FeatureREDUCE v1.10

User Manual v1.10.0

Riley & Bussemaker Labs

7/13/2015

Table of Contents

1.0 Introduction

Protein binding microarray (PBM) technology has been used to probe the DNA binding specificity of hundreds (thousands?) of transcription factors from a variety of organisms. Existing algorithms for analyzing such data either assume independence between nucleotides within the binding site, or assign a binding score to all possible bound sequences. In each case, the results are suboptimal. FeatureREDUCE combines the advantages of both approaches, achieving quantification of relative binding affinity at an unprecedented level of accuracy. Accounting explicitly for considerable technology-specific biases enables us to thermodynamically model dependencies that exist between nucleotide positions. The resulting sequence-to-affinity models are the first to accurately estimate affinities from PBM data for binding motifs up to 10nt long. In addition, we introduce a simple metric to help assess the quality of the PBM data and the derived affinities.

FeatureREDUCE can produce up to three models in one for each PBM experiment:

Model 1 – The FSAM (Feature Specific Affinity Model)

FeatureREDUCE builds on the biophysical modeling framework of the MatrixREDUCE algorithm. In MatrixREDUCE, the DNA sequence specificity of a given transcription factor is represented as a position-specific affinity matrix (PSAM), which is directly related to the differences in binding free energy associated with point mutations in the DNA sequence. Under the assumption of independence between nucleotide positions, the PSAM coefficients are directly inferred from a set of high-throughput measurements (mRNA expression, ChIP fold-enrichment, PBM intensity, etc.) and their associated cis-regulatory sequences, which can be much longer than the length of a single binding site. In the PSAM biophysical model, the affinities of all possible binding sites within the longer sequence are added up, under the assumption that saturation of binding is weak to moderate.

FeatureREDUCE extends MatrixREDUCE in seven distinct ways. First, it uses a more refined representation of binding specificity, in which dependencies between nucleotides are detected and modeled explicitly using additional free energy parameters. The resulting FSAM (feature-specific affinity model) can be used to predict the relative binding affinity for any oligomer of a specified length. Second, it accounts for certain biases that are specific to the PBM technology. Third, it uses a robust gradient-descent method to find the highest-affinity k-mer to be used as the seed. Fourth, it has the ability to detect a symmetric motif (common when the TF binds as a homodimer) and then generate a more accurate and robust symmetric model (with about half as many parameters). Fifth, it can also solve the nonlinear saturation model which includes the free-protein concentration parameter [P] in the objective function of the protein-DNA binding reaction at equilibrium. Finally, FeatureREDUCE employs robust regression techniques, which prevents over-fitting and allows for improved estimation of biophysical parameters.

Model 2 – The All-Kmer Model

The All-Kmer model is similar in concept to the model by Annala et al., but with some notable improvements. We use a robust regression framework that resists over-fitting, and also take into account that not all K-mers are well represented on the HK and ME microarray designs. Unlike the original UPBM designs, on the HK, and ME, and later UPBM PBM designs K-mers that contain poly-Gs are highly under-represented or not existent at all in one of the orientations. Also, FeatureREDUCE maximizes the pearson correlation with the straight intensities, while the Annala et al model maximizes the pearson with the log of the intensities. Since our All-Kmer models have a significantly reduced correlation with MITOMI Kds and ChIP-seq occupancy compared to the FSAMs, we believe that the All-Kmer models are fitting mostly PBM artifacts, which we can use when predicting PBM probe intensities.

Model 3 – The Combined FSAM and All-Kmer Model

The combined FSAM and All K-mer Model optimally combines the FSAM with the All-Kmer model and performs best when predicting PBM probe intensities. It's able to model both protein-DNA occupancy (with the FSAM) and the PBM artifacts that are consistent across the HK and ME designs (with the All-Kmer model). The All-Kmer Model component is only used to help model PBM probe intensities, and is disabled when predicting the affinity of genomic DNA.

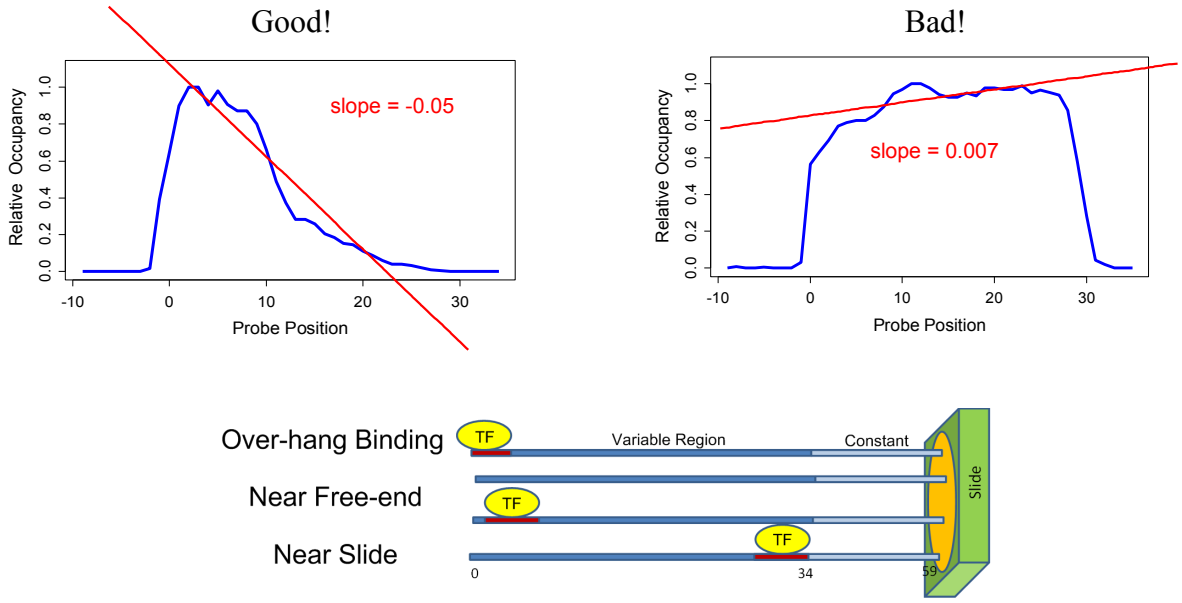
1.1 Components of an FSAM

PSAM (position-specific affinity matrix) – A biophysical positional-independence model that assumes that each nucleotide position within the footprint contributes independently to the binding strength, in which relative affinity parameters for individual nucleotide positions are multiplied to obtain the overall affinity. Thus, a PSAM is a numerical matrix of nucleotide affinities with one row for each nucleotide and one column for each position in the binding site, and the affinities are normalized so that the optimal-binding nucleotide has an affinity coefficient of 1. A PSAM is similar to a position weight matrix (PWM), which is widely used to discriminate putative binding sites from "background sequence" in a classification-based approach to sequence specificity.

Positional Bias Profile - Steric hindrance by the "carpet" of neighboring DNA molecules immobilized at each PBM spot can cause the affinity-contribution of a TF binding site to depend on its location within the PBM probe. To quantify this effect, we introduce an independent weighting factor for each offset of the TF footprint relative to the end of the probe. These spatial coefficients for each strand are estimated using a multivariate fit to the PBM intensities, which is alternated with re-estimation of the PSAM parameters, until convergence. The magnitude of the contribution to the PBM intensity of a given binding site can vary by an order of magnitude depending on its position within the probe.

In addition, the positional-occupancy profile can be used to help assess the quality of the PBM data and the inferred model. The left figure below shows how the positional-occupancy profile for the 10nt Cbf1 motif exhibits a strong preference for binding far from the substrate. (Position 0 is the nucleotide of the dsDNA probe farthest away from the substrate.) This indicates that this PBM experiment contains a strong TF-binding signal, which leads to a favorable signal-to-noise ratio and relative affinities that match MITOMI data ($R^2 = 0.96$). However, the positional-occupancy profile for the 10nt Pho4 motif on the right exhibits a slight preference for binding near the substrate, which indicates that

the Pho4 data has reduced post-wash, sequence-specific binding. The greatly reduced TF-binding signal results in a less favorable signal-to-noise ratio, making inference of 10-mer relative affinities less accurate when compared to MITOMI data ($R^2 = 0.82$).



Additional Features – FeatureREDUCE extends the positional-independence PSAM model to include possible higher-order "sequence features". For example, we can account for all adjacent nucleotide dependencies simultaneously by fitting a robust multivariate model in which a multiplicative correction parameter is estimated for each dinucleotide feature. With this inference framework, the FeatureREDUCE model pinpoints exactly where in the binding site the positional-independence assumption breaks down, with the corresponding energetic corrections.

2.0 Installation

FeatureREDUCE is a java application that interfaces to R using the rJava/JRI libraries. Some regressions are performed in R and others in java. FeatureREDUCE also makes use of the BioJava, GNU Trove, jFreeChart, Apache Commons, and iText classes. All of the java libraries below are included in the install (in the freduce/bin directory) except for the JVM.

The Java dependencies are:

Java Virtual Machine (JVM) v1.5 or higher
rJava/JRI 0.6-3
GNU Trove 3.0.2
Apache Commons
BioJava 1.5
jFreeChart 1.0.13
iText 5.1.0

The R dependencies are:

R 2.10.1 or later
Mass package 7.3-7 or later
Matrix package 0.999375-33 or later
nnls package 1.3 or later
rJava package 0.6-3 exactly!

Getting rJava to correctly install from the sources package can be tricky (since you have to get the compiler and linker options just right for your particular platform.) We provide the linux 64-bit binaries for your convenience in the freduce/R.packages directory.

Installing the rJava/JRI Linux 64-bit binaries package (recommended)
Unzip the rJava.06-3.linux.64bit.binaries.zip archive found in freduce/R.packages into the library directory where your R packages are installed (typically /usr/lib/R/site-library, /usr/lib/R/library, /usr/lib64/R/site-library, or /usr/lib64/R/library, etc.). Then change the RJAVA_HOME variable accordingly in the .envrc file.

Installing from the rJava/JRI source package
To install from source execute the command “sudo R CMD INSTALL rJava_0.6-3.tar.gz”. Note that this command requires that you have sudo privileges.

Testing the rJava/JRI install
To test that rJava/JRI are running correctly copy the "run2" shell script found in freduce/R.packages to the $RJAVA_HOME/rJava/jri directory (you should already see a "run" shell script in that same directory).

Then “cd” to $RJAVA_HOME/rJava/jri/examples and execute "../run2 rtest". You should see the output pasted below. Enter "q()" to quit rtest. The rtest java class tests the JRI API between Java and R. FeatureREDUCE uses this same API to interface between Java and the R runtime to perform regressions and get the results.

Output from successfully running 'rtest'

```
../run2 rtest
Creating Rengine (with arguments)

R version 2.14.1 (2011-12-22)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-redhat-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Rengine created, waiting for R
```

```
[VECTOR ([REAL* (5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.8, 4.8, 4.3, 5.8, 5.7, 5.4,
5.1, 5.7, 5.1, 5.4, 5.1, 4.6, 5.1, 4.8, 5.0, 5.0, 5.2, 5.2, 4.7, 4.8, 5.4, 5.2, 5.5, 4.9, 5.0, 5.5,
4.9, 4.4, 5.1, 5.0, 4.5, 4.4, 5.0, 5.1, 4.8, 5.1, 4.6, 5.3, 5.0, 7.0, 6.4, 6.9, 5.5, 6.5, 5.7, 6.3,
4.9, 6.6, 5.2, 5.0, 5.9, 6.0, 6.1, 5.6, 6.7, 5.6, 5.8, 6.2, 5.6, 5.9, 6.1, 6.3, 6.1, 6.4, 6.6, 6.8,
6.7, 6.0, 5.7, 5.5, 5.5, 5.8, 6.0, 5.4, 6.0, 6.7, 6.3, 5.6, 5.5, 5.5, 6.1, 5.8, 5.0, 5.6, 5.7, 5.7,
6.2, 5.1, 5.7, ... (50 more values follow))], [REAL* (3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1,
3.7, 3.4, 3.0, 3.0, 4.0, 4.4, 3.9, 3.5, 3.8, 3.8, 3.4, 3.7, 3.6, 3.3, 3.4, 3.0, 3.4, 3.5, 3.4, 3.2,
3.1, 3.4, 4.1, 4.2, 3.1, 3.2, 3.5, 3.6, 3.0, 3.4, 3.5, 2.3, 3.2, 3.5, 3.8, 3.0, 3.8, 3.2, 3.7, 3.3,
3.2, 3.2, 3.1, 2.3, 2.8, 2.8, 3.3, 2.4, 2.9, 2.7, 2.0, 3.0, 2.2, 2.9, 2.9, 3.1, 3.0, 2.7, 2.2, 2.5,
3.2, 2.8, 2.5, 2.8, 2.9, 3.0, 2.8, 3.0, 2.9, 2.6, 2.4, 2.4, 2.7, 2.7, 3.0, 3.4, 3.1, 2.3, 3.0, 2.5,
2.6, 3.0, 2.6, 2.3, 2.7, 3.0, 2.9, 2.9, 2.5, 2.8, ... (50 more values follow))], [REAL* (1.4, 1.4, 1.3,
1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.6, 1.4, 1.1, 1.2, 1.5, 1.3, 1.4, 1.7, 1.5, 1.7, 1.5, 1.0,
1.7, 1.9, 1.6, 1.6, 1.5, 1.4, 1.6, 1.6, 1.5, 1.5, 1.4, 1.5, 1.2, 1.3, 1.4, 1.3, 1.5, 1.3, 1.3, 1.3,
1.6, 1.9, 1.4, 1.6, 1.4, 1.5, 1.4, 4.7, 4.5, 4.9, 4.0, 4.6, 4.5, 4.7, 3.3, 4.6, 3.9, 3.5, 4.2, 4.0,
4.7, 3.6, 4.4, 4.5, 4.1, 4.5, 3.9, 4.8, 4.0, 4.9, 4.7, 4.3, 4.4, 4.8, 5.0, 4.5, 3.5, 3.8, 3.7, 3.9,
5.1, 4.5, 4.5, 4.7, 4.4, 4.1, 4.0, 4.4, 4.6, 4.0, 3.3, 4.2, 4.2, 4.2, 4.3, 3.0, 4.1, ... (50 more
values follow))], [REAL* (0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1, 0.2,
0.4, 0.4, 0.3, 0.3, 0.3, 0.2, 0.4, 0.2, 0.5, 0.2, 0.2, 0.4, 0.2, 0.2, 0.2, 0.2, 0.4, 0.1, 0.2, 0.2,
0.2, 0.2, 0.1, 0.2, 0.2, 0.3, 0.3, 0.2, 0.6, 0.4, 0.3, 0.2, 0.2, 0.2, 0.2, 1.4, 1.5, 1.5, 1.3, 1.5,
1.3, 1.6, 1.0, 1.3, 1.4, 1.0, 1.5, 1.0, 1.4, 1.3, 1.4, 1.5, 1.0, 1.5, 1.1, 1.8, 1.3, 1.5, 1.2, 1.3,
1.4, 1.4, 1.7, 1.5, 1.0, 1.1, 1.0, 1.2, 1.6, 1.5, 1.6, 1.5, 1.3, 1.3, 1.3, 1.2, 1.4, 1.2, 1.0, 1.3,
1.2, 1.3, 1.3, 1.1, 1.3, ... (50 more values follow))], [FACTOR
{levels=("setosa","versicolor","virginica"),ids=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2)}])]
has names:
Sepal.Length
Sepal.Width
Petal.Length
Petal.Width
Species
and once again from the list:
Sepal.Length
Sepal.Width
Petal.Length
Petal.Width
Species
[BOOLi* ]
F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F F T T
T F T F T F T F F T T T F T F F T F T T T T T T T T T T T T F F F F T F T T T F F F T F F F F F T F F T F T T
T T F T T T T T T F F T T T T T T F T T T T T T T T T T T T T T T T T T T T T T F T T T T T T T
[BOOLi* ]
isTRUE? true
[LIST [REAL* (1.0)]:[SYMBOL a],([LIST [STRING "foo"]:[SYMBOL b],([LIST [INT* (1, 2, 3, 4, 5)]:[SYMBOL
c],(null)])])]
Keys:
a
b
c
Contents:
[REAL* (1.0)]
[STRING "foo"]
[INT* (1, 2, 3, 4, 5)]
[REAL* (6.0)]
Parsing
Result = 140695430114488, running eval
Result = 140695430114104, building REXP
REXP result = [STRING "iris"]
Parsing
Result = 140695430304584, running eval
Result = 140695425626784, building REXP
REXP result = [VECTOR ([REAL* (5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.8, 4.8, 4.3,
5.8, 5.7, 5.4, 5.1, 5.7, 5.1, 5.4, 5.1, 4.6, 5.1, 4.8, 5.0, 5.0, 5.2, 5.2, 4.7, 4.8, 5.4, 5.2, 5.5,
4.9, 5.0, 5.5, 4.9, 4.4, 5.1, 5.0, 4.5, 4.4, 5.0, 5.1, 4.8, 5.1, 4.6, 5.3, 5.0, 7.0, 6.4, 6.9, 5.5,
6.5, 5.7, 6.3, 4.9, 6.6, 5.2, 5.0, 5.9, 6.0, 6.1, 5.6, 6.7, 5.6, 5.8, 6.2, 5.6, 5.9, 6.1, 6.3, 6.1,
6.4, 6.6, 6.8, 6.7, 6.0, 5.7, 5.5, 5.5, 5.8, 6.0, 5.4, 6.0, 6.7, 6.3, 5.6, 5.5, 5.5, 6.1, 5.8, 5.0,
5.6, 5.7, 5.7, 6.2, 5.1, 5.7, ... (50 more values follow))], [REAL* (3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4,
3.4, 2.9, 3.1, 3.7, 3.4, 3.0, 3.0, 4.0, 4.4, 3.9, 3.5, 3.8, 3.8, 3.4, 3.7, 3.6, 3.3, 3.4, 3.0, 3.4,
3.5, 3.4, 3.2, 3.1, 3.4, 4.1, 4.2, 3.1, 3.2, 3.5, 3.6, 3.0, 3.4, 3.5, 2.3, 3.2, 3.5, 3.8, 3.0, 3.8,
3.2, 3.7, 3.3, 3.2, 3.2, 3.1, 2.3, 2.8, 2.8, 3.3, 2.4, 2.9, 2.7, 2.0, 3.0, 2.2, 2.9, 2.9, 3.1, 3.0,
```

```
2.7, 2.2, 2.5, 3.2, 2.8, 2.5, 2.8, 2.9, 3.0, 2.8, 3.0, 2.9, 2.6, 2.4, 2.4, 2.7, 2.7, 3.0, 3.4, 3.1,
2.3, 3.0, 2.5, 2.6, 3.0, 2.6, 2.3, 2.7, 3.0, 2.9, 2.9, 2.5, 2.8, ... (50 more values follow))], [REAL*
(1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.6, 1.4, 1.1, 1.2, 1.5, 1.3, 1.4, 1.7, 1.5,
1.7, 1.5, 1.0, 1.7, 1.9, 1.6, 1.6, 1.5, 1.4, 1.6, 1.6, 1.5, 1.5, 1.4, 1.5, 1.2, 1.3, 1.4, 1.3, 1.5,
1.3, 1.3, 1.3, 1.6, 1.9, 1.4, 1.6, 1.4, 1.5, 1.4, 4.7, 4.5, 4.9, 4.0, 4.6, 4.5, 4.7, 3.3, 4.6, 3.9,
3.5, 4.2, 4.0, 4.7, 3.6, 4.4, 4.5, 4.1, 4.5, 3.9, 4.8, 4.0, 4.9, 4.7, 4.3, 4.4, 4.8, 5.0, 4.5, 3.5,
3.8, 3.7, 3.9, 5.1, 4.5, 4.5, 4.7, 4.4, 4.1, 4.0, 4.4, 4.6, 4.0, 3.3, 4.2, 4.2, 4.2, 4.3, 3.0, 4.1, ...
(50 more values follow))], [REAL* (0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1,
0.1, 0.2, 0.4, 0.4, 0.3, 0.3, 0.3, 0.2, 0.4, 0.2, 0.5, 0.2, 0.2, 0.4, 0.2, 0.2, 0.2, 0.2, 0.4, 0.1,
0.2, 0.2, 0.2, 0.2, 0.1, 0.2, 0.2, 0.3, 0.3, 0.2, 0.6, 0.4, 0.3, 0.2, 0.2, 0.2, 0.2, 1.4, 1.5, 1.5,
1.3, 1.5, 1.3, 1.6, 1.0, 1.3, 1.4, 1.0, 1.5, 1.0, 1.4, 1.3, 1.4, 1.5, 1.0, 1.5, 1.1, 1.8, 1.3, 1.5,
1.2, 1.3, 1.4, 1.4, 1.7, 1.5, 1.0, 1.1, 1.0, 1.2, 1.6, 1.5, 1.6, 1.5, 1.3, 1.3, 1.3, 1.2, 1.4, 1.2,
1.0, 1.3, 1.2, 1.3, 1.3, 1.1, 1.3, ... (50 more values follow))], [FACTOR
{levels=("setosa","versicolor","virginica"),ids=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,
2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2)}])]
Parsing
Result = 140695430304344, running eval
Result = 140695425626696, building REXP
REXP result = [STRING* ]
[0] "Sepal.Length"
[1] "Sepal.Width"
[2] "Petal.Length"
[3] "Petal.Width"
[4] "Species"
Parsing
Result = 140695430304200, running eval
Result = 140695418393056, building REXP
REXP result = [REAL* (-0.4563578917943141, 0.014964568959145245, -0.04045751623192635,
1.6567154544596057, 0.12917253631764955, -0.437941516145991, 0.5055877297503669, 0.4089953059474734,
-0.1231624146800834, 0.9571482972543192)]
-0.4563578917943141, 0.014964568959145245, -0.04045751623192635, 1.6567154544596057,
0.12917253631764955, -0.437941516145991, 0.5055877297503669, 0.4089953059474734, -0.1231624146800834,
0.9571482972543192

REXP result = [INT* (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)]
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
 [1] 0.3333333 0.6666667 1.0000000 1.3333333 1.6666667 2.0000000 2.3333333
 [8] 2.6666667 3.0000000 3.3333333
Now the console is yours ... have fun
rBusy(0)
> q()
```

Setting the Environment Variables
Execute "`source .envrc`" in a bash shell to add environment variables and augment the CLASSPATH with the necessary jar files for FeatureREDUCE. Note that you will probably need to change the R environment variables to match your particular installation:

Script `.envrc`:

```
# make sure that the necessary R environment variables are set
R_HOME=/usr/lib/R
export R_HOME

RJAVA_HOME=/usr/local/lib/R
export RJAVA_HOME

R_SHARE_DIR=/usr/share/R/share
export R_SHARE_DIR

LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$RJAVA_HOME/site-library/rJava/jri
export LD_LIBRARY_PATH

# augment the CLASSPATH with all the FeatureREDUCE dependencies
CLASSPATH=$CLASSPATH:$HOME/freduce/bin:$HOME/freduce/bin/biojava.jar:
$HOME/freduce/bin/Biojava.suppl.v1.01.jar:$HOME/freduce/bin/bytecode.jar:$HOME/freduce/bin/commons-
cli.jar:$HOME/freduce/bin/commons-collections-2.1.jar:$HOME/freduce/bin/commons-pool-1.1.jar:
$HOME/freduce/bin/trove-3.0.2.jar:$HOME/freduce/bin/jfreechart-1.0.13.jar:$HOME/freduce/bin/jcommon-
1.0.16.jar:$RJAVA_HOME/site-library/rJava/jri/JRI.jar:$HOME/freduce/bin/commons-lang-2.4.jar:
$HOME/freduce/bin/itextpdf-5.1.0.jar
export CLASSPATH

# Set the display if you're using Xvfb
DISPLAY=:1.0
export DISPLAY
```

3.0 Logos and Graphics

FeatureREDUCE generates sequence logos, FSAM logos, and plots of the positional bias profiles using the Biojava and jFreeChart libraries. The iText libraries are used to save PDF versions of the figures. Both the PNG and PDF versions can be found in the `[OUTPUT] → Directory` specified in the `INIT_FILE`. Use the optional parameter "`-displayMotifs No`" to turn off the generation, display, and saving of these logos.

If you want FeatureREDUCE to exit automatically after creating the models and logos (and not wait for the user to peruse and close the logo), you add the "`-batch`" option when calling FeatureREDUCE.

To generate and save the logos without displaying them, use Xvfb (X virtual frame buffer). This tool can configure a virtual monitor that doesn't actually display anything, but can still be used as an invisible sandbox to draw graphics. If you are sending your display to Xvfb then you will also want to include the "`-batch`" option when calling FeatureREDUCE.

Script `start.xvfb.bash`:

```
# run Xvfb to listen on ":1.0"
# to write graphics to this Xvfb instance set environment variable "DISPLAY=:1.0"
sudo Xvfb :1 -screen 0 1280x1024x24 &
```

Mybl2 Positional Bias Profile

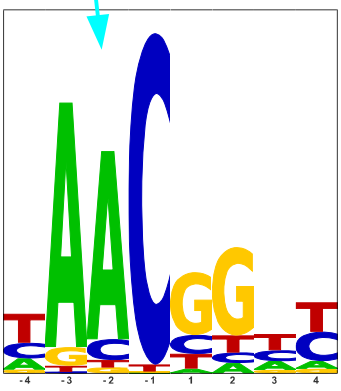
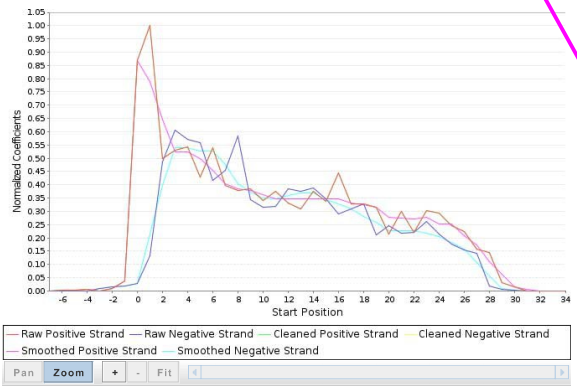
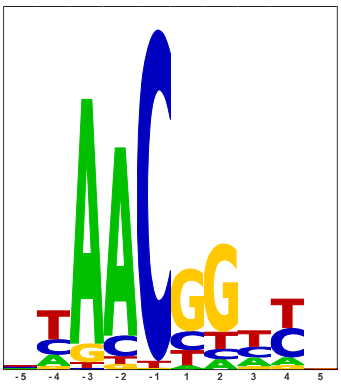
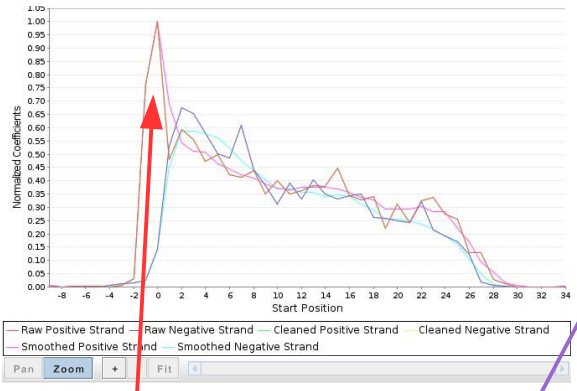
Positional Occupancy Bias Profile for the PSAM to the right. Position 0 is the nucleotide of the dsDNA probe farthest away from the substrate. Like many profiles from successful experiments, this one shows a consistent preference to bind away from the substrate. Also, there is considerable over-hang binding on the positive strand, which is inducing a moderate strand bias.

Both probe strands were used to train this model

8nt PSAM (Position Specific Affinity Matrix) Logo generated from the taacggt seed

R^2 before using the Positional Occupancy Bias Profile

R^2 after using the Occupancy Bias Profile

Regression	Seed	Strand	Logo	R-squared	Occupancy Bias	Bias R-squared
Hamming PSAM	taacggtt	BOTH		0.643212630889864		0.780200750645038
Hamming PSAM	1 + taacggtt + 1	BOTH		0.617912995084912		0.799382646956671

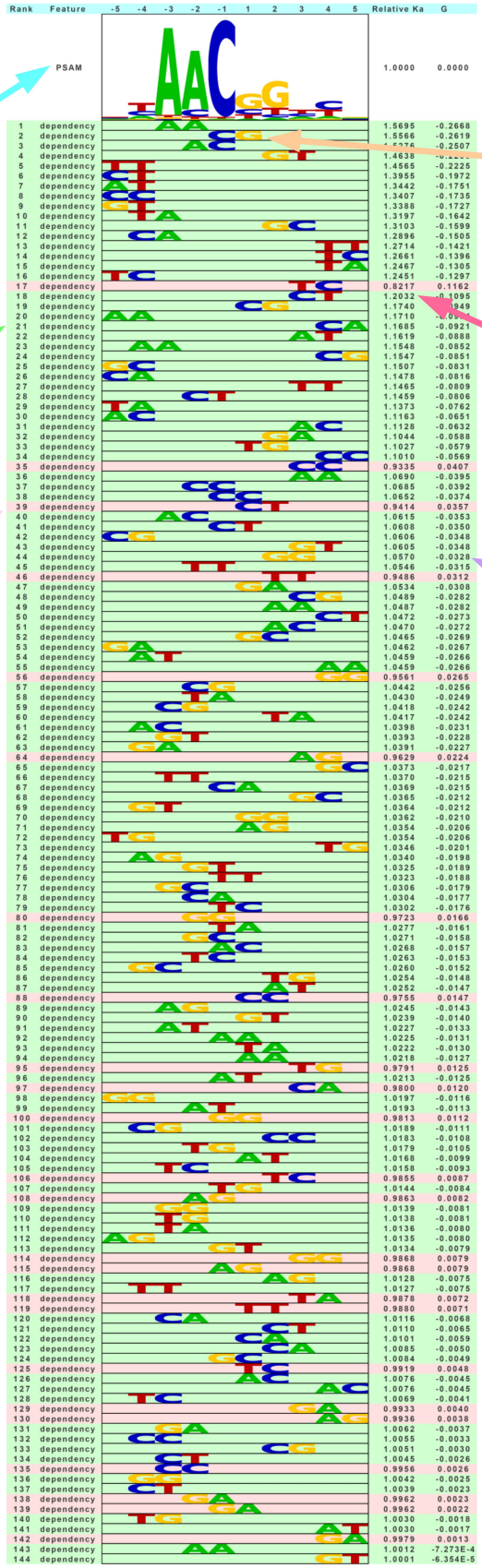
Highest affinity taacggt seed is discovered by performing robust multivariate regression over all 8-mers.

10bp PSAM is generated from the 8bp model by growing the motif on both sides. This 10bp PSAM shows very little specificity within the two additional columns.

Over-hang binding on the positive strand.

Final R^2 for the 10bp PSAM after using the Positional Occupancy Bias Profile

FSAM Logo For Mybl2



The PSAM (Position Specific Affinity Matrix) assumes independence between the nucleotide positions, and that each position contributes additively to the overall binding free energy

The positional-independence assumption breaks down the most in the center of the binding site for Mybl2.

A green background dinucleotide dependency increases the binding affinity

The relative Ka is the multiplicative factor for the dependency feature relative to the positional-independence model (PSAM). A binding site that contains this feature will have a relative affinity 1.2 times that predicted by the PSAM.

A red background dinucleotide dependency decreases the binding affinity

The ΔΔΔG is the change in the binding free energy (in units of kcals/mol) for this dependency feature relative to the positional-independence model (PSAM). A binding site that contains this feature will have a binding free energy 0.0328 kcals/mol less than that predicted by the PSAM.

The dinucleotide dependencies are sorted by descending ΔΔΔG magnitude.

4.0 Training Models from PBM data

All the necessary parameters can be set in the `INIT_FILE`, while some can optionally be specified on the command line (which then over-ride the parameters in the `INIT_FILE`). The command to invoke FeatureREDUCE for model training is:

```
FeatureReduce INIT_FILE -l LABEL -i INPUT_PROBE_FILE
```

The models will be saved in the `[OUTPUT] → Directory` specified in the `INIT_FILE`.

`LABEL` is any arbitrary label that will be prepended onto the names of the models. Examples include "`-l notPreprocessed`", "`-l spatialDetrended`", etc.

`INPUT_PROBE_FILE` is a tab-delimited file that, by default, has the same format as that found in the DREAM5 competition. To change the `INPUT_PROBE_FILE` format use the optional `-c` parameter described below. The default DREAM5 `INPUT_PROBE_FILE` format has the following column labels:

```
ID    Array Type  Sequence    Signal Mean Background Mean   Signal Median
Background Median Signal Std  Background Std    Flag
```

4.1 Examples

```
### Train all the models (TF_1 thru TF_66) and also train the All-Kmers model
java -Xmx5000M FeatureReduce dream.init -kmer -l dream -i DREAM5_PBM_Data_Needed_For_Predictions.txt
2>&1 | tee dream.training.all.out


### Train the model for TF_1 including the All-Kmers model
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -kmer -l dream -i
DREAM5_PBM_Data_Needed_For_Predictions.txt 2>&1 | tee dream.training.TF_1.out
```

4.2 Optional Parameters

`-s MOTIF_LENGTH` is the parameter to change the length of the motif from the default length of 8bp. Currently allowable `MOTIF_LENGTH` values range from 4 to 12, inclusive.

`-kmer` is a flag to turn on the training of the All-Kmers Model which greatly helps in the prediction of PBM probe intensities for similar PBM designs (eg. ME and HK designs).

`-ids TF_1 TF_2...` allows for training the models for a subset (and not all) of the PBM datasets.

`-c LABEL_COLUMN SEQUENCE_COLUMN INTENSITY_COLUMN` allows for multiple `INPUT_PROBE_FILE` formats. By default, FeatureREDUCE uses only columns 0, 2, and 5 of the DREAM5 format `INPUT_PROBE_FILE` for the protein label, probe sequence, and probe intensity, respectively, using 0-based indexing. (Generally, the signal median gives better results than the signal mean.) You can use the `-c` option to change which columns FeatureREDUCE uses. Here's an example:

```
### Use the 1st, 3rd, and 4th columns to train the model
java -Xmx5000M FeatureReduce badis09.init -c 0 2 3 -kmer -l dream -i TRAIN_FILE 2>&1 | tee
dream.training.all.out
```

4.3 Using A Priori Knowledge To Help Train the Models

Using a priori knowledge during training can help FeatureREDUCE generate a statistically more accurate affinity model.

4.31 A priori knowledge is that the protein binds as a homodimer, so enforce symmetry

Specifically, if you know that the protein binds as a homodimer (like bHLHs and bZIPs), then you can use that knowledge to force a symmetric motif which will be more accurate (symmetric motifs get a statistical accuracy boost by using both sides of the binding site simultaneously to train for each position).

To do this, you need to send an argument to FeatureREDUCE: "`-seedType rcPalindrome`" and set the `PalindromicSeedThresh` variable in the init file to something << 1.0 like this:

```
[REGRESSION]
PalindromicSeedThresh = 0.1
```

The `PalindromicSeedThresh` variable allows FeatureREDUCE to discover symmetric motifs while simultaneously allowing you to finely tune your preference for a palindromic seed. For example, say you are using 8-mers as seeds (default) and `PalindromicSeedThresh = 0.1`, then if the highest affinity palindrome has a relative affinity within 0.1 of the highest affinity 8mer then that 8-mer palindrome will be used as the seed. The default value is `PalindromicSeedThresh = 0.9`

If `PalindromicSeedThresh = 1.0`, then the highest affinity palindromic 8mer must also have the highest affinity relative to all possible 8mers as well in order to select it as the seed.

4.32 A priori knowledge is the sequence of the highest affinity binding site

From SELEX experiments, etc. it is sometimes known what the highest affinity binding site is for a TF-protein. You can use this prior knowledge to set the seed sequence for FeatureREDUCE. If the specified seed is symmetric, then this will also enforce symmetry in the discovered binding model.

To set the seed in FeatureREDUCE, you need to send it as an argument: "`-seed GTCACGTGAC`". (This is the highest affinity 8mer for Cbf1, a bHLH factor that binds as a homodimer and has a symmetric motif (confirmed by Maerkl and Quake, *Science* 2007)).

4.4 Output Files

All of the output files are in the `[OUTPUT] → Directory` specified in the `INIT_FILE`, and are prepended with the `LABEL` specified during the training of the models.

It is strongly recommended to have a different `LABEL` and `[OUTPUT] → Directory` for each probe intensities dataset (eg. with different microarray preprocessing). For example, for the set of probe intensities that have been spatially detrended use "`-l dream.spatialDetrended`" on the command

line, and "`Directory = dream.spatialDetrended`" in the init file. This will make it much easier to keep track of your different models.

All the possible output files that can be saved:

`LABEL.id.`topSeeds.txt – The top 50 highest-affinity seeds out of all K-mers of length K
`LABEL.id.`results.`N`nt.* - The plots of the PSAM logos and their Positional Bias Profiles
`LABEL.id.`psam.`N`nt.* - The files corresponding to the trained `N`nt PSAM
`LABEL.id.N`nt.psam.info – Text file that contains 3 values: seed used to train the PSAM model, R^2 before using the Positional Bias Profile, R^2 after using the profile
`LABEL.id.`fsam.`N`nt.* - The files corresponding to the trained `N`nt FSAM
`LABEL.id.N`nt.fsam.info – Text file that contains 3 values: seed used to train the FSAM model, R^2 before using the Positional Bias Profile, R^2 after using the profile
`LABEL.id.`kmers.`K`nt.positionalBias.* - The files corresponding to the `K`nt Positional Bias Profile for all `K`-mers of length `K` in the All-Kmers Model
`LABEL.id.`kmers.`K`nt.affinities.table - The affinity corrections for all `K`-mers of length `K` in the All-Kmers Model

5.0 Predicting PBM Probe Intensities

All the necessary parameters can be set in the `INIT_FILE`, while some can optionally be specified on the command line (which then over-ride the parameters in the `INIT_FILE`). The command to invoke FeatureREDUCE for probe intensity predictions is:

```
FeatureReduce INIT_FILE -l LABEL -kmer -a ANSWER_TEMPLATE -o PREDICTION_FILE
```

The models are loaded from the `[OUTPUT] → Directory` specified in the `INIT_FILE`. The probe intensities will be predicted for the sequences contained in the `ANSWER_TEMPLATE`, and saved to the `PREDICTION_FILE`.

`LABEL` is any arbitrary label that was used to name the models during training. Examples include "`-l notPreprocessed`", "`-l spatialDetrended`", etc.

`-kmer` is a flag to turn on scoring with the All-Kmers Model which greatly helps in the prediction of PBM probe intensities for similar PBM designs (eg. ME and HK designs). This scoring option is only available if the All-Kmers Model was trained.

`ANSWER_TEMPLATE` is the tab-delimited format from the DREAM5 competition that has the following column labels (with "Signal Mean" filled in as a '`?`') :

```
ID    Array Type  Sequence    Signal Mean
```

`PREDICTION_FILE` has the same format as `ANSWER_TEMPLATE` with the same column labels:

```
ID    Array Type  Sequence    Signal Mean
```

5.1 Examples

```
### Predict all the probe intensities (TF_1 - TF_66) and use the All-Kmers Models
java -Xmx5000M FeatureReduce dream.init -kmer -l dream -a DREAM5_PBM_TeamName_Predictions.txt -o
DREAM5_PBM_FeatureREDUCE_Predictions.all.txt  2>&1 | tee dream.predictions.all.out

### Predict the probe intensities for TF_1 and use the All-Kmers Model
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -kmer -l dream -a DREAM5_PBM_TeamName_Predictions.txt
-o DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee dream.predictions.TF_1.out

### Predict the probe intensities for TF_1 and DON'T use the All-Kmers Model
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -l dream -a DREAM5_PBM_TeamName_Predictions.txt -o
DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee dream.predictions.TF_1.out
```

5.2 Optional Parameters

`-kmer` is a flag to turn on scoring with the All-Kmers Model which greatly helps in the prediction of PBM probe intensities for similar PBM designs (eg. ME and HK designs). This scoring option is only available if the All-Kmers Model was trained.

`-noFeatures` is a flag to turn off scoring with the dinuleotides dependency corrections. When using this flag, FeatureREDUCE will use only the PSAM with the positional-bias profile, and optionally the All-Kmers Model.

`-ids TF_1 TF_2`... allows for scoring a subset (and not all) of the PBM datasets.

5.3 Optional Parameters Examples

```
### Predict probe intensities turning off Dinucleotide Features, still keeping the All-Kmers Model
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -noFeatures -kmer -l dream -a
DREAM5_PBM_TeamName_Predictions.txt -o DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee
dream.predictions.TF_1.out

### Predict probe intensities while Turning off Dinucleotide Features and the All-Kmers Model
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -noFeatures -l dream -a
DREAM5_PBM_TeamName_Predictions.txt -o DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee
dream.predictions.TF_1.out
```

6.0 Training Models from PBM data & Predicting PBM Probe Intensities Together

It's possible to combine the training and predicting of probe intensities into one invocation by combining the command line options together:

```
FeatureReduce INIT_FILE -kmer -l LABEL -i INPUT_PROBE_FILE -a ANSWER_TEMPLATE -o
PREDICTION_FILE
```

The models will be saved in the `[OUTPUT] → Directory` specified in the `INIT_FILE`. The probe intensities will be predicted for the sequences contained in the `ANSWER_TEMPLATE`, and saved to the `PREDICTION_FILE`.

6.1 Examples

```
### Train all the models (TF_1 thru TF_66) and also train the All-Kmers model, then predict all the
probe intensities (TF_1 thru TF_66) and use the All-Kmers Models in the predictions
java -Xmx5000M FeatureReduce dream.init -kmer -l dream -i DREAM5_PBM_Data_Needed_For_Predictions.txt -a
DREAM5_PBM_TeamName_Predictions.txt -o DREAM5_PBM_FeatureREDUCE_Predictions.all.txt 2>&1 | tee
dream.training.predicting.all.out


### Train the model for TF_1 including the All-Kmers model, then predict the probe intensities for TF_1
and use the All-Kmers Model in the predictions
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -kmer -l dream -i
DREAM5_PBM_Data_Needed_For_Predictions.txt -a DREAM5_PBM_TeamName_Predictions.txt -o
DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt 2>&1 | tee dream.training.predicting.TF_1.out
```

7.0 Predicting (non-PBM) Sequence Affinities

Again, all the necessary parameters can be set in the `INIT_FILE`, while some can optionally be specified on the command line (which then over-ride the parameters in the `INIT_FILE`). When predicting the affinities of genomic DNA, we don't wish to use the Positional-Bias Profile nor the All-Kmer Model. Therefore, the command to invoke FeatureREDUCE for sequence affinity predictions is:

```
FeatureReduce INIT_FILE -noPosBias -l LABEL -a ANSWER_TEMPLATE -o PREDICTION_FILE
```

The models are loaded from the `[OUTPUT] → Directory` specified in the `INIT_FILE`. The relative affinities will be predicted for the sequences contained in the `ANSWER_TEMPLATE`, and saved to the `PREDICTION_FILE`.

`-noPosBias` is a flag that turns off using the Positional-Bias Profile when predicting the relative affinities of non-PBM DNA.

`LABEL` is any arbitrary label that was used to name the models during training. Examples include "`-l notPreprocessed`", "`-l spatialDetrended`", etc.

`ANSWER_TEMPLATE` is the tab-delimited format from the DREAM5 competition that has the following column labels (with "Signal Mean" filled in as a '`?`'):

```
ID    Array Type  Sequence    Signal Mean
```

`PREDICTION_FILE` has the same format as `ANSWER_TEMPLATE` with the same column labels:

```
ID    Array Type  Sequence    Signal Mean
```

7.1 Examples

```
### Predict affinities for non-PBM DNA for all the IDs
java -Xmx5000M FeatureReduce dream.init -noPosBias -l dream -a DREAM5_PBM_TeamName_Predictions.txt -o
DREAM5_PBM_FeatureREDUCE_Predictions.all.txt  2>&1 | tee dream.predictions.all.out

### Predict affinities for non-PBM DNA for just TF_1
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -noPosBias -l dream -a
```

```
DREAM5_PBM_TeamName_Predictions.txt -o DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee
dream.predictions.TF_1.out
```

7.2 Optional Parameters

`-noFeatures` is a flag to turn off scoring with the dinuleotides dependency corrections. When using this flag, FeatureREDUCE will use only the PSAM with the positional-bias profile, and optionally the All-Kmers Model.

`-ids TF_1 TF_2...` allows for scoring a subset (and not all) of the PBM datasets.

7.3 Optional Parameters Examples

```
### Predict sequence affinities while turning off Dinucleotide Features (and the All-Kmers Model)
java -Xmx5000M FeatureReduce dream.init -ids TF_1 -noPosBias -noFeatures -l dream -a
DREAM5_PBM_TeamName_Predictions.txt -o DREAM5_PBM_FeatureREDUCE_Predictions.TF_1.txt  2>&1 | tee
dream.predictions.TF_1.out
```

8.0 Getting the Corresponding K-mer Affinities from an FSAM

It is often easier to work with K-mer affinities derived from an FSAM when predicting the affinity of genomic sequence. The K-mers will be the same length as the FSAM motif, and the table is sorted into descending relative-affinity order. The command to get the table of K-mer affinities from an FSAM file is:

```
FeatureReduce -fsam FSAM_FILE -affinitySphere FEATURE_THRESH TOTAL_AFFINITY_THRESH
OUTPUT_FILE
```

`FSAM_FILE` is a serialized FSAM java object which ends with a `.ser` extension. They are saved in the `[OUTPUT] → Directory` specified in the `INIT_FILE`.

`FEATURE_THRESH` is a decimal ≥ 0 that gives the ability to remove features that have little effect on the affinities (by removing features from the model you can improve performance when scoring very long K-mers). Typically this should be kept to 0 except for very long FSAM motifs.

`TOTAL_AFFINITY_THRESH` is a decimal ≥ 0 that removes K-mers from the table with very low affinity. This can improve performance and keep the file size smaller for motifs longer than 9bp.

`OUTPUT_FILE` is the file that the K-mer affinities are written to.

8.1 Examples

```
### Get the affinity sphere for all 10-mers with a relative affinity ≥ 0.01 of the highest affinity 10-
mer.
java -Xmx1000M FeatureReduce -fsam protein.fsam.ser -affinitySphere 0 0.01
protein.fsam.affinitySphere.01.table
```

9.0 Loading Models to View and Save Their Logos

With FeatureREDUCE you can load trained models to view and save their logos, manipulate the models, or predict the affinities of genomic DNA. You can also load popular PWM-format models and convert them to PSAMs.

9.1 Loading and Viewing FSAMs

The command to load a trained FSAM and view its FSAM Logo is:

```
### Just load an FSAM model for logo viewing
java FeatureReduce -fsam results.dream/dream.TF_1.fsam.10nt.ser
```

To save the Logo to file right-click over the Logo with the mouse and specify the "Save to File" option in the pop-up menu. Currently supported graphics formats are PNG and PDF. (If needed, the PDF format can be converted into any other format by many graphics software packages, including Acrobat Reader.) The command to dump all the FSAM parameters to STDOUT without viewing the FSAM Logo includes the optional parameter "`-displayMotifs No`" :

```
### Just dump the FSAM parameters without showing the Logo
java FeatureReduce -fsam results.dream/dream.TF_1.fsam.10nt.ser -displayMotifs No
```

9.2 Loading and Viewing PSAMs and PWMs

The command to load a PSAM and view its PSAM Logo is similar to that of loading an FSAM except that the file format of the PSAM must also be specified, since there are many supported formats, including popular PWM formats that can be converted to a PSAM. Here is an example that loads a PSAM to view its PSAM Logo:

```
### Just load a PSAM model to view the logo
java FeatureReduce -psam xml results.dream/dream.TF_1.psam.10nt.xml
```

The currently supported PSAM and PWM formats are:

`xml`	– a PSAM in BioJava 1.5 XML format
`ser`	– a FeatureREDUCE PSAM serialized Java object
`table`	– a FeatureREDUCE R-format PSAM table file
`mxr`	– a MatrixREDUCE XML-format PSAM file
`uniprobe`	– a UniProbe format PWM file
`jaspar`	– a Jaspar format PWM file

9.3 Saving PSAM, PWM, and FSAM Logos

To save any of the logos to file you can add the `-saveLogo LOGO_FILE` option. The `LOGO_FILE` must end with either a `.png` or `.pdf` extension. If you want FeatureREDUCE to exit automatically after creating the logo (and not wait for the user to peruse and close the logo), you add the "`-batch`" option to the command line.

9.4 Examples

```
### load a PSAM model to view and save the logo
java FeatureReduce -psam xml results.dream/dream.TF_1.psam.10nt.xml -saveLogo
dream.TF_1.psam.10nt.png

### load a PSAM model to create and save its logo (but not view it)
java FeatureReduce -psam xml results.dream/dream.TF_1.psam.10nt.xml -saveLogo
dream.TF_1.psam.10nt.png -batch

### load an FSAM model to view and save the logo
java FeatureReduce -fsam results.dream/dream.TF_1.fsam.10nt.ser -saveLogo
dream.TF_1.fsam.10nt.png

### load an FSAM model to create and save its logo (but not view it)
java FeatureReduce -fsam results.dream/dream.TF_1.fsam.10nt.ser -saveLogo
dream.TF_1.fsam.10nt.png -batch
```

10.0 Issues with Poly-C and Poly-G Motifs

The HK, ME, and later UPBM array designs all have a major “issue” that should be taken into consideration when analyzing poly-C and poly-G motifs. The initial designers of the PBM arrays noticed that poly-Gs on the coding strand greatly affected the efficiency of the ssDNA-synthesis step when creating the arrays, resulting in probes that are shorter than the desired length. A possible molecular explanation is that the contiguous run of Gs on closely packed, single-stranded DNA oligos are forming G-quadruplexes, essentially forming a knot. In order to remove this ssDNA-synthesis bias, the de Bruijn sequences are modified by removing many consecutive Gs of length 5 or more (e.g. GGGGG, GGGGGG, etc.). However, mostly-G sequences also affect the efficiency of ssDNA-synthesis somewhat, but are not removed from the probe sequences. Unfortunately, this situation can create some problems.

The following Table 10.1 lists the number of poly-G 6-mers, 7-mers, 8-mers, 9-mers, and 10-mers that are missing from the positive strand of the probe sequences. Table 10.2 lists the number of 8-mers that are found on only 1, 2, 4, 8, and 12 or less probes on the positive strand. Table 10.3 lists the number of 10-mers that are found on only 1, 2, 4, and 6 or less probes on the positive strand. Note that the reverse-complement poly-C K-mers are found (or not found) in equal numbers on the negative strand of the probe sequences. Also, the unmodified de Bruijn sequences should contain all 4^{10} (1,048,576) 10-mers exactly once and all 4^8 (65,536) 8-mers exactly 16 times on the positive strand.

Table 10.1: Missing K-mers on the positive strand

PBM Design	Missing 6-mers	Missing 7-mers	Missing 8-mers	Missing 9-mers	Missing 10-mers
UPBM-1	0	0	0	0	0
UPBM-2	0	0	0	0	0
UPBM-9	0	12	150	948	22,646
UPBM-11	1	18	170	975	23,023
HK	0	20	409	3,769	85,131
ME	1	21	412	4,920	272,199

Table 10.2: 8-mers found on only N probes or less on the positive strand

PBM Design	8-mers on ≤ 1 Probes	8-mers on ≤ 2 Probes	8-mers on ≤ 4 Probes	8-mers on ≤ 8 Probes	8-mers on ≤ 12 Probes
UPBM-1	0	0	0	0	2
UPBM-2	0	0	0	0	1
UPBM-9	200	206	208	208	253
UPBM-11	202	208	208	208	253
HK	695	876	999	1,032	2,170
ME	728	912	1,007	1,236	5,531

Table 10.3: 10-mers found on only N probes or less on the positive strand

PBM Design	10-mers on ≤ 1 Probes	10-mers on ≤ 2 Probes	10-mers on ≤ 4 Probes	10-mers on ≤ 6 Probes
UPBM-1	1,048,572	1,048,576	1,048,576	1,048,576
UPBM-2	1,048,572	1,048,576	1,048,576	1,048,576
UPBM-9	945,030	1,046,234	1,048,576	1,048,576
UPBM-11	945,122	1,046,209	1,048,575	1,048,576
HK	963,441	1,048,576	1,048,576	1,048,576
ME	772,670	1,047,427	1,048,568	1,048,576

Table 10.4: PBM-ME Design - the 412 missing 8-mers on the positive strand

AAACGGGG	AAAGGGGG	AAAGGGGT	AACGGGGG	AACGGGGT	AAGGGGAA	AAGGGGAC	AAGGGGCA
AAGGGGGG	AAGGGGGT	AAGGGGTA	AAGGGGTC	AAGTGGGG	AATGGGGG	AATGGGGT	ACAGGGGA
ACATGGGG	ACCGGGGA	ACCGGGGG	ACCTGGGG	ACGCGGGG	ACGGGGAA	ACGGGGAT	ACGGGGCA
ACGGGGGA	ACGGGGGC	ACGGGGGG	ACGGGGGT	ACGGGGTA	ACGGGGTG	ACGTGGGG	ACTAGGGG
ACTGGGGC	ACTGGGGG	AGAGGGGG	AGATGGGG	AGCCGGGG	AGCGGGGT	AGGGGAAG	AGGGGACA
AGGGGACC	AGGGGACG	AGGGGAGC	AGGGGATA	AGGGGCCA	AGGGGGAG	AGGGGGAT	AGGGGGCA
AGGGGGCG	AGGGGGCT	AGGGGGGA	AGGGGGGC	AGGGGGGG	AGGGGGGT	AGGGGGTA	AGGGGGTC
AGGGGGTT	AGGGGTAC	AGGGGTTA	AGGTGGGG	ATACGGGG	ATAGGGGA	ATAGGGGG	ATATGGGG
ATCCGGGG	ATCGGGGG	ATCTGGGG	ATGCGGGG	ATGGGGAA	ATGGGGCA	ATGGGGCG	ATGGGGGA
ATGGGGGC	ATGGGGGG	ATGGGGGT	ATGGGGTA	ATTCGGGG	ATTGGGGA	ATTGGGGG	CAAAGGGG
CAAGGGGT	CACCGGGG	CACGGGGG	CACGGGGT	CACTGGGG	CAGGGGAA	CAGGGGAC	CAGGGGCT
CAGGGGGG	CAGGGGGT	CAGGGGTC	CATAGGGG	CATGGGGA	CATGGGGG	CCAGGGGA	CCAGGGGG
CCGGGGCT	CCGGGGGC	CCGGGGGG	CCGGGGGT	CCGGGGTC	CCTGGGGG	CGACGGGG	CGAGGGGA
CGAGGGGG	CGATGGGG	CGCCGGGG	CGCGGGGG	CGCTGGGG	CGGGGACG	CGGGGATG	CGGGGCAT
CGGGGCGG	CGGGGGAG	CGGGGGAT	CGGGGGCA	CGGGGGCC	CGGGGGCG	CGGGGGCT	CGGGGGGA
CGGGGGGC	CGGGGGGG	CGGGGGGT	CGGGGGTA	CGGGGGTC	CGGGGGTT	CGGGGTAG	CGGGGTCA
CGGGGTCG	CGGGGTCT	CGGGGTGC	CGGGGTGG	CGTCGGGG	CGTGGGGC	CGTGGGGT	CTAAGGGG
CTAGGGGG	CTCGGGGG	CTGGGGAA	CTGGGGAC	CTGGGGCC	CTGGGGCT	CTGGGGGA	CTGGGGGC
CTGGGGGG	CTGGGGGT	CTTAGGGG	CTTGGGGG	GAAGGGGA	GAAGGGGT	GACCGGGG	GACGGGGG
GACTGGGG	GAGCGGGG	GAGGGGAT	GAGGGGGA	GAGGGGGC	GAGGGGGG	GATCGGGG	GATGGGGC
GATGGGGG	GCAAGGGG	GCAGGGGA	GCAGGGGC	GCAGGGGG	GCCCGGGG	GCCGGGGG	GCCGGGGT
GCCTGGGG	GCGAGGGG	GCGGGGGA	GCGGGGGC	GCGGGGGG	GCGGGGTA	GCGGGGTG	GCTAGGGG
GCTCGGGG	GCTGGGGC	GCTGGGGG	GGAAGGGG	GGAGGGGG	GGATGGGG	GGCAGGGG	GGCCGGGG
GGCGGGGC	GGCGGGGG	GGGCGGGG	GGGGAAAC	GGGGAAAT	GGGGAACG	GGGGAAGG	GGGGAATC
GGGGAATG	GGGGACAT	GGGGACCT	GGGGACGG	GGGGACGT	GGGGAGCG	GGGGAGCT	GGGGAGGT
GGGGAGTT	GGGGATAC	GGGGATAT	GGGGATCA	GGGGATCG	GGGGATGT	GGGGATTC	GGGGCAAA
GGGGCACT	GGGGCAGC	GGGGCAGT	GGGGCATA	GGGGCCAC	GGGGCCAG	GGGGCCAT	GGGGCCCG
GGGGCCCT	GGGGCCGA	GGGGCCGC	GGGGCCGG	GGGGCCTA	GGGGCCTG	GGGGCGAG	GGGGCGCA
GGGGCGGG	GGGGCGTA	GGGGCTAG	GGGGCTCG	GGGGCTCT	GGGGCTGC	GGGGCTGT	

GGGGCTTA GGGGCTTG GGGGGAAA GGGGGAAT GGGGGACC GGGGGACG GGGGGAGA GGGGGAGC GGGGGAGG GGGGGATA GGGGGATC
GGGGGATG GGGGGATT GGGGGCAA GGGGGCAC GGGGGCAG GGGGGCAT GGGGGCCA GGGGGCCG GGGGGCCT GGGGGCGC GGGGGCGG
GGGGGCGT GGGGGCTA GGGGGCTC GGGGGCTG GGGGGCTT GGGGGGAA GGGGGGAC GGGGGGAG GGGGGGAT GGGGGGCA GGGGGGCC
GGGGGCCG GGGGGGCT GGGGGGGA GGGGGGGC GGGGGGGG GGGGGGGT GGGGGGTA GGGGGGTC GGGGGGTG GGGGGGTT GGGGGTAA
GGGGGTAC GGGGGTAG GGGGGTAT GGGGGTCA GGGGGTCC GGGGGTCG GGGGGTCT GGGGGTGG GGGGGTGT GGGGGTTA GGGGGTTC
GGGGGTTG GGGGGTTT GGGGTAAT GGGGTACA GGGGTACT GGGGTAGA GGGGTAGC GGGGTAGG GGGGTATA GGGGTATT GGGGTCAC
GGGGTCAG GGGGTCCA GGGGTCCG GGGGTCGC GGGGTCGT GGGGTCTA GGGGTCTT GGGGTGAA GGGGTGCA GGGGTGGA GGGGTGGT
GGGGTTAT GGGGTTGG GGGGTTGT GGGGTTTG GGTGGGGC GGTGGGGG GGTGGGGT GGTTGGGG GTAAGGGG GTAGGGGC GTAGGGGG
GTCCGGGG GTCGGGGG GTGGGGCA GTGGGGCG GTGGGGGG GTGGGGGT GTGGGGTG GTTCGGGG GTTGGGGG GTTTGGGG TAAAGGGG
TAAGGGGG TAATGGGG TACAGGGG TACCGGGG TACGGGGC TACGGGGG TAGGGGAG TAGGGGAT TAGGGGCA TAGGGGGA TAGGGGGC
TAGGGGGG TAGGGGGT TAGGGGTT TATAGGGG TATGGGGG TATTGGGG TCACGGGG TCATGGGG TCCAGGGG TCCCGGGG TCCGGGGG
TCGAGGGG TCGGGGAC TCGGGGCC TCGGGGGA TCGGGGGC TCGGGGGG TCGGGGGT TCGGGGTC TCTCGGGG TCTGGGGC TCTGGGGG
TGACGGGG TGCAGGGG TGCGGGGG TGCGGGGT TGCTGGGG TGGAGGGG TGGGGAAA TGGGGAAC TGGGGAAG TGGGGCAA TGGGGCAC
TGGGGCCG TGGGGCGA TGGGGCGT TGGGGCTT TGGGGGAA TGGGGGAC TGGGGGAT TGGGGGCA TGGGGGCC TGGGGGCT TGGGGGGA
TGGGGGGC TGGGGGGG TGGGGGGT TGGGGGTA TGGGGGTC TGGGGGTG TGGGGGTT TGGGGTAA TGGGGTGC TGGGGTGG TGGTGGGG
TGTAGGGG TGTCGGGG TGTGGGGG TGTTGGGG TTAGGGGG TTATGGGG TTCGGGGG TTCGGGGT TTCTGGGG TTGAGGGG TTGGGGAT
TTGGGGGA TTGGGGGC TTGGGGGG TTGGGGGT TTTGGGGG

Table 10.5: PBM-HK Design - The 409 missing 8-mers on the positive strand

AAAGGGGC AAAGGGGG AACCGGGG AACGGGGG AAGGGGCA AAGGGGCG AAGGGGCT AAGGGGGA AAGGGGGC AAGGGGGG AAGGGGGT
AAGGGGTC AATAGGGG AATGGGGC AATGGGGG AATTGGGG ACACGGGG ACAGGGGG ACAGGGGT ACCAGGGG ACCGGGGA ACCGGGGG
ACCGGGGT ACCTGGGG ACGGGGAG ACGGGGAT ACGGGGCT ACGGGGGA ACGGGGGC ACGGGGGT ACGGGGTG ACGGGGTT
ACGTGGGG ACTGGGGG ACTGGGGT ACTTGGGG AGAAGGGG AGACGGGG AGAGGGGG AGATGGGG AGCAGGGG AGGCGGGG AGGGGAAC
AGGGGACT AGGGGATC AGGGGCAC AGGGGCAG AGGGGCCA AGGGGCCG AGGGGCGA AGGGGCGC AGGGGGAA AGGGGGAC AGGGGGAG
AGGGGGAT AGGGGGCA AGGGGGCT AGGGGGGA AGGGGGGC AGGGGGGG AGGGGGGT AGGGGGTA AGGGGGTC AGGGGGTG AGGGGGTT
AGGGGTCT AGGGGTTA AGGGGTTC AGGGGTTG AGTAGGGG AGTGGGGG AGTTGGGG ATAAGGGG ATACGGGG ATAGGGGC ATAGGGGG
ATCAGGGG ATCGGGGA ATCGGGGG ATCGGGGT ATCTGGGG ATGGGGCG ATGGGGCT ATGGGGGA ATGGGGGC ATGGGGGG ATGTGGGG
ATTAGGGG ATTCGGGG ATTGGGGA ATTGGGGC ATTGGGGG ATTTGGGG CAAGGGGC CAATGGGG CACAGGGG CACCGGGG CACGGGGG
CACGGGGT CAGGGGGA CAGGGGGC CAGGGGGG CAGGGGGT CAGGGGTC CAGGGGTT CATAGGGG CATGGGGC CATGGGGG CATTGGGG
CCAGGGGG CCGGGGGA CCGGGGGC CCGGGGGT CGAGGGGA CGAGGGGC CGAGGGGG CGCAGGGG CGCGGGGG CGCTGGGG CGGGGAAG
CGGGGACA CGGGGACT CGGGGAGT CGGGGATG CGGGGCAC CGGGGCCA CGGGGCTC CGGGGGAA CGGGGGAC CGGGGGAG CGGGGGAT
CGGGGGCA CGGGGGCT CGGGGGGA CGGGGGGG CGGGGGGT CGGGGGTA CGGGGGTC CGGGGGTG CGGGGTAA CGGGGTAG CGGGGTTG
CGTGGGGC CGTGGGGG CTAAGGGG CTAGGGGG CTATGGGG CTCGGGGA CTCGGGGG CTGAGGGG CTGGGGAC CTGGGGGA CTGGGGGG
CTGGGGGT CTGGGGTG CTGGGGTT CTTCGGGG CTTGGGGA CTTGGGGG CTTTGGGG GAAAGGGG GAAGGGGG GAAGGGGT GAATGGGG
GACCGGGG GACGGGGG GACGGGGT GAGCGGGG GAGGGGGA GAGGGGGG GAGGGGGT GATAGGGG GATTGGGG GCAGGGGA GCAGGGGG
GCAGGGGT GCATGGGG GCCGGGGA GCCTGGGG GCGAGGGG GCGGGGAG GCGGGGGA GCGGGGGG GCGTGGGG GCTCGGGG
GCTGGGGG GGCAGGGG GGGGAAAT GGGGAACG GGGGAACT GGGGAATT GGGGACAG GGGGACCG GGGGACCT GGGGACGA GGGGACGC
GGGGACTA GGGGACTC GGGGACTT GGGGAGAG GGGGAGCA GGGGAGCT GGGGAGTT GGGGATAG GGGGATCG GGGGATGA GGGGATGC
GGGGATTA GGGGATTC GGGGCACA GGGGCACG GGGGCACT GGGGCCAA GGGGCCAC GGGGCCAG GGGGCCAT GGGGCCGA GGGGCCGT
GGGGCCTC GGGGCGAC GGGGCGAG GGGGCGCA GGGGCGCT GGGGCGGC GGGGCGTA GGGGCGTT GGGGCTAG GGGGCTAT GGGGCTCT
GGGGCTGA GGGGCTGC GGGGGAAA GGGGGAAC GGGGGAAG GGGGGAAT GGGGGACA GGGGGACC GGGGGACG GGGGGAGA GGGGGAGC
GGGGGAGG GGGGGAGT GGGGGATA GGGGGATC GGGGGATG GGGGGATT GGGGGCAA GGGGGCAC GGGGGCAG GGGGGCAT GGGGGCCA
GGGGGCCG GGGGGCCT GGGGGCGC GGGGGCTA GGGGGCTC GGGGGCTG GGGGGCTT GGGGGGAA GGGGGGAC GGGGGGAG GGGGGGAT
GGGGGGCA GGGGGGCC GGGGGGCG GGGGGGCT GGGGGGGA GGGGGGGC GGGGGGGG GGGGGGGT GGGGGGTA GGGGGGTC GGGGGGTG
GGGGGGTT GGGGGTAA GGGGGTAC GGGGGTAG GGGGGTAT GGGGGTCC GGGGGTCG GGGGGTCT GGGGGTGC GGGGGTGG GGGGGTGT
GGGGGTTA GGGGGTTC GGGGGTTG GGGGTAAG GGGGTACA GGGGTACT GGGGTAGA GGGGTAGC GGGGTAGT GGGGTCAC GGGGTCCG
GGGGTCCT GGGGTCTA GGGGTCTC GGGGTGAA GGGGTGAG GGGGTGAT GGGGTGCT GGGGTGGA GGGGTGGC GGGGTGTA GGGGTTAA
GGGGTTGA GGGGTTGG GGGGTTGT GGTGGGGG GTAAGGGG GTAGGGGA GTCGGGGG GTCGGGGT GTGAGGGG GTGGGGGA GTGGGGGC
GTGGGGGG GTGGGGGT GTTAGGGG GTTGGGGG GTTGGGGT GTTTGGGG TAACGGGG TAAGGGGA TAAGGGGC TAAGGGGG TAATGGGG
TACGGGGA TACGGGGG TAGGGGCG TAGGGGGA TAGGGGGG TAGGGGGT TATCGGGG TATGGGGG TCACGGGG TCAGGGGA TCAGGGGG
TCATGGGG TCCGGGGG TCCTGGGG TCGAGGGG TCGGGGAA TCGGGGAC TCGGGGGA TCGGGGGC TCGGGGGG TCGGGGTA TCGGGGTG
TCGGGGTT TCTAGGGG TCTCGGGG TCTGGGGA TCTGGGGG TGAAGGGG TGACGGGG TGAGGGGC TGAGGGGG TGATGGGG TGCCGGGG
TGCGGGGG TGCTGGGG TGGGGAAA TGGGGATC TGGGGCCA TGGGGCCT TGGGGCGC TGGGGCGG TGGGGGAA TGGGGGAC TGGGGGAG
TGGGGGAT TGGGGGCA TGGGGGCC TGGGGGCT TGGGGGGA TGGGGGGC TGGGGGGG TGGGGGGT TGGGGGTA TGGGGGTC TGGGGGTT
TGGGGTAC TGGGGTGA TGGGGTTG TGTCGGGG TGTGGGGA TGTGGGGG TGTTGGGG TTACGGGG TTAGGGGC TTAGGGGG TTGGGGAA
TTGGGGAC TTGGGGAT TTGGGGCA TTGGGGCG TTGGGGGA TTGGGGGC TTGGGGGG TTGGGGGT TTGGGGTA TTGTGGGG TTTGGGGA
TTTGGGGC TTTGGGGG

10.1 Issue 1 with Poly-C and Poly-G Motifs

There is a known PBM artifact whereby the sequence composition of the probes affects the degree of ssDNA-synthesis for each probe, and this can never be removed fully from the experimental design. Attempts can be made to try to normalize out this artifact, but no method is perfect. If the proteins are not properly prepped and will stick to most any DNA oligo, then the signal partially or fully

degenerates to a preference between short and long probes, or single- and double-stranded DNA. This signal may manifest itself into a poly-C or poly-G motif that the protein presumably prefers over all other double-stranded motifs. For this reason, poly-C and poly-G motifs that have a low self-R^2 (i.e. - explain a low percentage of the variance using the biophysical model) should be treated as suspect (see Figure S4).

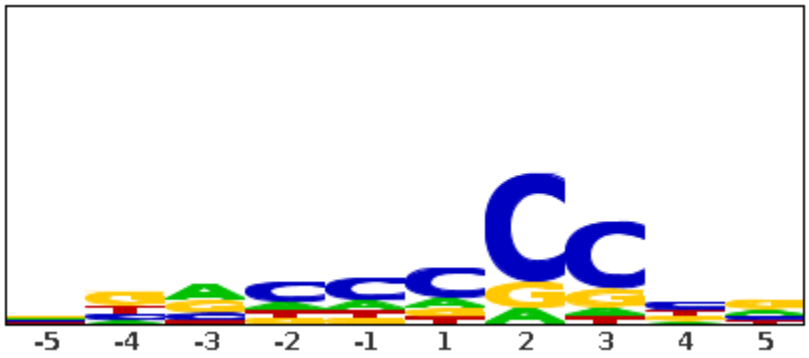
10.2 Issue 2 with Poly-C and Poly-G Motifs

Another problem arises if the protein really does prefer a poly-C or poly-G motif. (e.g. – Zif268 binds to a poly-G motif and αCP3 binds to a poly-C motif. However, we must remember that the popular orientation of a dual-strand binding motif is arbitrary and that any motif that is poly-G on one strand is necessarily poly-C on the other.) After the de Bruijn sequences are modified, the positive strand is greatly depleted of poly-G sequences, and likewise the negative strand is greatly depleted of poly-C sequences. This can induce a strong strand bias in the binding data (see Figure S4) and reduces the statistical accuracy of the affinity model. This also makes it difficult to impossible to properly calculate the relative Positional Bias Profiles for both strands when one of the strands is greatly depleted of the binding motif. In short, the strand that is greatly depleted of the binding motif cannot be trusted since it contains very low binding signal above the noise, and therefore only the other strand has a favorable signal-to-noise ratio.

10.3 Partial Solution

To deal with these issues, FeatureREDUCE detects whether the highest-affinity motif contains 5 or more consecutive Gs or Cs. If the highest-affinity motif contains 5 or more consecutive Cs then only the positive strand is used to generate the biophysical model. Likewise, if the highest-affinity motif contains 5 or more consecutive Gs then only the negative strand is used to generate the model (see Figure S7). In either case, FeatureREDUCE will warn you that there are possible “issues” with this motif.

This strategy removes the problematic strand from the model, but cannot remove the noise from that bad strand from the data. Therefore, it's unavoidable that these models will have a less favorable signal-to-noise ratio compared to other models that can effectively use the binding signal from both strands. Finally, when under low signal-to-noise ratios, FeatureREDUCE may not be able to accurately calculate the dinucleotide affinity corrections. In this case, the biophysical model will consist of just the positional-independence model (PSAM) and the Positional Bias Profile.



Above is an example Sequence Logo for a preferred poly-C motif that was generated by FeatureREDUCE from real PBM data using only the positive strand. With this inferred binding motif, the positive-strand self-R^2 was .17 while the negative strand self-R^2 was .05, which is indicative of a strong strand bias. Because of the low self-R^2s this biophysical model suffers from a poor signal-to-noise ratio on the negative strand. Also, with the apparent preference for poly-Cs, it's possible that this motif is partially (or fully) confounded by the PBM artifacts that resulted from the unequal lengths of the PBM probes (due to biases in the ssDNA-synthesis step).

11.0 Configuration File – `INIT_FILE`

The configuration (init) file contains many options that affect the training and prediction methods of FeatureREDUCE. Most of them should not be changed by the end-user. However, here are listed some important parameters in the `INIT_FILE` that the user may consider changing to suite his/her needs.

11.1 Output Parameters

```
[OUTPUT]
Directory = results.dream
DisplayLogos = yes
```

11.2 All-Kmers Model Parameters

```
[AFFINITY MODELS]
GetAllKmerModel = Yes
KmerModelLengths = 4 5 6 7 8
KmerModelPosBiases = uniform uniform uniform uniform uniform
NonNegativeRegression = false false false false false
IncludeRevComps = false false false false false
TopPercentageKmersForScoring = 1.0 1.0 1.0 0.05 0.02
IncludeNegativeInTopPercentage = Yes Yes Yes Yes Yes
```

11.3 FSAM Model Parameters

```
[REGRESSION]
RankedKmerForSeed = 1
PalindromicSeedThresh = 1.1

[AFFINITY MODELS]
SeedMotifMinExtensionLength = 1
SeedMotifMaxExtensionLength = 1
Get_NNDD_AdditiveAffinities = Yes
```