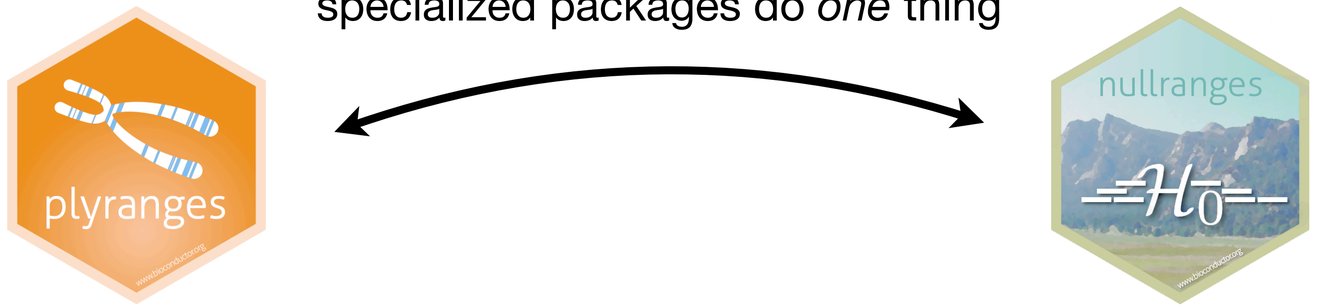
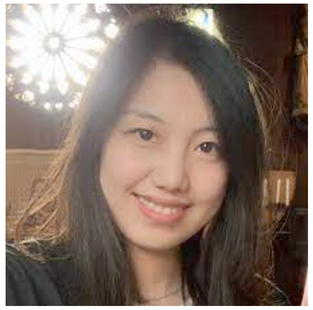
Modular framework:
specialized packages do *one* thing



Tidy enrichment analysis with *plyranges* and *nullranges*

Michael Love ~ October, 2023

***nullranges*: `matchRanges()` and `bootRanges()`**


Eric Davis, Wancen Mu,
Doug Phanstiel and myself



Mikhail Dozmorov, Stuart Lee, Tim Triche,
others from **#nullranges** on Bioc Slack

nullranges.github.io/nullranges/
tidyomics.github.io/tidy-ranges-tutorial/

***matchRanges*: Generating null hypothesis genomic ranges via covariate-matched sampling**

Eric S. Davis[1], Wancen Mu[2], Stuart Lee[3], Mikhail G. Dozmorov[4,5], Michael I. Love[2,6*], Douglas H. Phanstiel[1,7,8,9,10*]

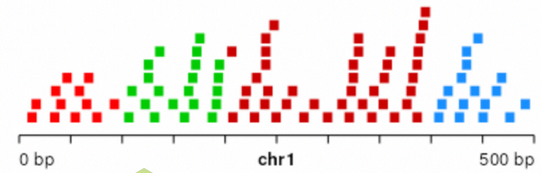
Availability and implementation
https://nullranges.github.io/nullranges

Deriving biological insights from genomic data commonly requires comparing attributes of selected genomic loci to a null set of loci. The selection of this null set is non trivial, as it requires careful consideration of potential covariates, a problem that is exacerbated by the non-uniform distribution of genomic features including genes, enhancers, and transcription factor binding sites. Propensity score-based covariate matching methods allow selection of null sets from a pool of possible items while controlling for multiple covariates; however, existing packages do not operate on genomic data classes and can be slow for large data sets making them difficult to integrate into genomic workflows. To address this, we developed *matchRanges*, a propensity score-based covariate matching method for the efficient and convenient generation of matched null ranges from a set of background ranges within the Bioconductor framework.

bootRanges: Flexible generation of null sets of genomic ranges for hypothesis testing

Wancen Mu[1], Eric Davis[2], Stuart Lee[5], Mikhail Dozmorov[6], Douglas H. Phanstiel[2,3], and Michael I. Love [*1,4]

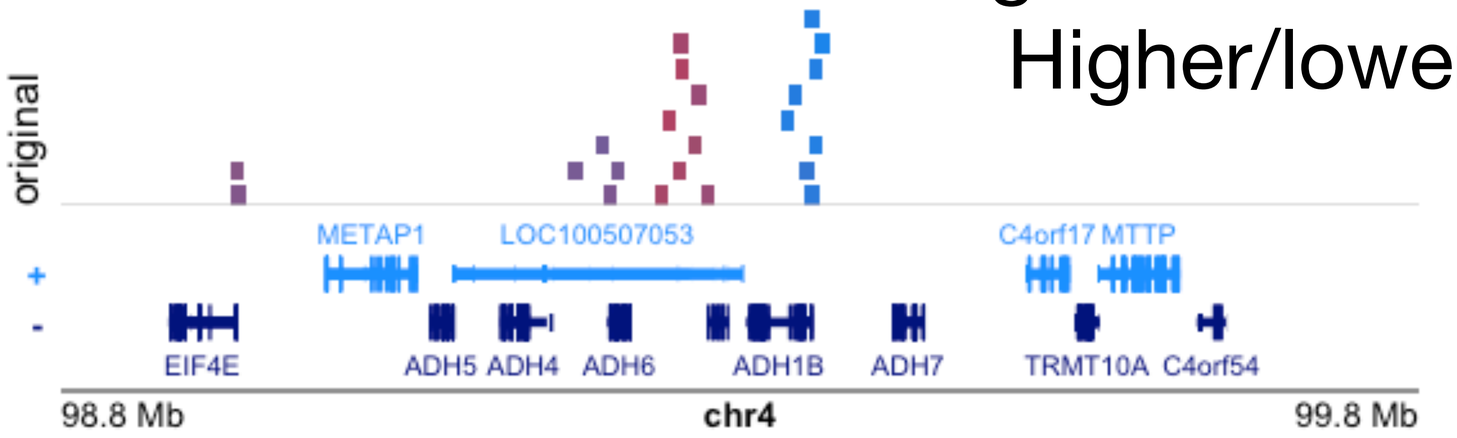
[1]Department of Biostatistics,
[2]Curriculum in Bioinformatics and Computational Biology,
[3]Thurston Arthritis Research Center, Department of Cell Biology & Physiology, Lineberger Comprehensive Cancer Center, Curriculum in Genetics & Molecular Biology, and
[4]Department of Genetics, University of North Carolina-Chapel Hill, NC 27599
[5]Genentech, South San Francisco, CA, USA
[6]Department of Biostatistics, Department of Pathology, Virginia Commonwealth University, Richmond, VA 23298, USA



Can visualize null range set with
plotgardener, by Nicole Kramer, *et al.*

Funded by CZI EOSS,
NIH: R35-GM128645,
R01-HG009937,
T32-GM067553

Consider a set of features (“original”)
e.g. ATAC-seq peaks, color by score



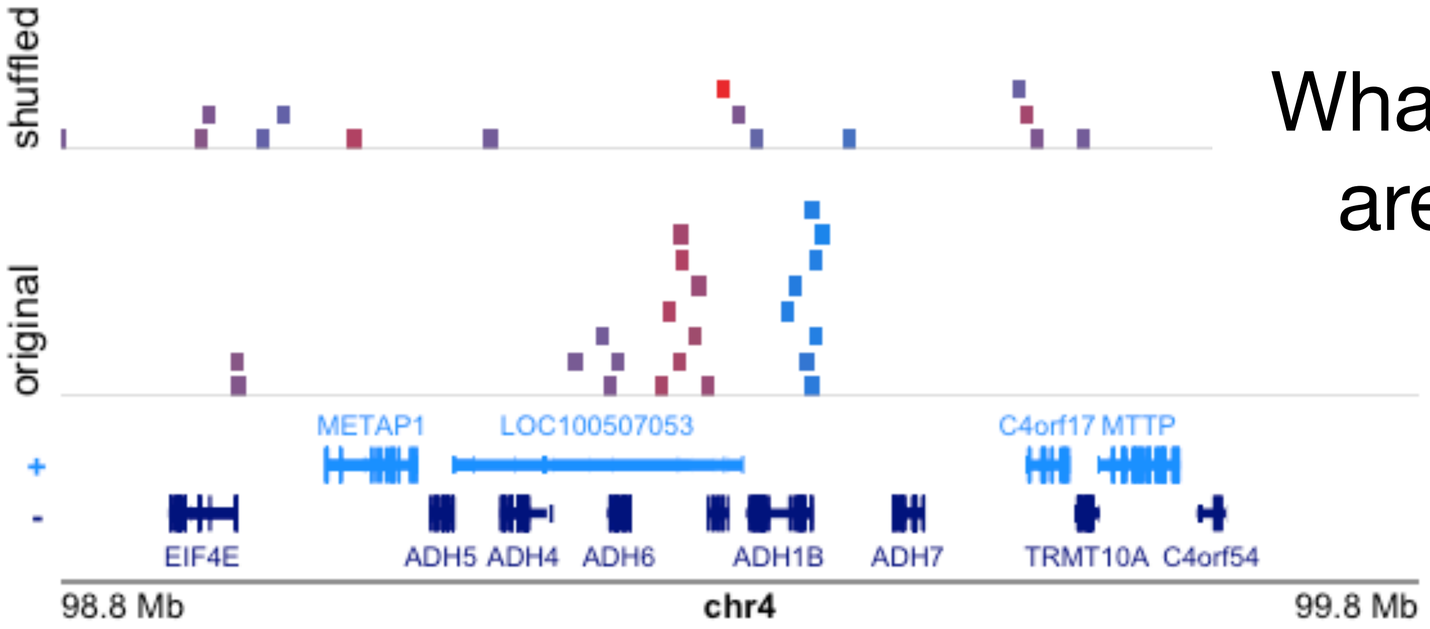
Are these closer to genes than expected? Higher/lower scores?

nullranges 1.5.5 | Get started | Reference

Related work

For general considerations of generation of null feature sets or segmentation for enrichment or colocalization analysis, consider the papers of De, Pedersen, and Kechris (2014), Haiminen, Mannila, and Terzi (2007), Huen and Russell (2010), Ferkingstad, Holden, and Sandve (2015), Dozmorov (2017), Kanduri et al. (2019) (with links in references below). Other Bioconductor packages that offer randomization techniques for enrichment analysis include LOLA (Sheffield and Bock 2016) and regioneR (Gel et al. 2016). Methods implemented outside of Bioconductor include GAT (Heger et al. 2013), GSC (Bickel et al. 2010), GREAT (McLean et al. 2010), GenometriCorr (Favorov et al. 2012), ChIP-Enrich (Welch et al. 2014), and OLOGRAM (Ferré et al. 2019).

Many have noted, shuffling position not a good model



What two things are missing?

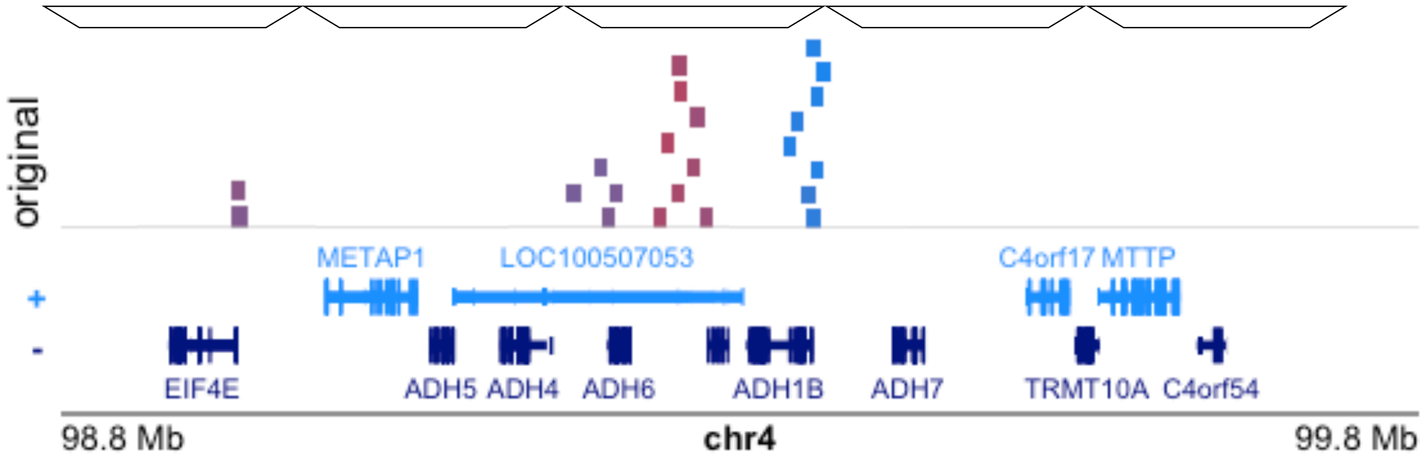
The Annals of Applied Statistics
2010, Vol. 4, No. 4, 1660–1697
DOI: 10.1214/10-AOAS363

SUBSAMPLING METHODS FOR GENOMIC INFERENCE[1]

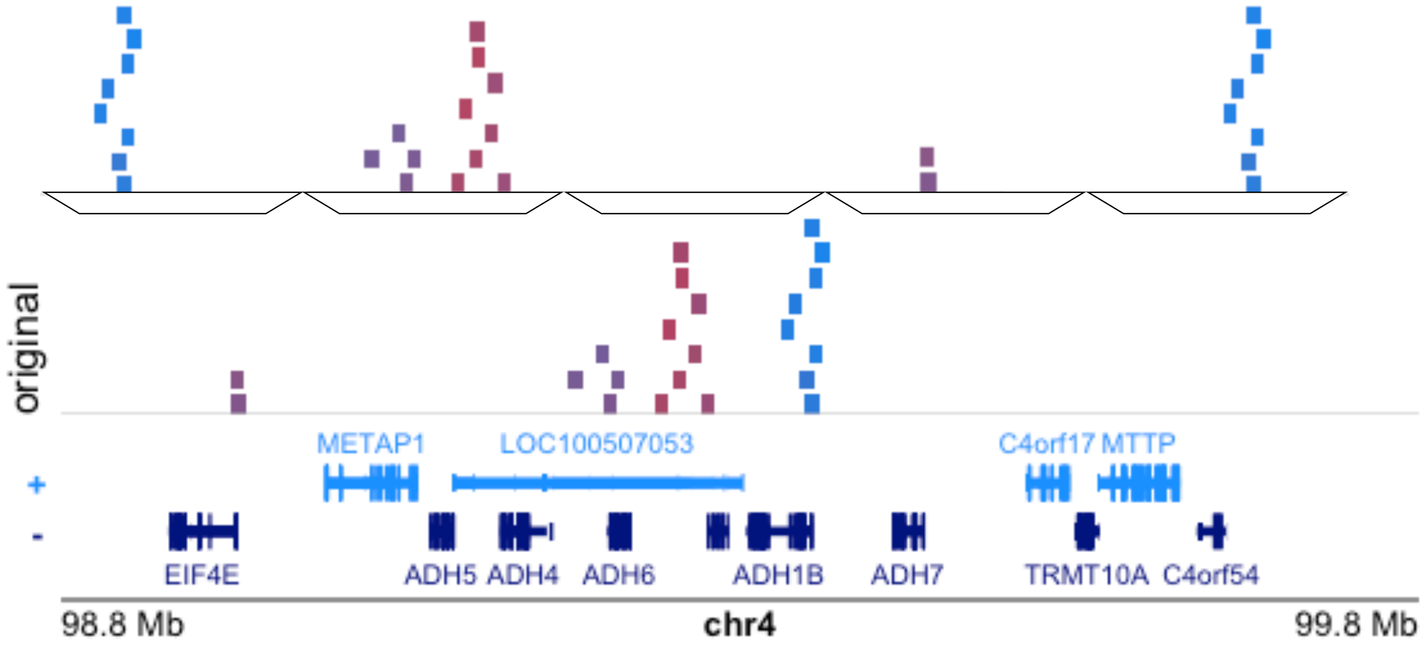
BY PETER J. BICKEL*, NATHAN BOLEY*, JAMES B. BROWN*,
HAIYAN HUANG* AND NANCY R. ZHANG*

University of California at Berkeley, University of California at Berkeley, University of California at Berkeley, University of California at Berkeley, and Stanford University

Idea: sample blocks with replacement



What is preserved?



An actual block bootstrap of a larger region

Size of resampling blocks

Iterations

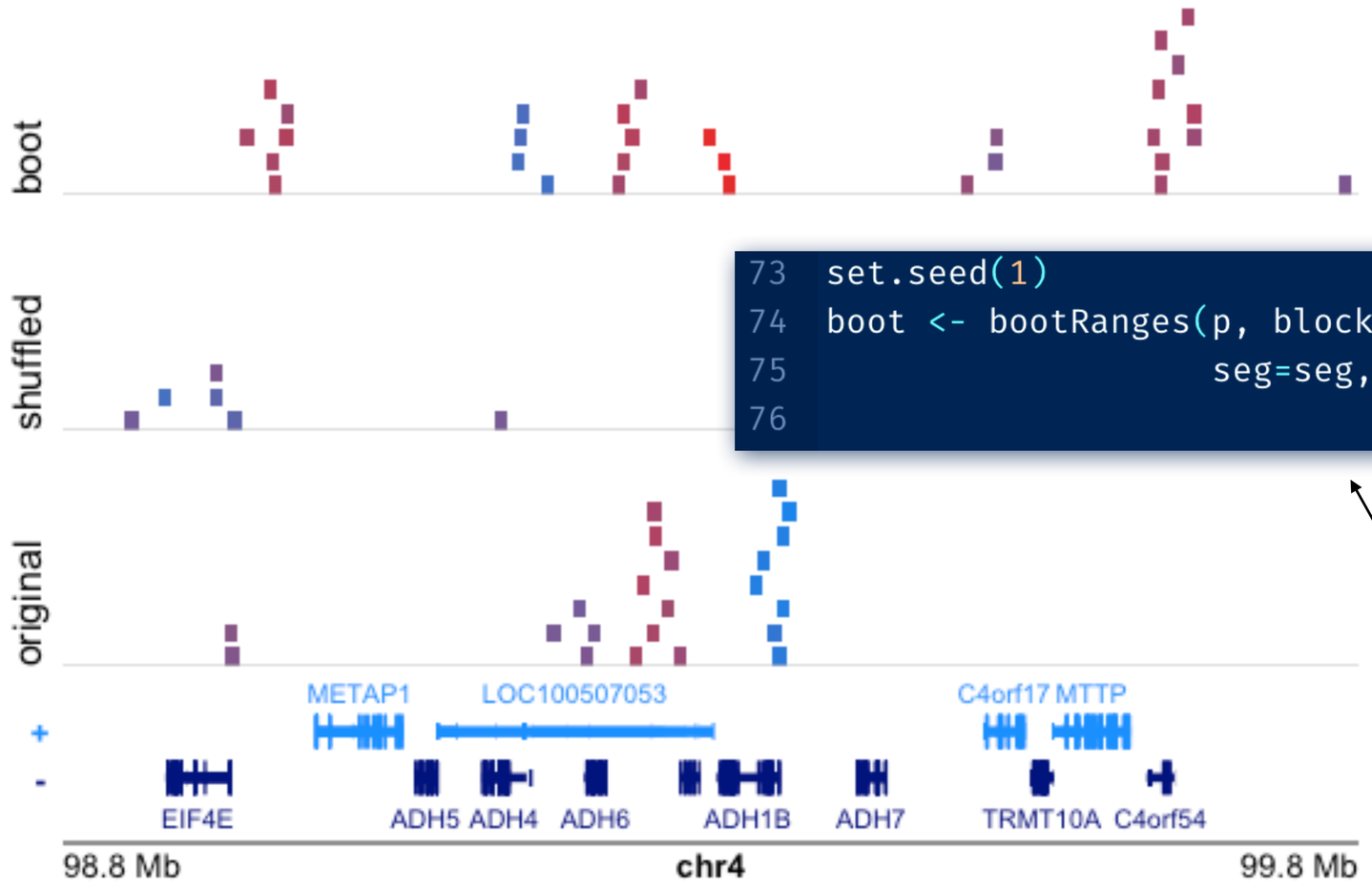
```
73  set.seed(1)
74  boot <- bootRanges(p, blockLength=1e5, R=1,
75                     seg=seg, proportionLength=FALSE)
76
```

peaks

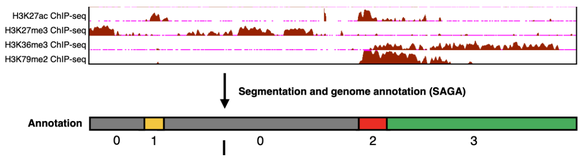
p

g

genes

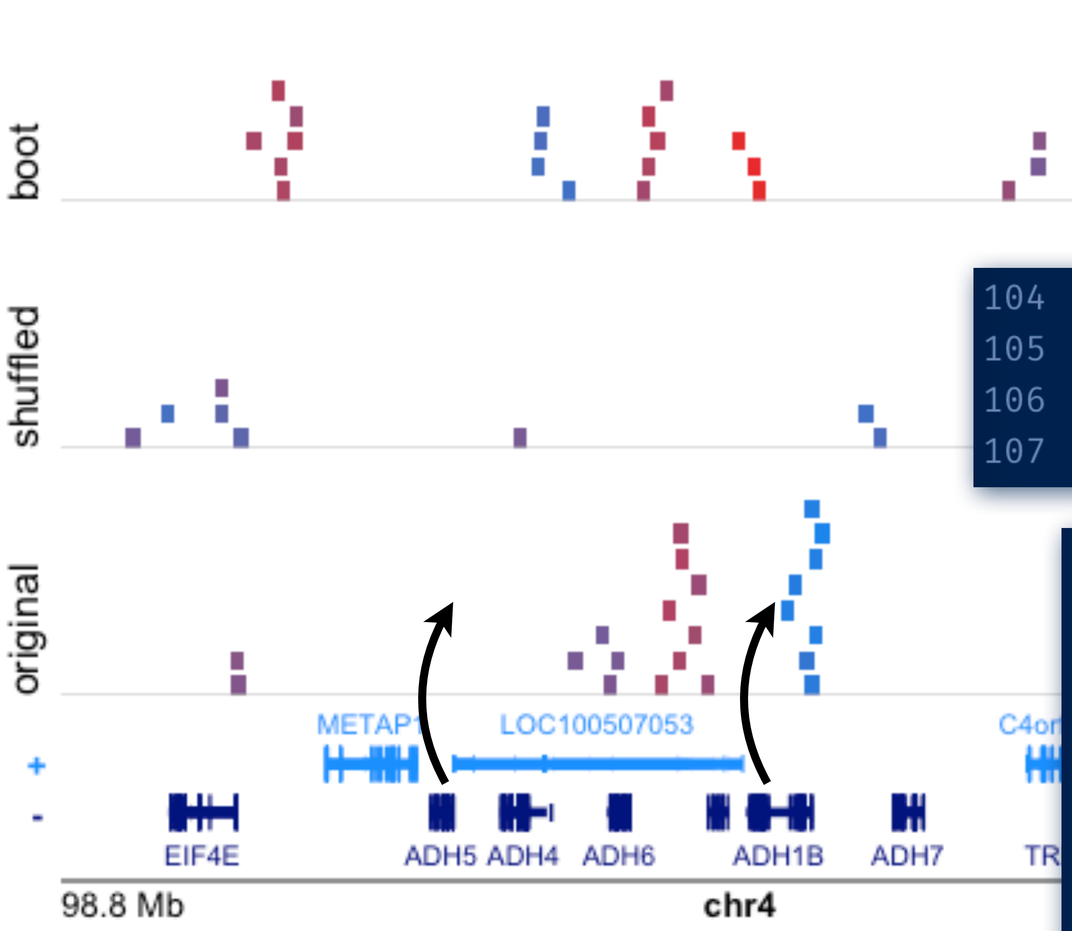


Genome segmentation, see vignette



Libbrecht, Chan, Hoffman 2021

Overlaps, count or other statistics




```
104  g %>%
105    join_overlap_left(p) %>%
106    group_by(symbol) %>% # per gene symbol
107    summarize(sum_score = sum(score))
```

```
DataFrame with 16 rows and 2 columns
                symbol sum_score
           <character> <numeric>
124              ADH1A   1.50756
125              ADH1B -32.93319
126              ADH1C -32.93319
127               ADH4         NA
128               ADH5         NA
```

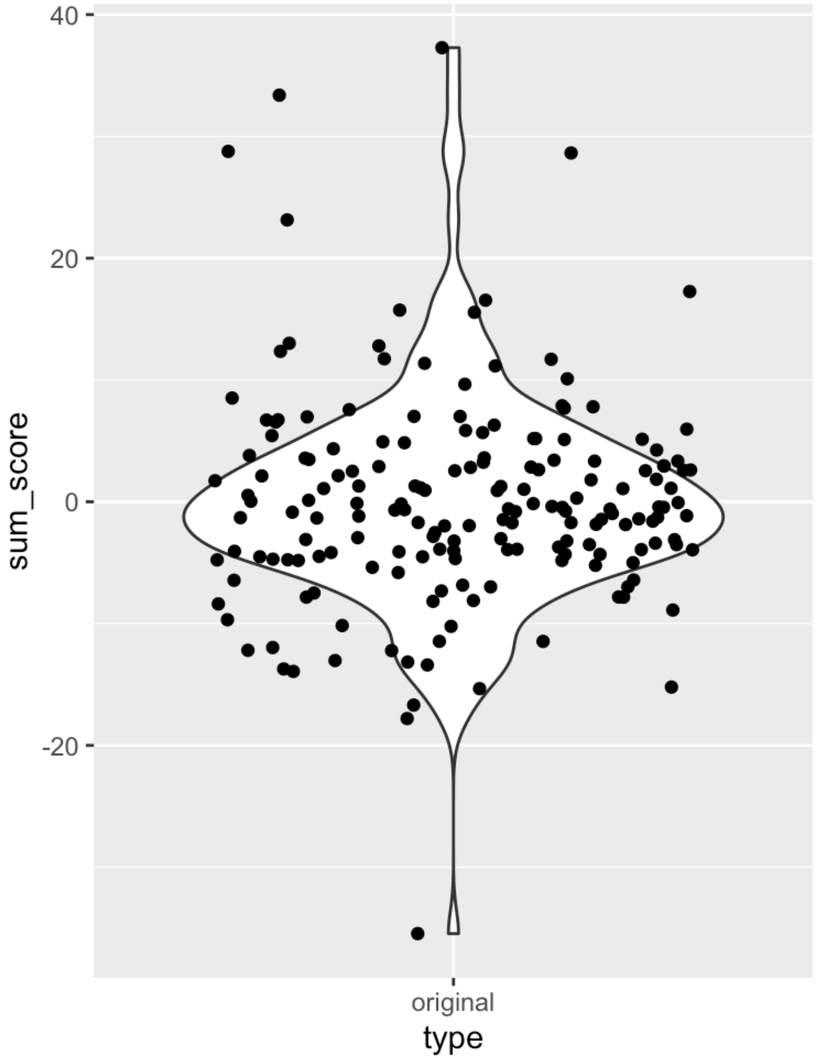
Because we simulate the data, we can sample more draws from distribution

```
126  niter <- 50
127  sim_list <- replicate(niter, {
128    makeClusterRanges(chrom, rng_big, n=300, lambda=5, seqlens)
129  })
130  sim_long <- bind_ranges(sim_list, .id="iter")
```

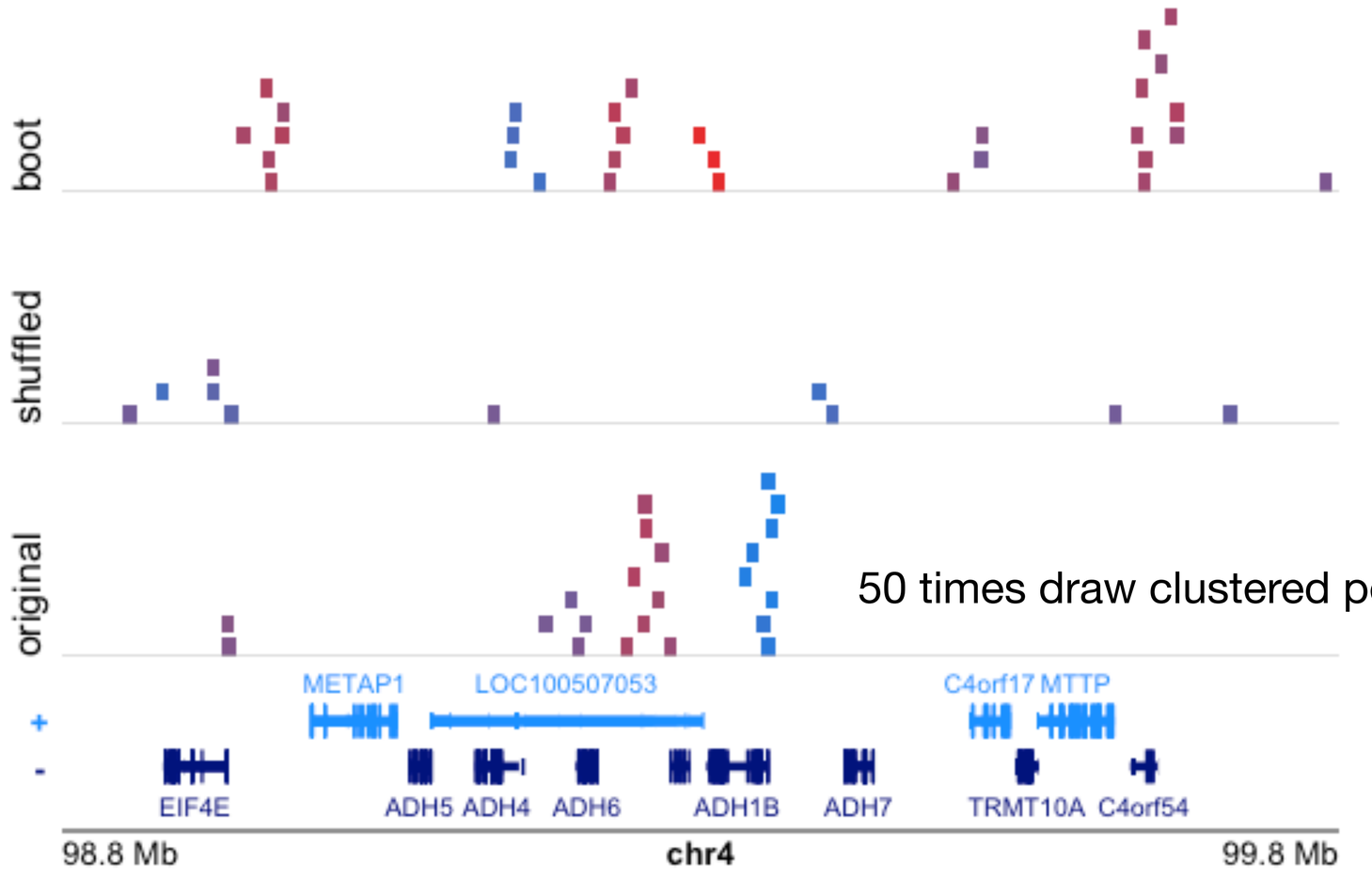
```
132  g %>%
133    join_overlap_inner(sim_long) %>%
134    mutate(type = "original") %>%
135    group_by(symbol, iter, type) %>%
136    summarize(sum_score = sum(score))
```

Adds an iteration column

“inner” removes no overlaps



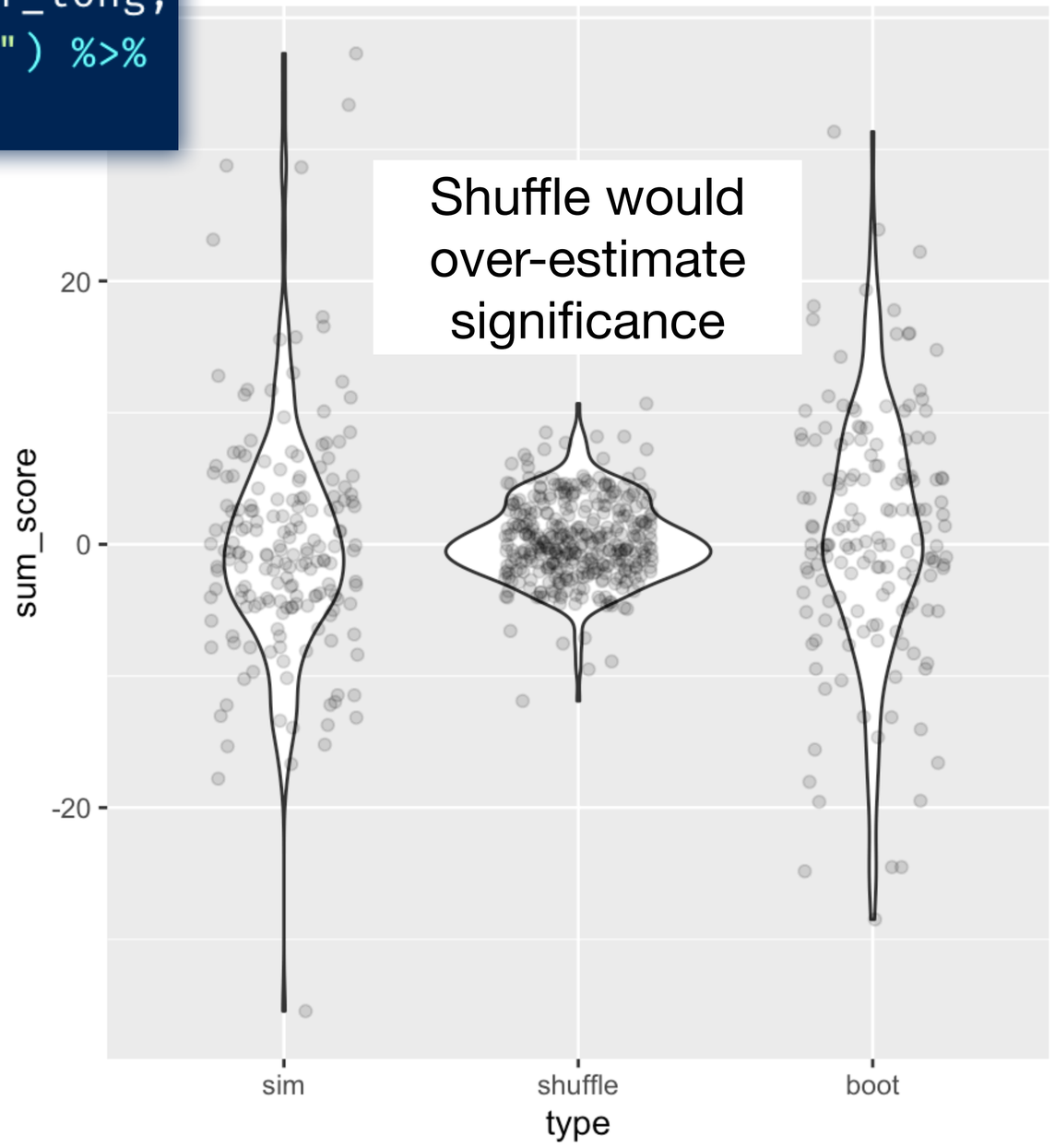
50 times each from original



50 times draw clustered peaks

```r
all <- bind_ranges(sim=sim_long, shuffle=shuf_long,
                   boot=boot_long, .id="type") %>%
  mutate(type = factor(type, levels=lvls))
```

```r
g %>%
  join_overlap_inner(all) %>%
  group_by(symbol, iter, type) %>%
  summarize(sum_score = sum(score)) %>%
  as_tibble() %>%
  ggplot(aes(type, sum_score)) +
  geom_violin() +
  geom_jitter(width=.25, alpha=.15)
```



What is missing?

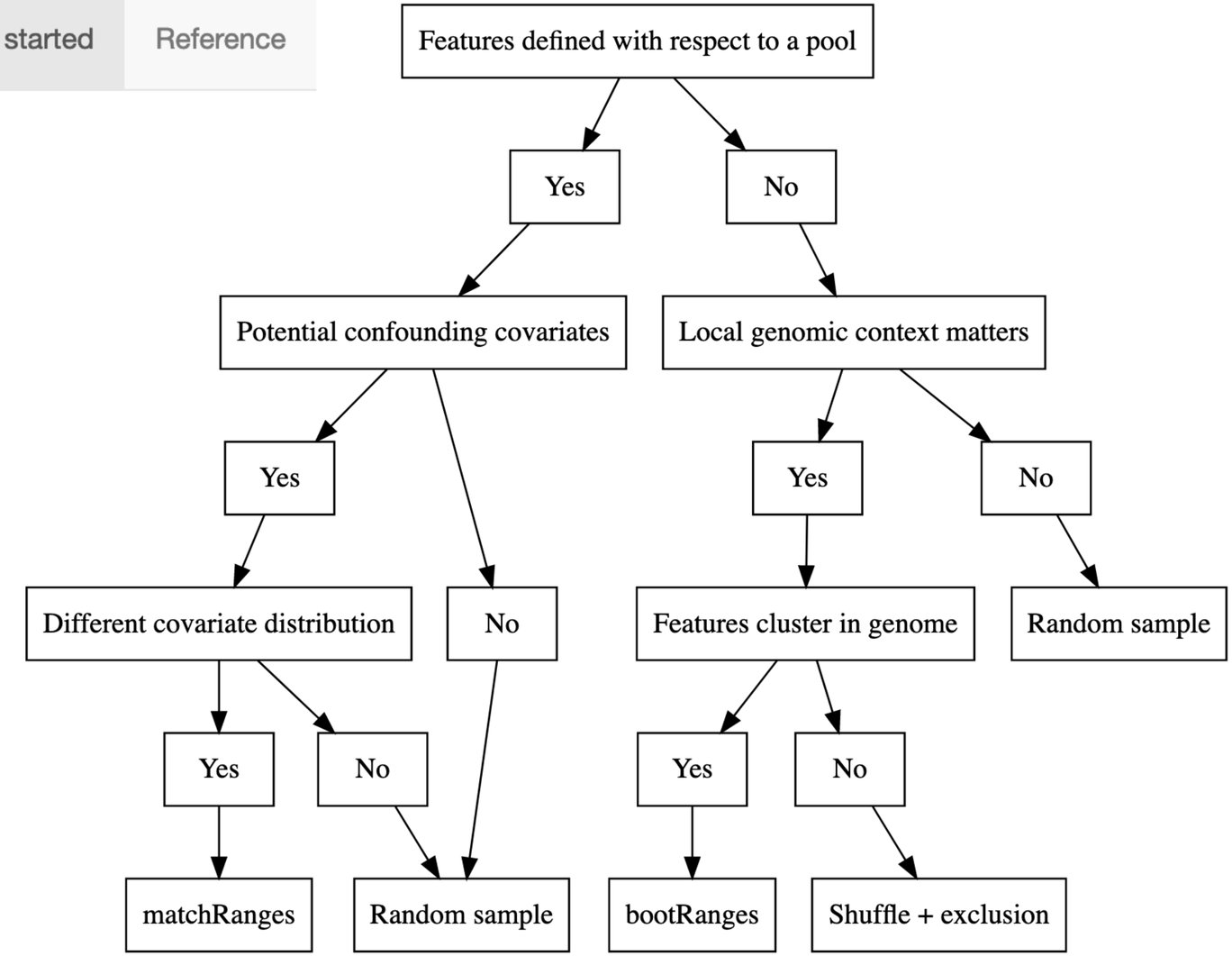
```r
tidyr::complete(
  symbol, iter, type
  fill=list(sum_score = 0)
)
```

Review mechanics of bootstrapping

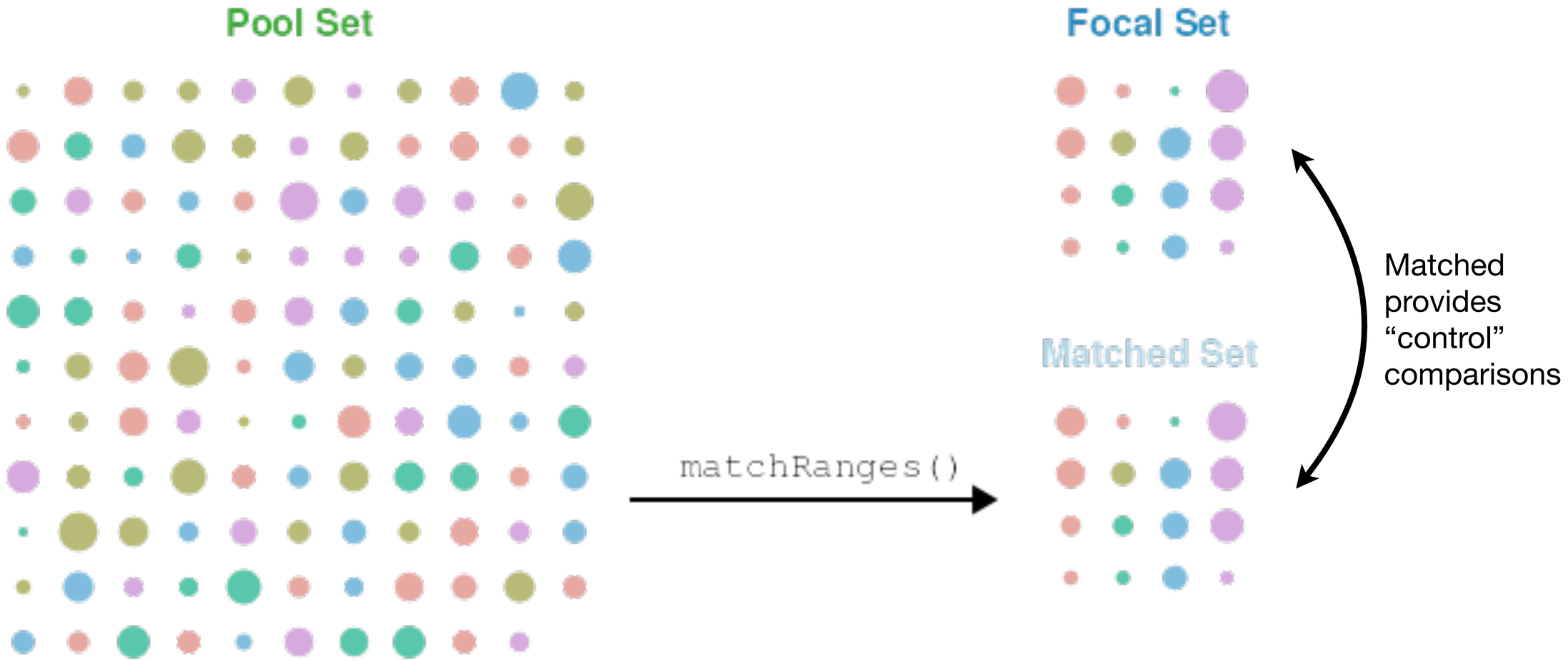
- Avoid copy/paste code with `bind_ranges()` specifying `.id`
  - Alternatively set observed to `iter=0`
- A single `join_overlap`
- Followed by `group_by(iter)` and `summarize()` and `complete()`

Which null generating method to choose?

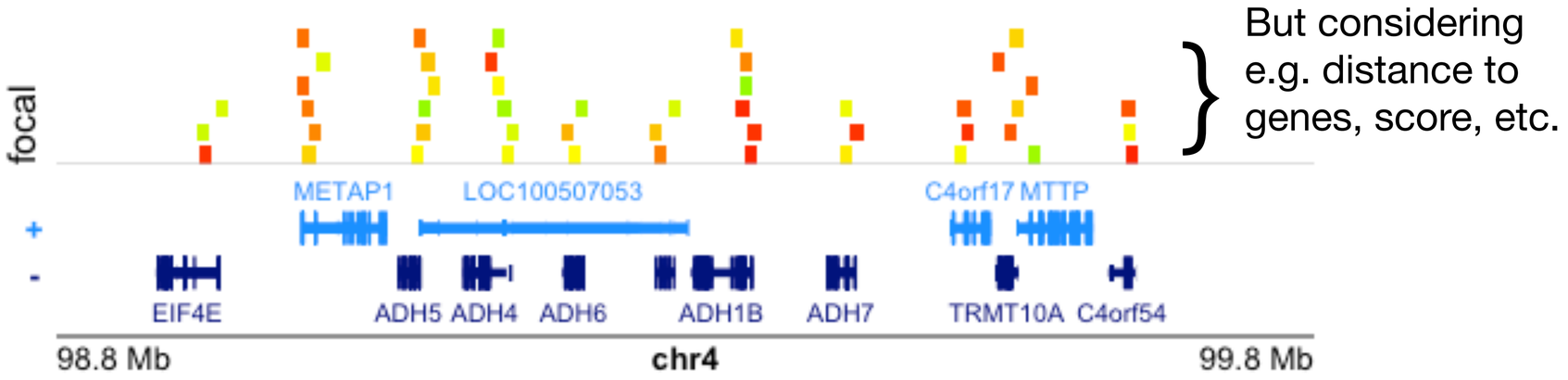
nullranges 1.5.5 | Get started | Reference



Quick introduction to matching



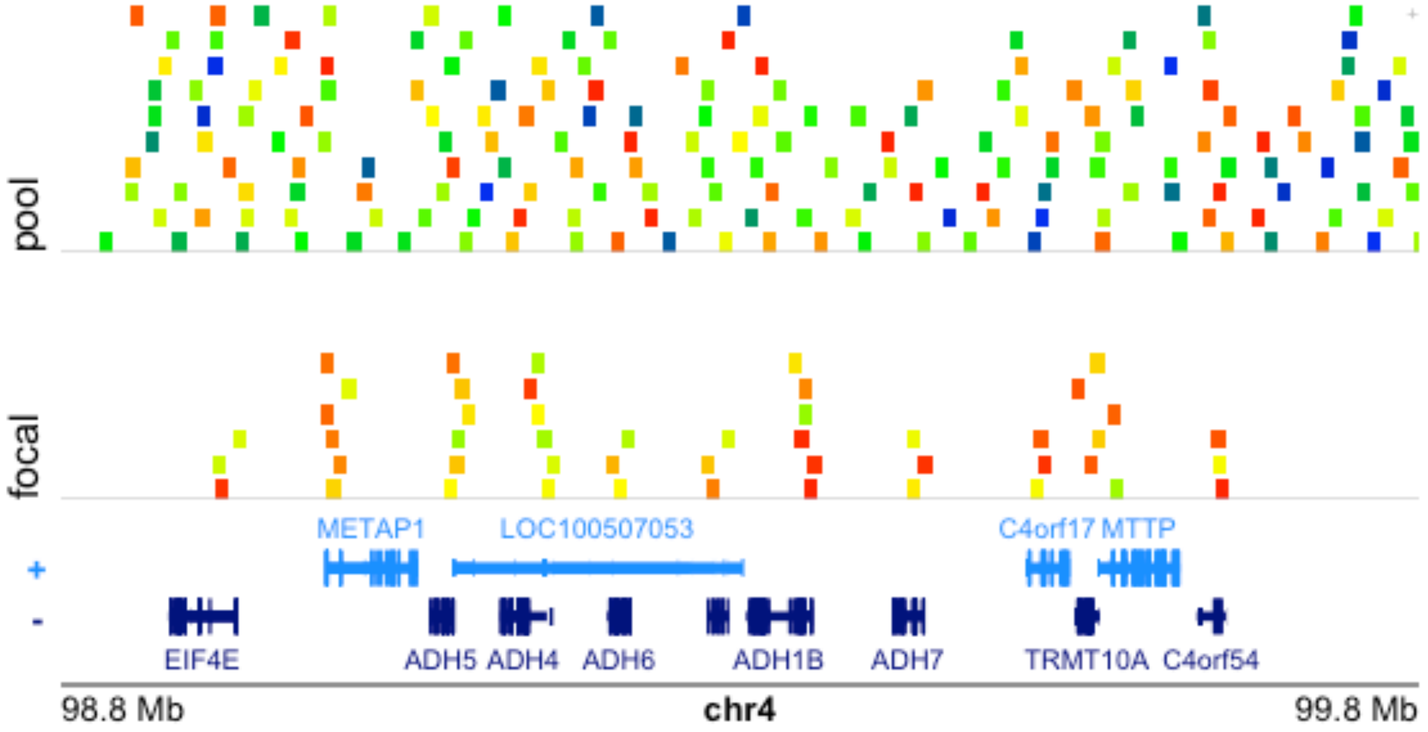
Want: compare focal set to some background ranges, in terms of internal properties, or overlap



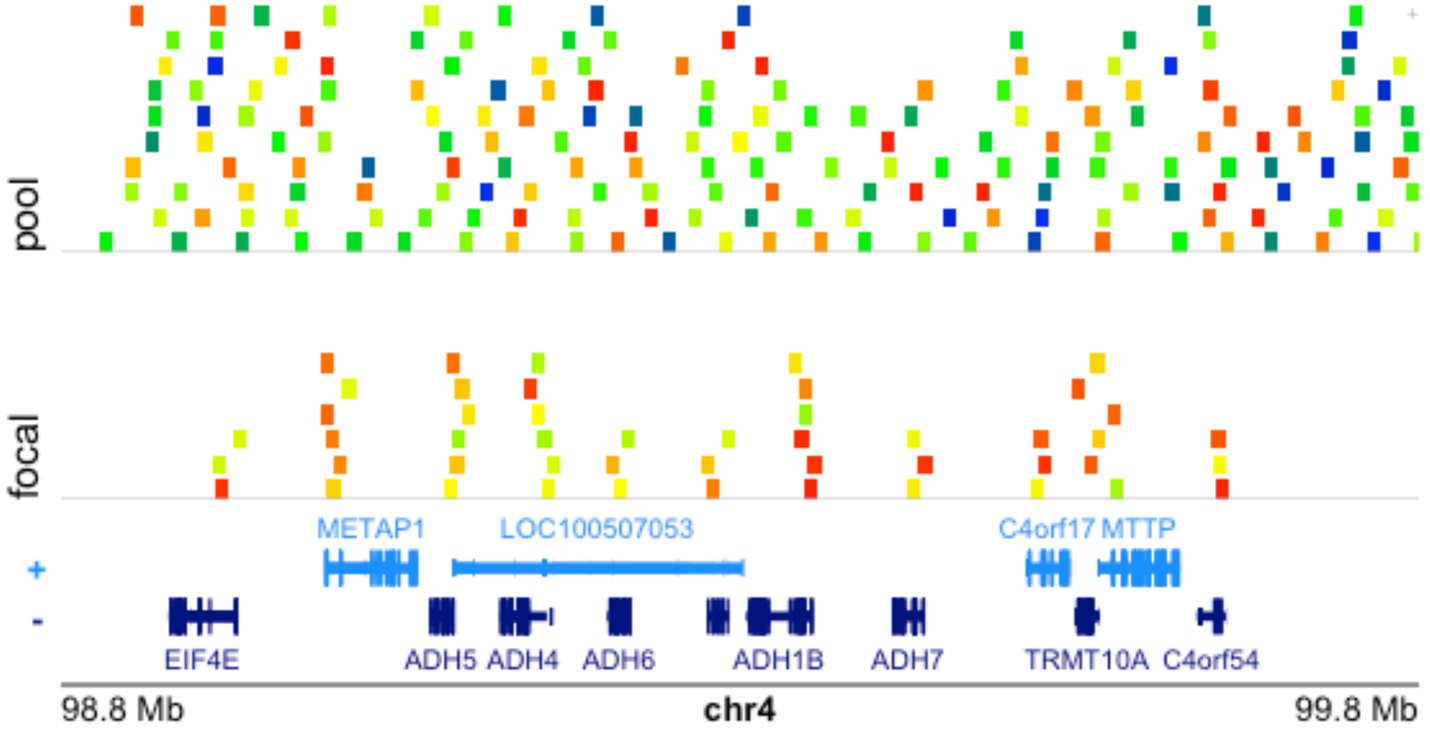
But considering e.g. distance to genes, score, etc.

Suppose a much larger pool to select from

```
205  # add another feature: distance to nearest TSS
206  tss <- g %>% anchor_5p() %>% mutate(width=1)
207
208  both <- bind_ranges(focal = focal, pool = pool, .id="type") %>%
209    add_nearest_distance(tss) %>%
210    mutate(log10dist = log10(distance + 1000))
```

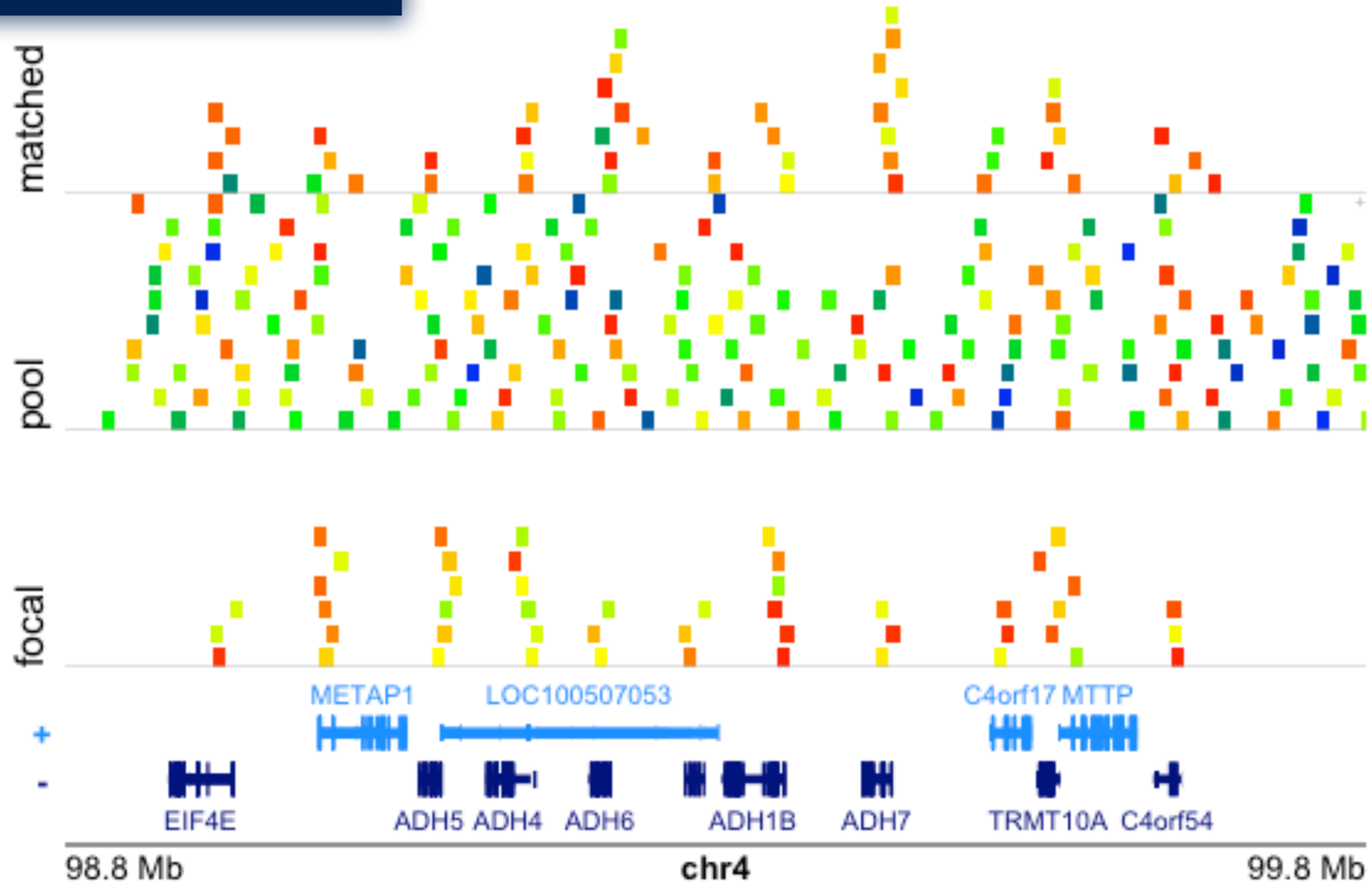


```
m <- both %>% {
   matchRanges(filter(., type=="focal"),
               filter(., type=="pool"),
               covar=~score + log10dist,
               method="nearest", replace=TRUE)
}
```

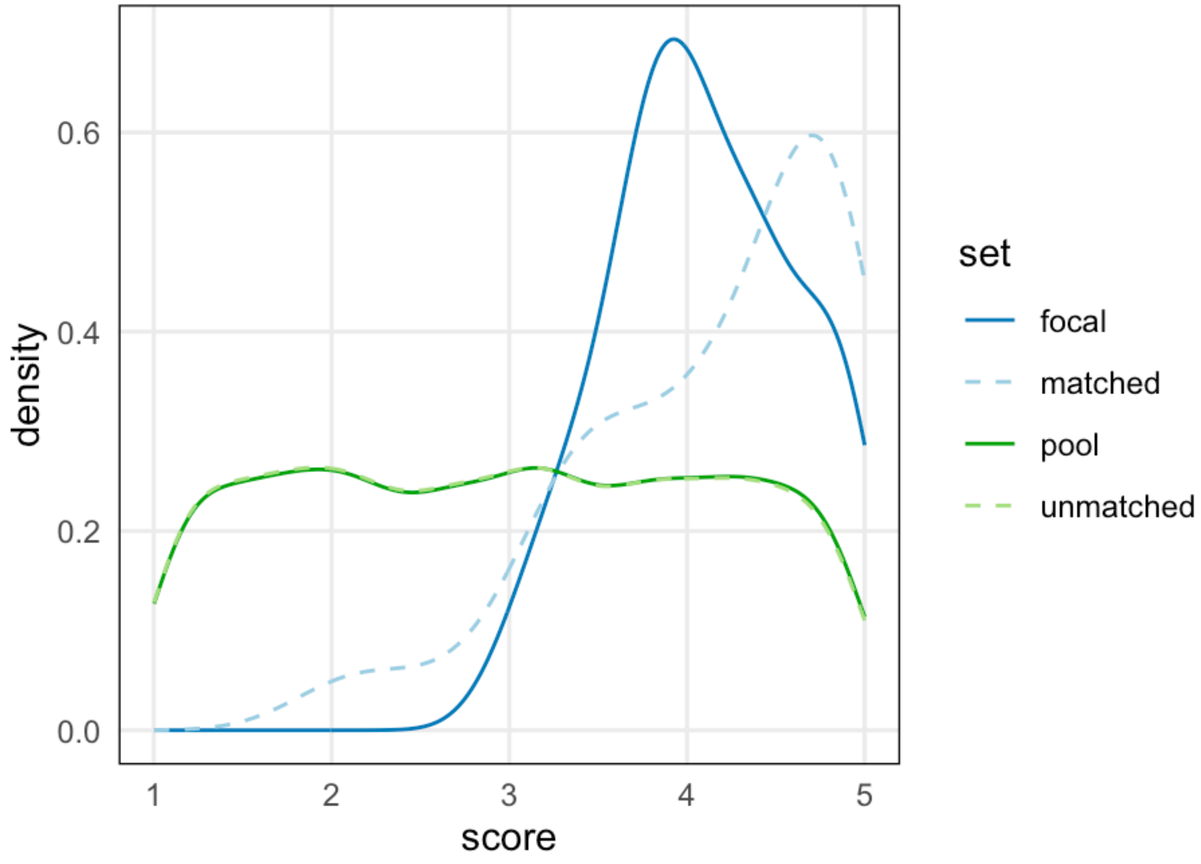

pool
focal
METAP1
LOC100507053
C4orf17
MTTP
+
-
EIF4E
ADH5
ADH4
ADH6
ADH1B
ADH7
TRMT10A
C4orf54
98.8 Mb
chr4
99.8 Mb

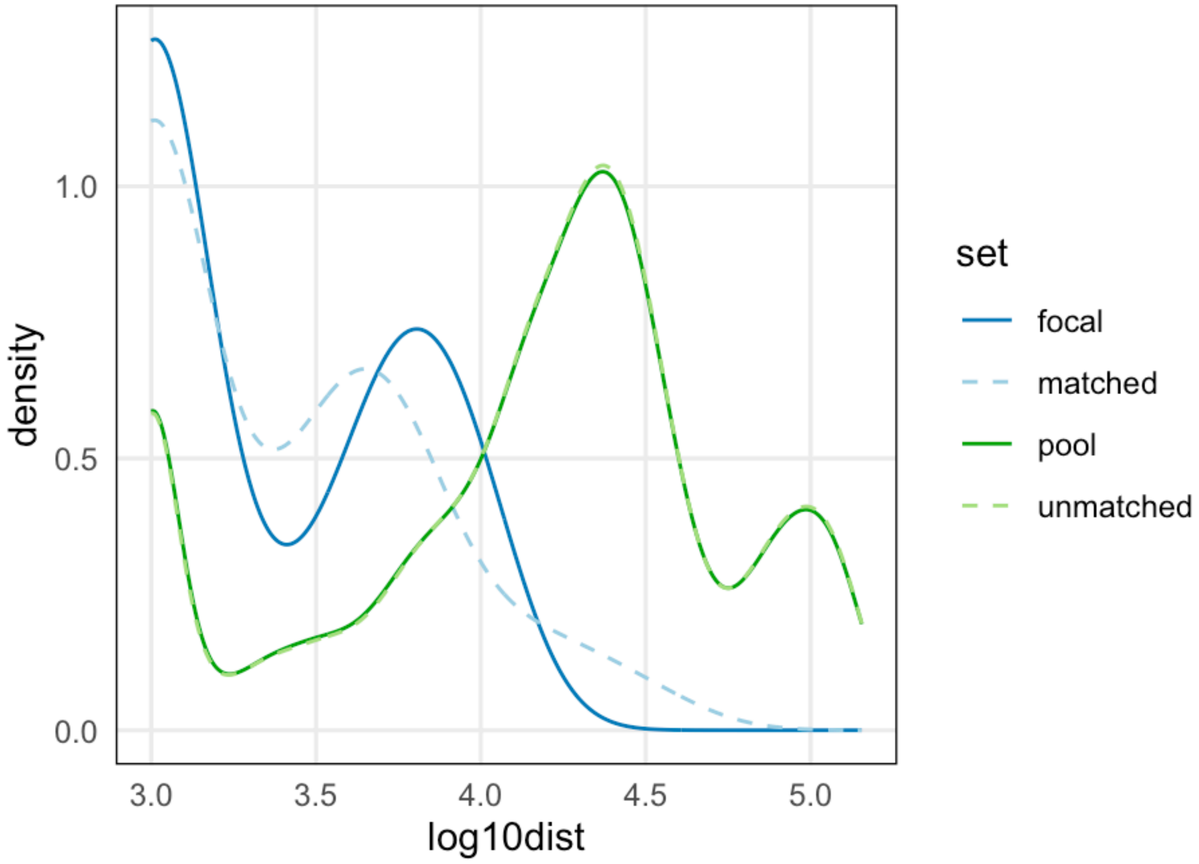
```r
m <- both %>% {
   matchRanges(filter(., type=="focal"),
               filter(., type=="pool"),
               covar=~score + log10dist,
               method="nearest", replace=TRUE)
}
```

method = c("nearest", "rejection", "stratified")



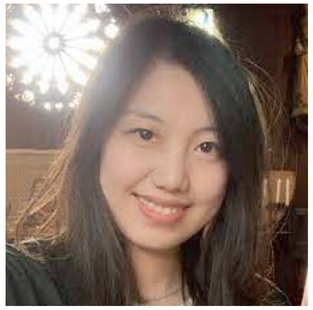
Bigger the pool, better covariate balance






```
plotCovariate(m, covar="score") +
  plotCovariate(m, covar="log10dist")
```

***nullranges*: `matchRanges()` and `bootRanges()`**


Eric Davis, Wancen Mu,
Doug Phanstiel and myself



Mikhail Dozmorov, Stuart Lee, Tim Triche,
others from **#nullranges** on Bioc Slack

nullranges.github.io/nullranges/
tidyomics.github.io/tidy-ranges-tutorial/

***matchRanges*: Generating null hypothesis genomic ranges via covariate-matched sampling**

Eric S. Davis[1], Wancen Mu[2], Stuart Lee[3], Mikhail G. Dozmorov[4,5], Michael I. Love[2,6*], Douglas H. Phanstiel[1,7,8,9,10*]

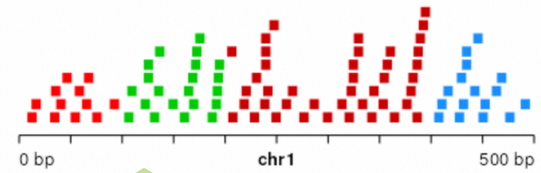
Availability and implementation
https://nullranges.github.io/nullranges

Deriving biological insights from genomic data commonly requires comparing attributes of selected genomic loci to a null set of loci. The selection of this null set is non trivial, as it requires careful consideration of potential covariates, a problem that is exacerbated by the non-uniform distribution of genomic features including genes, enhancers, and transcription factor binding sites. Propensity score-based covariate matching methods allow selection of null sets from a pool of possible items while controlling for multiple covariates; however, existing packages do not operate on genomic data classes and can be slow for large data sets making them difficult to integrate into genomic workflows. To address this, we developed *matchRanges*, a propensity score-based covariate matching method for the efficient and convenient generation of matched null ranges from a set of background ranges within the Bioconductor framework.

bootRanges: Flexible generation of null sets of genomic ranges for hypothesis testing

Wancen Mu[1], Eric Davis[2], Stuart Lee[5], Mikhail Dozmorov[6], Douglas H. Phanstiel[2,3], and Michael I. Love [*1,4]

[1]Department of Biostatistics,
[2]Curriculum in Bioinformatics and Computational Biology,
[3]Thurston Arthritis Research Center, Department of Cell Biology & Physiology, Lineberger Comprehensive Cancer Center, Curriculum in Genetics & Molecular Biology, and
[4]Department of Genetics, University of North Carolina-Chapel Hill, NC 27599
[5]Genentech, South San Francisco, CA, USA
[6]Department of Biostatistics, Department of Pathology, Virginia Commonwealth University, Richmond, VA 23298, USA



Can visualize null range set with
plotgardener, by Nicole Kramer, *et al.*

Funded by CZI EOSS,
NIH: R35-GM128645,
R01-HG009937,
T32-GM067553