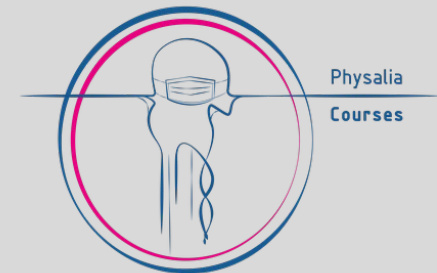


Hi-C processing

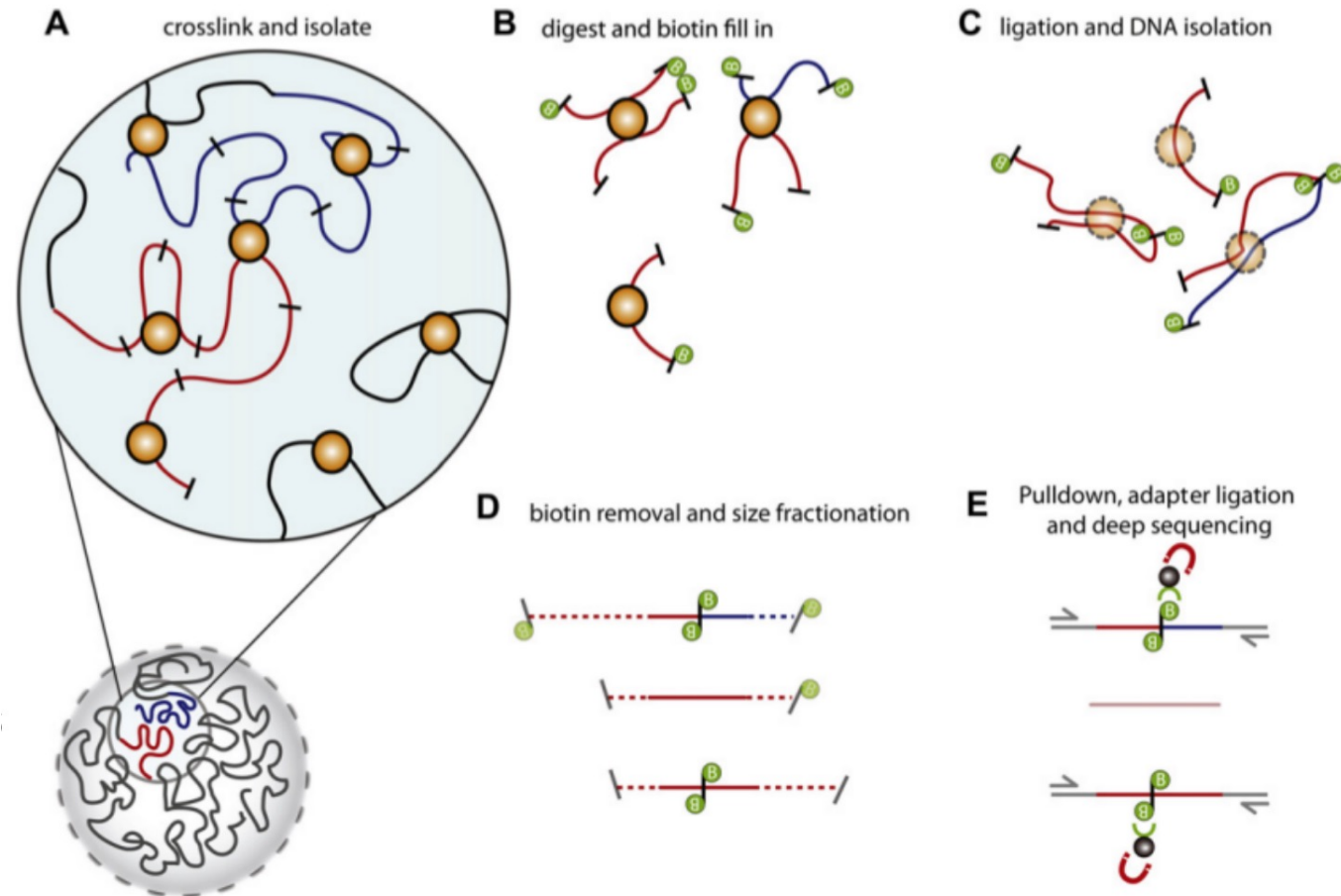
Epigenomics Data Analysis

Jacques Serizay

Physalia 2025

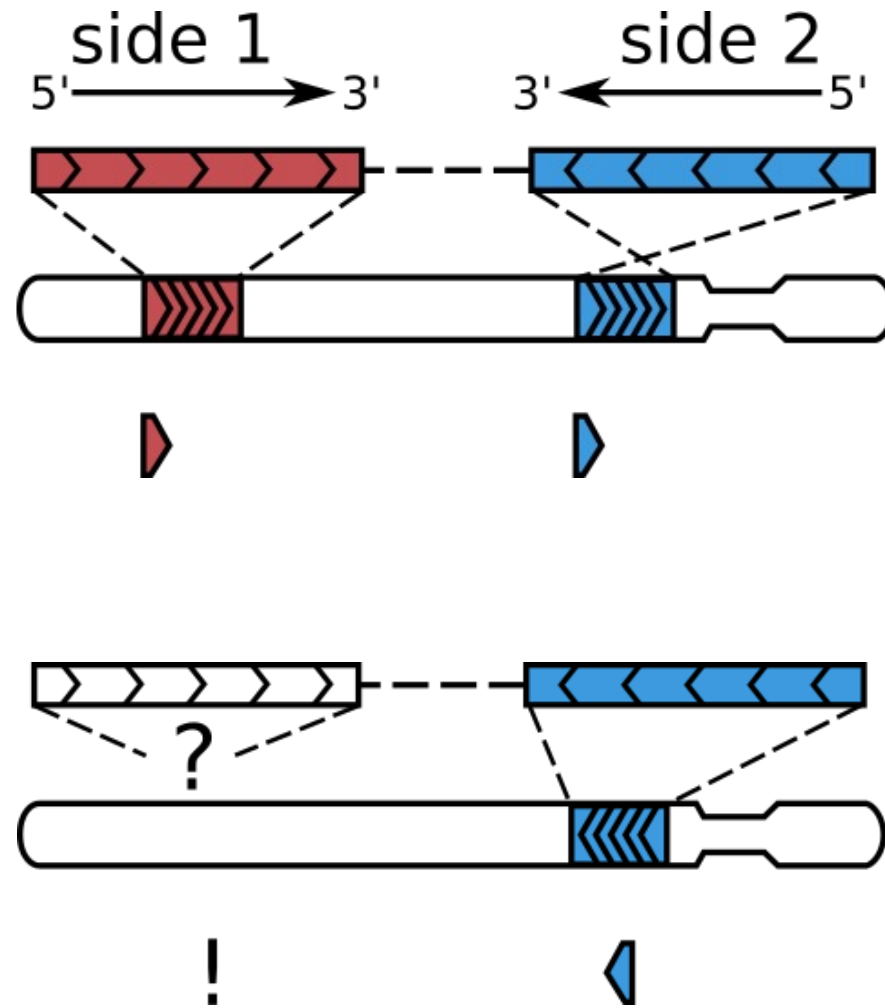


Hi-C experimental workflow



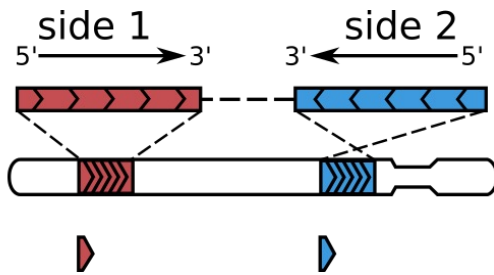
Belton et al., Methods 2012

Hi-C computational workflow



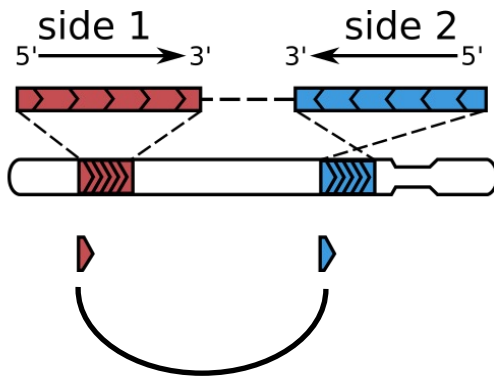
Open2C et al., Biorxiv 2023

Hi-C computational workflow



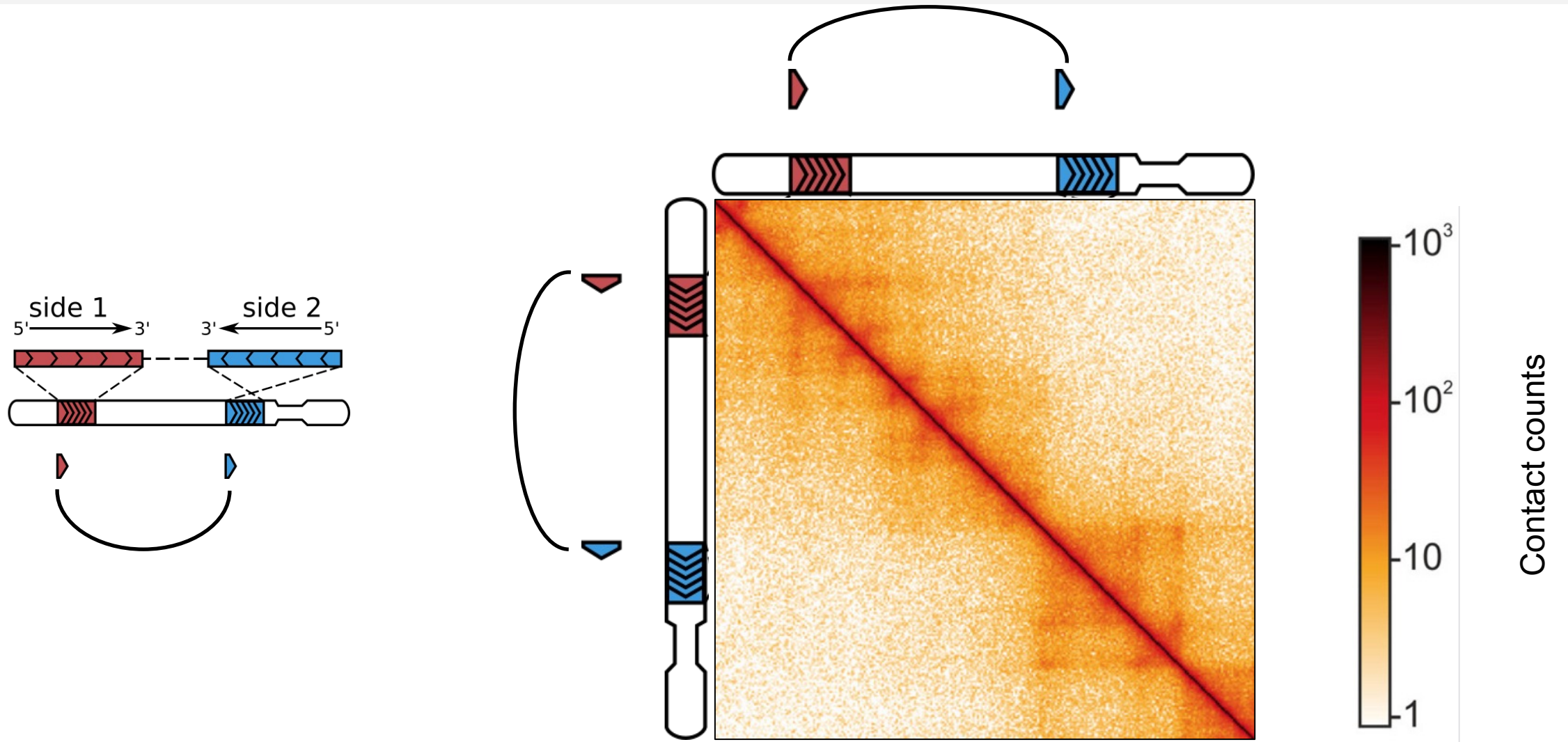
Open2C et al., Biorxiv 2023

Hi-C computational workflow



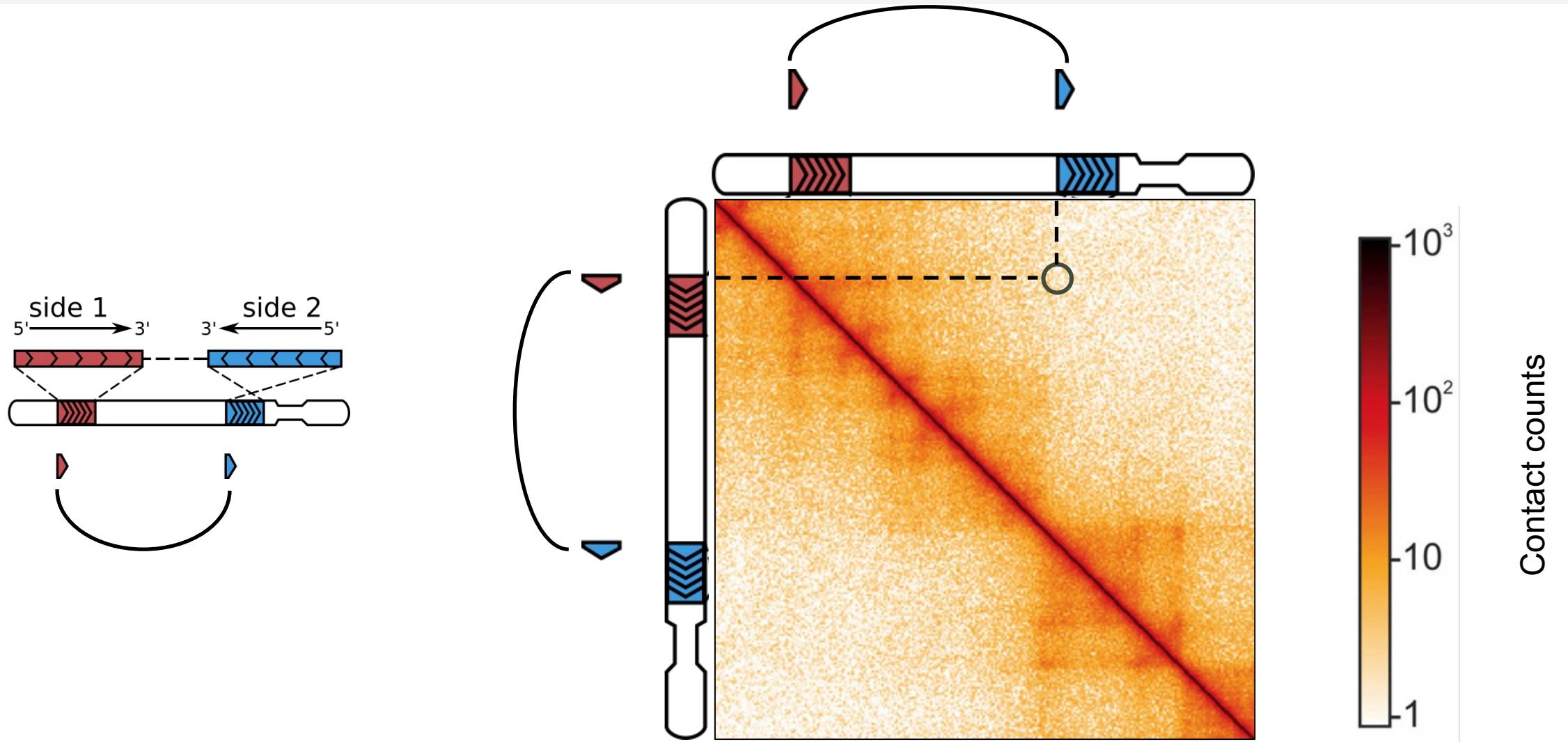
Open2C et al., Biorxiv 2023

Hi-C computational workflow



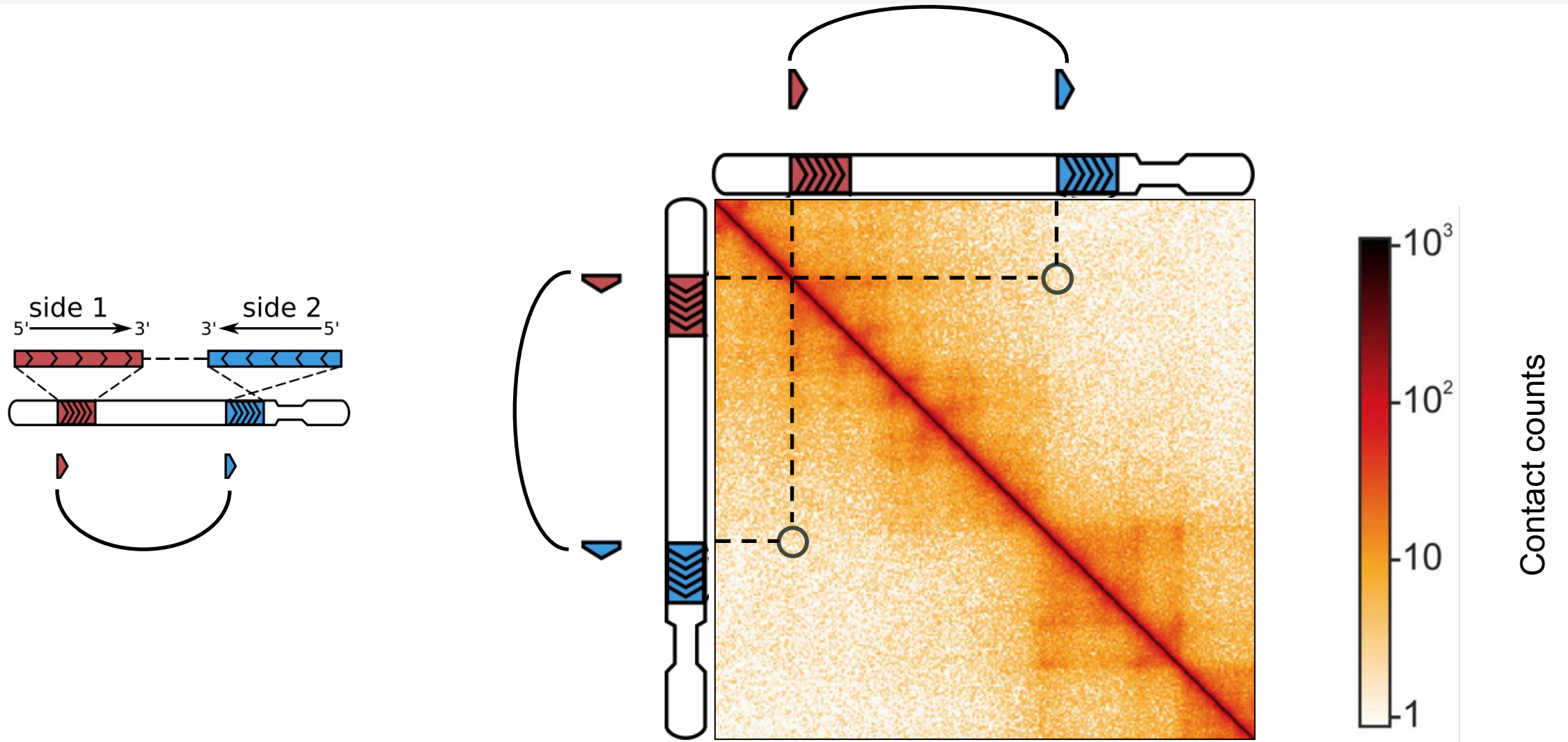
Open2C et al., Biorxiv 2023

Hi-C computational workflow



Open2C et al., Biorxiv 2023

Hi-C computational workflow



Open2C et al., Biorxiv 2023

Hi-C pipeline



Get .bcl files



Create fastq files



QC: remove/trim low quality reads



Align fastq to BAM



Filter duplicates, artifacts, ...



Generate tracks



Assay-specific downstream analysis

Hi-C pipeline



Get .bcl files



Convert fastq files



QC: remove low quality reads



Align reads



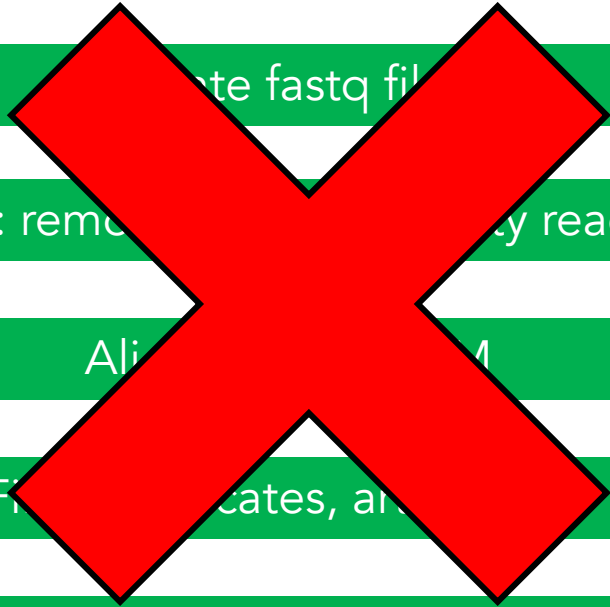
Filter duplicates, and



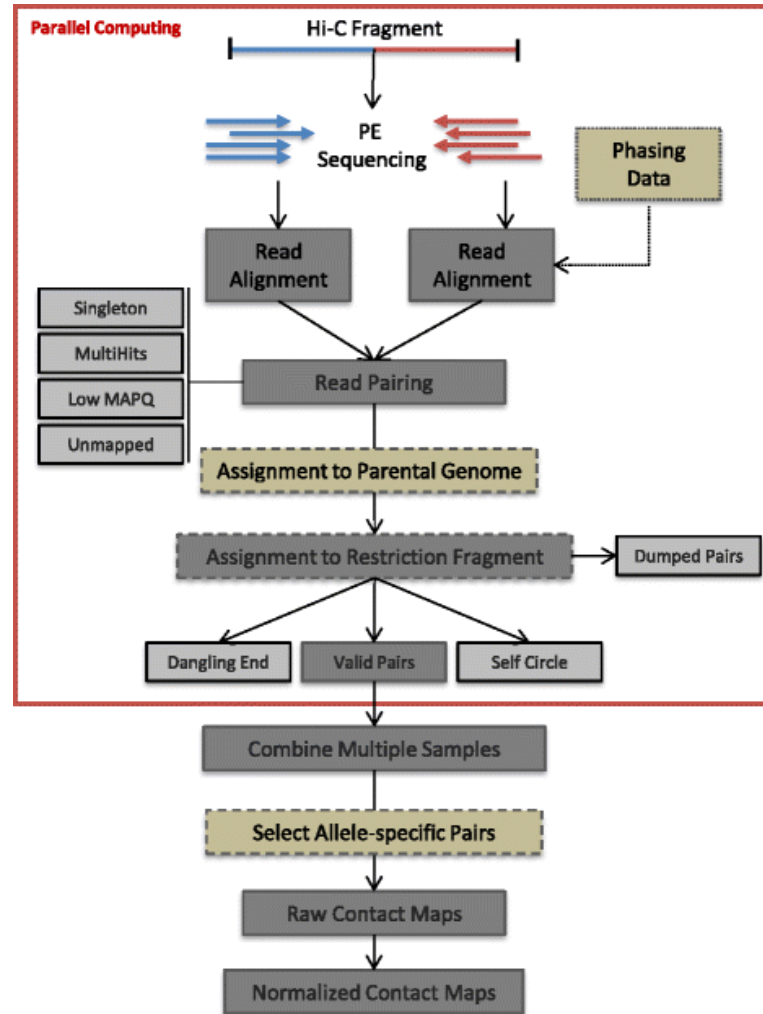
Generate tracks



Assay-specific downstream analysis

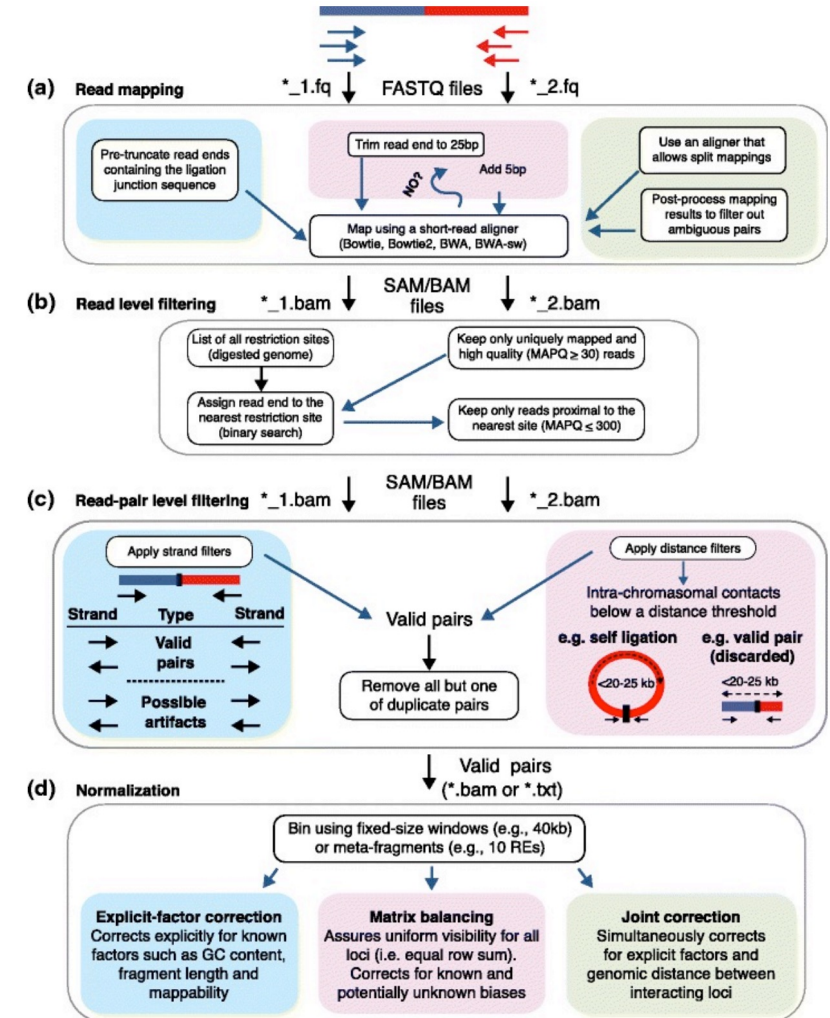


Hi-C pipeline(s)



Servant et al., Genome Biol. 2015

Ay & Noble Genome Biol 2015



Hi-C pipeline(s) – 2015

Tool	Short-read aligner(s)	Mapping improvement	Read filtering	Read-pair filtering	Normalization	Visualization	Confidence estimation	Implementation language(s)
HiCUP [46]	Bowtie/Bowtie2	Pre-truncation	✓	✓	—	—	—	Perl, R
Hiclib [47]	Bowtie2	Iterative	✓ ^a	✓	Matrix balancing	✓	—	Python
HiC-inspector [131]	Bowtie	—	✓	✓	—	✓	—	Perl, R
HIPPIE [132]	STAR	✓ ^b	✓	✓	—	—	—	Python, Perl, R
HiC-Box [133]	Bowtie2	—	✓	✓	Matrix balancing	✓	—	Python
HiCdat [122]	Subread	— ^c	✓	✓	Three options ^d	✓	—	C++, R
HiC-Pro [134]	Bowtie2	Trimming	✓	✓	Matrix balancing	—	—	Python, R
TADbit [120]	GEM	Iterative	✓	✓	Matrix balancing	✓	—	Python
HOMER [62]	—	—	✓	✓	Two options ^e	✓	✓	Perl, R, Java
Hicpipe [54]	—	—	—	—	Explicit-factor	—	—	Perl, R, C++
HiBrowse [69]	—	—	—	—	—	✓	✓	Web-based
Hi-Corrector [57]	—	—	—	—	Matrix balancing	—	—	ANSI C
GOTHIC [135]	—	—	✓	✓	—	—	✓	R
HiTC [121]	—	—	—	—	Two options ^f	✓	✓	R
chromoR [59]	—	—	—	—	Variance stabilization	—	—	R
HiFive [136]	—	—	✓	✓	Three options ^g	✓	—	Python
Fit-Hi-C [20]	—	—	—	—	—	✓	✓	Python

^aHiclib keeps the reads with only one mapped end (single-sided reads) for use in coverage computations

^bHIPPIE states that it rescues chimeric reads. No details are given

^cHiCdat reports no substantial improvement in successfully aligned read pairs when iterative mapping in Hiclib is used for *Arabidopsis thaliana* Hi-C data

^dHiCdat provides three options for normalization: coverage and distance correction, HiCNorm and ICE

^eHOMER provides two options for normalization: simpleNorm corrects for sequencing coverage only and norm corrects for coverage plus the genomic distance between loci

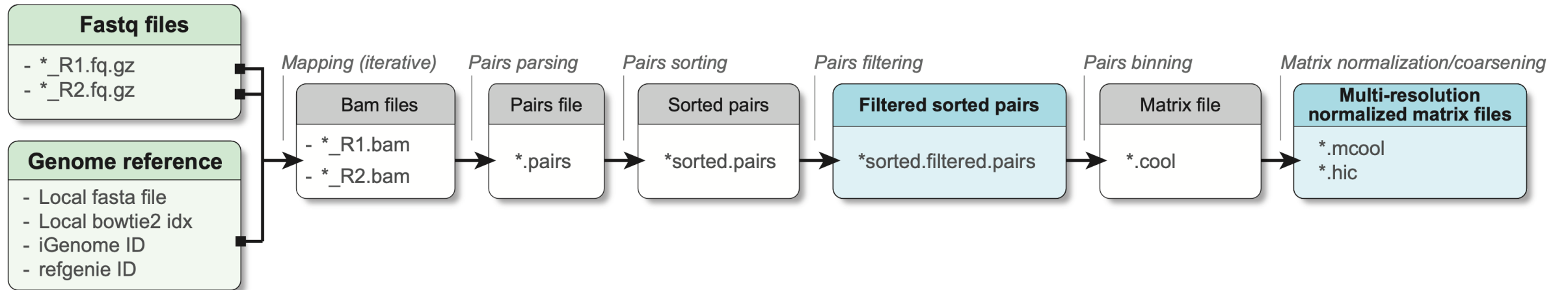
^fHiTC provides two options for normalization: normLGF implements HiCNorm and normICE implements ICE algorithm from Hiclib

^gHiFive provides three options - Probability, Express, and Binning - for normalization. The Express and Binning algorithms correspond to matrix balancing and explicit-factor correction schemes, respectively

Servant et al., *Genome Biol.* 2015

Ay & Noble *Genome Biol* 2015

Common Hi-C processing steps



Open2C collective

Welcome to Open2C!

We are the Open Chromosome Collective (Open2C)

Welcome to Open2C!

We are a group of computational biologists working on chromosome structure and interested in developing open source tools for the wider community.

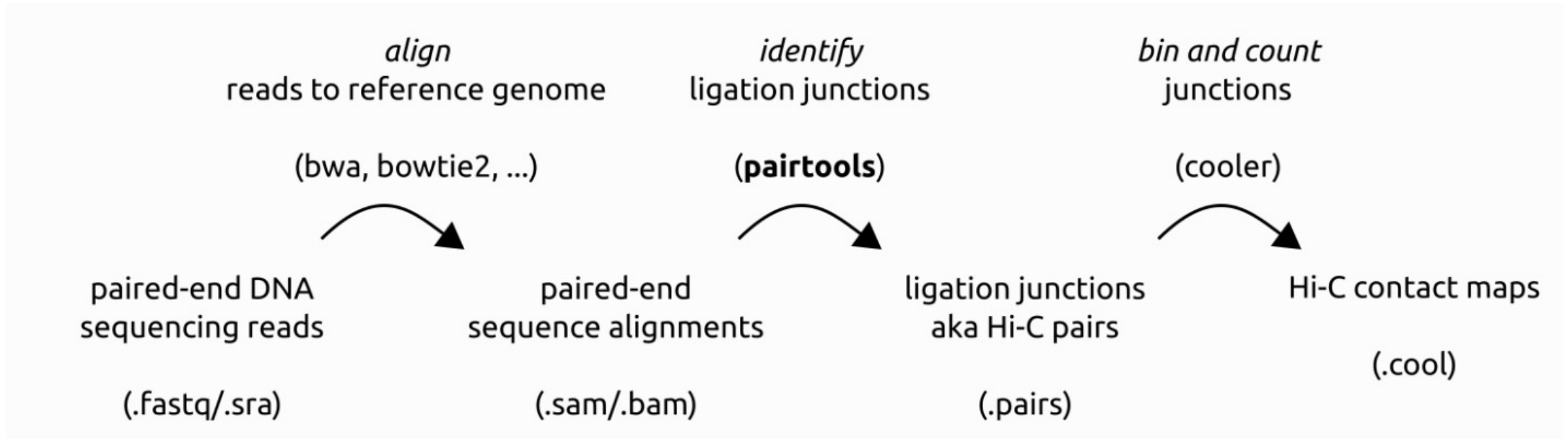
We are particularly interested in the 3D organization of chromosomes, and most of the tools are focused on the analysis of data obtained using Hi-C and related technologies. We like our tools to be easy to use, flexible, to facilitate active development of novel analytical approaches, and scalable, to make use of the latest and largest datasets.

✂ For analysis of Hi-C (and related) data, we have a suite of tools:

- [cooler](#)* for generation, normalization and storage of interaction matrices
- [pairtools](#)* for analysis and filtering of mapped reads
- [cooltools](#)* for extracting and quantifying features from Hi-C data, including: contacts vs. distance, compartments and saddles, insulation, and peaks
- [coolpup.py](#) for flexible pileup analysis
- [distiller](#) - a Nextflow pipeline for automation of Hi-C data processing, from reads to Cooler files
- [quaich](#) - a Snakemake pipeline for automated Hi-C postprocessing using **cooltools** and **coolpup.py**

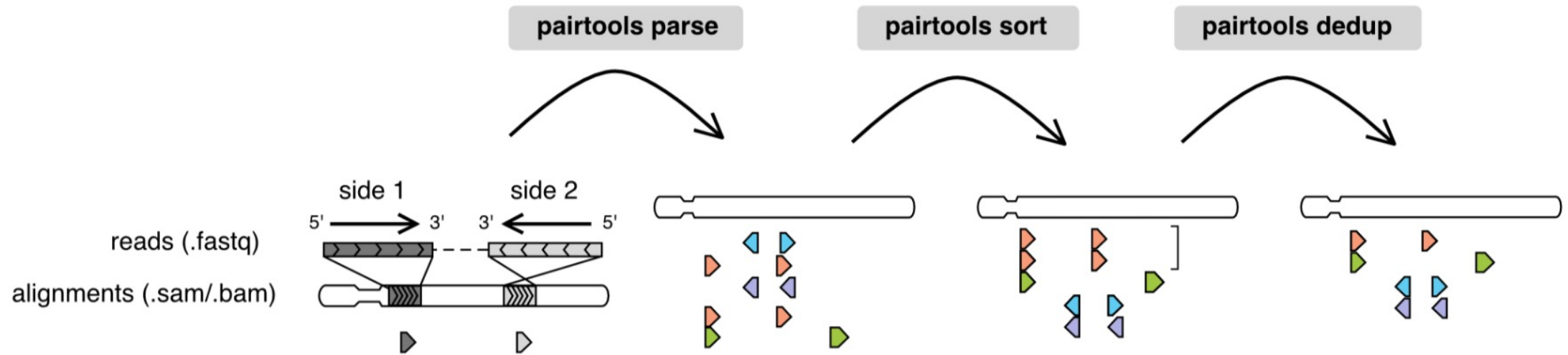
<https://github.com/open2c/>

Pairtools



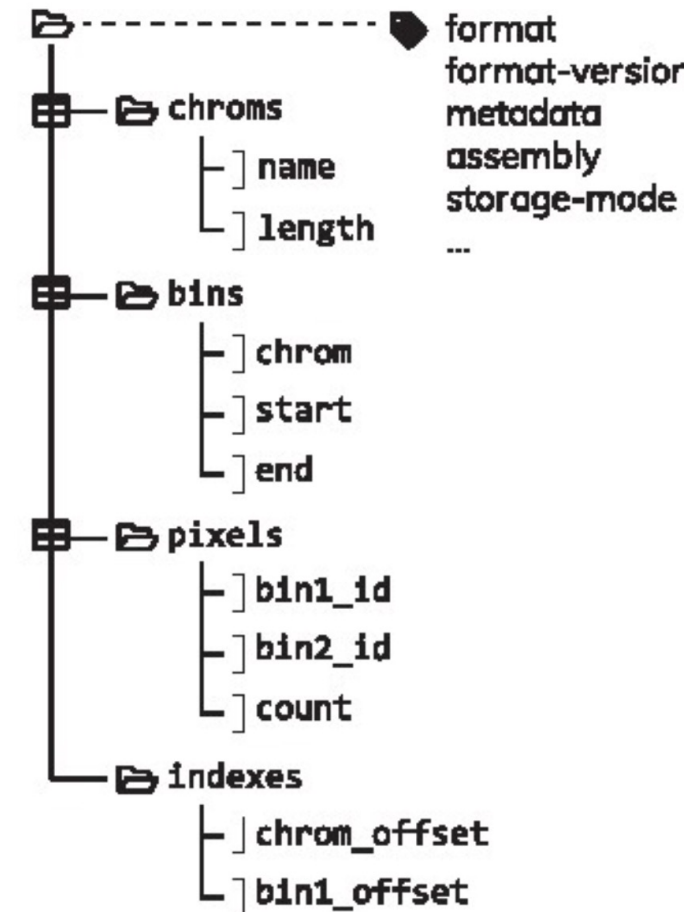
<https://pairtools.readthedocs.io/en/latest/>

Pairtools



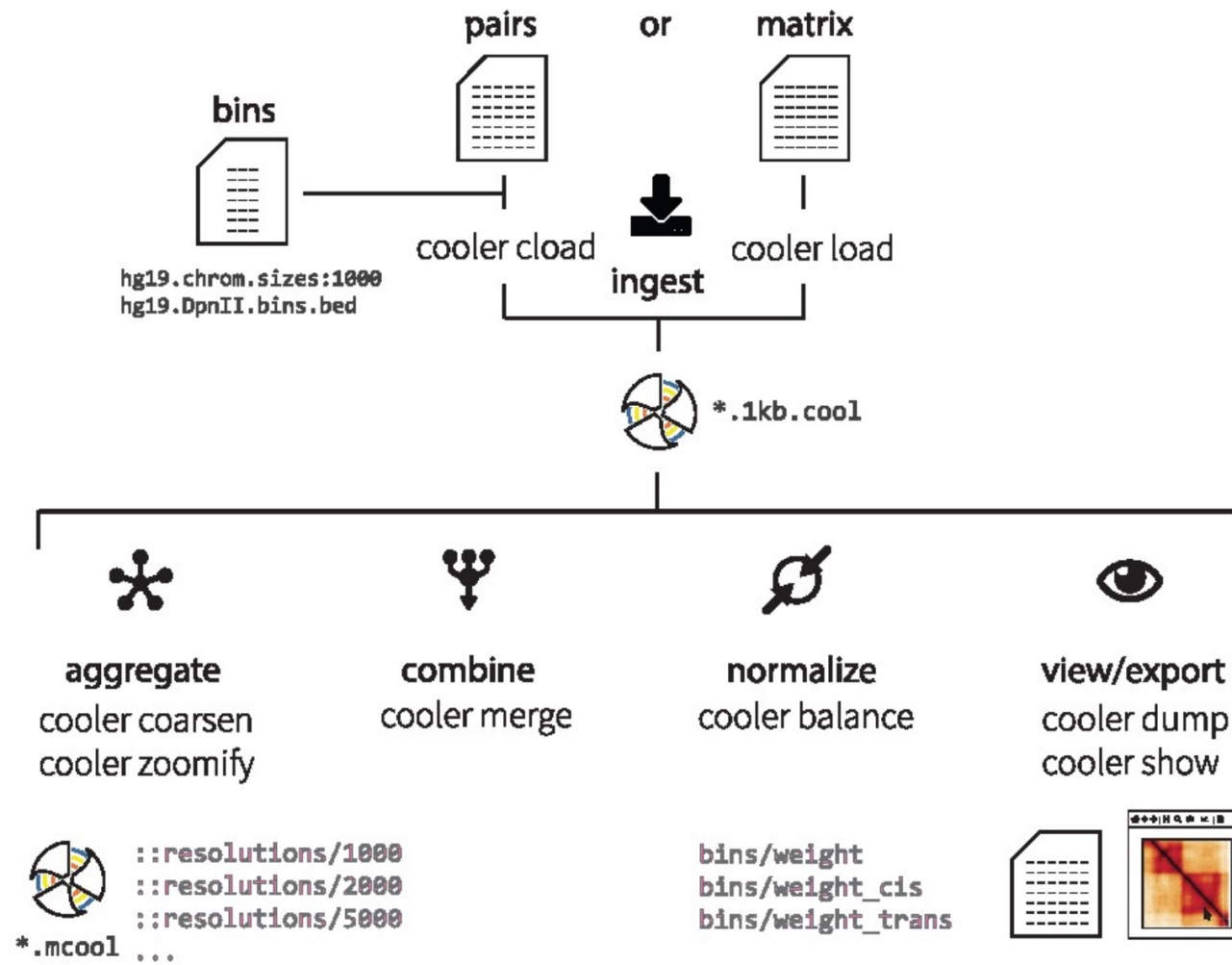
<https://pairtools.readthedocs.io/en/latest/>

Cooler



<https://cooler.readthedocs.io/en/latest/>

Cooler



<https://cooler.readthedocs.io/en/latest/>

Handling Hi-C data in R

Welcome

Preamble

Fundamentals concepts

[Chapter 1 – Hi-C pre-processing steps](#)

[Chapter 2 – Hi-C data structures in R](#)

[Chapter 3 – Manipulating Hi-C data in R](#)

[Chapter 4 – Hi-C data visualization](#)

In-depth Hi-C analysis

[Chapter 5 – Matrix-centric analysis](#)

[Chapter 6 – Interactions-centric analysis](#)

[Chapter 7 – Finding topological features in Hi-C](#)

Advanced Hi-C topics

[Chapter 8 – Data gateways: accessing public Hi-C data portals](#)

[Chapter 9 – Interoperability: using HiCExperiment with other R packages](#)

[Workflow 1: Distance-dependent interactions across yeast mutants](#)

[Workflow 2: Chromosome compartment cohesion upon mitosis entry](#)

[Workflow 3: Inter-centromere interactions in yeast](#)

Orchestrating Hi-C analysis with Bioconductor

Package: OHCA

Authors: Jacques Serizay [aut, cre]

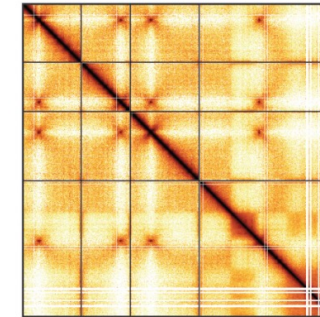
Compiled: 2024-11-11

Package version: 1.3.0

R version: **R Under development (unstable) (2024-10-21 r87258)**

BioC version: **3.21**

License: MIT + file LICENSE



Welcome

This is the landing page of the “**Orchestrating Hi-C analysis with Bioconductor**” book. **The primary aim of this book is to introduce the R user to Hi-C analysis.** This book starts with key concepts important for the analysis of chromatin conformation capture and then presents `Bioconductor` tools that can be leveraged to process, analyze, explore and visualize Hi-C data.

<https://www.nature.com/articles/s41467-024-44761-x>

Granges vs GInteractions

```
> gr = exons(TxDb.Hsapiens.UCSC.hg19.knownGene); gr
```

GRanges with 289969 ranges and 1 metadata column:

	seqnames	ranges	strand	exon_id
	<Rle>	<IRanges>	<Rle>	<integer>
[1]	chr1	[11874, 12227]	+	1
[2]	chr1	[12595, 12721]	+	2
[3]	chr1	[12613, 12721]	+	3
...	...	...	...	...
[289967]	chrY	[59358329, 59359508]	-	277748
[289968]	chrY	[59360007, 59360115]	-	277749
[289969]	chrY	[59360501, 59360854]	-	277750

seqlengths:

chr1	chr2 ...	chrUn_gl000249
249250621	243199373 ...	38502

GRanges

- length(gr); gr[1:5]
- seqnames(gr)
- start(gr)
- end(gr)
- width(gr)
- strand(gr)

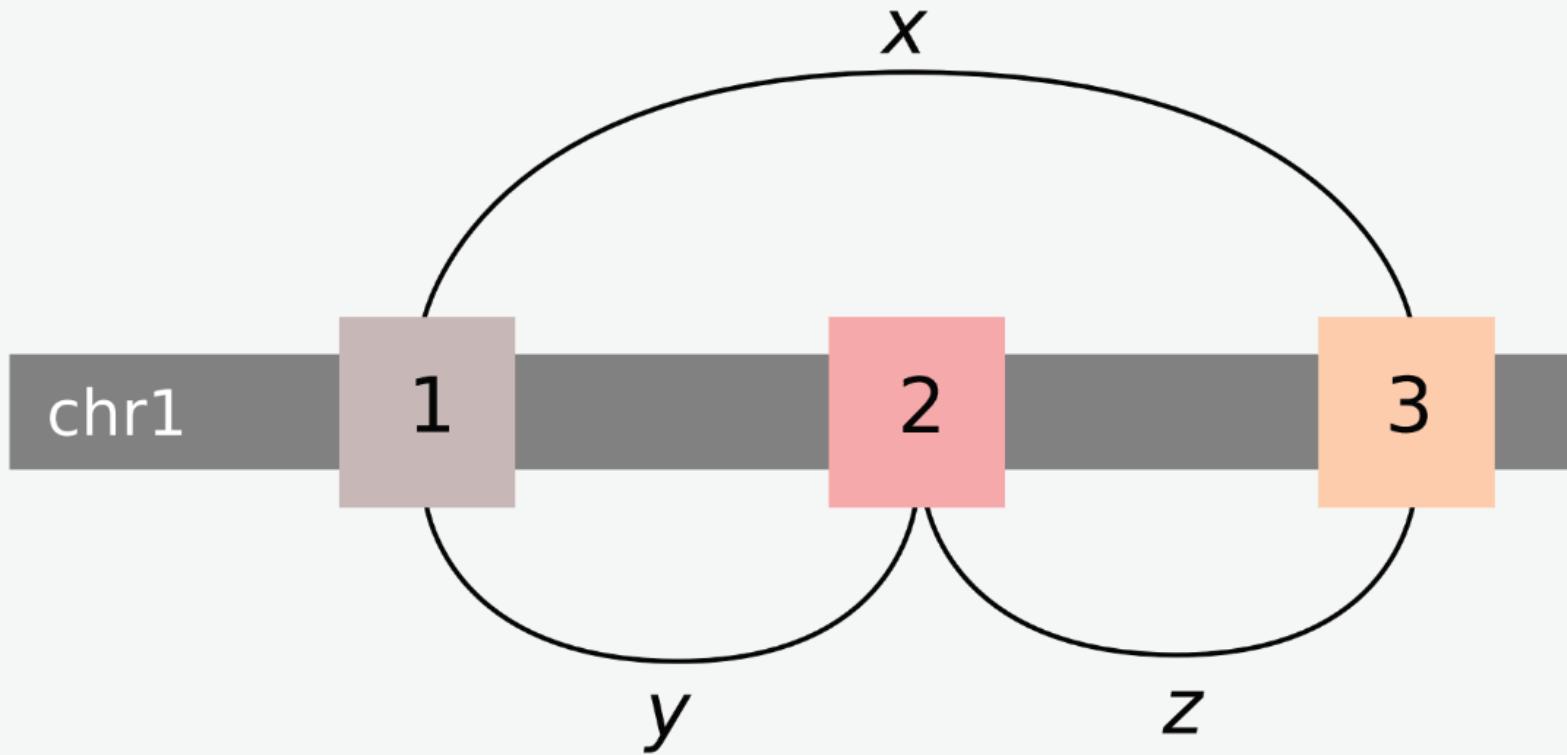
DataFrame

- mcols(gr)
- gr\$exon_id

Seqinfo

- seqlevels(gr)
- seqlengths(gr)
- genome(gr)

Granges vs GInteractions



GInteractions

x	1	3	chr1 [10, 15]
y	1	2	chr1 [30, 35]
z	2	3	chr1 [50, 55]

@anchor1 @anchor2 @regions

Handling Hi-C files in R

<ContactFile> .(m)cool, .hic, HiC-Pro

ContactFile:

- path to disk-stored cool file
- (resolution)
- (path to disk-stored pairs file)
- (metadata)

availableResolutions(*File)
availableChromosomes(*File)

import(*File, focus = "II:1-10000", resolution = 2000)

<PairsFile>

PairsFile:

- path to disk-stored pairs file
- (metadata)

import(PairsFile)
<GInteractions>

Coercion

as: "GInteractions"
as: "InteractionSet"
as: "ContactMatrix"
as: "matrix"
as: "data.frame"

Methods

refocus()
zoom()
"[", "\$"
anchors()
regions()
seqinfo()

<HiCExperiment>

fileName() <character>

- path to disk-stored cool file

pairsFile() <character>

- path to disk-stored pairs file

focus() <character>

- Imported genomic locus

resolutions() <numeric>

- 1000
- 2000
- 4000
- 8000
- ...

resolution()
<numeric>

interactions()
<GInteractions>

...
<GRanges>

anchors(x)[[1]]
<GRanges>

scores(), scores<-
<SimpleList>

...
<GRanges> <Num>

scores(x, 2)
<numeric>

metadata(),
metadata<-
<list>

- log
- processing steps
- biological
information
- ...

topologicalFeatures(),
topologicalFeatures<-
<SimpleList>

- loops
- borders
- compartments
- viewpoints
- ...