Shazam: Inferring SHM targeting models

Namita Gupta

2017-06-09

Contents

The targeting model is the background likelihood of a particular mutation, based on the surrounding sequence context as well as the mutation itself. The model is inferred from observed mutations in the data. The model can then be transformed into a distance function to compare Ig sequences of a given dataset based on the likelihood of the observed mutations. This is done via the following steps:

1. Infer a substitution model, which is the likelihood of a base mutating to each other base given the microsequence context.
2. Infer a mutability model, which is likelihood of a given base being mutated given the microsequence context and substitution model.
3. Visualize the mutability model to identify hot and cold spots.
4. Calculate a nucleotide distance matrix based on the underlying SHM models.

Example data

A small example Change-O database is included in the `shazam` package. Inferring a targeting model requires the following fields (columns) to be present in the Change-O database:

- `SEQUENCE_ID`
- `SEQUENCE_IMGT`
- `GERMLINE_IMGT_D_MASK`
- `V_CALL`

```
# Load example data
library(shazam)
data(ExampleDb, package="alakazam")
```

Infer targeting model (substitution and mutability)

The function for inferring substitution rates (`createSubstitutionMatrix`) counts the number of mutations from a given base to all others occurring in the center position for all 5-mer motifs in the dataset. The `model` argument of `createSubstitutionMatrix` allows the user to specify whether to count all mutations, or just silent mutations to infer the model. Column names for the sample sequence, germline sequence, and V call can also be passed in as parameters if they differ from Change-O defaults. Additionally, the `multipleMutation` parameter determines handling of multiple

mutations in a single 5-mer: `independent` treats each mutation independently and `ignore` entirely disregards 5-mers with multiple mutations.

The function for inferring a mutability model (`createMutabilityMatrix`) counts the number of mutations in all 5-mer motifs of the dataset, and depends upon the inferred substitution rates. Furthermore, the same parameters available for inferring the substitution rates are also available to adjust this function.

The inferred substitution and mutability matrices returned by the above functions only account for unambiguous 5-mers. However, there may be cases in which the user may need the likelihood of a mutation in a 5-mer with ambiguous characters. Each of the above functions has a corresponding function (`extendSubstitutionMatrix` and `extendMutabilityMatrix`) to extend the matrix to infer 5-mers with Ns by averaging over all corresponding unambiguous 5-mers.

These extended substitution and mutability matrices can be used to create an overall SHM targeting matrix (`createTargetingMatrix`), which is the combined probability of mutability and substitution.

```
# Create substitution model using silent mutations
sub_matrix <- createSubstitutionMatrix(ExampleDb, model="S")
# Create mutability model using silent mutations
mut_matrix <- createMutabilityMatrix(ExampleDb, sub_matrix, model="S")

# Extend models to include ambiguous 5-mers
sub_matrix <- extendSubstitutionMatrix(sub_matrix)
mut_matrix <- extendMutabilityMatrix(mut_matrix)

# Create targeting model matrix from substitution and mutability matrices
tar_matrix <- createTargetingMatrix(sub_matrix, mut_matrix)
```

All of the above steps can be combined by using the single function `createTargetingModel` to infer a `TargetingModel` object directly from the dataset. Additionally, it is generally appropriate to consider the mutations within a clone only once. Consensus sequences for each clone can be generated using the `collapseClones` function.

```
# Collapse sequences into clonal consensus
clone_db <- collapseClones(ExampleDb, nproc=1)
```

```
## When both includeAmbiguous and breakTiesStochastic are FALSE, ties are broken in the order
```

```
## Collapsing clonal sequences...
```

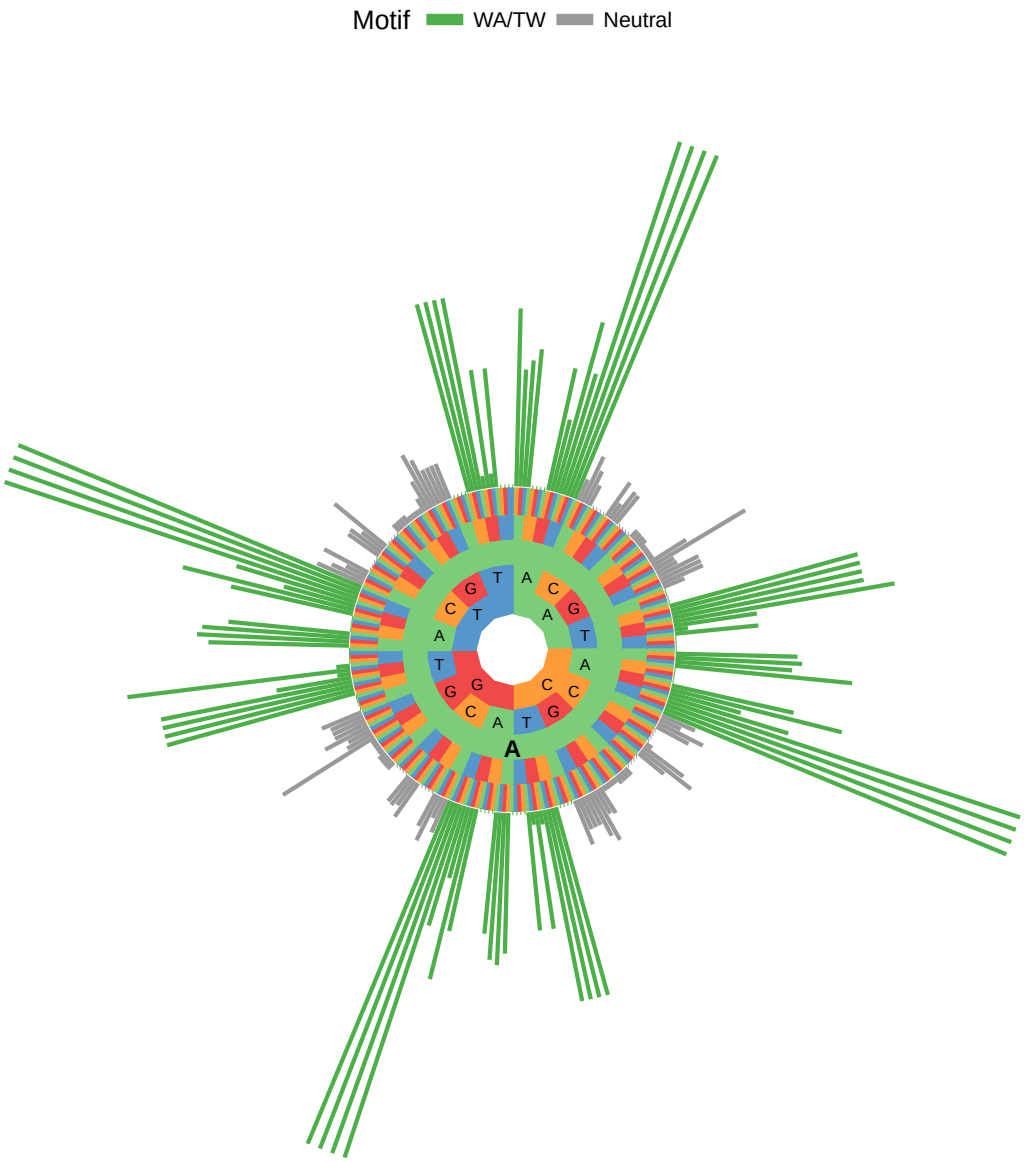
```
# Create targeting model in one step using only silent mutations
# Use consensus sequence input and germline columns
model <- createTargetingModel(clone_db, model="S", sequenceColumn="CLONAL_SEQUENCE",
                              germlineColumn="CLONAL_GERMLINE")
```

Visualize targeting model

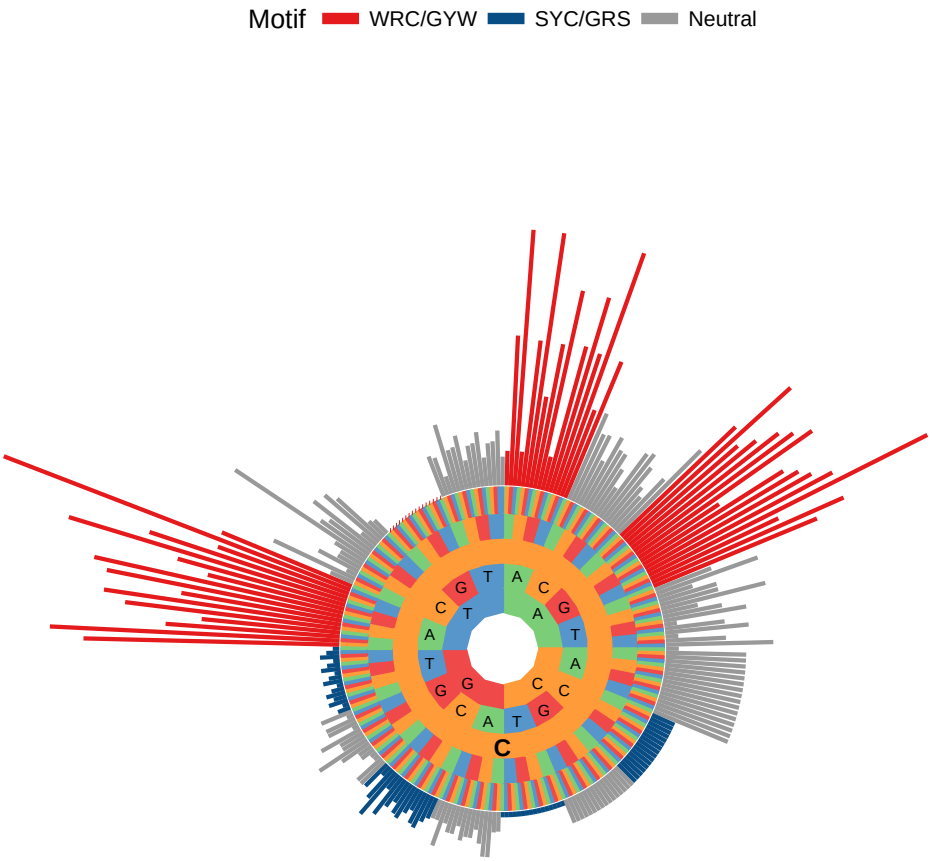
The visualization of a dataset's underlying SHM mutability model can be used to investigate hot and cold spot motifs. The length of the bars on the plot of mutability rates corresponds to the likelihood of a given base in the given 5-mer being mutated. The plotting function `plotMutability`

has an argument `style` to specify either a hedgehog plot (circlular) or barplot diplay of 5-mer mutability rates. If the mutability for only specific bases is required, this can be specified via the `nucleotides` argument.

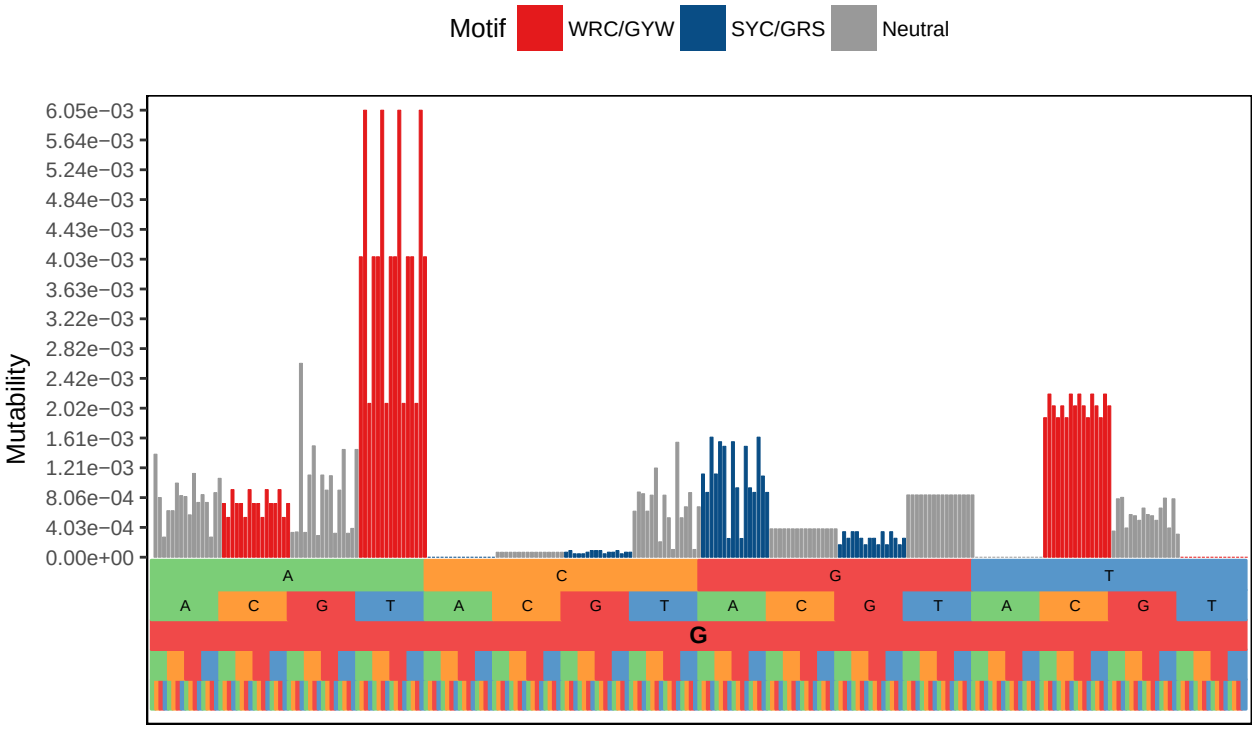
```
# Generate hedgehog plot of mutability model
plotMutability(model, nucleotides="A", style="hedgehog")
```



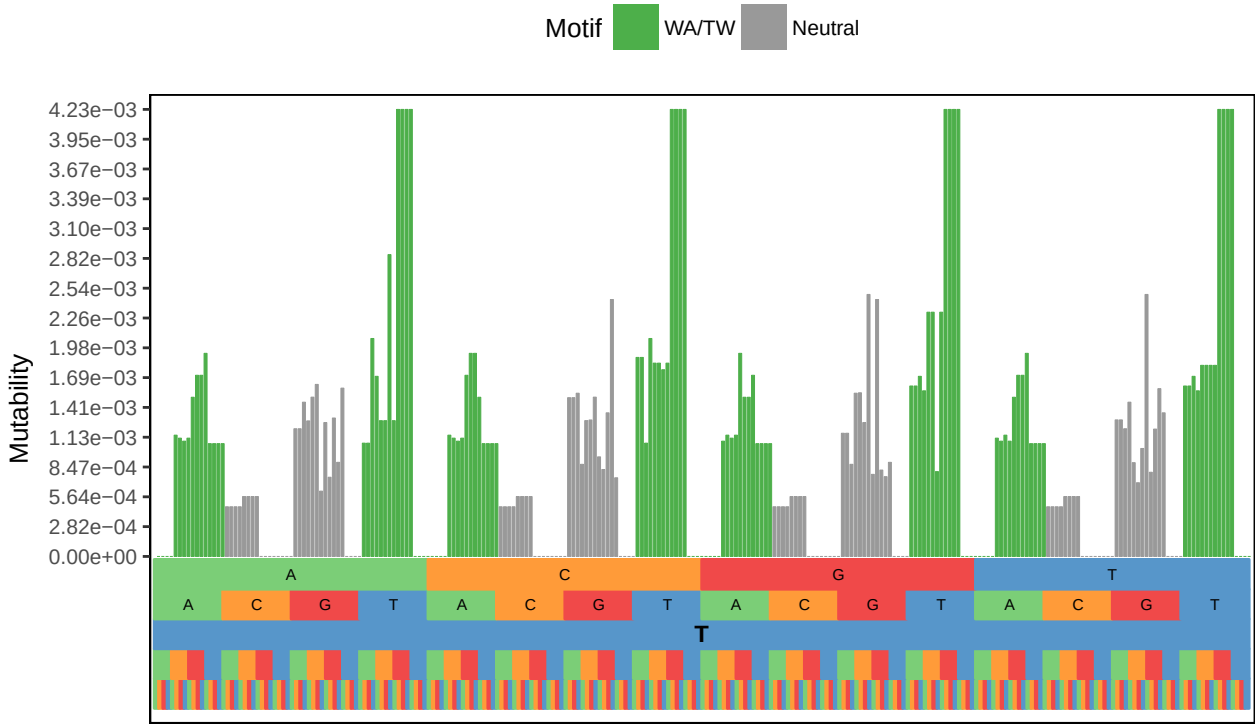

```
plotMutability(model, nucleotides="C", style="hedgehog")
```




```
# Generate bar plot of mutability model
plotMutability(model, nucleotides="G", style="bar")
```



```
plotMutability(model, nucleotides="T", style="bar")
```



Calculate targeting distance matrix

In the Change-O pipeline, the `hs5f` cloning method rely on an inferred targeting model. If users wish to use a targeting model inferred from their data to assign distance between sequences for clonal grouping, then the observed SHM targeting rates must be transformed into distances. The `calcTargetingDistance` function returns a matrix of distances between each 5-mer and each corresponding mutation of the center base. This matrix can also be generated and written directly to a file using the function `writeTargetingDistance`.

```
# Calculate distance matrix
dist <- calcTargetingDistance(model)
```