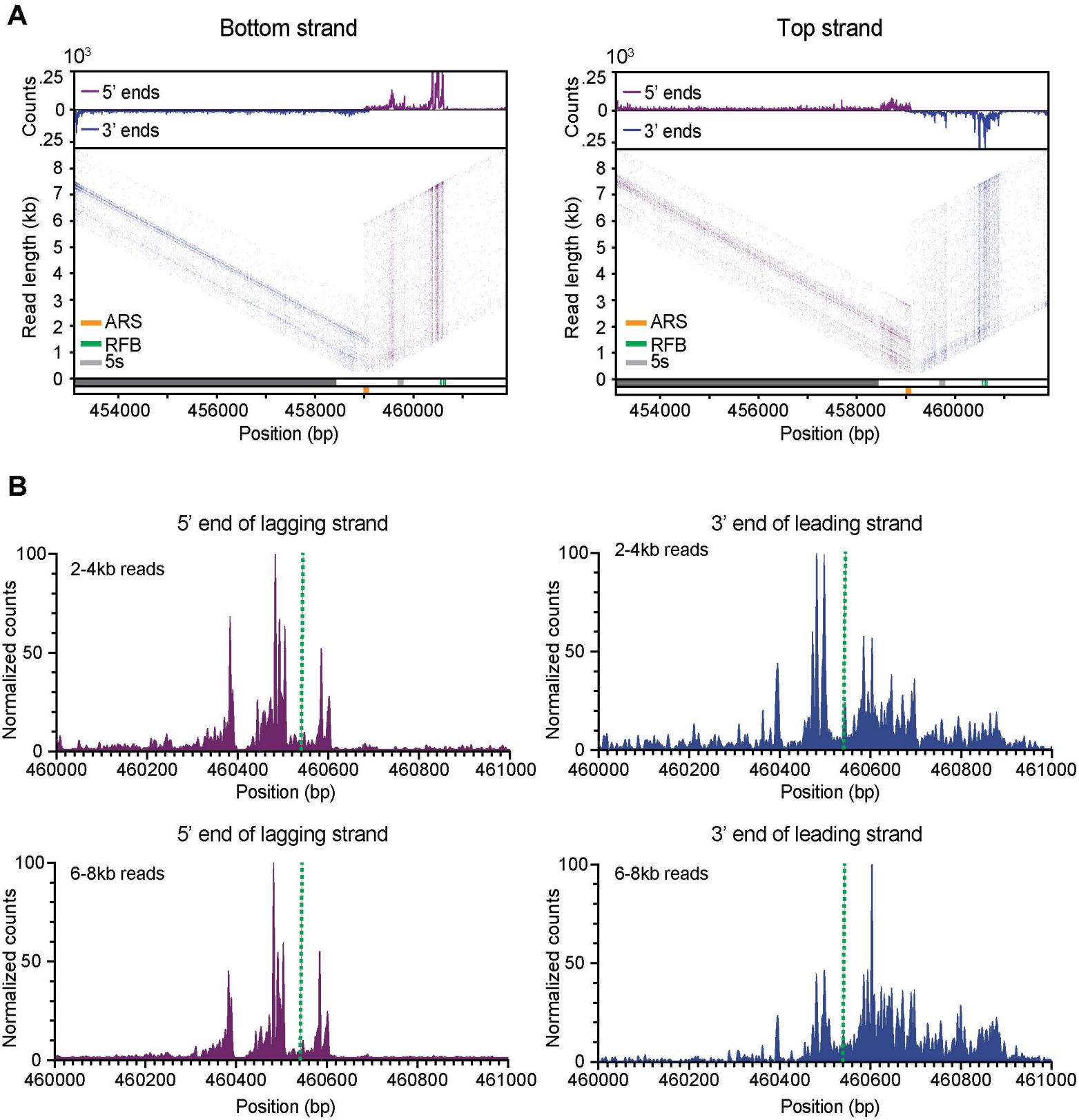
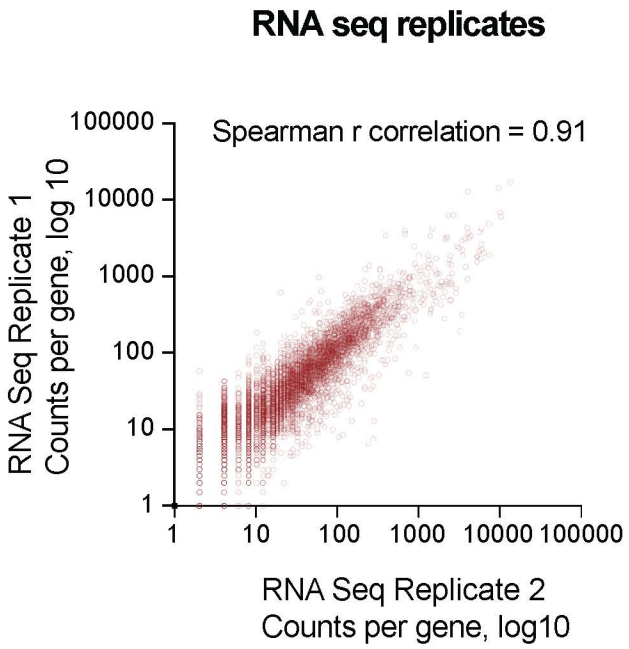

A
Bottom strand
Top strand
10^3
Counts
.25
0
.25
5' ends
3' ends
Read length (kb)
8
7
6
5
4
3
2
1
0
ARS
RFB
5s
454000
456000
458000
460000
Position (bp)
B
5' end of lagging strand
3' end of leading strand
2-4kb reads
6-8kb reads
Normalized counts
100
50
0
460000
460200
460400
460600
460800
461000
Position (bp)

Figure S11

TTAATGCTAGTTCGCCGCCTCGAAAATGAAAAGAGAAGAAGAGACCAGAAGAGCA
ACTTCGCATTGcCACTGTTTATTTCTTTAAGAACATCTTCAATACTAAATAAGACAAC
CCATCTTCAGTTATTTTTACTCTTGGTAGAGCCCAACTTAGcATCCTTTTGAACCTCA
AGAACTTAAAATG**GCTGCTGCTACTTCTACTAAGAAGTTGCACAAGGAACCAGCT**
ACTTTGATTAAGGCTATTGATGGTGATACTGTTAAGTTGATGTACAAGGGTCAAC
CAATGACTTTCAGATTGTTGTTGGTTGATACTCCAGAAACTAAGCACCCAAAGAA
GGGTGTTGAAAAGTACGGTCCAGAAGCTTCTGCTTTCACTAAGAAGATGGTTGAA
AACGCTAAGAAGATTGAAGTTGAATTCGATAAGGGTCAAAGAACTGATAAGTAC
GGTAGAGGTTTGGCTTACATTTACGCTGATGGTAAGATGGTTAACGAAGCTTTGG
TTAGACAAGGTTTGGCTAAGGTTGCTTACGTTTACAAGCCAAACAACACTCACGA
ACAACACTTGAGAAAGTCTGAAGCTCAAGCTAAGAAGGAAAAGTTGAACATTTG
GTCTGAAGATAACGCTGATTCTGGTCAAgattacaaggatcacgatggcggtagtggcagcAT
GTCTCAACAGTCTAGCTCTCCAACAAAAGAGGATAATAACTCCAGCTCCCCCGTTG
TGCCTAATCCTGATTCTGTTCCACCACAGCTTTCTTCCCCAGCTCTATTTTATAGCT
CCTCTTCATCACAAGGTGATATCTATGGTCGCAACAATAGCCAGAACTTAAGTCAG
GGAGAGGGAAACATCAGAGCTGCTATAGGTTCTTCTCCACTAAATTTTCCATCTTCT
TCCCAAAGACAAAATTCCGATGTTTTCCAATCTCAAGGCAGACAGGGCAGAATTCG
TTCTTCTGCGAGCGCTTCTGGAAGGTCTAGATATCATTCTGATCTGAGAAGTGATA
GAGCACTGCCTACTTCTTCCTCTTCTTTAGGCCGTAATGGTCAAAACCGTGTACAC
ATGCGAAGAAATGATATTCATACATCTGATTTATCCTCTCCAAGAAGAATTGTGGAT
TTTGATACTAGATCCGGCGTGAATACTTTaGATACTTCTTCTTCTTCTGCTCCTCCAT
CGGAAGCCAGTGAACCCTTGAGAATAATTTGGGGTA

Figure S12


RNA seq replicates
Spearman r correlation = 0.91
RNA Seq Replicate 1
Counts per gene, log 10
100000
10000
1000
100
10
1
1
10
100
1000
10000
100000
RNA Seq Replicate 2
Counts per gene, log10