Shazam: Quantifing selection pressures

Namita Gupta

2016-02-20

Contents

BASELINe quantifies selection pressure by calculating the posterior probability density function (PDF) based on observed mutations compared to expected mutation rates derived from an underlying SHM targeting model. Selection is quantified via the following steps:

1. Load a Change-O tab-delimited database file.
2. Calculate the selection scores and group by relevant fields for comparison.
3. Plot and compare selection scores of different groups of sequences.

Load Change-O data

A small example Change-O tab-delimited database file is included in the `shazam` package. The example dataset consists of a subset of Ig sequencing data from an influenza vaccination study (Laserson and Vigneault et al., PNAS, 2014). The data include sequences from multiple time-points before and after the subject received an influenza vaccination. Quantifying selection requires the following fields (columns) to be present in the Change-O file: `SEQUENCE_IMGT` and `GERMLINE_IMGT_D_MASK`.

```
# Set example data
library(shazam)
db <- InfluenzaDb
```

Calculate selection scores and group by relevant fields for comparison

The first step to calculate selection is to call `calcDBClonalConsensus` to collapse sequences by the `CLONE` column, if the data has been divided into clonal groups.

```
library(shazam)

# Collapse clonal groups into single sequences
db_clone <- collapseByClone(db, regionDefinition=IMGT_V_NO_CDR3,
                            expandedDb=TRUE, nproc=1)
```

Following construction of an effective sequence for each clone, the observed and expected mutation counts are calculated for each sequence of the `CLONAL_SEQUENCE` column adding in the previous step. `calcDBObservedMutations` is used to calculate the number of observed mutations and `calcDBExpectedMutations` calculates the expected frequency of mutations. The underlying targeting model for calculating expectation can be specified using the `targetingModel` parameter. In the example below, the default `HS5FModel` is used. Column names for sequence and germline sequence may also be passed in as parameters if they differ from Change-O defaults.

Mutations are counted by these functions separately for complementarity determining (CDR) and framework (FWR) regions. If Ig sequences are not gapped according to IMGT numbering, the region boundaries may be defined using the `regionDefinition` parameter. There are four built-in region definitions in the `shazam` package:

- `IMGT_V` is all regions in the V-segment grouped as either CDR or FWR.
- `IMGT_V_BY_REGIONS` is all CDR and FWR regions in the V-segment treated as individual regions.
- `IMGT_V_NO_CDR3` (the default value of `regionDefinition`) is all regions in the V-segment, excluding CDR3, grouped as either CDR or FWR.
- `IMGT_V_BY_REGIONS_NO_CDR3` is all regions in the V-segment, excluding CDR3, treated as individual regions.

```r
# Count observed mutations and add OBSERVED columns to db_clone
db_obs <- calcDBObservedMutations(db_clone,
                                  sequenceColumn="CLONAL_SEQUENCE",
                                  regionDefinition=IMGT_V_NO_CDR3, nproc=1)
# Calculate expected mutations and add EXPECTED columns to db_obs
db_exp <- calcDBExpectedMutations(db_obs,
                                  sequenceColumn="CLONAL_SEQUENCE",
                                  targetingModel=HS5FModel,
                                  regionDefinition=IMGT_V_NO_CDR3, nproc=1)
```

The counts of observed and expected mutations can be combined to test for selection using `calcBaseline`. The statistical framework used to test for selection based on mutation counts can be specified using the `testStatistic` parameter. If the previous three functions have not yet been called on the input `data.frame` (as demonstrated by having columns beginning with `OBSERVED` and `EXPECTED`), this function will calculate observed and expected mutations after collapsing by the `CLONE` column.

```r
# Calculate selection scores using the output from calcDBExpectedMutations
baseline <- calcBaseline(db_exp, testStatistic="focused",
                         regionDefinition=IMGT_V_NO_CDR3,
                         nproc=1)

# Subset the original data to two time-points and switched isotypes
db_sub <- subset(db, CPRIMER %in% c("IGHA", "IGHG") &
                     BARCODE %in% c("RL013", "RL018"))
# Calculate selection scores from scratch on subset
```

```
baseline_sub <- calcBaseline(db_sub, testStatistic="focused",
                             regionDefinition=IMGT_V_NO_CDR3, nproc=1)
```

Selection scores can be used to compare selection pressures between different samples. In the example influenza dataset, the `BARCODE` field corresponds to samples taken at different timepoints before and after an influenza vaccination. Additionally, the `CPRIMER` field specifies the isotype of the sequence. The `groupBaseline` function convolves the Baseline PDFs of individual sequences (or clones) to get a combined PDF. The field(s) by which to group the sequences are specified with the `groupBy` parameter. The `groupBaseline` function automatically calls `summarizeBaseline` to generate summary statistics based on the requested groupings, and populates the `stats` slot of the input `Baseline` object with the number of sequences with observed mutations for each region, selection scores, 95% confidence intervals, and p-values for each grouping.

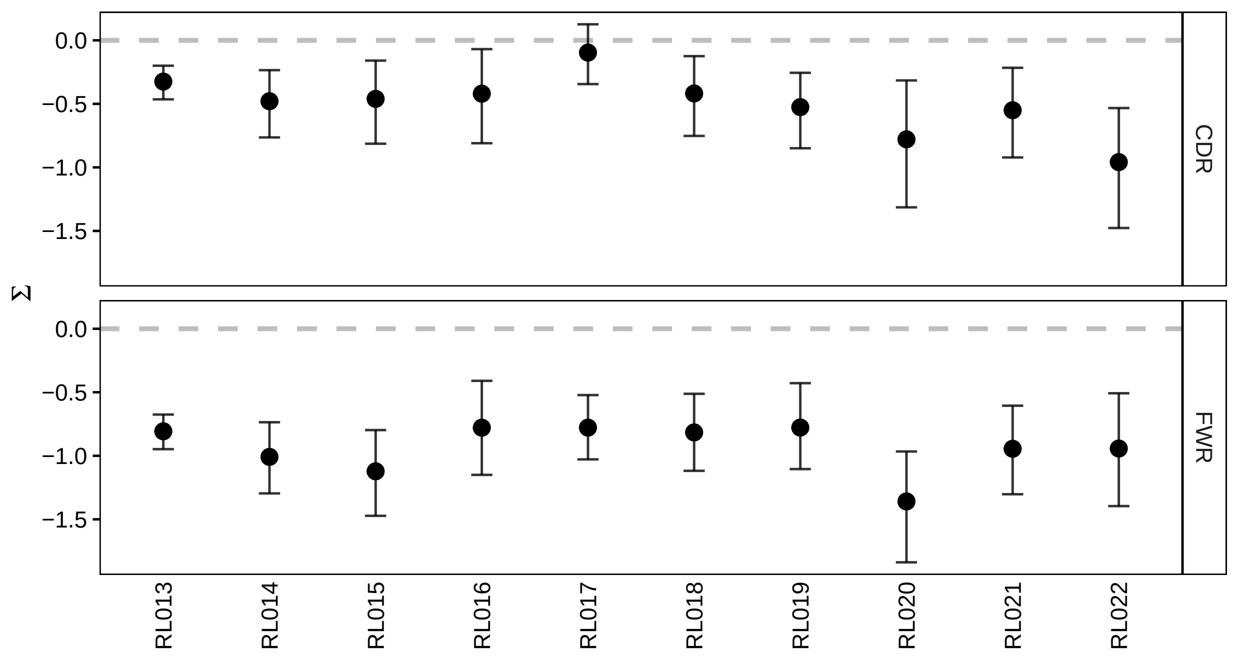
```
# Combine selection scores by time-point
baseline_one <- groupBaseline(baseline, groupBy=c("BARCODE"))

# Combine selection scores by time-point and isotype
baseline_two <- groupBaseline(baseline, groupBy=c("BARCODE", "CPRIMER"))
baseline_sub <- groupBaseline(baseline_sub, groupBy=c("BARCODE", "CPRIMER"))
```

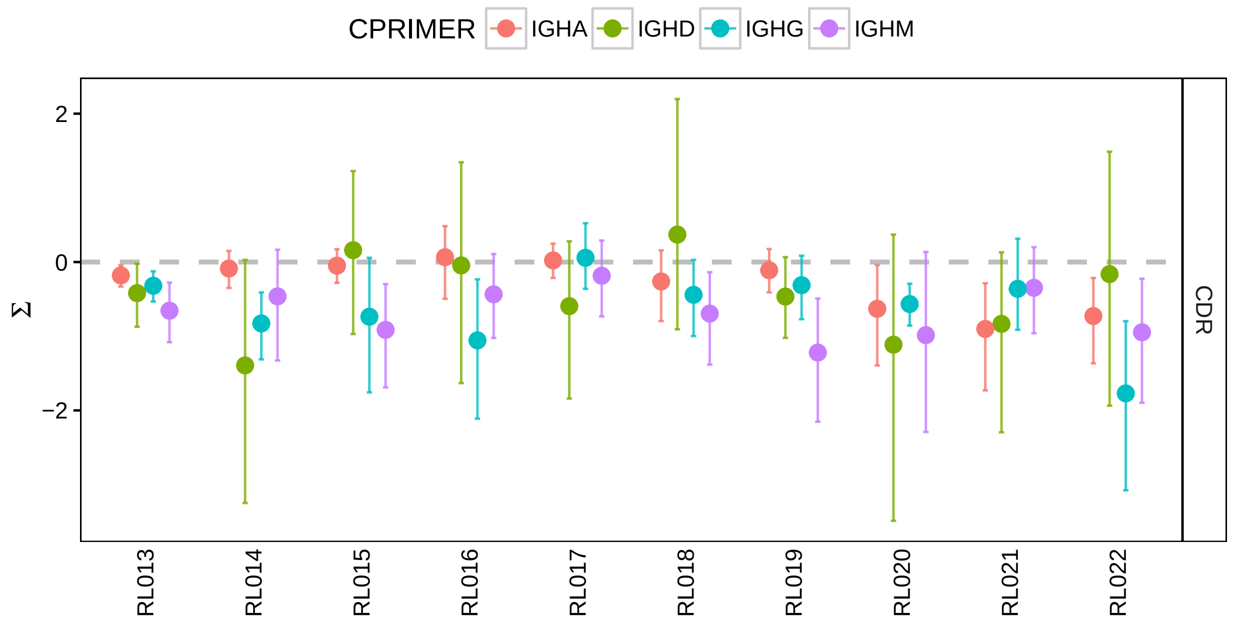
Plot and compare selection scores of different groups of sequences

`plotBaselineSummary` plots the mean and confidence interval of selection scores for the given groups. The `idColumn` argument specifies the field that contains identifiers of the groups of sequences. If there is a secondary field by which the sequences are grouped, this can be specified using the `groupColumn`. This secondary grouping can have a user-defined color palette passed into `groupColors` or can be separated into facets by setting the `facetBy="group"`. The `subsetRegions` argument can be used to visualize selection of specific regions. Several examples utilizing these different parameters are provided below.

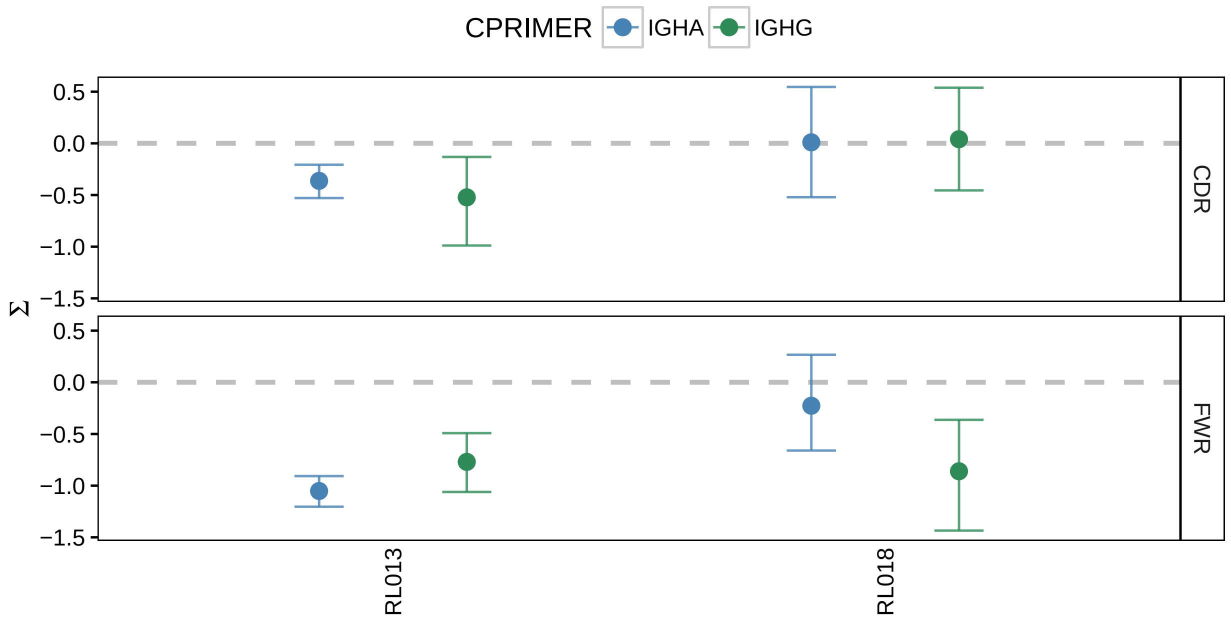
```
# Plot mean and confidence interval by time-point
plotBaselineSummary(baseline_one, "BARCODE")
```



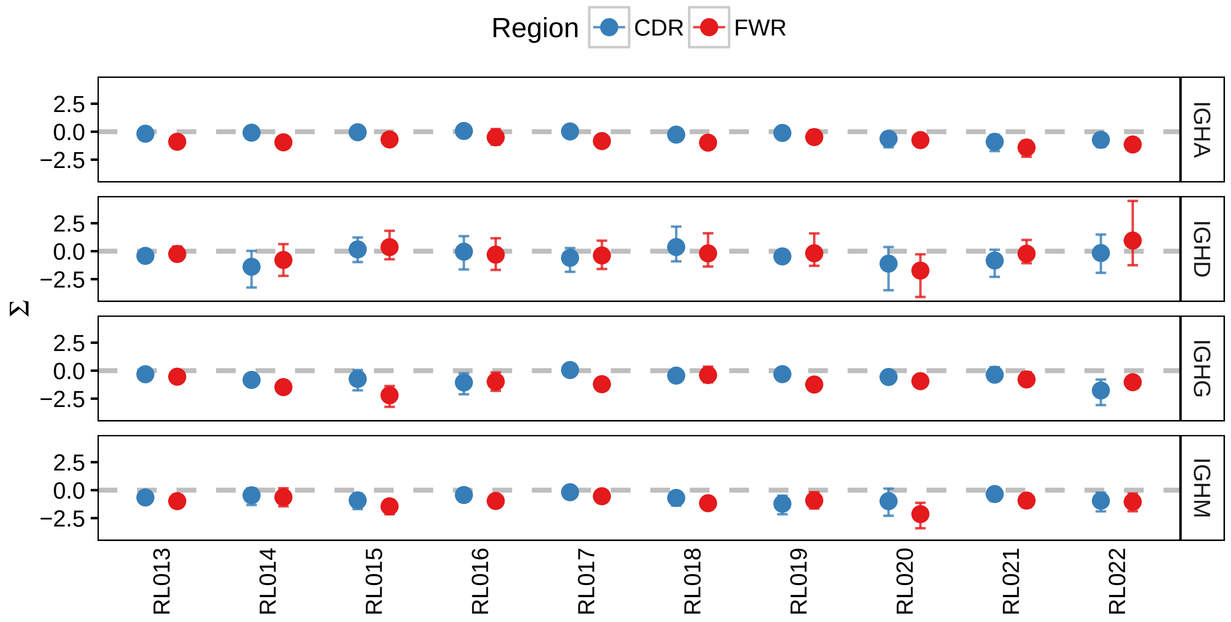
```
# Plot selection scores by time-point and isotype for only CDR
plotBaselineSummary(baseline_two, "BARCODE", "CPRIMER", subsetRegions="CDR")
```



```
# Plot only two time-points and recolor isotypes
group_colors <- c("IGHM"="darkorchid", "IGHD"="firebrick",
                  "IGHG"="seagreen", "IGHA"="steelblue")
plotBaselineSummary(baseline_sub, "BARCODE", "CPRIMER", groupColors=group_colors)
```



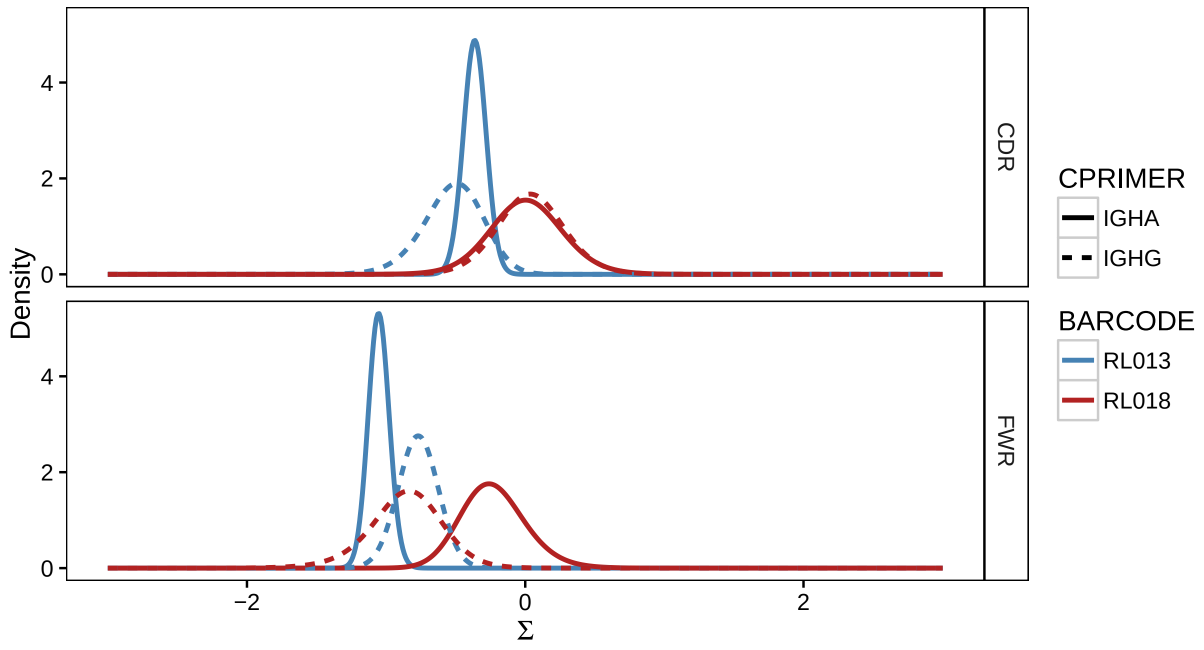
```
# Group by CDR/FWR and facet by isotype
plotBaselineSummary(baseline_two, "BARCODE", "CPRIMER", facetBy="group")
```



`plotBaselineDensity` plots the full `Baseline` PDF of selection scores for the given groups. The parameters are the same as those for `plotBaselineSummary`. However, rather than plotting the mean and confidence interval, the full density function is shown.

```
# Plot selection PDFs for a subset of the data
group_colors <- c("RL013"="steelblue", "RL018"="firebrick")
```

```
plotBaselineDensity(baseline_sub, "CPRIMER", "BARCODE",
                    groupColors=group_colors, sigmaLimits=c(-3, 3))
```


Density
CDR
FWR
Σ
CPRIMER
IGHA
IGHG
BARCODE
RL013
RL018