

SUPPLEMENTAL INFORMATION

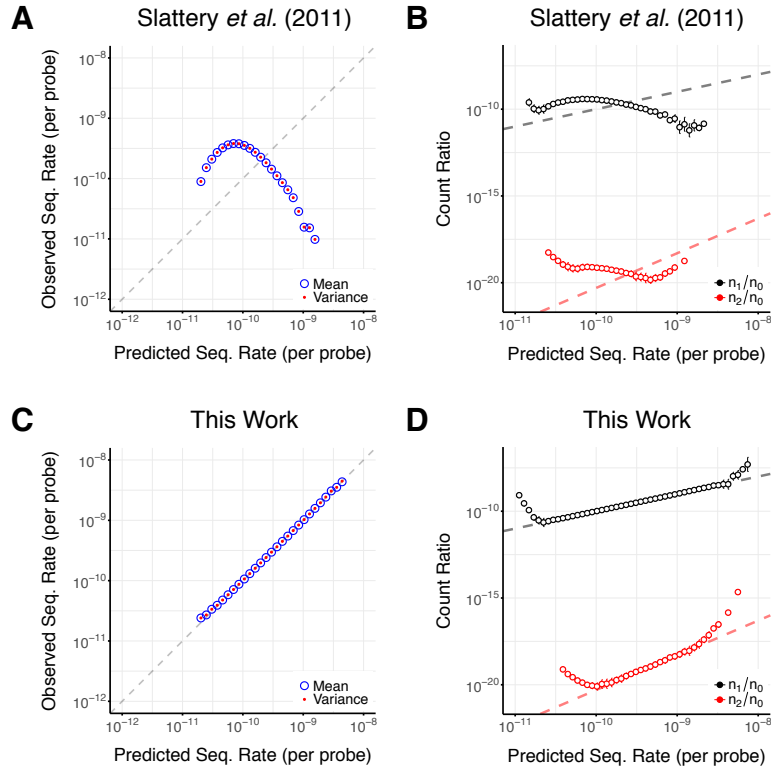
Accurate and Sensitive Quantification of Protein-DNA Binding Affinity

Chaitanya Rastogi, H. Tomas Rube, Judith F. Kribelbauer, Justin Crocker, Ryan E. Loker,
Gabiella D. Martini, Oleg Laptenko, William Freed-Pastor, Carol L. Prives, David L. Stern,
Richard S. Mann, Harmen J. Bussemaker

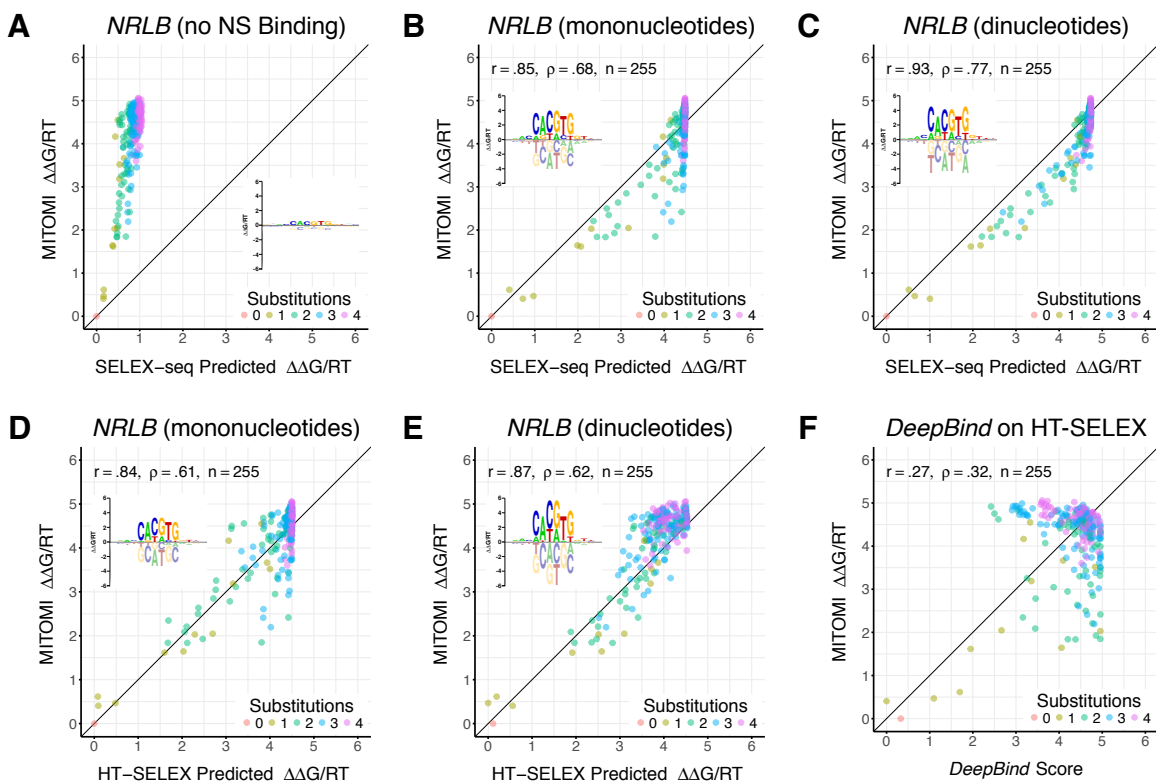
Part A – Supplemental Figures.....	2
Part B – Supplemental Methods.....	36

PART A

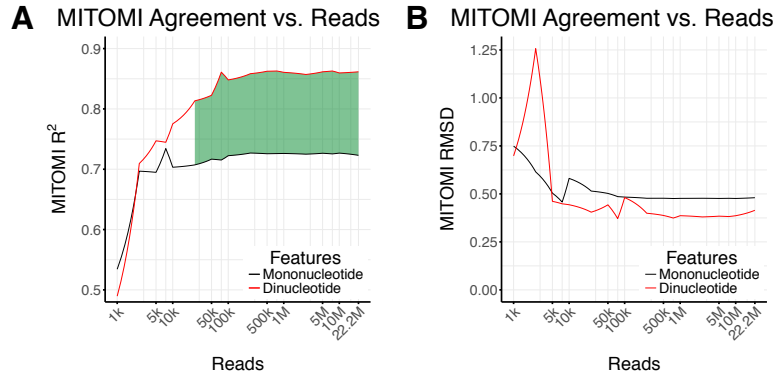
SUPPLEMENTAL FIGURES



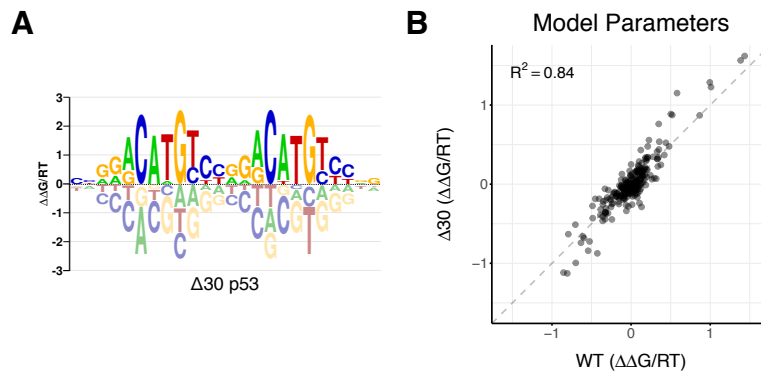
Supplemental Figure S1. Characterization of models representing biases in R0. Methods to assess the ability of the Markov model approach of Slattery, *et al.* (1) and the improved bias model from this work to accurately parameterize biases in R0 at the level of the entire probe. Both models are trained and tested on R0 data from Slattery, *et al.* (1). **(A)** Probes are binned according to their model predicted sequencing rate (x-axis), allowing the computation of the observed probe count mean (blue circles) and variance (red dots) in each bin. Expected mean and variance (grey dashed line) assumes Poisson statistics. **(B)** Probes are binned according to their model predicted sequencing rate (x-axis), allowing the computation of the number of times every probe within a bin was observed (n_0 : no counts, n_1 : one count, n_2 : two counts); error bars are indicated by vertical lines. Expected ratios (dashed lines) assume Poisson statistics. **(C-D)** Idem, for the improved bias model used in this work.



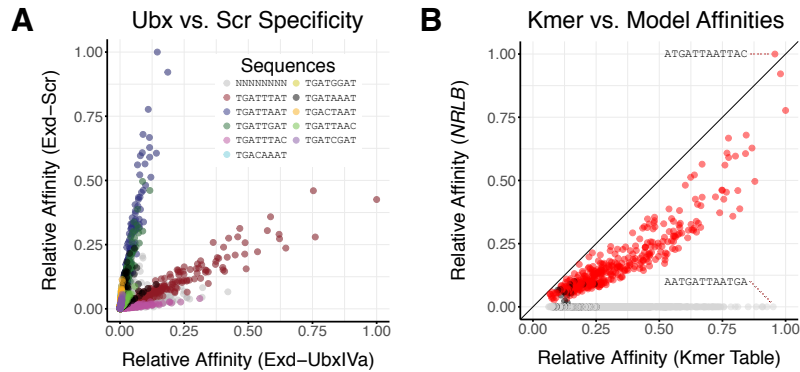
Supplemental Figure S2. Model predictions vs. MITOMI derived $\Delta\Delta G$'s for human MAX. Comparison between MITOMI-derived binding free energies (y-axis) for 255 different DNA ligands and the same values as predicted by *NRLB* models (x-axis) (**A-E**) or by a *DeepBind* model trained on R1 HT-SELEX data from (2) (**F**). MITOMI binding free energy predictions made by *NRLB* models trained on R1 SELEX-seq data using (**A**) mononucleotide features and excluding nonspecific binding, (**B**) mononucleotide features including nonspecific binding, and (**C**) mono- and di-nucleotide features with nonspecific binding. MITOMI binding free energy predictions made by *NRLB* models with nonspecific binding trained on R1 HT-SELEX data using (**D**) mononucleotide features and (**E**) mono- and di-nucleotide features. In panels A-E, the energy logo representation (3) of the *NRLB* model is inset. Color denotes the number of substitutions relative to the optimal sequence. Model predictions are made using the full sequence used in the MITOMI assay. MITOMI sequence and binding free energy data from (4). In panels A-C, the R1 SELEX-seq data for human MAX was generated as part of (5) but sequenced as part of the present study. In panels D-F, the R1 HT-SELEX data for human MAX from Jolma, *et al.* (6). Pearson (r) and Spearman rank correlation (ρ), along with the number of data points (n), are indicated in panels B-F.



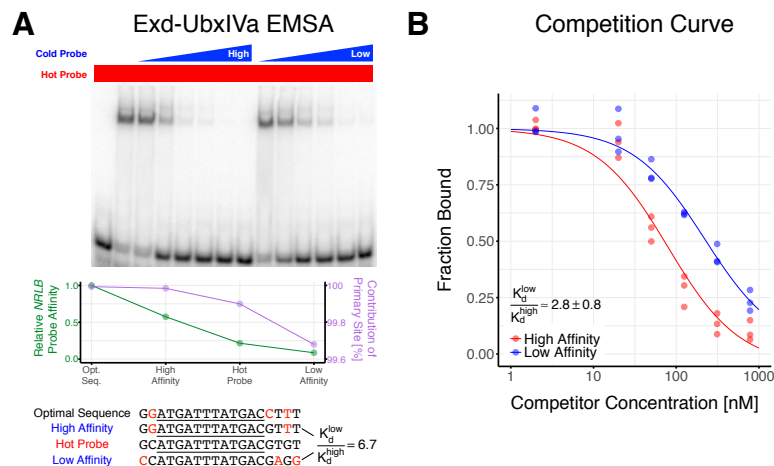
Supplemental Figure S3. Relationship between *NRLB* performance and the number of reads in the R1 library. **(A)** Agreement between MITOMI-derived binding free energies and relative affinities predicted by mononucleotide (black) and dinucleotide (red) *NRLB* models trained on R1 SELEX-seq data for human MAX at varying degrees of subsampling from the original dataset (22M reads). The area between mononucleotide and dinucleotide curves is shaded green if the difference between the mononucleotide and dinucleotide R^2 is statistically significant ($p < 0.05$). **(B)** Root-mean-square deviation (RMSD) between MITOMI-derived binding free energies and relative affinities predicted by the models in A. The R1 SELEX-seq data for human MAX was generated as part of (5) but sequenced as part of the present study.



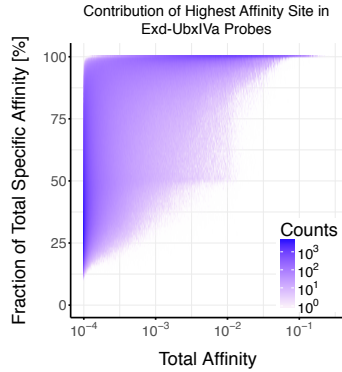
Supplemental Figure S4. *NRLB* model fit to $\Delta 30$ p53. **(A)** Energy logo representation for an *NRLB* model with dinucleotide features trained on R1 SELEX-seq data for C-terminally truncated ($\Delta 30$) p53. **(B)** *NRLB* model parameters for full-length p53 (cf. Figure 2D) are similar to those of the $\Delta 30$ model.



Supplemental Figure S5. *NRLB* models succinctly capture information within oligomer tables. **(A)** Comparison of 12mer affinities as predicted by truncated versions of the Exd-Scr and Exd-UbxIVa *NRLB* models in Figure 3A. Points are colored by the presence of known Exd-Hox 8mer binding sites from (1) and reveal similar differences in binding preferences for the two proteins which were originally uncovered in the same study. **(B)** Comparison of Exd-Scr 12mer affinities as predicted by the same truncated model in panel A and by an oligomer enrichment table built on the same data using the method of Slattery, *et al.* (1). All 12-mers for which oligomer enrichment table predictions exist are shown; 12-mers are shown in grey if they contain a match to the Exd-Scr 8-mer (WRATWDAT) and in red if the match occurs at the correct offset (NWRATWDATNNN). The comparison highlights the inconsistent offsets and biased affinity estimates associated with oligomer enrichment tables. R1 Exd-Hox SELEX-seq data from Slattery, *et al.* (1) was used in the construction of the enrichment table.



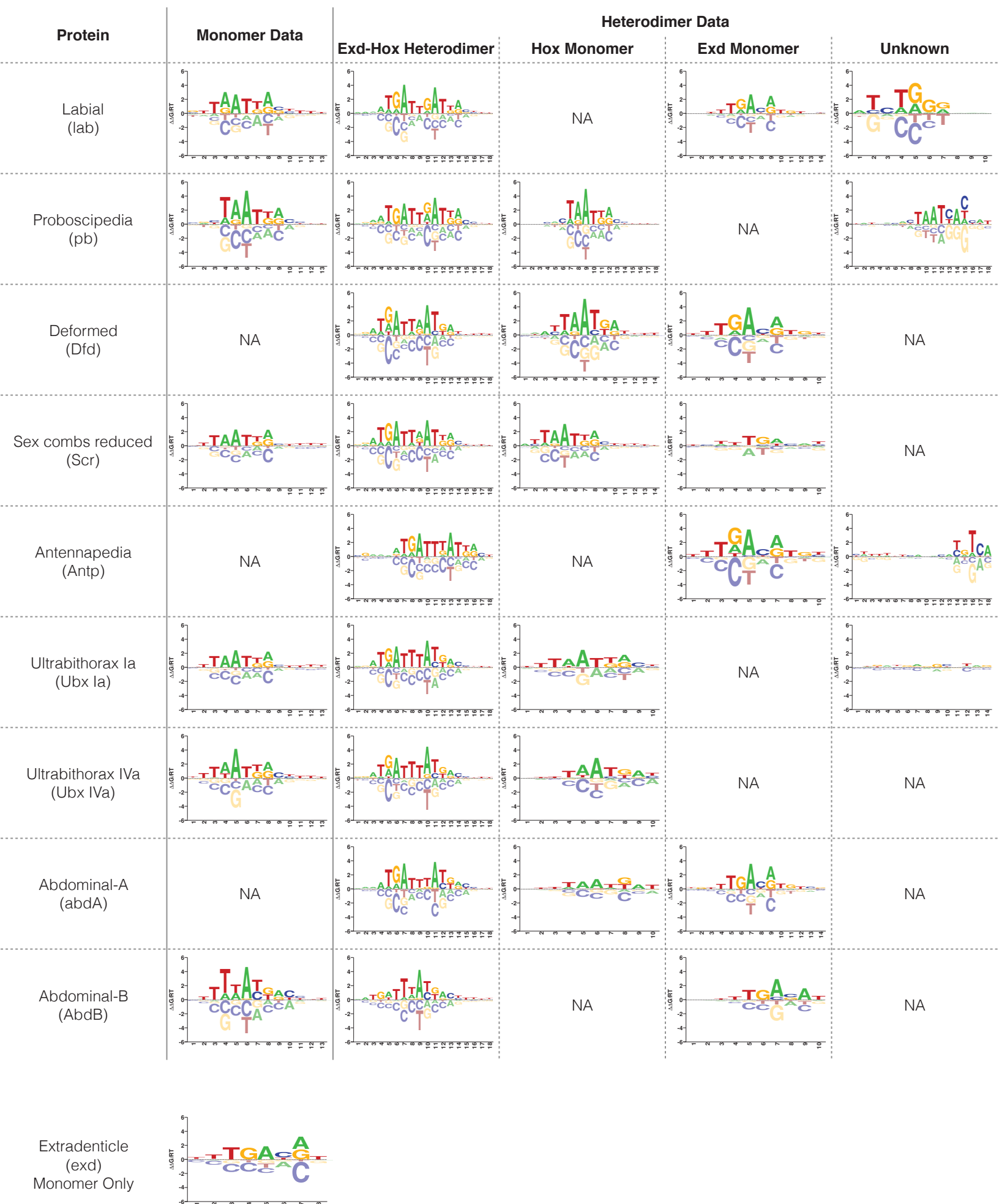
Supplemental Figure S6. Competitive EMSAs confirm additional flanking specificity in Exd-UbxIVa. **(A)** Competitive EMSA used to confirm the additional flanking specificity outside the 12mer core in Exd-UbxIVa with probe relative affinities predicted by our *NRLB* model for Exd-UbxIVa (cf. Figure 3A). **(B)** Binding curves fit to quantified competitive EMSA data (panel A and Table S2) confirm the contribution of flanking basepairs outside the 12mer core in Exd-UbxIVa predicted by the *NRLB* model.

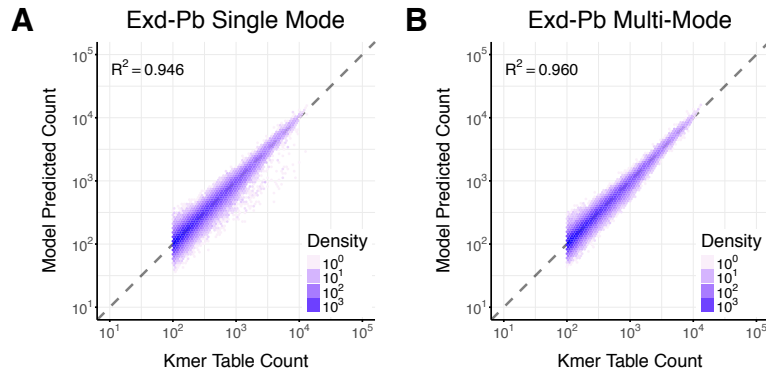


Supplemental Figure S7. Multiple suboptimal sites contribute to the binding affinity of low-affinity probes. Density plot showing the relationship between model-predicted total affinity (specific and non-specific) and the fraction of this affinity contributed by the highest specific affinity site in the probe (y-axis) for all observed R1 probes using a dinucleotide *NRLB* model for Exd-UbxIVa (cf. Figure 3A). R1 SELEX-seq data for Exd-UbxIVa from Slattery, *et al.* (1).

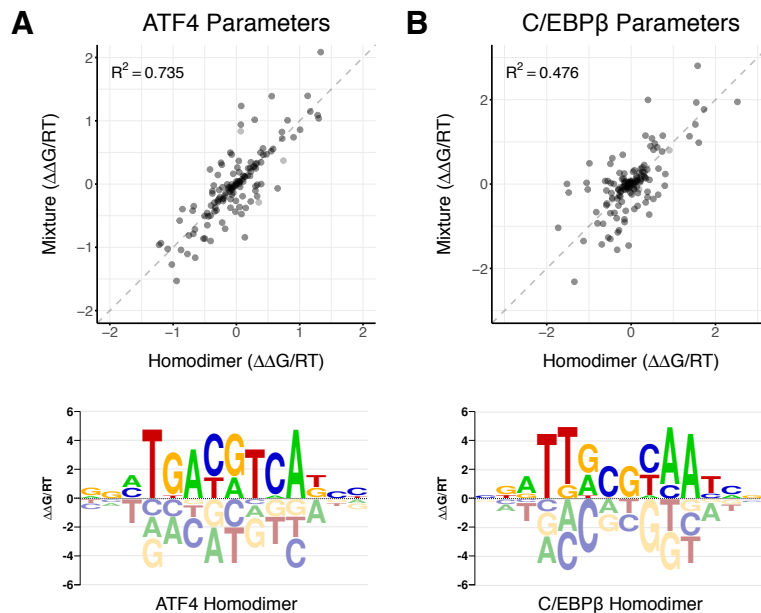
Figure shown on Subsequent Pages

Supplemental Figure S8. Overview of *NRLB* models for Hox monomers and Exd-Hox heterodimers. Hox monomer motifs are derived from 13 bp dinucleotide single-mode models fit on R1 Hox monomer data from Slattery, *et al.* (1). R1 Exd-Hox heterodimer data also from Slattery, *et al.* (1) (except Pb, which is from this work) was used to build either two or three mode dinucleotide models of varying footprint sizes based on heuristic criteria (see Methods). Single-mode dinucleotide model for Exd monomers was built using R1 Exd monomer data from this work.

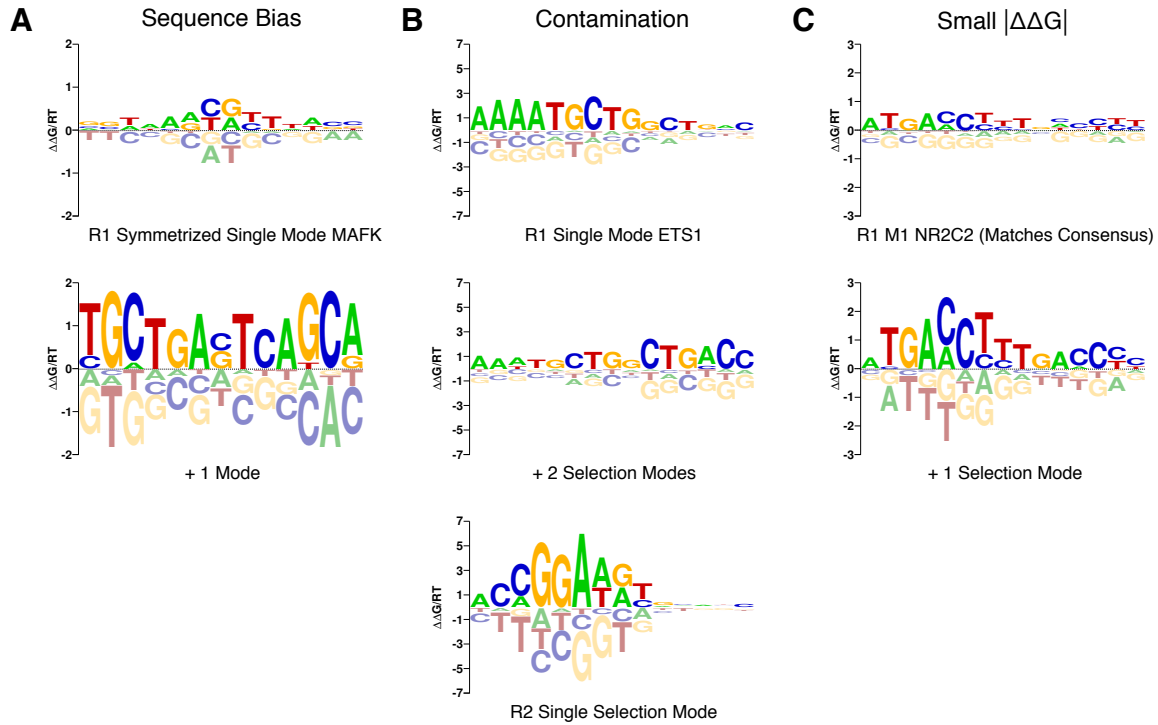




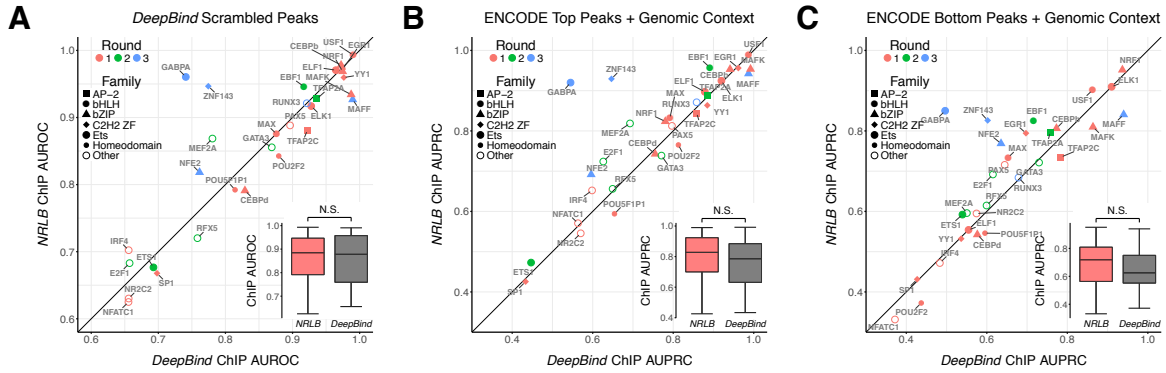
Supplemental Figure S9. Performance of single mode and multiple binding mode models. **(A)** Density plot of observed 10mer counts with 100 or more observations ($n = 299k$) in R1 SELEX-seq data for Exd-Pb as predicted by a single-mode dinucleotide NRLB model fit to the same data (c.f. Figure 4A). The model accurately represents the observed data except for a sparse cloud of off-diagonal points. **(B)** Idem, for a three-mode dinucleotide NRLB model fit to the same data (c.f. Figure 4A). This model resolves the discrepancy in Panel A, explaining the data with higher, statistically significant accuracy ($p < 1 \times 10^{-16}$, Fischer's r-to-z transform).



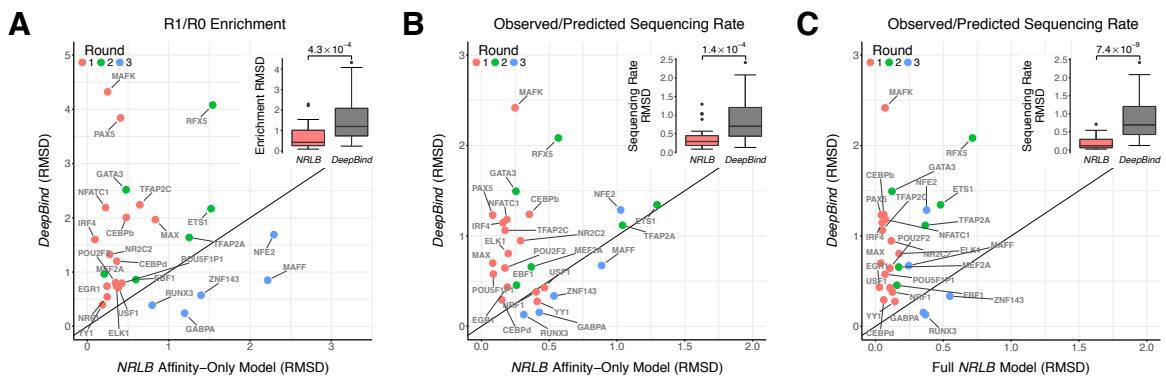
Supplemental Figure S10. ATF4 and CEBP β motifs. **(A)** (Top) Comparison between *NRLB* model parameters for an ATF4 homodimer model built from R1 SELEX-seq data for ATF4 only (x-axis and energy logo representation below) and parameters from an ATF4 homodimer model built using SELEX-seq data from a mixture of ATF4 and C/EBP β (cf. Figure 4B). **(B)** Idem, for C/EBP β homodimer model built from R1 SELEX-seq data for C/EBP β only.



Supplemental Figure S11. Contaminants detected in HT-SELEX data. **(A)** (Top) The best symmetric single-mode model and (Bottom) the primary mode of the best two-mode symmetric model fit to R1 HT-SELEX data for MAFK (6). Single-mode NRLB models can discover motifs that appear to represent sequence bias rather than the binding specificity of the TF in question. In some cases, additional binding modes can account for these biases. **(B)** (Top) Best single-mode model fit and (Middle) the primary mode of the best three-mode model fit to R1 HT-SELEX data for ETS1 (6). (Bottom) Best single-mode model fit on R2 of the same dataset. In line with other analyses (2), some HT-SELEX datasets appear to contain contaminants or display poor enrichment, which forced us to use later-round data to build models that capture the binding specificity of the TF in question. To ensure that the correct motif was discovered, the primary mode of each model was checked to see if it matches a known consensus sequence for the TF. **(C)** (Top) The best single-mode model and (Bottom) primary mode of the best two-mode model fit to R1 HT-SELEX data for NR2C2 (6) that matches its consensus (AGGTCA). In some cases, *NRLB* models contain modes that match the expected consensus sequence for the given TF but display unusually low specificity. In such cases, multiple binding modes can rescue the model. In the above analyses, only mononucleotide models trained on HT-SELEX data were used, and “best” refers to the highest-likelihood model from a collection trained with different hyperparameters (see Methods).



Supplemental Figure S12. ChIP-seq classification performance of *NRLB* and *DeepBind* models trained on HT-SELEX data. Each point represents the performance of models constructed using the respective algorithms when classifying ENCODE ChIP-seq peaks, as measured by the appropriate area under the curve (AUC) metric for the particular method used to construct positive and negative sets (see Methods). **(A)** Area under the receiver operating characteristic curve (AUROC) comparison using the “*DeepBind* method”. **(B)** Area under the precision-recall curve (AUPRC) comparison using the “ENCODE Top 500 method.” **(C)** Similar analysis using the “ENCODE Bottom 500 method.” Statistical significance was assessed using a Mann-Whitney U-test. *NRLB* models match *DeepBind* model performance when classifying ENCODE ChIP-seq peaks using models trained on HT-SELEX data, regardless of the AUC metric and method used to construct positive and negative sets.

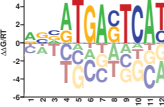
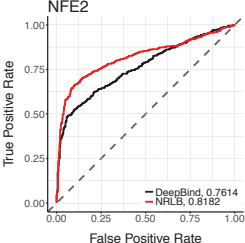
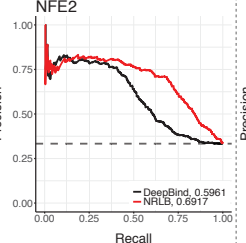
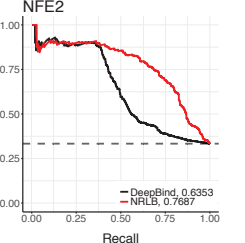
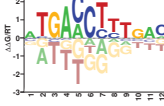
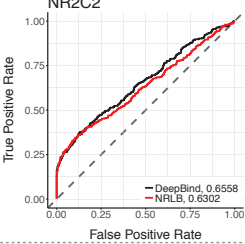
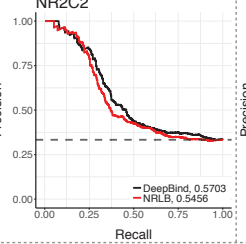
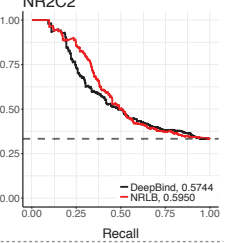
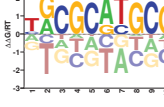
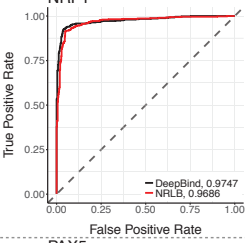
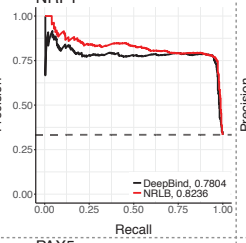
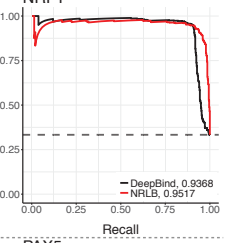

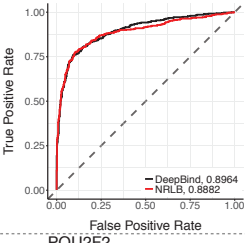
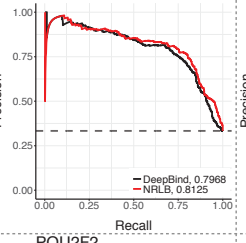
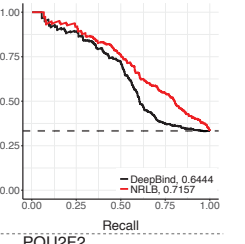
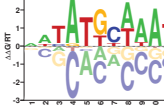
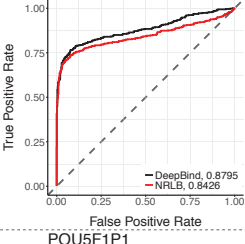
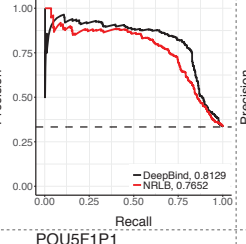
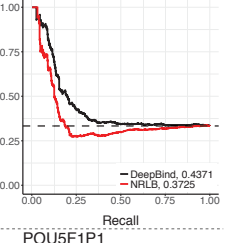
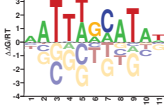
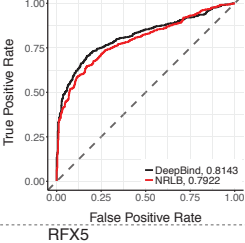
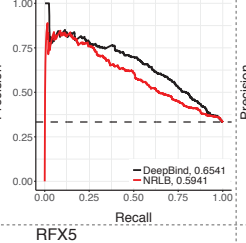
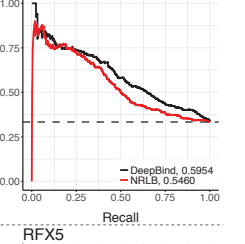
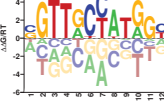
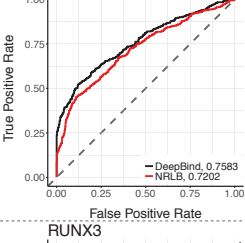
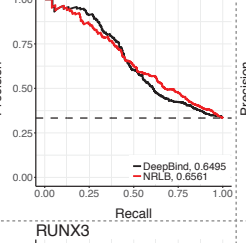
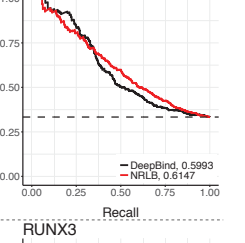
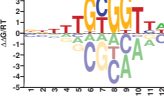
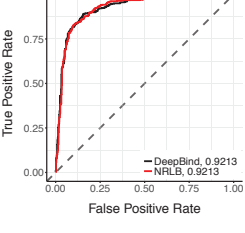
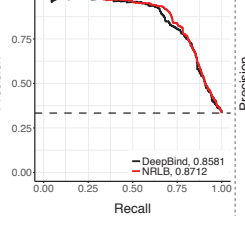
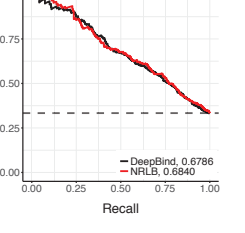


Supplemental Figure S13. Quantitative performance of *NRLB* and *DeepBind* models trained on HT-SELEX data. Each point represents the performance of models constructed using the respective algorithms on HT-SELEX data from (7) in explaining the observed R1 probe frequencies from a more deeply sequenced technical replicate of the same dataset (8), as measured by different metrics. **(A)** Root-mean-square deviation (RMSD) between the observed and predicted enrichment of probe counts from R0 to R1. **(B)** RMSD between observed and predicted R1 probe sequencing rate. In panels A and B, *NRLB* predictions were made only with the binding model, ignoring R0 bias. **(C)** The same analysis in panel B using the full *NRLB* model for predictions. See Methods for more details.

Figure Shown on Subsequent Pages

Supplemental Figure S14. *NRLB* models trained on HT-SELEX data. Collection of mononucleotide *NRLB* fits to HT-SELEX datasets used in the CHIP-seq peak classification analyses in Figures 2E and S12. Here, “round” refers to the HT-SELEX enrichment round used to train either model. “*DeepBind* Detectors” lists the number of motif detectors used in the corresponding model as reported by (2). “*NRLB* Modes” lists the number of binding modes used to learn the *NRLB* model. “*NRLB* Motif” displays the primary binding mode of the *NRLB* model trained on the HT-SELEX data. “ROC”, “Top PRC”, and “Bottom PRC” display the individual area under the curve (AUC) performance metric for both *DeepBind* and *NRLB* models using the “*DeepBind* Method,” the “ENCODE Top 500 Method,” and the “ENCODE Bottom 500 Method,” respectively.

Protein	Family	DeepBind Round	DeepBind Detectors	NRLB Round	NRLB Modes	NRLB Motif	ROC	Top PRC	Bottom PRC
C/EBP β	bZIP	4	10	1	1				
C/EBP δ	bZIP	2	7	1	2				
E2F1	E2F	3	11	2	2				
EBF1	bHLH	3	9	2	3				
EGR1	C2H2 ZF	2	11	1	2				
ELF1	Ets	6	9	1	1				
ELK1	Ets	2	16	1	2				
ETS1	Ets	3	16	2	1				

Protein	Family	DeepBind Round	DeepBind Detectors	NRLB Round	NRLB Modes	NRLB Motif	ROC	Top PRC	Bottom PRC
NFE2	bZIP	4	6	3	1				
NR2C2	Nuclear Receptor	3	15	1	2				
NRF1	bZIP/CNC	2	2	1	2				
PAX5	Paired Box	3	11	1	2				
POU2F2	Homeo-domain	2	13	1	2				
POU5F1P1	Homeo-domain	2	16	1	2				
RFX5	RFX	4	6	2	2				
RUNX3	Runt	3	9	3	2				

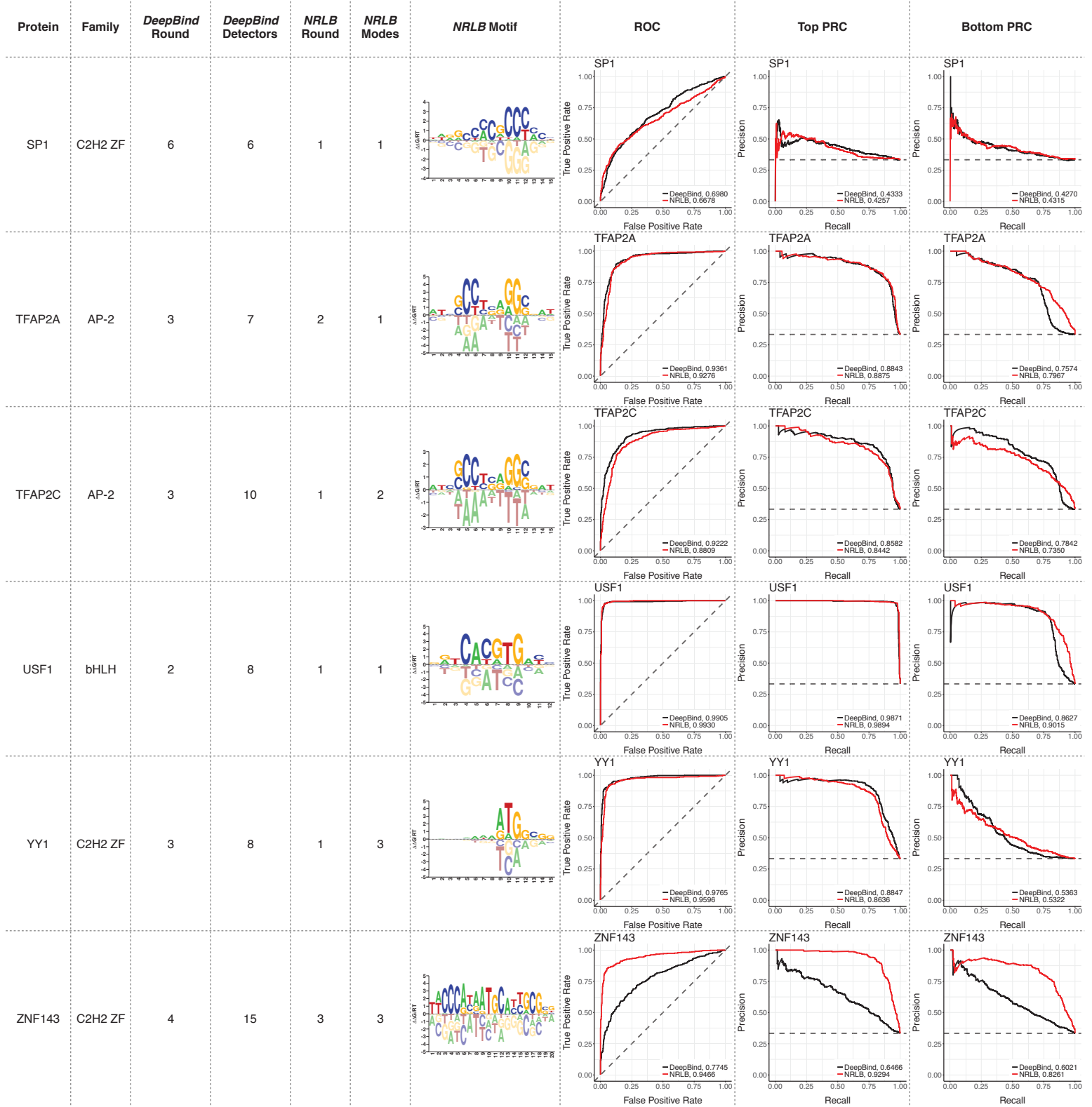
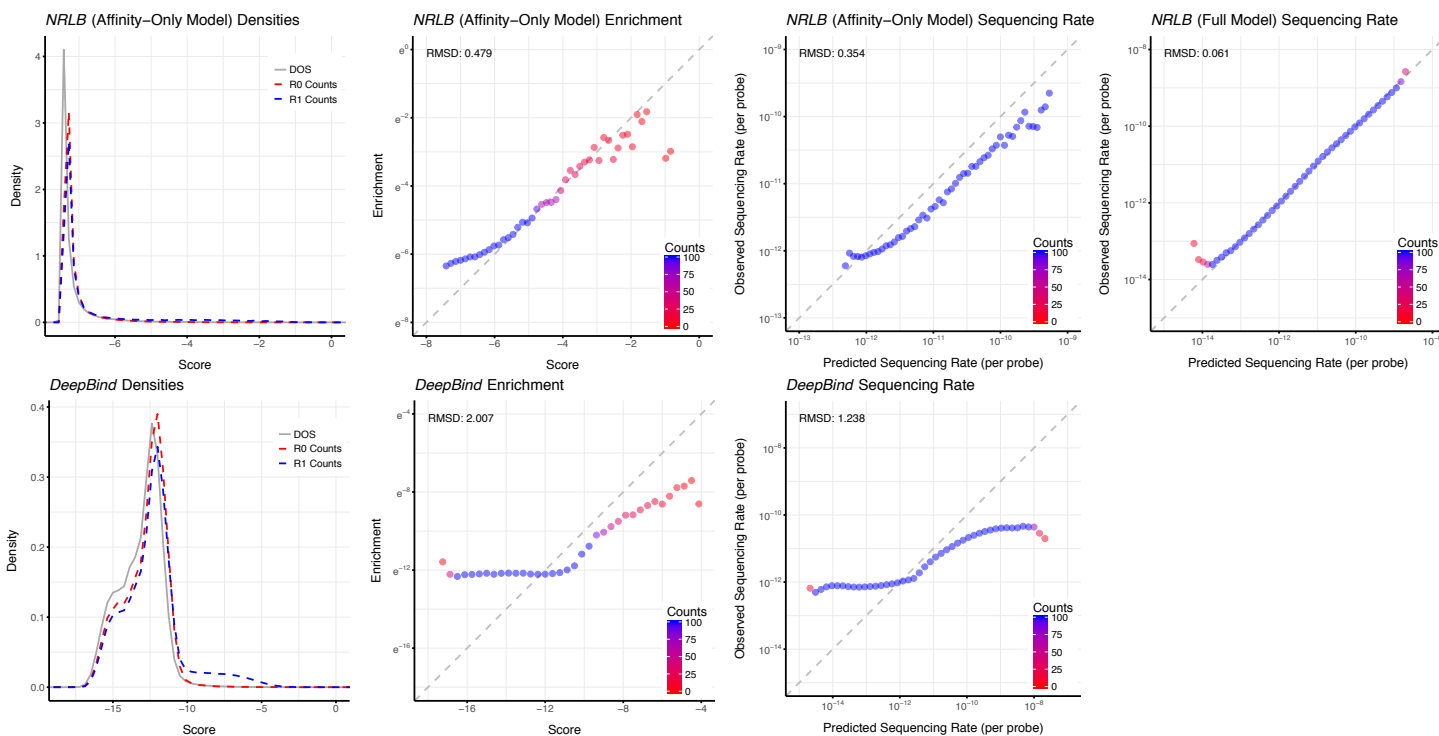


Figure Shown on Subsequent Pages

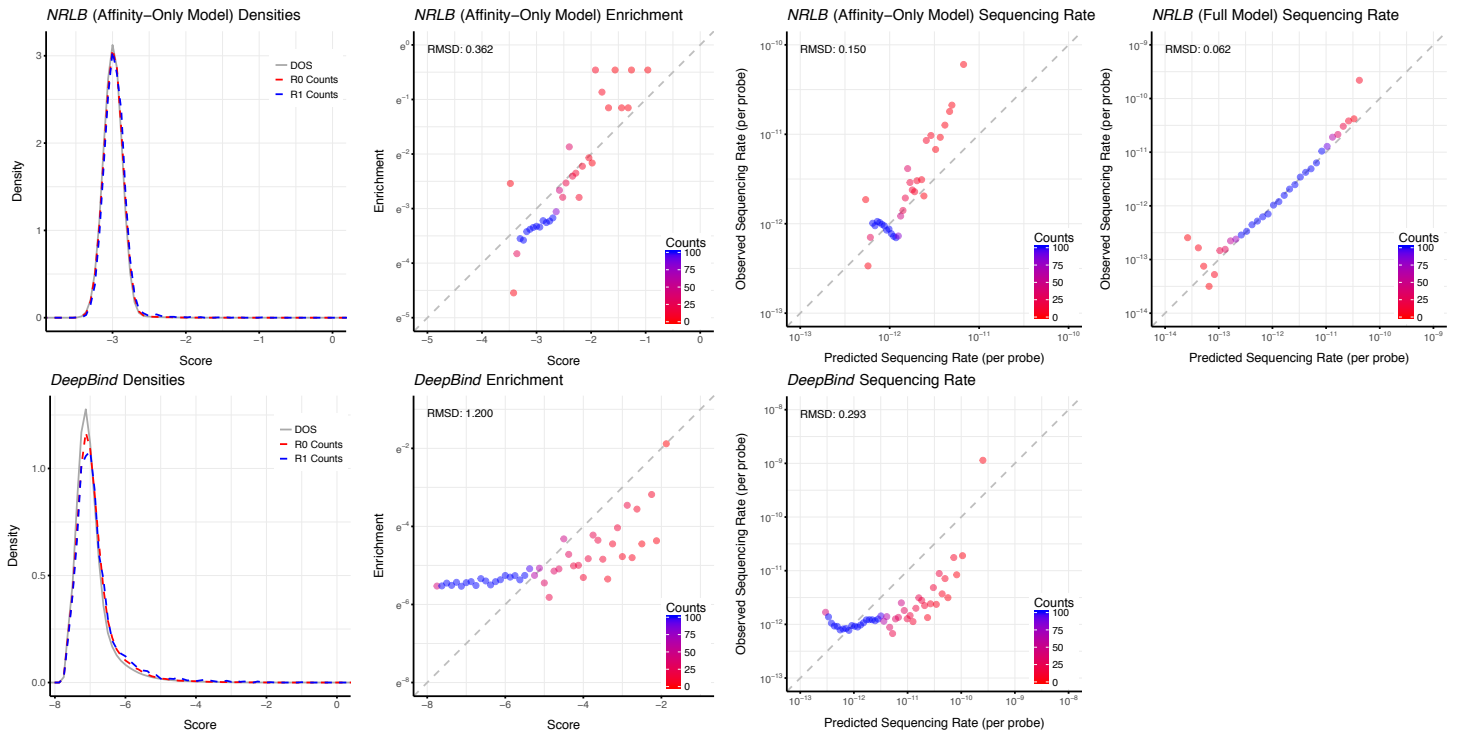
Supplemental Figure S15. Quantitative metrics for *NRLB* and *DeepBind* models trained on HT-SELEX data. Detailed plots underlying the analyses in Figures 2F and S13. The models used here are trained on the HT-SELEX dataset (7); *NRLB* models are shown in S13. Here, "*DeepBind* Round" and "*NRLB* Round" refers to the HT-SELEX enrichment round used to train either model. From left to right for each protein and model, the columns are: (i) the density of states for the given model, as estimated using the Wang-Landau method (9); (ii) a comparison of the observed and predicted enrichment of probe counts from R0 to R1; (iii) and (iv) a comparison of the observed and predicted R1 probe frequency. For columns (ii) and (iii), *NRLB* predictions were made only with the binding model, ignoring R0 bias. For columns (ii-iv), points are colored blue if they have at least 100 observations in each bin; if the number of observations is lower, the point's color follows the color scale. Root-mean-square deviation (RMSD) is computed only for points with at least 50 observations.

Densities, Enrichments, and Observed/Predicted Sequencing Rates for *NRLB* and *DeepBind* HT-SELEX Models

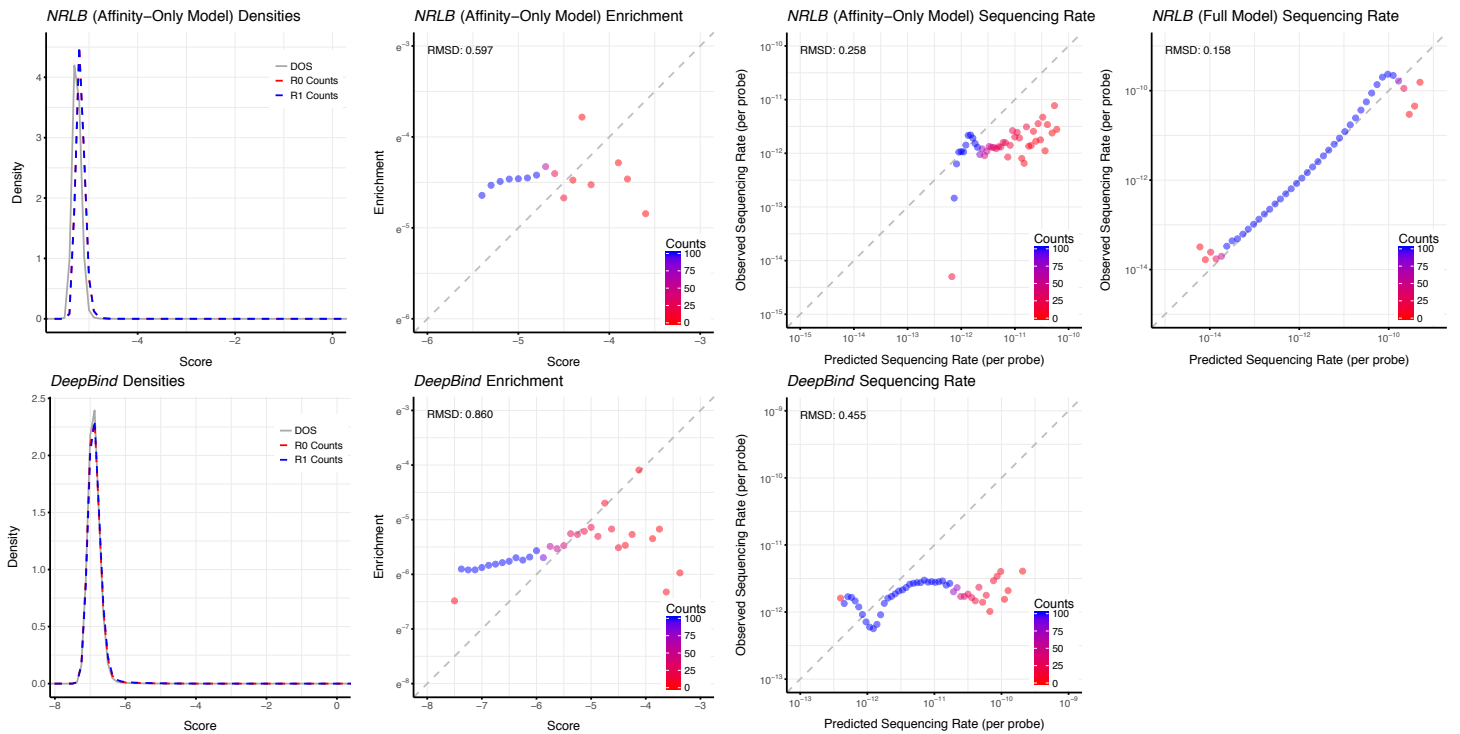
CEBPB; *DeepBind* Round: 4, *NRLB* Round: 1



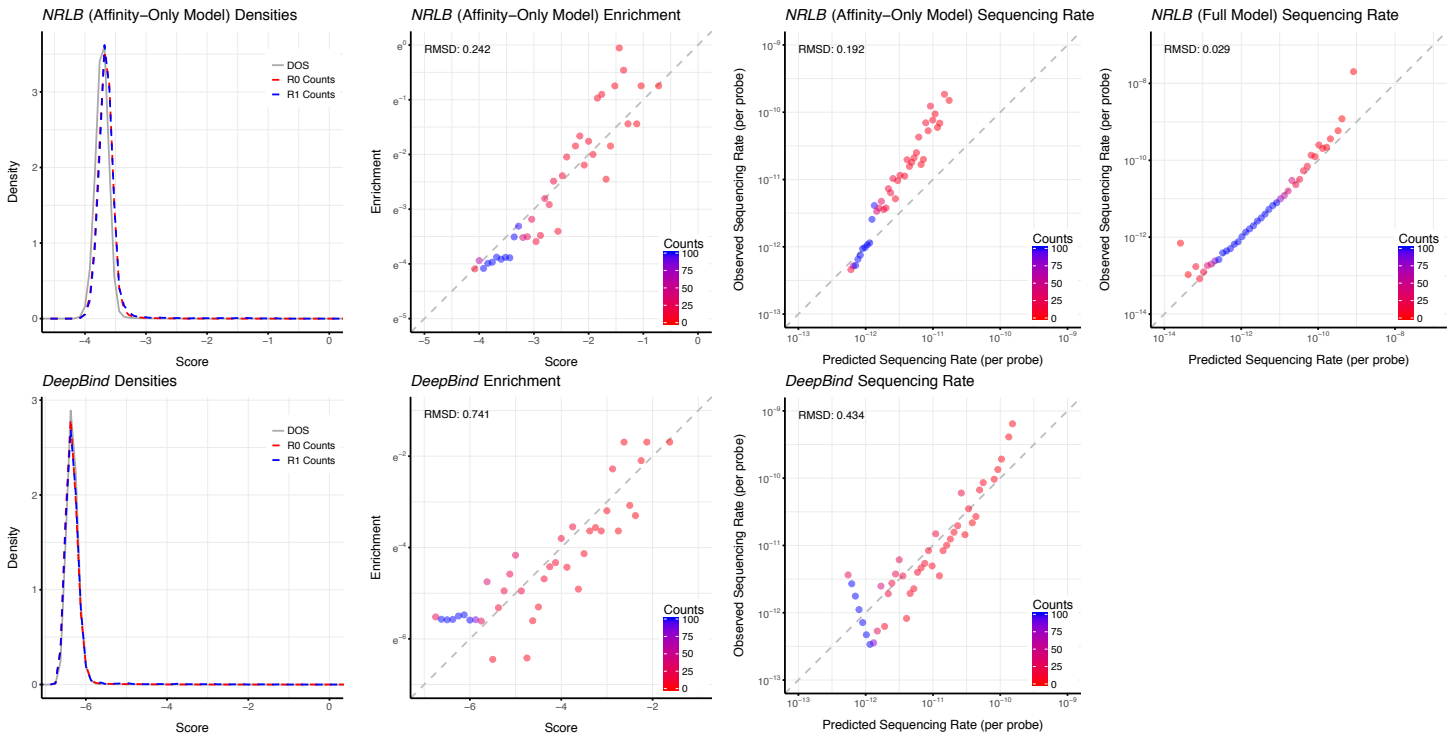
CEBPD; *DeepBind* Round: 2, *NRLB* Round: 1



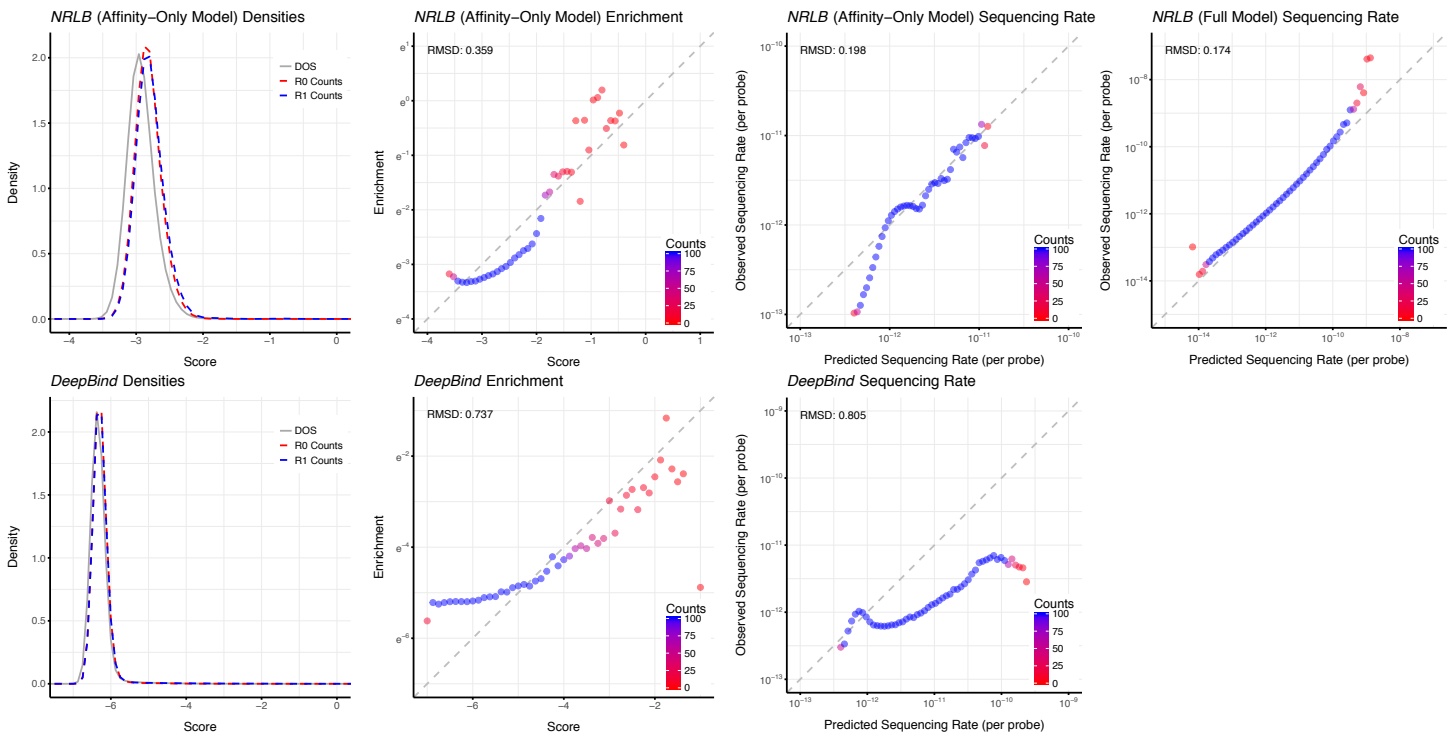
EBF1; *DeepBind* Round: 3, *NRLB* Round: 2



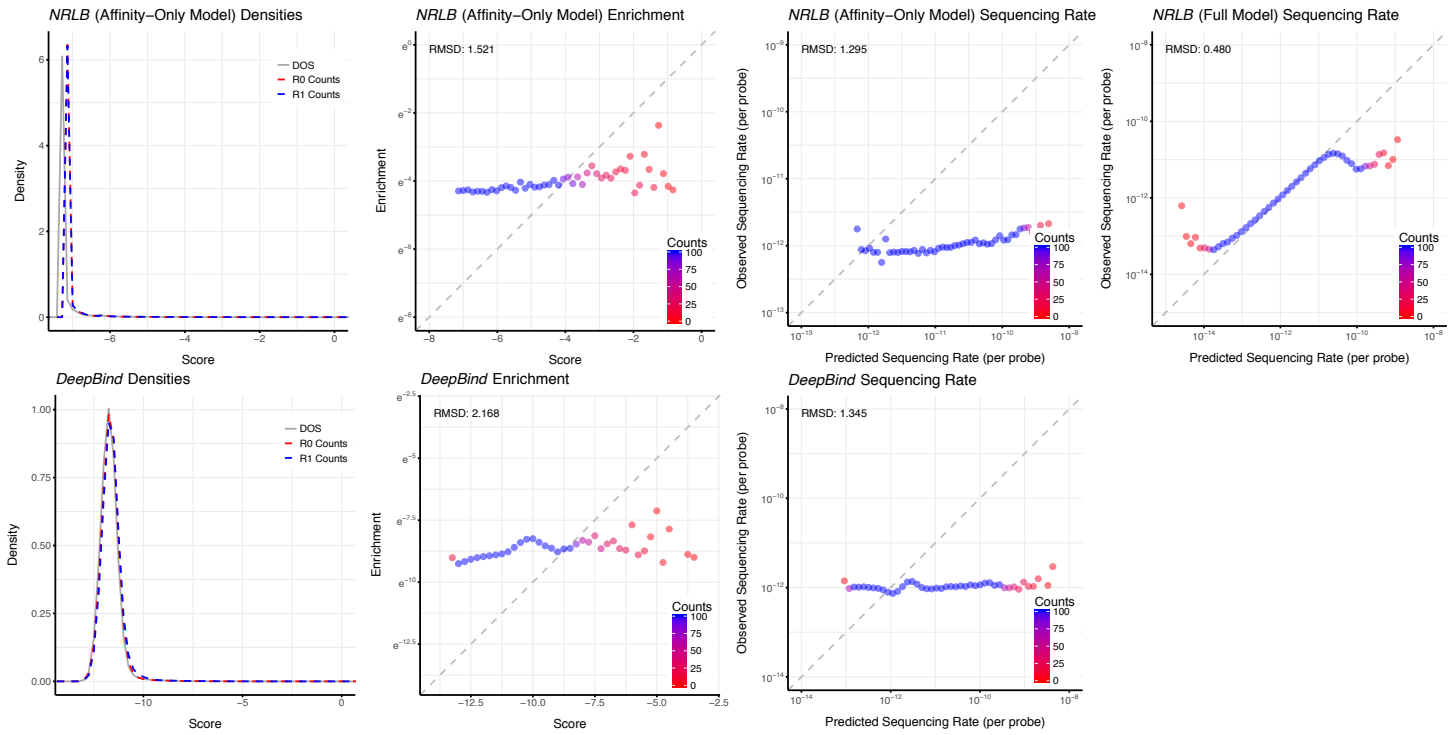
EGR1; *DeepBind* Round: 2, *NRLB* Round: 1



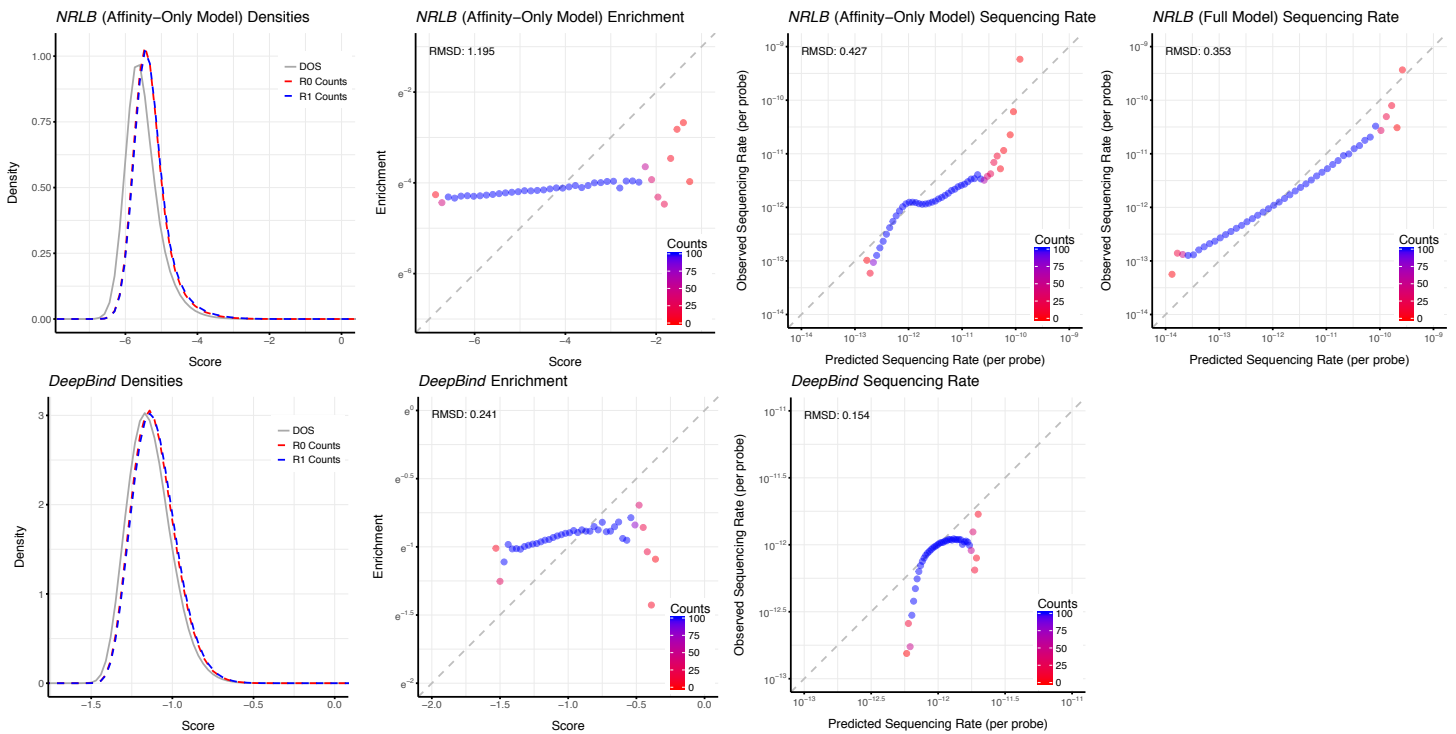
ELK1; *DeepBind* Round: 2, *NRLB* Round: 1



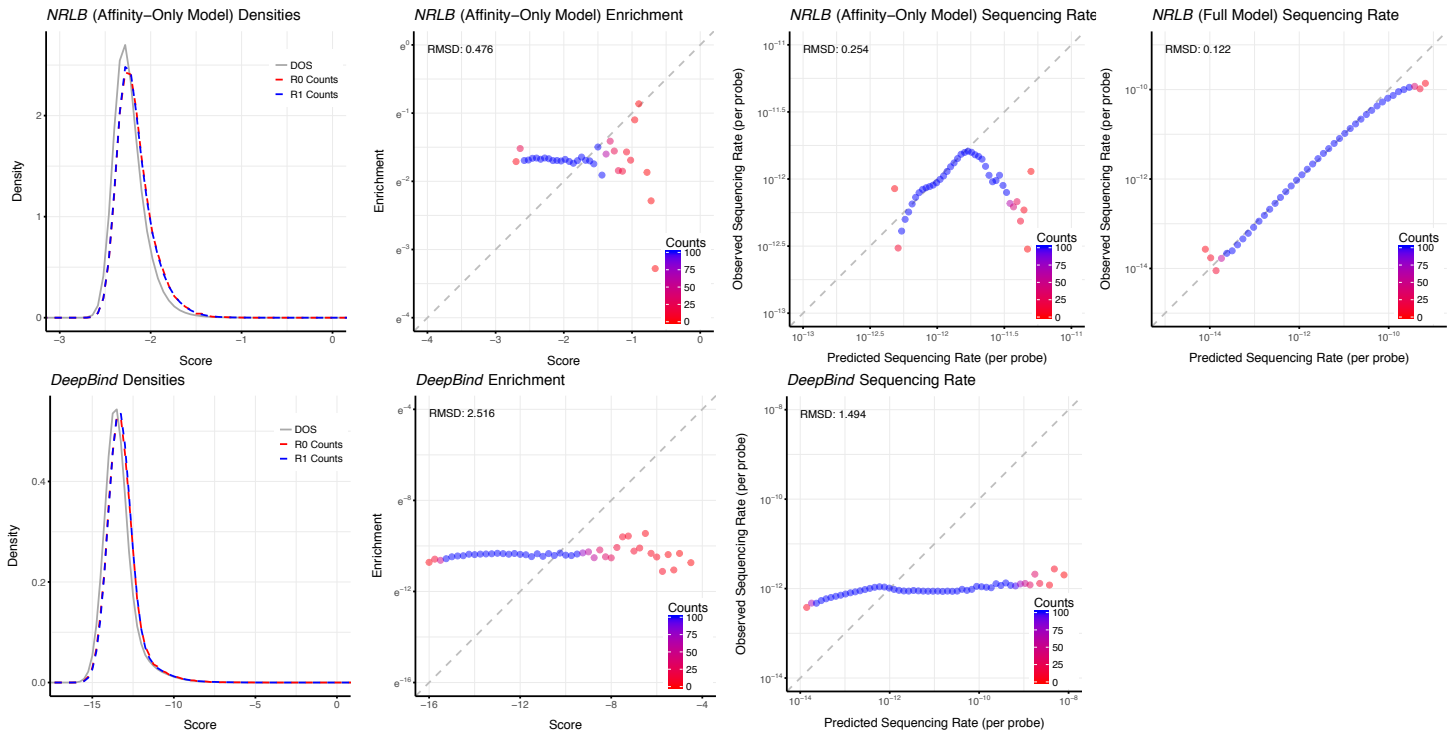
ETS1; DeepBind Round: 3, NRLB Round: 2



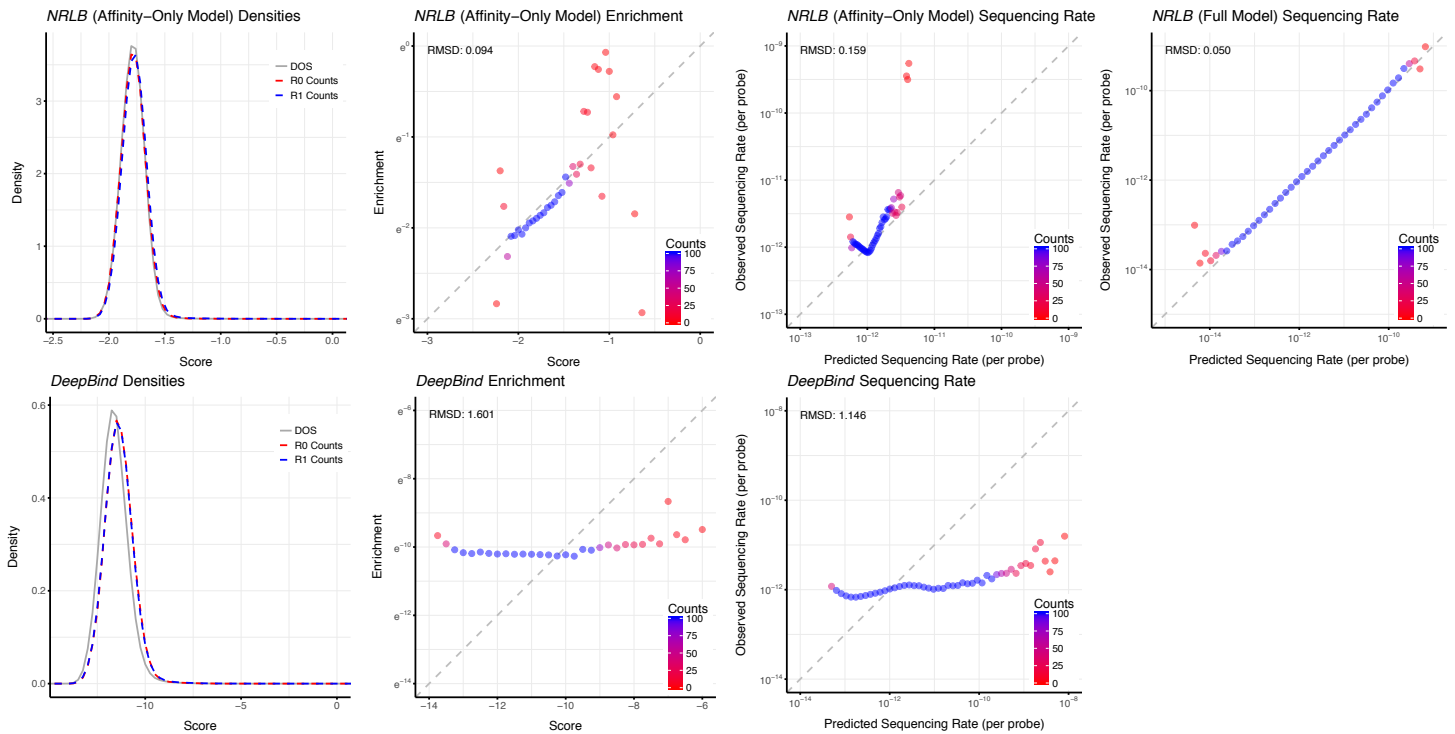
GABPA; DeepBind Round: 4, NRLB Round: 3



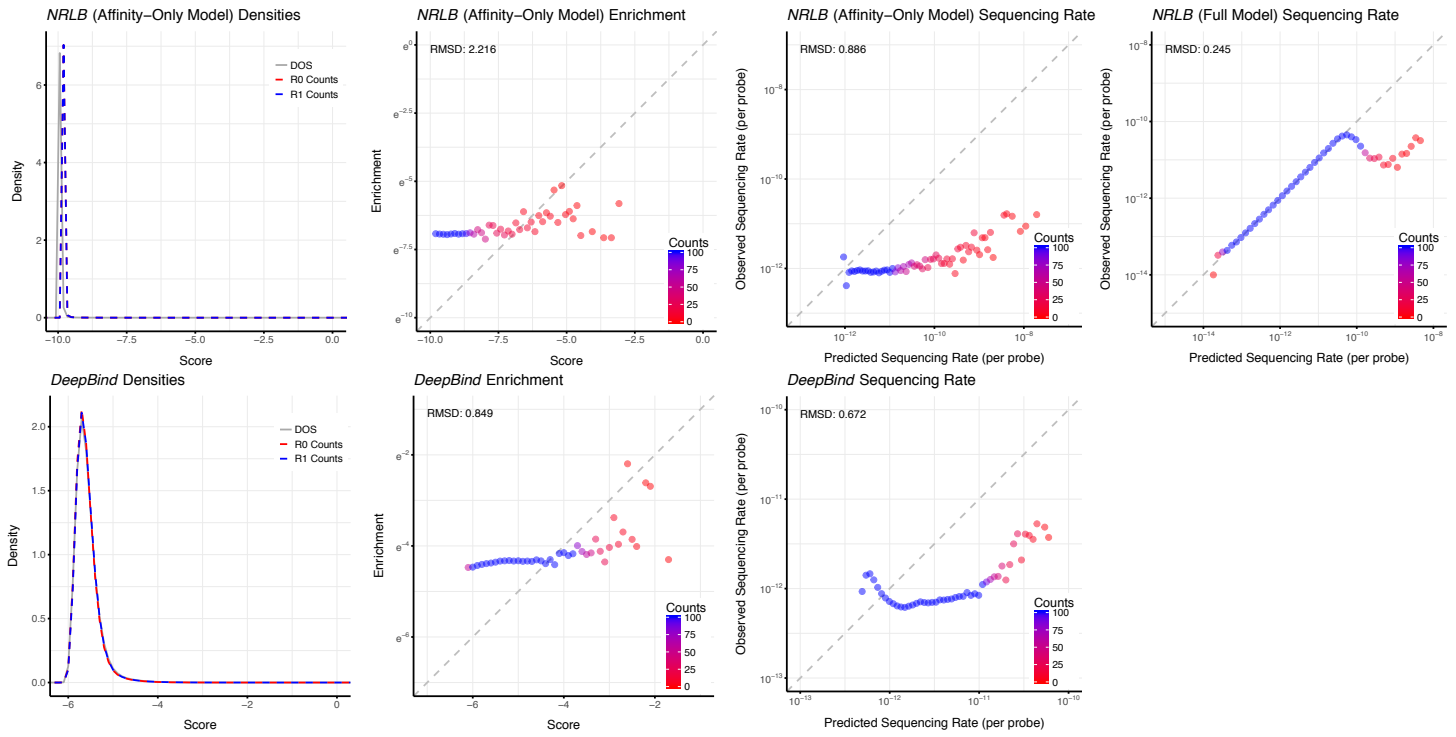
GATA3; *DeepBind* Round: 4, *NRLB* Round: 2



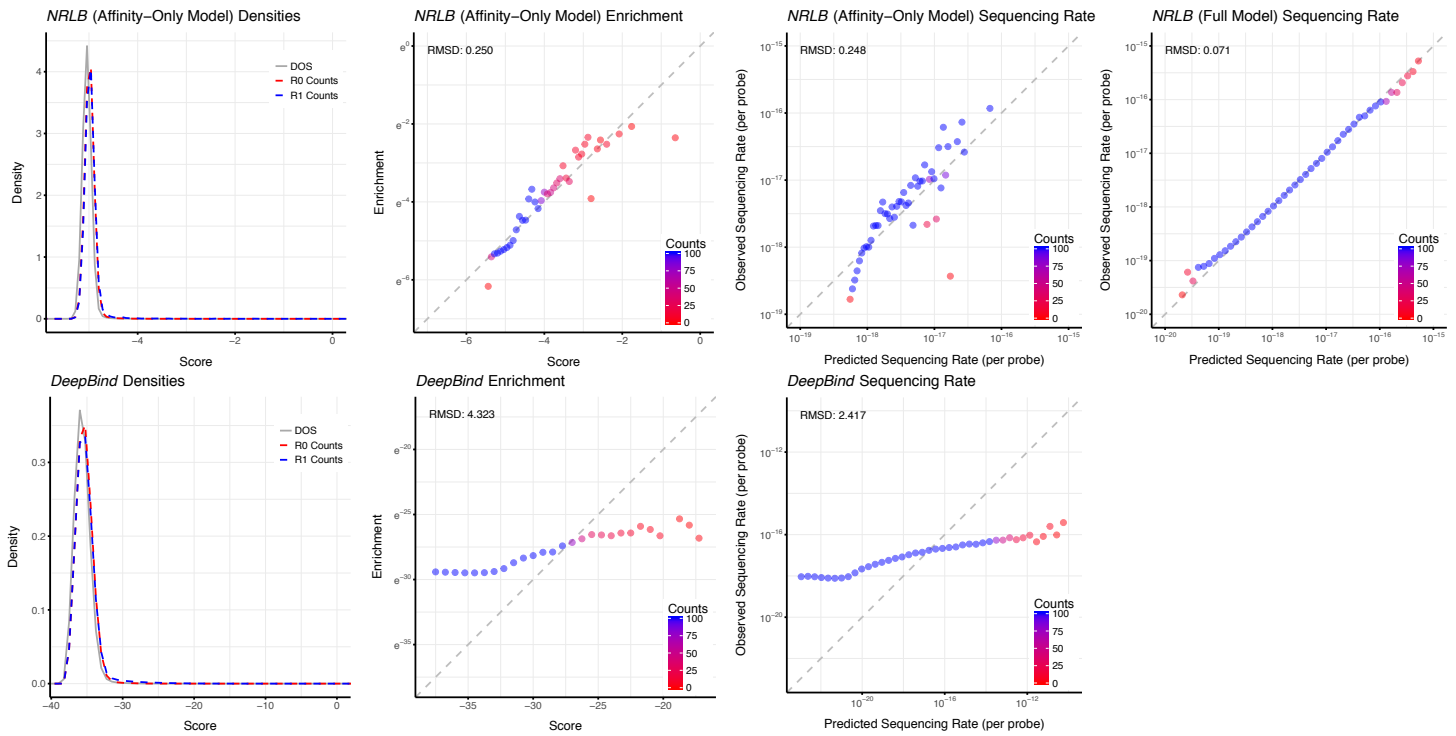
IRF4; *DeepBind* Round: 3, *NRLB* Round: 1



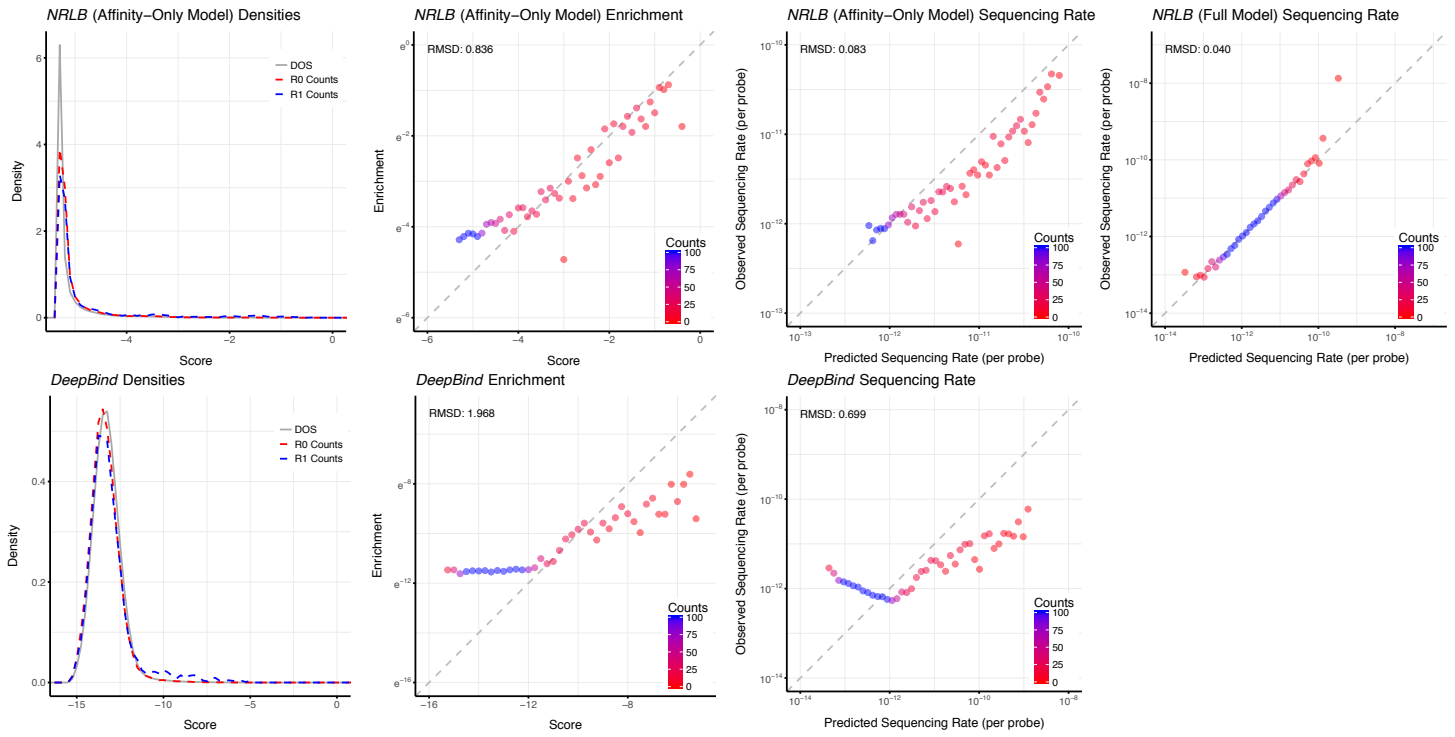
MAFF; DeepBind Round: 4, NRLB Round: 3



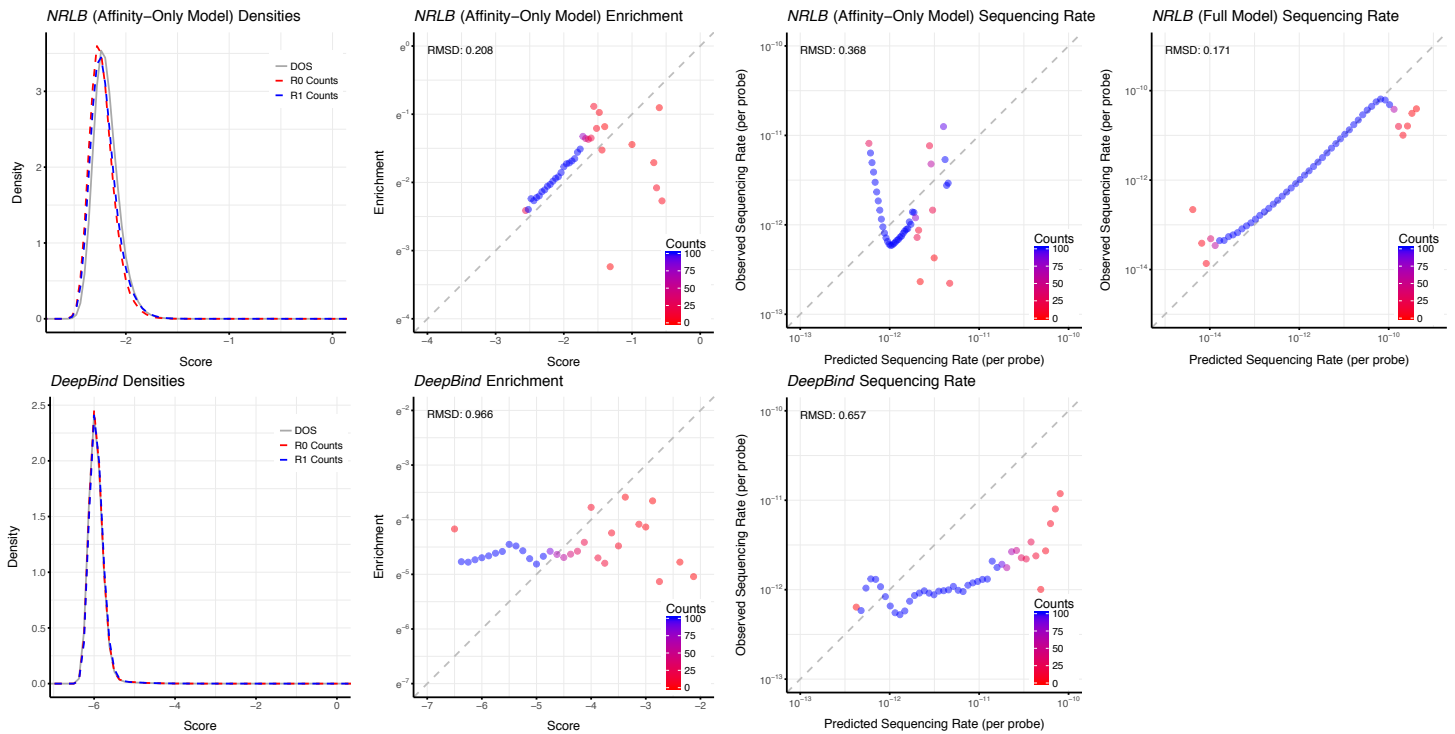
MAFK; DeepBind Round: 5, NRLB Round: 1



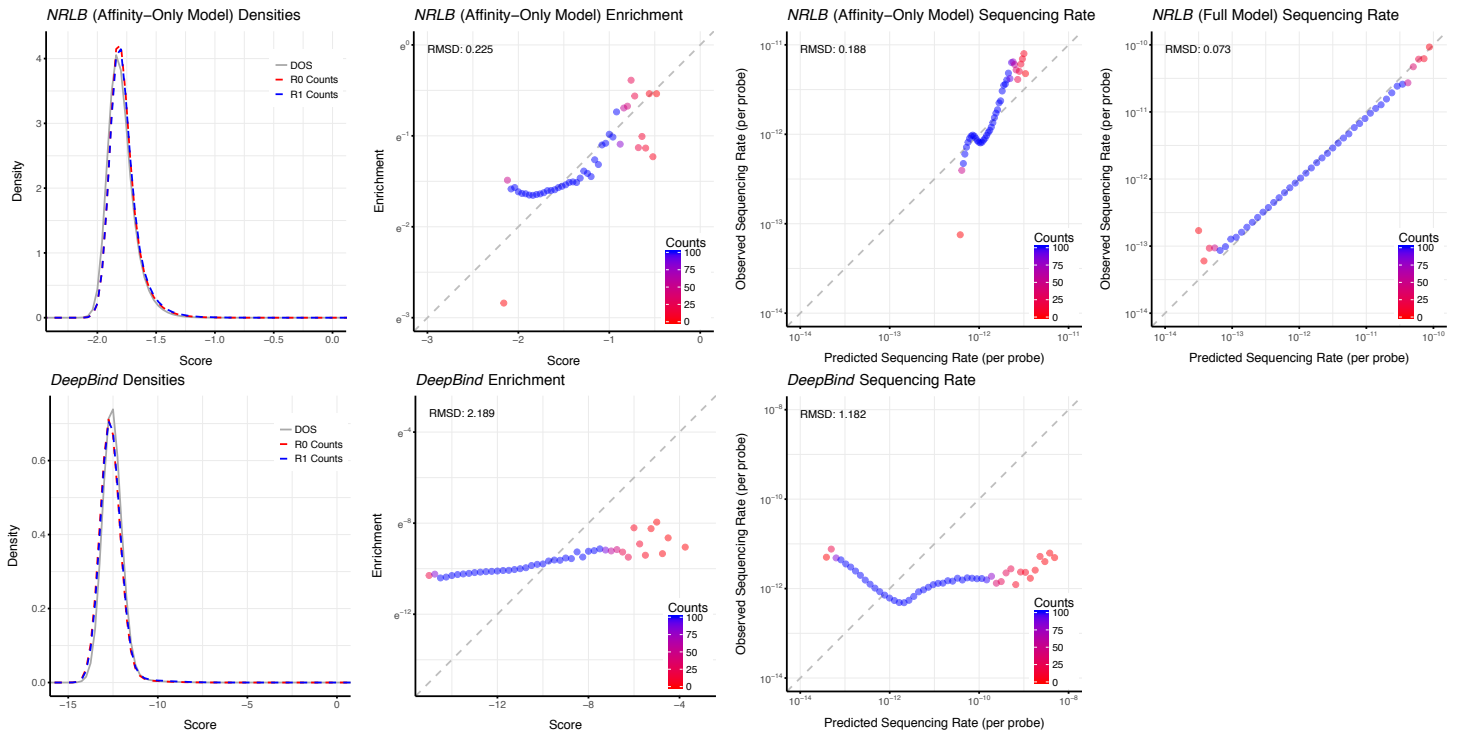
MAX; DeepBind Round: 4, NRLB Round: 1



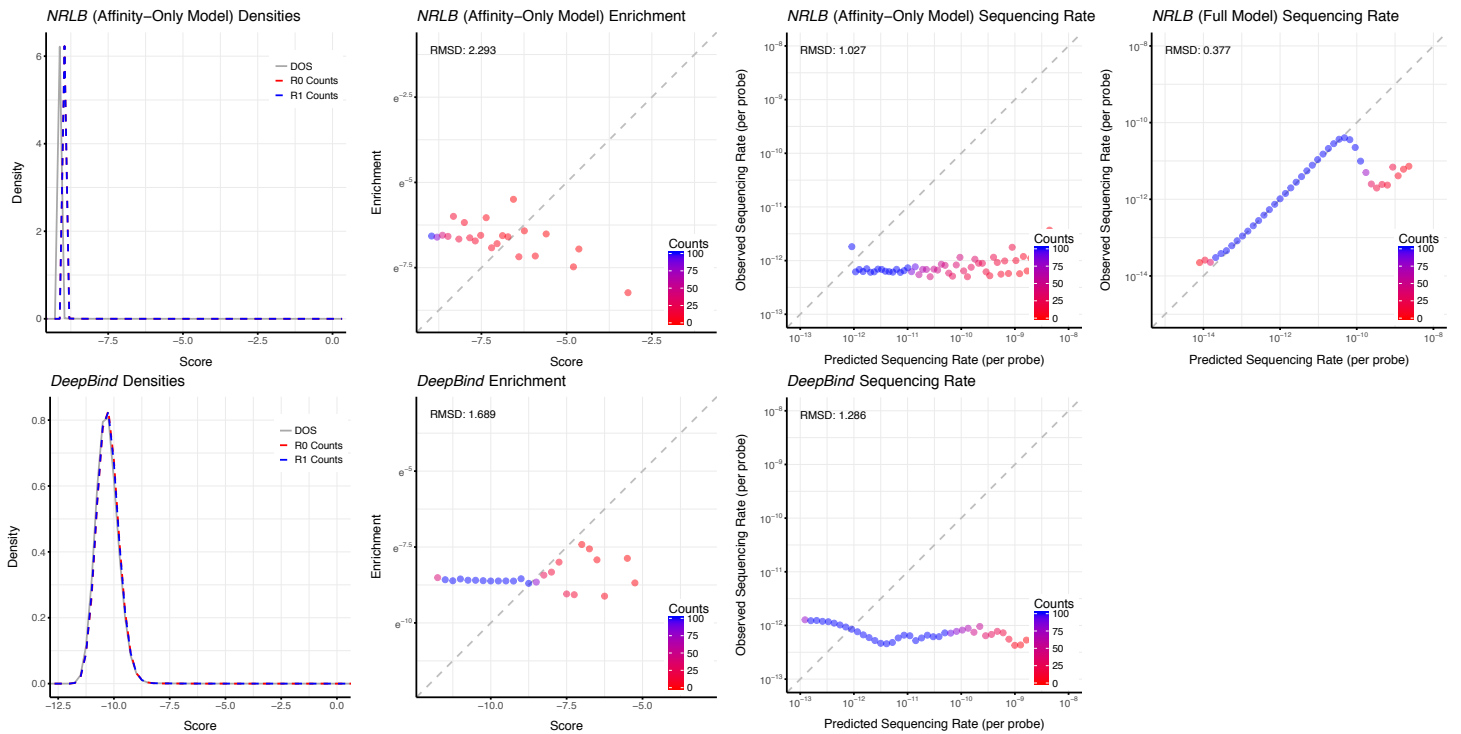
MEF2A; DeepBind Round: 4, NRLB Round: 2



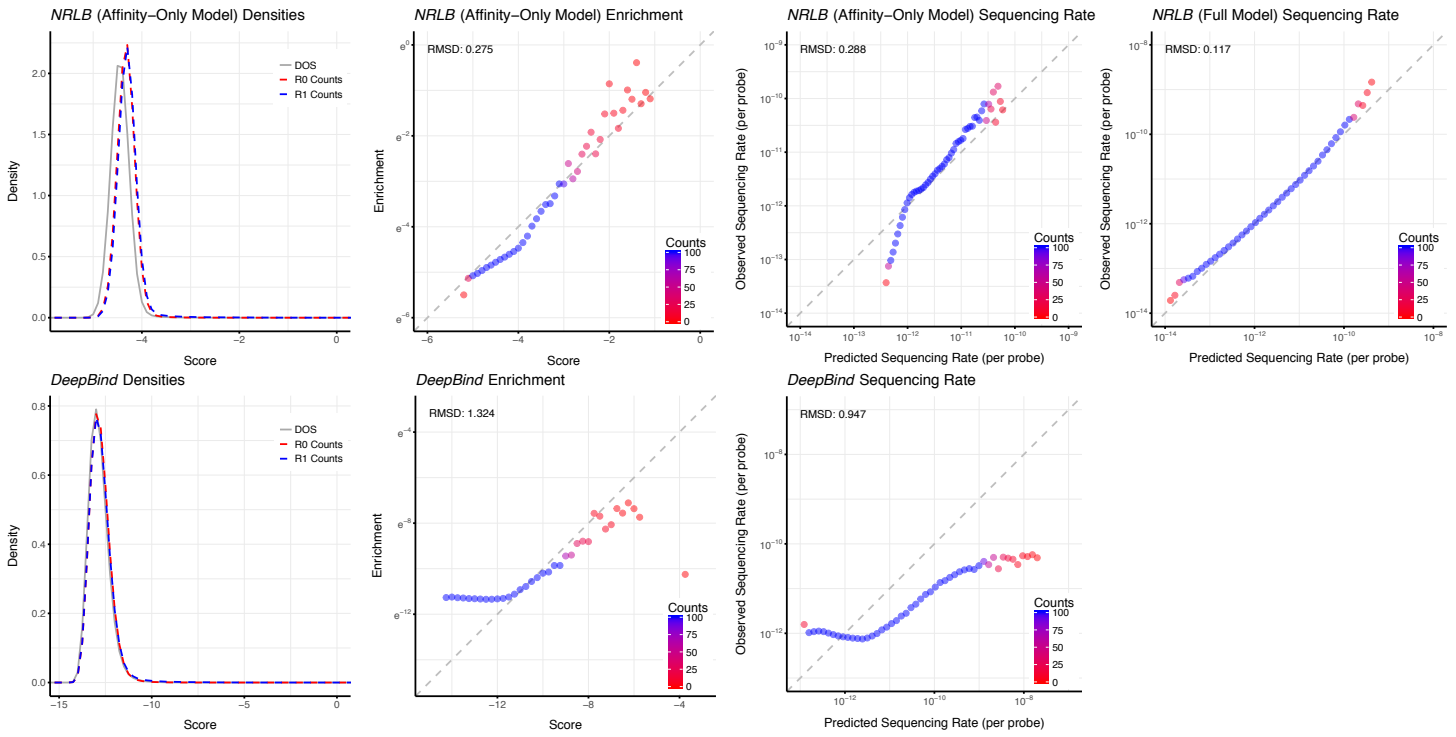
NFATC1; DeepBind Round: 4, NRLB Round: 1



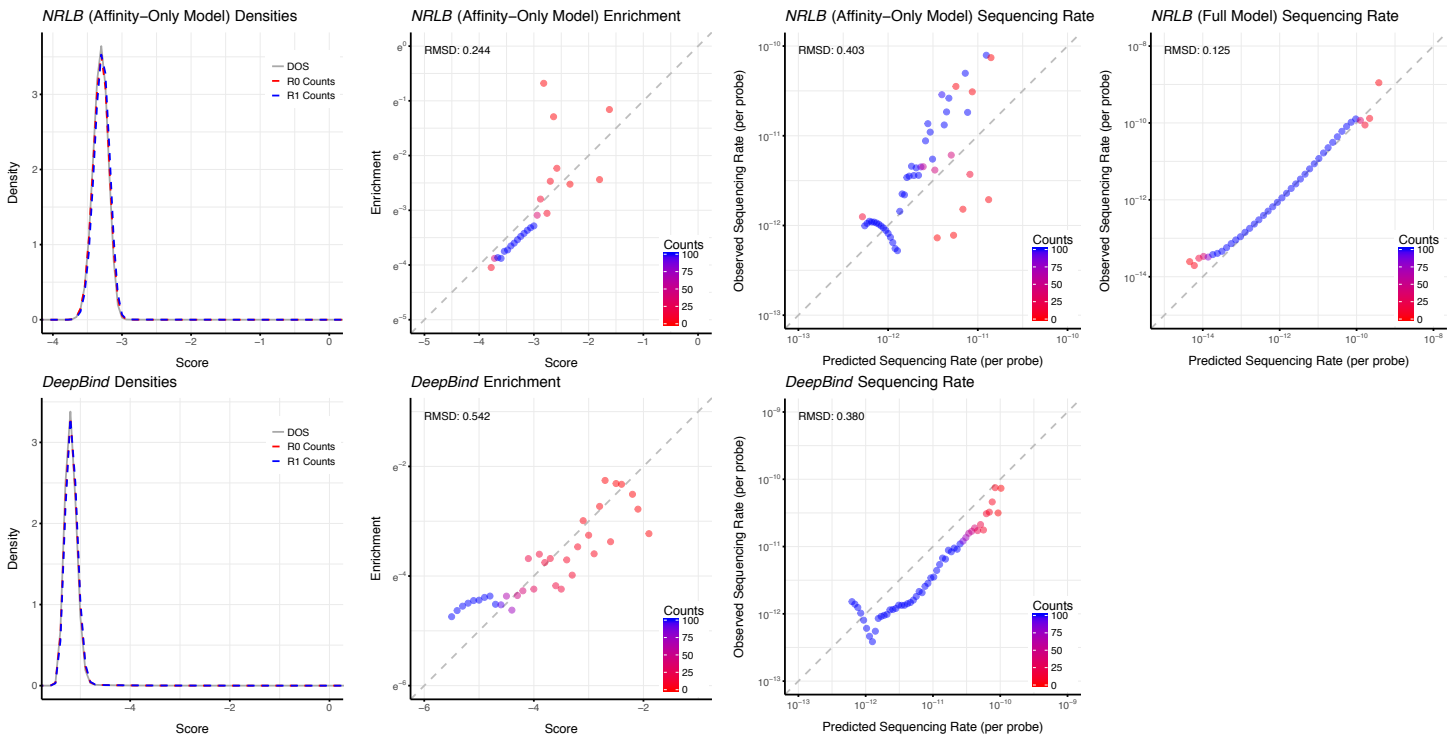
NFE2; DeepBind Round: 4, NRLB Round: 3



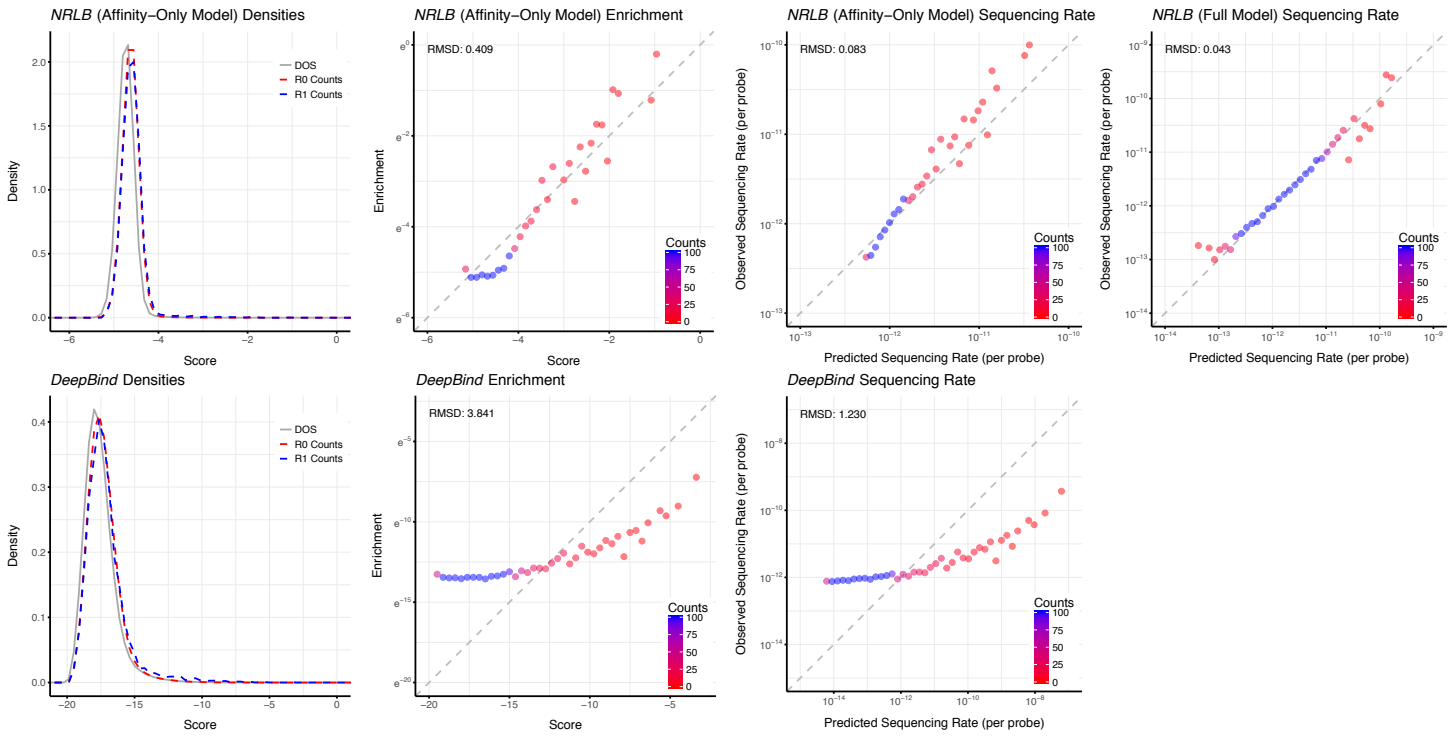
NR2C2; DeepBind Round: 3, NRLB Round: 1



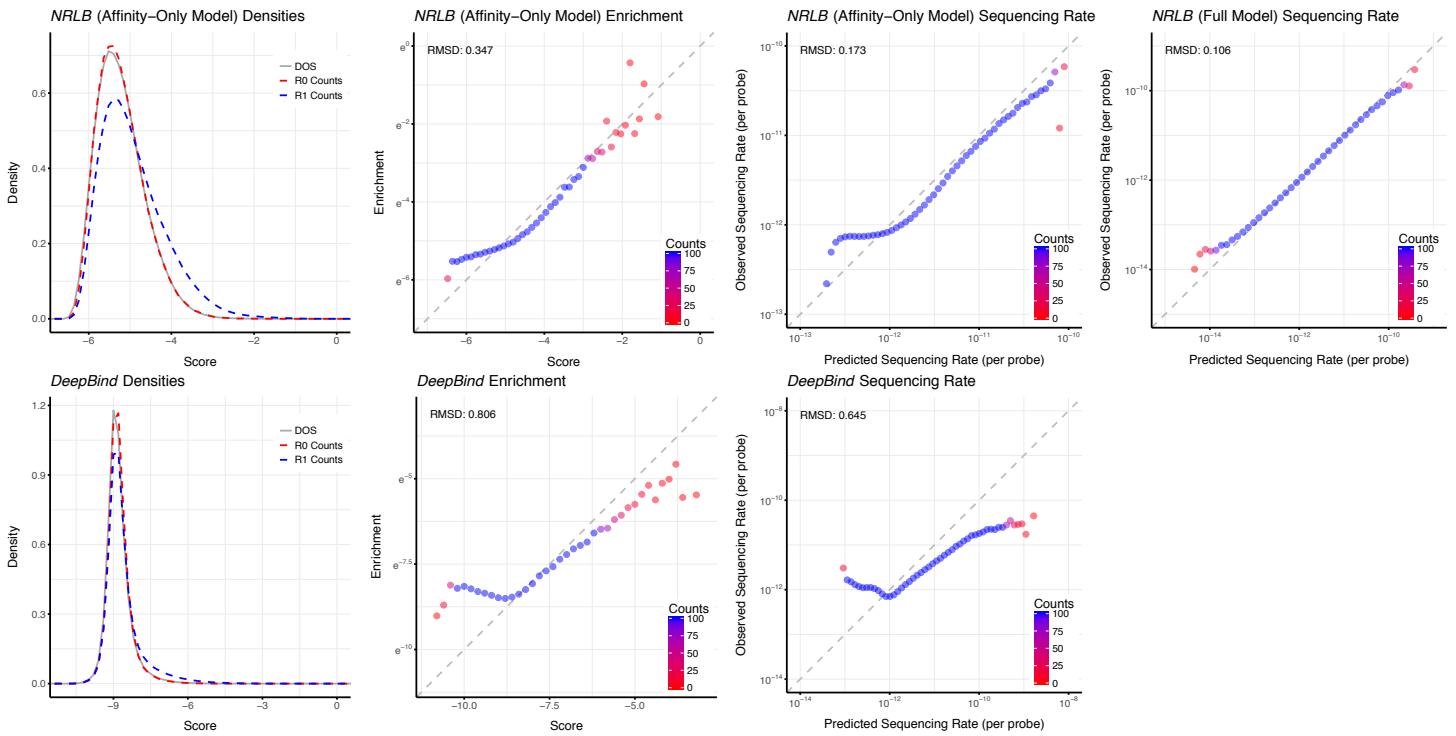
NR1; DeepBind Round: 2, NRLB Round: 1



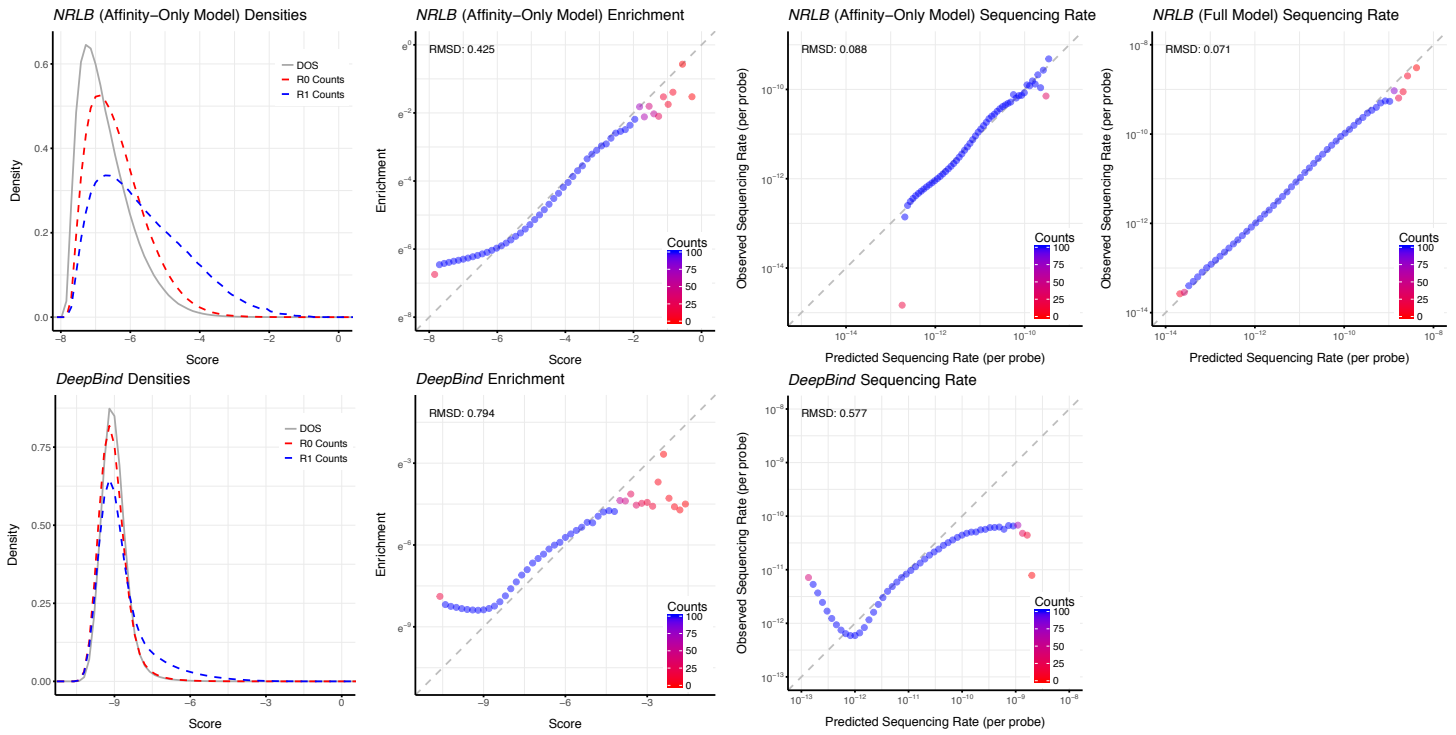
PAX5; DeepBind Round: 3, NRLB Round: 1



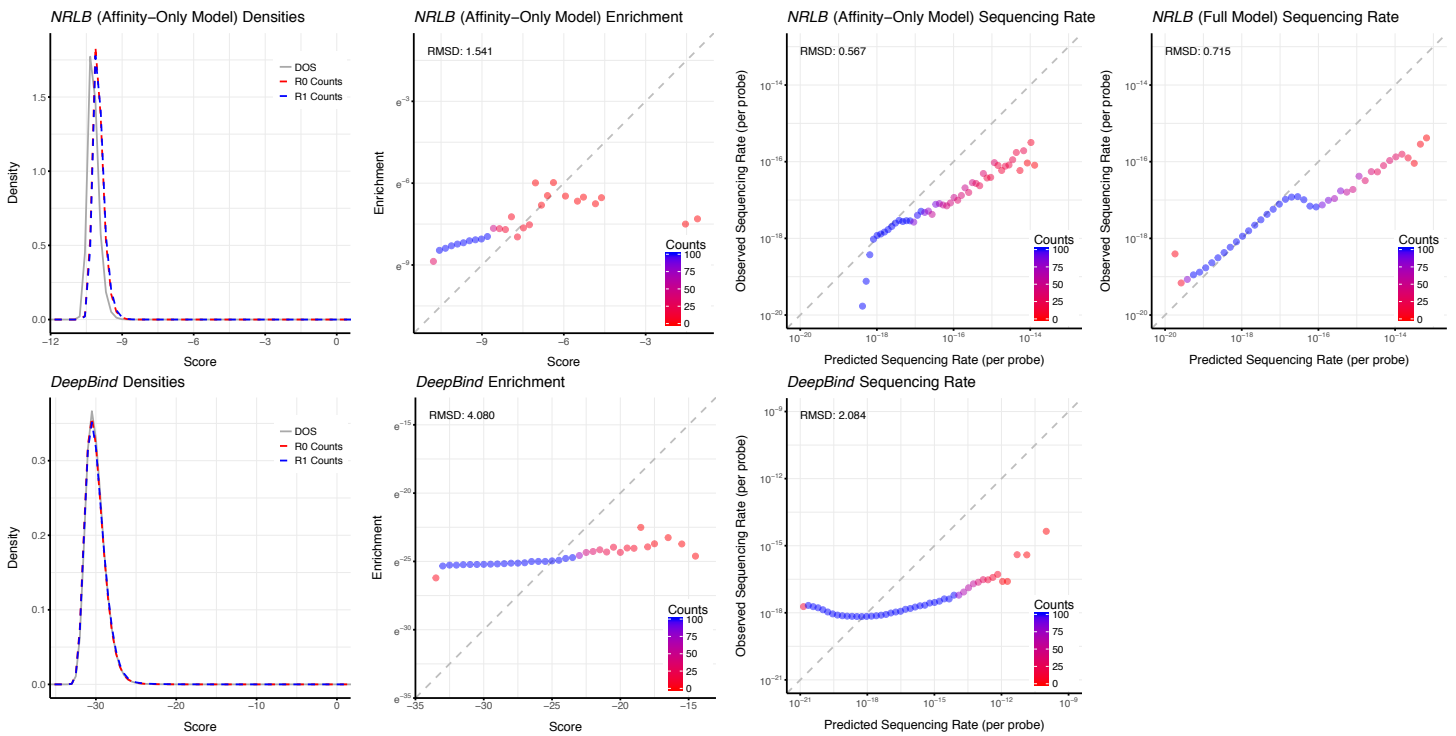
POU2F2; DeepBind Round: 2, NRLB Round: 1



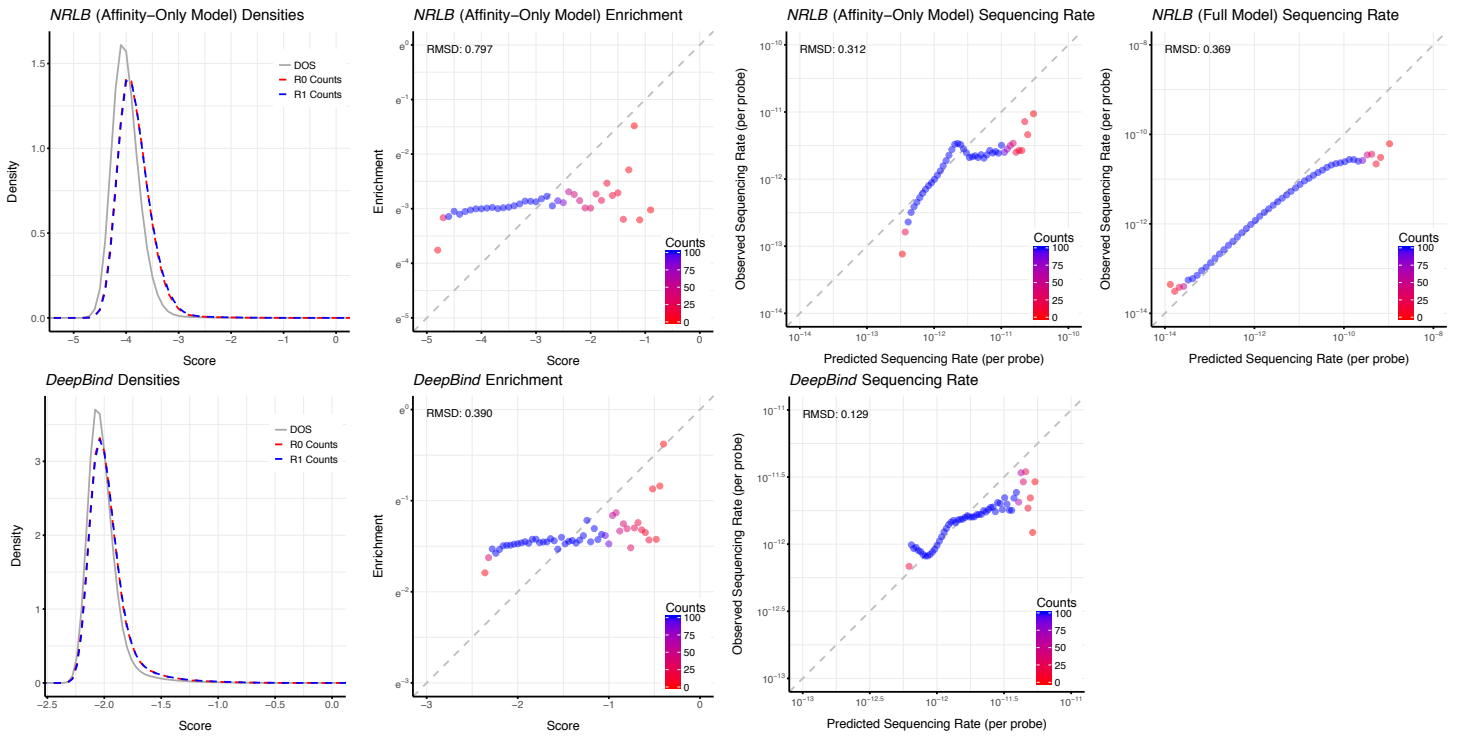
POU5F1P1; DeepBind Round: 2, NRLB Round: 1



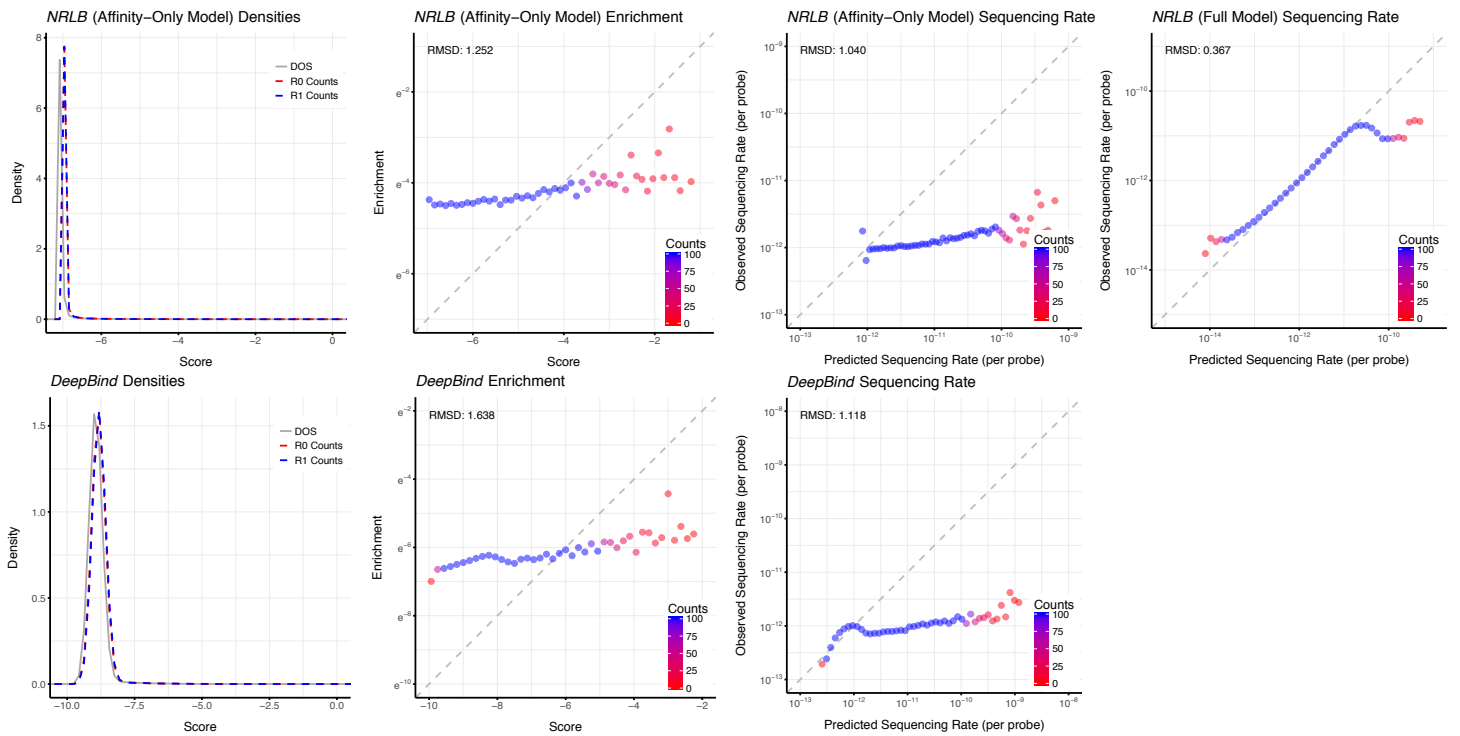
RFX5; DeepBind Round: 4, NRLB Round: 2



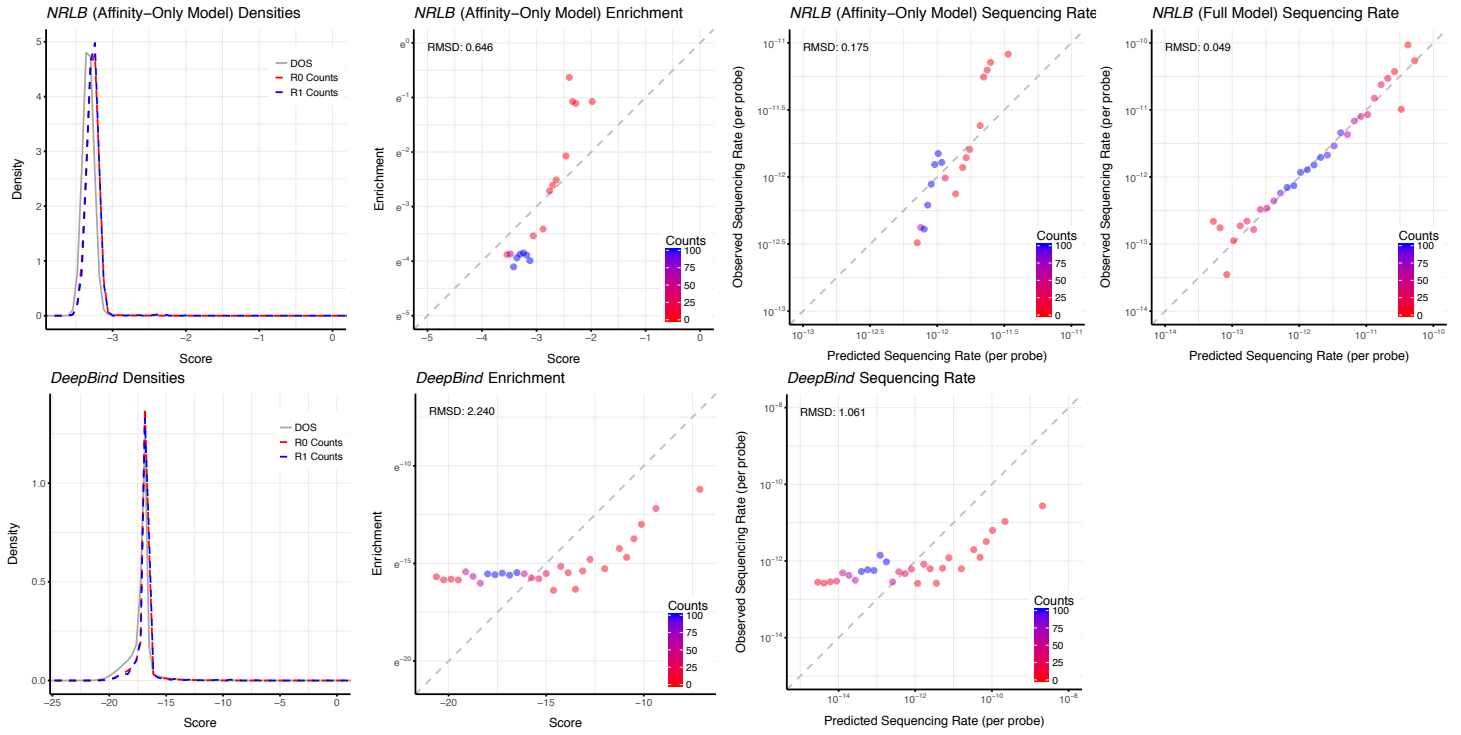
RUNX3; DeepBind Round: 3, NRLB Round: 3



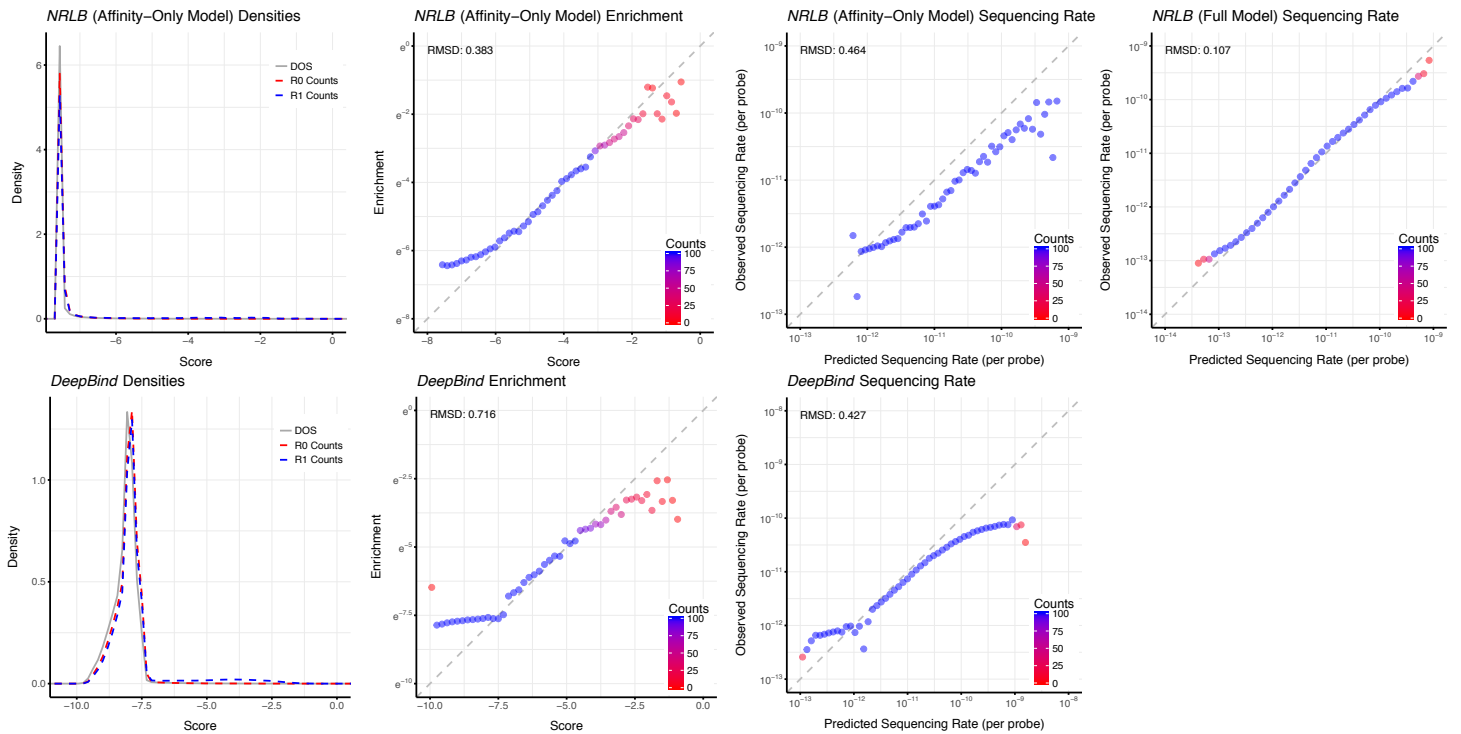
TFAP2A; DeepBind Round: 3, NRLB Round: 2



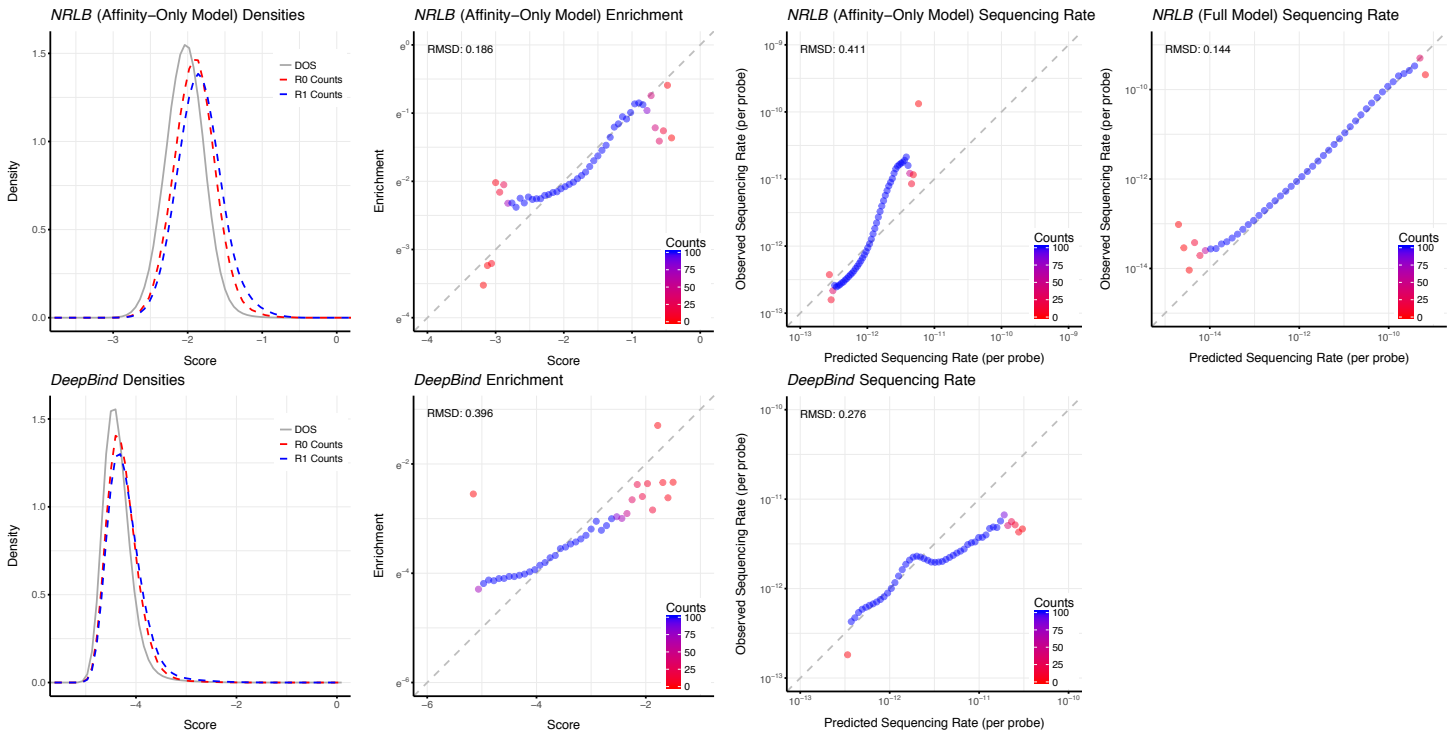
TFAP2C; DeepBind Round: 3, NRLB Round: 1



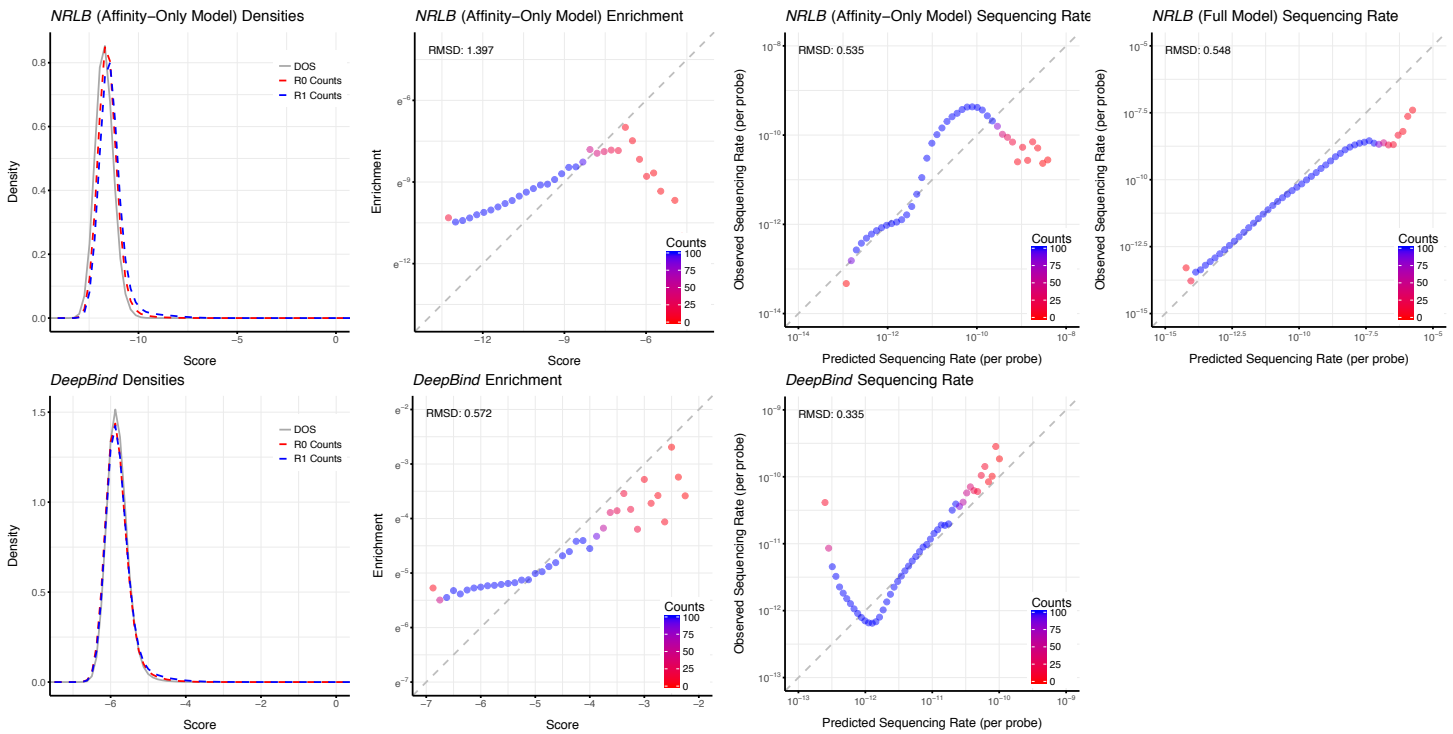
USF1; DeepBind Round: 2, NRLB Round: 1

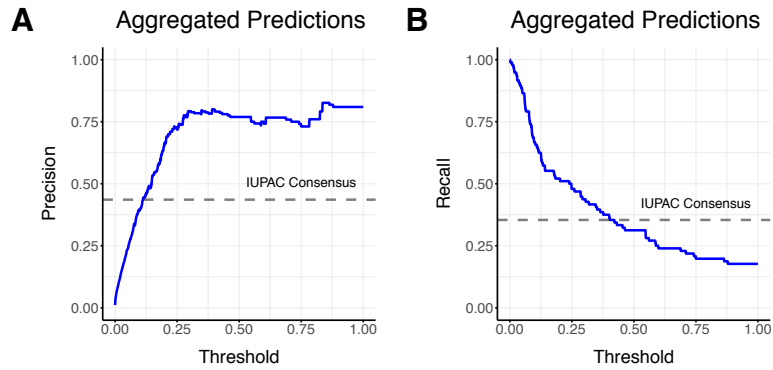


YY1; DeepBind Round: 3, NRLB Round: 1

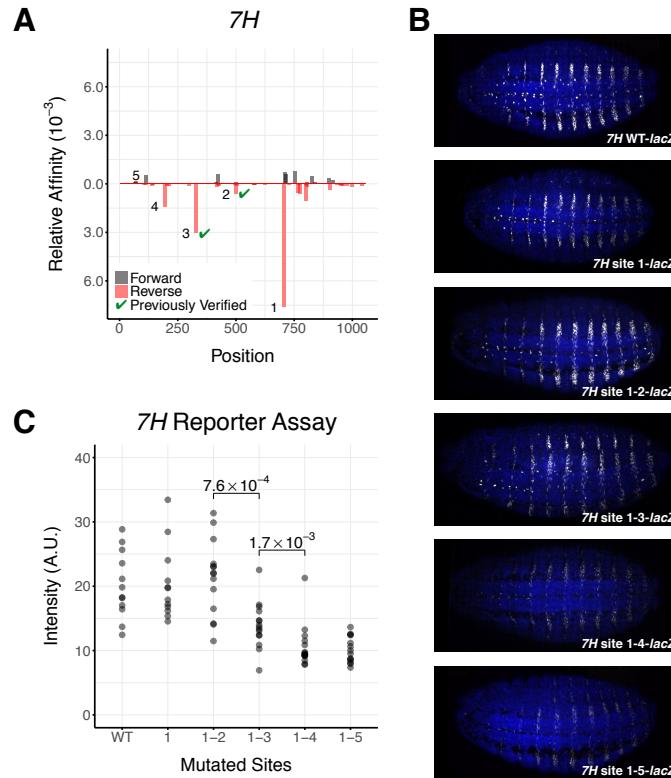


ZNF143; DeepBind Round: 4, NRLB Round: 3

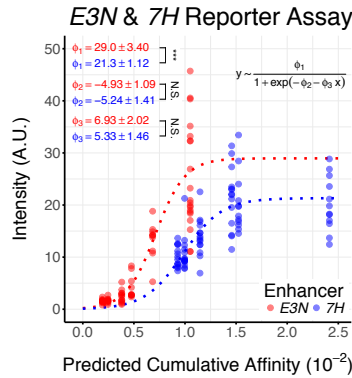




Supplemental Figure S16. *NRLB* models outperform consensus matching methods in identifying functional binding sites in enhancers. **(A)** Precision and **(B)** recall as a function of the relative affinity cutoff (threshold) for *NRLB* models for Hox and Exd-Hox (blue line) and consensus matching methods (grey dashed line) on the task of identifying 96 functionally validated binding sites across 21 curated *D. melanogaster* enhancer elements. Hox and Exd-Hox models trained on data from Slattery, *et al.* (1) (cf. Figure S8). See Methods for details on the construction and identification of positive and negative sites within enhancers. See Table 4 for a complete list of enhancers and sites.



Supplemental Figure S17. Functional validation of ultra-low affinity sites in the *svb* enhancer element *7H* as predicted by *NRLB*. **(A)** *NRLB* model predicted relative affinities for Exd-UbxIVa (y-axis) at all potential binding sites on the forward (grey) and reverse strands (red) within the *shavenbaby* (*svb*) enhancer element *7H* in *D. melanogaster* as a function of position in the element (x-axis). Sites indicated by a green check mark were functionally validated in a previous study (10). Numbers correspond to the order in which sites were mutated. *NRLB* model was trained on R1 SELEX-seq data for Exd-UbxIVa from Slattery, *et al.* (1) and shown in Figure S8. **(B)** Expression (white) of *7H::lacZ* reporter constructs where the binding sites identified in panel A were sequentially mutated. WT indicates the wild-type *7H* enhancer element, while site 1, site 1-2, etc. indicate the mutations of site 1, sites 1 and 2, etc. **(C)** Reporter expression level (y-axis) for every reporter construct (x-axis) as quantitated from panel B. Each point represents the reporter expression level of a single embryo (see Methods). Mutation of sites 3 and 4 demonstrate statistically significant changes in reporter intensity (Mann-Whitney U Test). A decrease in expression level is observed after the top three sites are mutated, likely due to saturation.



Supplemental Figure S18. Nonlinear fits to reporter expression data for the *svb* enhancer elements *E3N* and *7H*. Joint comparison between the *NRLB* predicted Exd-UbxIVa cumulative affinities for various *E3N* and *7H* reporter constructs (x-axis) and reporter expression level (y-axis) as quantitated from Figures 6C and S17B. Each point represents the reporter expression level of a single embryo (see Methods) for either an *E3N* (red) or *7H* (blue) construct. Red and blue dashed lines and coefficients correspond to independent nonlinear least-square model fits between predicted cumulative affinity and reporter expression for *E3N* (red) and *7H* (blue), respectively. Nonlinear fits assume a logistic model of expression saturation (equation). The F-test for individual parameters was applied to test if the parameters learned from each fit independently were significantly different. While the fit parameters ϕ_2 and ϕ_3 are similar, the scale parameter ϕ_1 , corresponding to the overall intensity or brightness of the experiments, is significantly different. The *NRLB* model for Exd-UbxIVa was trained on R1 SELEX-seq data from Slattery, *et al.* (1) and shown in Figure S8.

REFERENCES

1. Slattery M, *et al.* (2011) Cofactor binding evokes latent differences in DNA binding specificity between Hox proteins. *Cell* 147(6):1270-1282.
2. Alipanahi B, Delong A, Weirauch MT, & Frey BJ (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat Biotechnol* 33(8):831-838.
3. Foat BC, Morozov AV, & Bussemaker HJ (2006) Statistical mechanical modeling of genome-wide transcription factor occupancy data by MatrixREDUCE. *Bioinformatics* 22(14):e141-149.
4. Maerkl SJ & Quake SR (2007) A systems approach to measuring the binding energy landscapes of transcription factors. *Science* 315(5809):233-237.
5. Zhou T, *et al.* (2015) Quantitative modeling of transcription factor binding specificities using DNA shape. *Proc Natl Acad Sci U S A* 112(15):4654-4659.
6. Jolma A, *et al.* (2013) DNA-binding specificities of human transcription factors. *Cell* 152(1-2):327-339.
7. Jolma A, *et al.* (2013) DNA-binding specificities of human transcription factors. *Cell* 152(1-2):327-339.
8. Yang L, *et al.* (2017) Transcription factor family-specific DNA shape readout revealed by quantitative specificity models. *Mol Syst Biol* 13(2):910.
9. Wang F & Landau DP (2001) Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys Rev Lett* 86(10):2050-2053.
10. Crocker J, *et al.* (2015) Low affinity binding site clusters confer hox specificity and regulatory robustness. *Cell* 160(1-2):191-203.

PART B

SUPPLEMENTAL METHODS

Supplemental Methods for “Accurate and Sensitive Quantification of Protein-DNA Binding Affinity”

Chaitanya Rastogi, H. Tomas Rube, and Harmen J. Bussemaker

February 7, 2018

Contents

1	Modeling Methodology	2
1.1	The Multinomial Representation of SELEX Data	2
1.2	R0 Bias Model	4
1.3	<i>NRLB</i>	5
2	Numerical Methods	15
2.1	Model Evaluation	15
2.2	Dynamic Programming Techniques	18
2.3	Invariances and Symmetries of the Likelihood	25
2.4	Parameter Conditioning	31
2.5	Parameter Error	33
3	R0 Bias Model Validation	35
3.1	Synthetic Data	35
3.2	Real Data	36
A	R0 Bias Model Feature Selection	40
B	Minimization Methods	41
B.1	Optimization Algorithms	42
B.2	Jacobian Based Symmetries	47
	References	49

1 Modeling Methodology

Here, we will introduce our general multinomial approach for modeling SELEX reads and two adaptations: the R0 Bias model and *No Read Left Behind*, or *NRLB*. The modeling philosophy of both will be exposed in great detail and also cover the feature sets and fitting strategies used by them.

1.1 The Multinomial Representation of SELEX Data

A typical SELEX sequencing library consists of DNA probes isolated from multiple experiments, each with a unique set of barcodes that flank an l -bp long region of random DNA. After sequencing, probes from a specific experiment and enrichment round r can be filtered and aggregated via the unique barcode, producing data with the structure in Table 1. The kmer counting method of [1]

Table 1: An Example of SELEX Data

Probe	R1 Count
GTGATTTATTGTTATT	17
ATGATTTATGGTTTTT	16
GTTTTATGATTTATTA	15
GATGATTTATTATCCT	15
GATAATGATTTATTAT	15
AGATTTTGATTTATTA	15
TTTTTTTGATTTATTA	14
⋮	⋮

R1 Exd-UbxIVa SELEX-seq data from [1].

discussed in the previous chapter builds count tables based on data similar to what is shown. Ideally, we would want a model that represents observed probe counts rather than an intermediary such as kmer counts.

One way to represent the observed SELEX counts is through a multinomial distribution over the entire universe of 4^l probes. While this appears to place an unnecessary burden on the model (especially for large l), a typical SELEX library contains only a fraction of all affinity-selected probes – therefore, it is possible that some probes are unobserved in the dataset due to stochastic effects. With a multinomial distribution, the observed count c_i in round r can be related to a model-predicted observation frequency p_i for all sequences S_i as follows:

$$P(\text{data}) \propto \prod_i p_i^{c_i} \quad (1)$$

If multiple rounds of selection and sequencing were performed, we can simulta-

neously model all rounds:

$$P(\text{data}) \propto \prod_{r \in \{\text{rounds}\}} \prod_i p_{i,r}^{c_{i,r}} \quad (2)$$

We would like the model-predicted frequencies p_i in (1) to correspond to biophysically relevant measures of TF-DNA interaction such as the equilibrium association constant K_a . The total count of sequence S_i after $r \geq 1$ rounds of selection is given by

$$c_{i,r} = N_r(S_i)c_{i,r-1} \quad (3)$$

where $N_r(S_i)$ is the the occupancy of sequence S_i in round r . The probability of observing sequence S_i after r rounds of selection is then

$$p_{i,r} = \frac{c_{i,r}}{\sum_{i'} c_{i',r}} = \frac{N_r(S_i)c_{i,r-1}}{\sum_{i'} N_r(S_{i'})c_{i',r-1}} \quad (4)$$

Recursively applying (3), we see that

$$c_{i,r} = \prod_{r=1}^r N_r(S_i)c_{i,0}$$

and

$$p_{i,r} = \frac{\prod_{r=1}^r N_r(S_i)c_{i,0}}{\sum_{i'} \prod_{r=1}^r N_r(S_{i'})c_{i',0}}$$

In the low free protein concentration limit, $N(S) \approx [P]K_a(S)$ [2]. Additionally, if we make the mild assumption that the association constant $K_a(S_i)$ is identical for all rounds, the above reduces to

$$p_{i,r} = \frac{K_a(S_i)^r c_{i,0}}{\sum_{i'} K_a(S_{i'})^r c_{i',0}} = \frac{p_{i,0}(\kappa_i)^r}{Z} \quad (5)$$

for round r . Without loss of generality, we can represent $c_{i,0}$ as $p_{i,0}$, or the probability that sequence S_i will be found in the initial pool – this bias can be represented either by the Markov model method described previously or any other suitable approach. Letting $\kappa_i = K_a(S_i)$ and $Z = \sum_i p_{i,0}\kappa_i^r$ (also known as the partition function), SELEX data in round r can be modeled as

$$P(\text{data} | \vec{\kappa}) \propto \prod_i \left(\frac{p_{i,0}\kappa_i^r}{Z} \right)^{c_{i,r}} \quad (6)$$

κ_i in (5) represents the selection rate of the probe. While many models can be used to represent κ_i , fundamentally, it represents the biophysical activity of

the TF in question. To accurately infer this activity, we need to know where the TF is bound. In most SELEX-seq experiments, the overall probe (including the fixed flank regions) is significantly larger than the protein-DNA interface. As there are no physical constraints forcing the protein to bind at a specific location within the probe, let alone the variable region, the binding interface could lie *anywhere* on the sequence S_i . In general, if we are considering a binding interface of length k , there are $l - k + 1$ potential binding sites, or views, on the forward strand alone. Additionally, as the protein does not view any physical separation between the variable region and the fixed flanking region, binding sites that bleed into the fixed regions should also be considered. Lastly, as binding can take place on either strand, potential binding sites on both strands must be considered. Given this, we will restrict ourselves to parameterizations where κ_i is represented by the sum of the affinities of all potential views within probe ([2, 3]). However, other models can be built within the same framework and are discussed below.

1.2 R0 Bias Model

The multinomial framework presented in (1) can be used to directly model R0 SELEX data and create a representation for $p_{i,0}$. Extending the kmer-based Markov model from [1], we can relate the observed R0 count of a probe to its predicted probability $p_{i,0}$ with a log-linear model w_i over the probe’s k -mer composition:

$$w_i = \exp \left[\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi} \right] \quad (7)$$

and

$$p_{i,0} = \frac{w_i}{Z_0} = \frac{w_i}{\sum_i w_i} \quad (8)$$

Here, ϕ refers to one of the 4^k possible k -mers, Φ_0 is the set of all kmer features used, $X_{i\phi}$ is the number of observations of each k -mer in sequence S_i , β_ϕ is the contribution of every k -mer to the overall observation probability, and Z_0 is the partition function. The feature set Φ_0 can be extended such that ϕ covers all kmers from length 1 through length k , however this made it difficult to converge on a solution (see Appendix A).

$X_{i\phi}$ represents the count of kmer ϕ in the forward strand within the variable region of the probe. However, it is possible that the large regions of fixed DNA (barcodes, adaptors) flanking the random region contribute to biases in the initial pool even though they remain unchanged throughout the SELEX process. To account for their effect, $X_{i\phi}$ can be computed by including the $k - 1$ bases flanking the random region.

Incorporating $p_{i,0}$ with (1) results in a function of the parameters $\vec{\beta}$ given

the observed data, or the likelihood, which is given by

$$\mathcal{L}(\vec{\beta} | \text{data}) = P(\text{data} | \vec{\beta}) \propto \prod_i p_{i,0}^{c_{i,0}} = \prod_i \left(\frac{w_i}{Z_0} \right)^{c_{i,0}}, \quad (9)$$

Given this statistical model and our observed data, we can find the optimal parameter values that maximize the likelihood - this optimization process is called Maximum Likelihood Estimation (MLE). When trying to find the optimal parameters $\vec{\beta}$, it is easier to work with the log of the likelihood:

$$\log \mathcal{L} = \sum_i c_{i,0} \log w_i - n_0 \log Z_0, \quad (10)$$

where $n_0 = \sum_i c_{i,0}$ is the total number of reads. Lastly, the gradient of the log likelihood is useful for gradient-based optimizers such as L-BFGS (see Appendix B), and is given by

$$\frac{\partial \log \mathcal{L}}{\partial \beta_\phi} = \sum_i c_{i,0} X_{i\phi} - \frac{n_0}{Z_0} \sum_i X_{i\phi} \exp \left[\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi} \right] \quad (11)$$

Cross validation methods are used to select the optimal k and whether or not flanking regions should be considered. Cross validation is performed either with another R0 dataset (generally part of the same series of experiments, but with a different barcode) or by holding out half of the data.

1.3 *NRLB*

Model Description

We can couple the framework of (6) with a biophysical feature-based model for κ_i to understand the impact of sequence variation on binding affinity in terms of interpretable features [2, 3]. We called this model *No Read Left Behind*, or *NRLB*, for its ability to build these feature based models on all SELEX data without filtering probes.¹ In *NRLB*, the free energy of a particular view (binding interface) v within the probe is modeled as a series of linear contributions from DNA sequence derived features ϕ . Additionally, we can simultaneously consider the impact of nonspecific binding interactions and multiple binding modes m on the overall selection rate of the probe. In general, κ_i is modeled as follows:

$$\begin{array}{l} \text{Affinity for one binding} \\ \text{interface } v \end{array} \exp \left[\sum_{\phi \in \Phi} \beta_\phi X_{i\phi} \right] \quad (12)$$

¹It should be noted that this is a property of the framework itself, rather than this specific model; however the selection model was named *NRLB*.

Total affinity for one mode, all views v

$$\sum_v \exp \left[\sum_{\phi \in \Phi} \beta_\phi X_{iv\phi} \right] \quad (13)$$

Total affinity for one mode, all views v and nonspecific binding β_{NS}

$$\sum_v \exp \left[\sum_{\phi \in \Phi} \beta_\phi X_{iv\phi} \right] + e^{\beta_{NS}} \quad (14)$$

Total affinity for multiple modes m , all views v and nonspecific binding β_{NS}

$$\sum_m \sum_{v \in V_m} \exp \left[\sum_{\phi \in \Phi_m} \beta_\phi X_{imv\phi} \right] + e^{\beta_{NS}} \quad (15)$$

Henceforth, we will continue to use the general case (15) for all derivations. The quantities used above are defined as follows:

- m the binding mode.
- v the position of the k -bp view (binding interface) in the probe. As different modes within a model can have different k 's, the total number of views per mode could vary. V_m represents the set of all views in mode m .
- ϕ a feature. Φ_m represents the set of all features in mode m , while the function $m(\phi)$ represents the mode feature ϕ belongs to.
- β_ϕ contribution of feature ϕ to the binding free energy in units of $\Delta\Delta G/RT$.
- $X_{iv\phi}$ value of feature ϕ in interface v for mode $m(\phi)$ in S_i . Also called the data.
- β_{NS} Nonspecific binding for the overall model. While there is a nonspecific binding contribution for every view in every mode, it is possible to combine all these terms into a single, global term:

$$\sum_m \sum_{v \in V_m} \left(e^{\sum_{\phi \in \Phi_m} \beta_\phi X_{iv\phi}} + \boxed{e^{\beta_{m, NS}}} \right) \rightarrow \sum_m \sum_{v \in V_m} e^{\beta_{m, NS}} = e^{\beta_{NS}}$$

It must be stressed that alternative binding modes, including nonspecific binding, may *not* represent actual modes of binding. It is very likely that the additional modes might act as ‘garbage collectors,’ cleaning up unwanted sequence biases accumulated through excessive PCR amplification or other contaminants introduced through the experimental process.

In a multiple binding mode model, every mode has an effective scaling parameter, or a ‘mode relative affinity’ γ_m , that is useful in understanding its

Table 2: Binding Mode Relative Affinities

Binding Mode	Optimal Sequence	γ_m	Normalized γ_m
Mode 1	ACCTGC	105.42	0.28
Mode 2	ATGAAT	371.85	1.00
Nonspecific	N/A	1.92	5.17×10^{-3}

Hypothetical mode relative affinities γ_m for a model with two binding modes and nonspecific binding. Mode 2 is the primary mode.

contribution to κ_i relative to the other modes. Effectively, γ_m represents the difference in the optimal affinities for each mode - for example, consider a model with two binding modes and nonspecific binding as given in Table 2. γ_m is then the affinity contribution (12) of the optimal sequence for the given mode. The mode with the highest possible affinity contribution is called the ‘primary mode.’ Normalizing the γ_m values results in a standardized representation of mode strength.

Similar to the R0 bias model, we will work with the log likelihood

$$\log \mathcal{L} = \sum_i c_{i,1} \log(p_{i,0} \kappa_i) - n \log Z. \quad (16)$$

and its gradient

$$\frac{\partial \log \mathcal{L}}{\partial \beta_\phi} = \sum_i \frac{c_{i,1}}{\kappa_i} \frac{\partial \kappa_i}{\partial \beta_\phi} - \frac{n}{Z} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\phi} \quad (17)$$

where

$$\frac{\partial \kappa_i}{\partial \beta_\phi} = \begin{cases} \sum_{v \in V_m(\phi)} X_{iv\phi} e^{\sum_{\phi' \in \Phi_m(\phi)} \beta_{\phi'} X_{iv\phi'}} & \phi \neq \text{NS} \\ e^{\beta_{\text{NS}}} & \phi = \text{NS} \end{cases}$$

The Hessian is a useful tool for functional analysis and forms a critical component of the gradient-based minimizers used to optimize *NRLB* (Appendix B). It can be evaluated with

$$\begin{aligned} \frac{\partial^2 \log \mathcal{L}}{\partial \beta_\phi \partial \beta_\psi} &= \sum_i \left(\frac{c_{i,1}}{\kappa_i} \frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} - \frac{c_{i,1}}{\kappa_i^2} \frac{\partial \kappa_i}{\partial \beta_\psi} \frac{\partial \kappa_i}{\partial \beta_\phi} \right) \\ &\quad - \frac{n}{Z} \sum_i p_{i,0} \frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} + \frac{n}{Z^2} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\psi} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\phi} \end{aligned} \quad (18)$$

where the gradients are the same as before and where

$$\frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} = \begin{cases} \sum_{v \in V_m(\phi)} X_{iv\phi} X_{iv\psi} e^{\sum_{\phi' \in \Phi_m(\phi)} \beta_{\phi'} X_{iv\phi'}} & m(\phi) = m(\psi) \\ 0 & m(\phi) \neq m(\psi) \end{cases}$$

Table 3: Data Matrix $X_{N,o}$ for ATGGATC

	1	2	3	4	5	6	7
A	1	0	0	0	1	0	0
C	0	0	0	0	0	0	1
G	0	0	1	1	0	0	0
T	0	1	0	0	0	1	0

Description of Features

So far, our discussion of *NRLB* has remained agnostic with regards to the types of features ϕ used, with the one caveat that all features must map sequence identity to a numeric ‘data’ value X_ϕ . However, features must be carefully selected in order to maximize model interpretability without sacrificing computability (Chapter 2.2). To this end, two major feature classes will be considered: those with discrete nucleotide identities and those with continuous physical and geometric parameters.

Nucleotide Identities

Transcription factor binding is primarily driven by direct readout, or the ‘recognition’ of nucleotide-specific hydrogen bond donors, acceptors, and nonpolar groups by protein amino acid residues [4]. In direct readout, the amino acid residues recognize specific nucleotide bases at consistent positions relative to the binding interface. Features utilizing this position-specific nucleotide information can be used to construct simple models with straightforward biophysical and structural interpretations [2, 3]. In the following discussion, σ denotes the number of interacting nucleotides, or the order, of the feature.

The simplest such features consist of the nucleotide ($\sigma = 1$) identity at every position within the protein binding interface [5, 6, 7, 2]. Along with the nucleotide identity, these features have an additional position parameter o that represents the offset of the nucleotide within the binding interface. For example, consider the 7 bp binding interface below:

ATGGATC
 $o =$ 0 1 2 3 4 5 6

At each position o , there are a set of four nucleotide parameters $\{A, C, G, T\}$, each with its own β_ϕ . Different nucleotide parameters β_ϕ and data X_ϕ can be uniquely addressed using a combination of nucleotide identity N and position o , and can be conveniently represented as a matrix (Tables 3 and 4). As the β_ϕ parameters in *NRLB* constitute free energies, the matrix of nucleotide parameters can be visualized as an energy logo [2]. The complexity of position-specific nucleotide features can be extended by accounting for the interaction of amino acid residues with bases at two different locations within the binding interface ($\sigma = 2$). The simplest features within this class account for nucleotide

Table 4: PSAM $\beta_{N,o}$

	1	2	3	4	5	6	7
A	$\beta_{A,1}$	$\beta_{A,2}$	$\beta_{A,3}$	$\beta_{A,4}$	$\beta_{A,5}$	$\beta_{A,6}$	$\beta_{A,7}$
C	$\beta_{C,1}$	$\beta_{C,2}$	$\beta_{C,3}$	$\beta_{C,4}$	$\beta_{C,5}$	$\beta_{C,6}$	$\beta_{C,7}$
G	$\beta_{G,1}$	$\beta_{G,2}$	$\beta_{G,3}$	$\beta_{G,4}$	$\beta_{G,5}$	$\beta_{G,6}$	$\beta_{G,7}$
T	$\beta_{T,1}$	$\beta_{T,2}$	$\beta_{T,3}$	$\beta_{T,4}$	$\beta_{T,5}$	$\beta_{T,6}$	$\beta_{T,7}$

identity at adjacent positions o and $o + 1$, and are called nearest neighbor or ‘dinucleotide’ interactions [8, 3, 9]. As such, there are a total of 16 dinucleotides:

AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT

As each feature spans two consecutive positions, there are a total of $k - 1$ dinucleotide positions within a binding interface of length k .

Non-nearest neighbor features further generalize dinucleotides to account for longer range interactions at positions o and $o + s$, where $s > 1$. Significantly, non-nearest neighbor interactions have the same dimensionality per position as dinucleotide features despite spanning a longer physical range. For example, consider the non-nearest neighbor features for $s = 2$:

ANA, ANC, ANG, ANT, CNA, CNC, CNG, CNT
GNA, GNC, GNG, GNT, TNA, TNC, TNG, TNT

Here, N represents aNy base. As such, there are only $k - s$ positions for any non-nearest feature set included in the model. Despite the statistical and numerical advantages afforded by a compact parameterization, such features have not been properly explored by other methods [10], which favor far more complex higher-order feature sets.

Higher-order features account for the interaction between three or more adjacent nucleotides within the binding interface. Previously used features include trinucleotides ($\sigma = 3$; [11]) and tetranucleotides ($\sigma = 4$; [12]). Unlike the non-nearest neighbor features, the dimensionality of these higher order features grows exponentially (4^σ) - possibly resulting in a significant overparameterization of the feature space (Appendix A), introducing both convergence issues and error-prone parameter estimates. For the remainder of this study, we will focus on nucleotide and dinucleotide features and leave the implementation of non-nearest neighbor features for the future.

Physical and Geometric Parameters

Studies have shown that nucleotide interactions can impact DNA conformation [4, 13]. The recognition of this sequence-dependent 3D structure of DNA, or shape readout, has been shown to contribute to binding [14] and used to build more complete models of transcription factor affinity [11, 15]. In shape readout, protein residues recognize modulations to DNA structure at specific positions within the binding interface. Position-specific shape features can be used in

Table 5: DNA Shape Table from [16]

Pentamer	MGW [Å]	Roll [°]	Helical Twist [°]	Propeller Twist [°]
⋮			⋮	
AGCTC	4.63	-3.25	37.66	-0.83
AGCTG	4.8	-2.49	37.48	-1.38
AGCTT	4.14	-3.24	32.14	-1.56
⋮			⋮	

conjunction with nucleotide features to construct models with direct structural interpretations.

Recently, a high-throughput, *in silico* method was developed to estimate local DNA shape parameters using pentamer DNA sequences [16]. This method relates DNA shape parameter values at a given offset o to the local pentamer sequence context centered at o through pre-computed lookup tables. For example, consider the sequence below:

GACTC**AGCTG**GGTTCC
 ↑
 o

The pentamer sequence highlighted in red, starting at $o - 2$ and ending at $o + 2$, can be used to extract values for DNA minor groove width (MGW), roll, helical twist, and propeller twist at position o from the shape table (Table 5). For every shape feature, k DNA shape parameters β_ϕ can be added for a binding interface of length k and data X_ϕ is the shape value at a given position within the interface. However, knowledge of two additional bases flanking the interface is required to extract shape values at its edge, owing to the unique configuration of the pentamer table. While *NRLB* was designed to incorporate shape features, their impact will not be explored in this work.

Minimization and Model Selection

In designing *NRLB*, we wanted a method that builds maximally interpretable binding models with minimal prior information about the protein’s binding preferences. The latter goal is especially relevant in *de novo* motif discovery settings where there is no knowledge of a protein’s binding preferences. Ideally, experimental data should guide the inference and selection of these models; the most natural framework for achieving this goal given our model (6) is maximum likelihood estimation (MLE) and its associated metric of success, the log likelihood. However, due to the non-convexity of *NRLB*’s likelihood (Chapter 2.3, Appendix B), models with the best likelihood may not be physically accurate or relevant, especially when constructed on low-quality data (Figure 1). To further complicate matters, model performance and interpretability can be sensitive to

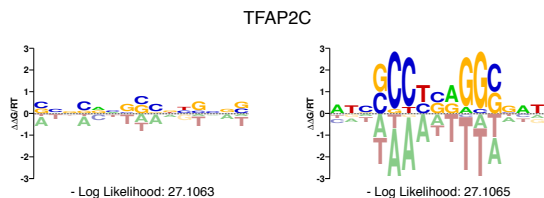


Figure 1: Energy logos of the primary modes and the negative log-likelihood per read of two TFAP2C *NRLB* models. Both models were fit with two reverse complement symmetric modes of length $k = 15$ with nonspecific binding. (Left) $l_f = 2$ (Right) $l_f = 4$; this model corresponds to the known TFAP2C recognition mode.

Table 6: *NRLB* Hyperparameters

Symbol	Definition
m_t	Total number of recognition modes
k_0	Length of kmer features used in R0 Bias model
k or k_m	Binding interface length for mode m
l_f	Length of flanking region
M	Symmetry configuration
NS	Is nonspecific binding used?
Di	Are dinucleotide features included?
For Multiple Binding Mode Models Only:	
k_m^0	Starting binding interface length for mode m
k_m^f	Final binding interface length for mode m

hyperparameter settings (Table 6) and the random initialization used by the minimization algorithms.

To address these issues, special minimization methods were developed to ‘lead’ the L-BFGS optimization algorithm (Appendix B) to consistently produce models concordant with our biological (heuristic) understanding of binding activity with minimal post hoc analysis. Each methodology addresses a unique pathology of both the model and SELEX data, and are combined to produce a robust and powerful inference framework.

Single Mode Hyperparameter and Shift Search

We readily find acceptable single mode models when training on high-quality datasets; however, multiple selection modes may required if contamination or biases exist within the data. Surprisingly, even in the simplest case where a single mode, nucleotide-only model is fit to high quality data, there is no guarantee that the first converged model (or solution) is either the most optimal or in-

terpretable. In fact, most initial solutions inadequately capture the preferences within the protein binding interface. Generally, a sweep over the hyperparameters l_f and k_m must be performed in order to identify optimal settings (Figure 2).

In theory, for a given set of hyperparameters, the minimization methods will find a binding model that captures a majority of the sequence specificity contained within the true protein binding interface. However, in practice, *NRLB* can represent the true binding interface at any offset *within* the binding models it learns, and therefore ignore significant binding preferences. Additionally, there is no way to ‘anchor’ the appropriate offset without providing prior information. This complication arises from the non-convexity of the *NRLB* objective function (Appendix B), and traditionally would require an exhaustive and computationally expensive search for all possible solutions to identify the model with the appropriate offset. Fortunately, the optimal offset can be discovered by ‘hopping’ from one solution to another by shifting the position specific parameters to the left and right. Starting from an initial solution found from an unseeded fit, the parameters can be cycled to the left and right by one base and used as a seed for a new fit. This process can be repeatedly applied, incrementing or decrementing a ‘shift offset’ with every application (Figure 3), and allows us to rapidly explore the space of all solutions. Once a series of solutions have been found, the model with the highest likelihood is selected.

Fitting Dinucleotide Parameters

Adding dinucleotide features to the optimal nucleotide-only model found from a single mode hyperparameter and shift search (described above) can be used to seed a nucleotide and dinucleotide model fit. This seed contains nonzero values only for nucleotide and nonspecific binding parameters. Seeding in this manner not only speeds up convergence, but attempts to find a model that corresponds to the best (and hopefully most concordant) nucleotide-only model fit. After the initial solution with nucleotide and dinucleotide features is found, the same shift search described above can be used to identify additional, related models.

Iterative Mode Discovery

When building models with multiple recognition modes, we rely on the methods described above to iteratively add and ‘discover’ additional modes. The methodology is outlined in detail below.

1. **Fit Nucleotide Only Models** Regardless of the desired feature set, all multiple mode fits begin by learning nucleotide-only models using the following process.
 - (a) A single mode fit with length k_1^0 (the starting length of the first mode) is fit using the shift search method described above. The shift with the highest likelihood is selected.
 - (b) The nucleotide parameters for the previous fit are frozen and an additional mode with length k_2^0 is added. The model is fit using the

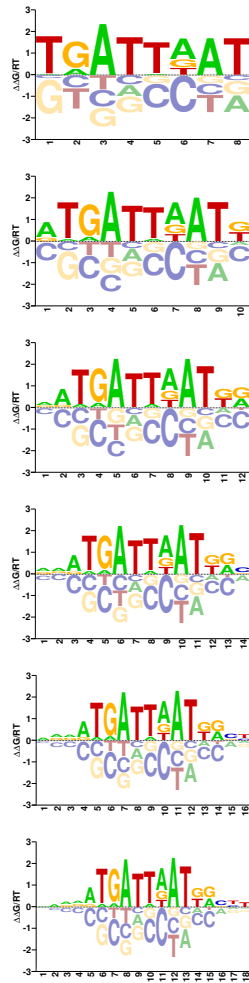


Figure 2: Energy logos for various single mode nucleotide-only Exd-Scr models with nonspecific binding. As k_m is increased from 8 bp to 18 bp in 2 bp increments, l_f is increased from 0 bp to 6 bp in 1 bp increments.

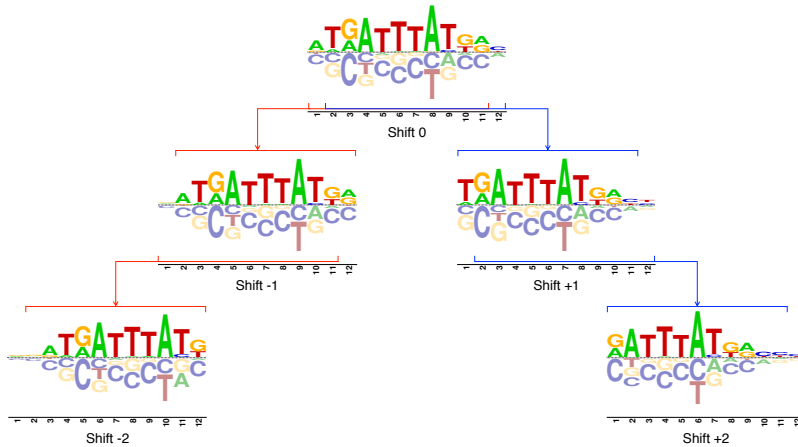


Figure 3: Energy logos for different shift offsets for single mode nucleotide-only Exd-UbxIVa models with nonspecific binding. Red and blue brackets indicate positions used to seed fits.

- shift search symmetry method applied *only* to this new mode. The shift that yields the highest likelihood is selected.
- (c) The model is refit after all nucleotide parameters from the previous modes in the model are unfrozen.
 - (d) The process returns to step (b) for each additional mode until the desired number of modes is reached.
2. **Expand Binding Interface** Once the desired number of modes is reached, the binding interface of each mode m can be iteratively increased to the desired maximum length k_m^f . In each step, a new nucleotide fit is seeded by adding one additional position to the left and right of the model learned in the previous step. This ‘growth’ step is simultaneously performed for all modes until all modes have achieved their target length.
 3. **Add Dinucleotide Features** If dinucleotide features are necessary, each mode’s nucleotide parameters can be used as seeds for a nucleotide and dinucleotide fit as described above.

All steps throughout this process maintain the symmetry status **M**.

Seeding

In some instances, it may be useful to see if the data supports any hypotheses regarding particular recognition modes. These hypotheses can be easily tested by initializing the optimization algorithm with a best guess for the hypothesis (a parameter seed) and comparing it with the final converged solution. This

seeding-and-converging process is straightforward when dealing with a single recognition mode, but requires an intermediate step to work effectively with multiple mode models. Each mode’s relative affinity parameter γ_m defines the relative contribution of each mode to the overall selection rate κ_i . The contribution of each mode is somewhat dataset dependent, and can vary based on TF concentration and other experimental factors. As such, these scaling parameters must be set appropriately when seeding a multiple mode model to prevent the solver from drifting to another solution; however, it is not possible to know the relative scaling between seeded modes beforehand. This difficulty can be addressed with ‘mode regression,’ an inference methodology where only the mode relative affinities γ_m are learned while the seeded feature parameters remain frozen for every recognition mode. Mode regression effectively optimizes (6) over γ_m , with the selection rate given by

$$\kappa_i = \sum_m \gamma_m \sum_{v \in V_m} e^{\sum_{\phi \in \Phi_m} \beta_\phi X_{iv\phi}} + \gamma_{NS} e^{\beta_{NS}} \quad (19)$$

Once the appropriate γ_m values are learned, they can be absorbed into the nucleotide parameters of mode m (Chapter 2.3) and all feature parameters can be unfrozen and fit as before.

2 Numerical Methods

We will focus on elucidating numerical methods for evaluating the likelihoods and their derivatives for both the R0 Bias model and *NRLB* here. Dynamic programming methods used to evaluate partition functions for both models, as well as techniques to condition *NRLB* parameters and compute their errors, will be discussed at length. In addition, we will explore the implications of *NRLB*’s various symmetries and invariances.

2.1 Model Evaluation

R0 Bias Model

We begin with methods for evaluating the log likelihood for an R0 Bias model with k_0 -mer features:

$$\log \mathcal{L} = \sum_i c_{i,0} \log w_i - n_0 \log Z_0$$

Here, n is simply the total number of reads in the R0 library. We can split up the evaluation of the log likelihood into two components: one which deals with the ‘data’ term $c_{i,0} \log w_i$, and the other which deals with the partition function Z_0 . Substituting the definition of w_i (7) into the data term gives

$$\sum_i c_{i,0} \log w_i = \sum_i c_{i,0} \log \left(e^{\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi}} \right) = \sum_i c_{i,0} \sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi}$$

Swapping the summation order gives

$$\sum_i c_{i,0} \sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi} = \sum_{\phi \in \Phi_0} \beta_\phi \underbrace{\sum_i c_{i,0} X_{i\phi}}_{\vec{D}} = \vec{\beta} \cdot \vec{D} \quad (20)$$

The data term reduces to a dot product between the parameter vector $\vec{\beta}$ and the ‘data’ vector \vec{D} that represents the total k -mer counts within the dataset. As \vec{D} is constant for any dataset, (20) can be computed rapidly. The partition function can be evaluated using dynamic programming techniques (see below).

The gradient of the log likelihood of the R0 Bias model is then given by

$$\frac{\partial \log \mathcal{L}}{\partial \beta_\phi} = D_\phi - \underbrace{\frac{n_0}{Z_0} \sum_i X_{i\phi} e^{\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi}}}_{\nabla_\phi Z_0} \quad (21)$$

The first term is simply element ϕ of the data vector from before, while the second term is the weighted average of feature ϕ of the partition function Z_0 , or $\nabla_\phi Z_0$. Once again, dynamic programming techniques will be used to evaluate $\nabla_\phi Z_0$.

NRLB

We begin by discussing the log likelihood:

$$\log \mathcal{L} = \sum_i c_i \log(p_{i,0} \kappa_i) - n \log Z.$$

As before, the evaluation of the data and partition function terms can be separated; however, unlike the R0 Bias model, brute force must be used to compute the data term for all observed S_i (where $c_i \neq 0$). The ‘sliding window’ sums over all windows in κ_i can be efficiently computed using bit-shifting techniques. Dynamic programming will be used to evaluate the partition function:

$$\begin{aligned} Z &= \sum_i p_{i,0} \left(\sum_m \sum_{v \in V_m} e^{\sum_{\phi \in \Phi_m} \beta_\phi X_{iv\phi}} + e^{\beta_{NS}} \right) \\ &= \sum_i \sum_m \sum_{v \in V_m} p_{i,0} e^{\sum_{\phi \in \Phi_m} \beta_\phi X_{iv\phi}} + \sum_i p_{i,0} e^{\beta_{NS}} \end{aligned}$$

Swapping the summation order and applying $p_{i,0} = w_i/Z_0$ and $\sum_i p_{i,0} = 1$ gives

$$Z = \frac{1}{Z_0} \sum_m \sum_{v \in V_m} \underbrace{\sum_i w_i e^{\sum_{\phi \in \Phi_m} \beta_\phi X_{iv\phi}}}_{Z_{mv}} + e^{\beta_{NS}} \quad (22)$$

The evaluation of Z_{mv} by dynamic programming methods will be discussed later.

Applying similar simplifications to the gradient of the log likelihood (17) gives

$$\frac{\partial \log \mathcal{L}}{\partial \beta_\phi} = \sum_i \frac{c_i}{\kappa_i} \frac{\partial \kappa_i}{\partial \beta} - \frac{n}{ZZ_0} \sum_{v \in V_{m(\phi)}} \underbrace{\sum_i w_i X_{iv\phi} e^{\sum_{\phi' \in \Phi_{m(\phi)}} \beta_{\phi'} X_{iv\phi'}}}_{\nabla Z_{mv\phi}} \quad (23)$$

$$\frac{\partial \log \mathcal{L}}{\partial \beta_{NS}} = e^{\beta_{NS}} \cdot \sum_i c_i \left(\frac{1}{\kappa_i} - \frac{1}{Z} \right) \quad (24)$$

The particular form of (24) is useful for reducing precision error during computation. Similar to the R0 Bias model, $\nabla Z_{mv\phi}$ is the feature weighted average of the partition function.

Simplifying the Hessian reduces its complexity and surfaces significant commonalities with the function and gradient evaluation of the log-likelihood. The (ϕ, ψ) component of the Hessian is given by

$$\underbrace{\frac{\partial^2 \log \mathcal{L}}{\partial \beta_\phi \partial \beta_\psi}}_{(\nabla^2 \log \mathcal{L})_{(\phi, \psi)}} = \sum_i \left(\frac{c_i}{\kappa_i} \frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} - \frac{c_i}{\kappa_i^2} \frac{\partial \kappa_i}{\partial \beta_\psi} \frac{\partial \kappa_i}{\partial \beta_\phi} \right) - \frac{n}{Z} \sum_i p_{i,0} \frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} + \frac{n}{Z^2} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\psi} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\phi}$$

The four terms above can be classified as either second derivatives or outer products of gradients. The second derivative terms are block diagonal in the modes:

$$\frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} = \begin{cases} \sum_{v \in V_{m(\phi)}} X_{iv\phi} X_{iv\psi} e^{\sum_{\phi' \in \Phi_{m(\phi)}} \beta_{\phi'} X_{iv\phi}} & m(\phi) = m(\psi) \\ 0 & m(\phi) \neq m(\psi) \end{cases}$$

This block structure decouples the second derivative of each mode from the rest, simplifying the the partition function term:

$$\frac{n}{Z} \sum_i p_{i,0} \frac{\partial^2 \kappa_i}{\partial \beta_\phi \partial \beta_\psi} = \frac{n}{ZZ_{R0}} \sum_{v \in V_{m(\phi)}} \underbrace{\sum_i w_i X_{iv\phi} X_{iv\psi} e^{\sum_{\phi' \in \Phi_{m(\phi)}} \beta_{\phi'} X_{iv\phi'}}}_{\nabla^2 Z_{mv\phi\psi}} \quad (25)$$

$\nabla^2 Z_{mv\phi\psi}$ is zero if $m(\phi) \neq m(\psi)$. The outer product terms can be written as

$$\sum_i \frac{k_i}{\kappa_i^2} \frac{\partial \kappa_i}{\partial \beta_\psi} \frac{\partial \kappa_i}{\partial \beta_\phi} = \left(\sum_i \frac{k_i}{\kappa_i^2} \nabla \kappa_i \otimes \nabla \kappa_i \right)_{(\phi, \psi)} \quad (26)$$

$$\frac{n}{Z^2} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\psi} \sum_i p_{i,0} \frac{\partial \kappa_i}{\partial \beta_\phi} = \frac{n}{Z^2 Z_{R0}^2} (\nabla Z \otimes \nabla Z)_{(\phi, \psi)} \quad (27)$$

Table 7: Shared *NRLB* Functional Elements

Function	κ_i	Z	$\nabla\kappa_i$	∇Z	$\nabla^2\kappa_i$	$\nabla^2 Z$
$\log \mathcal{L}$	X	X				
$\nabla \log \mathcal{L}$	X	X	X	X		
$\nabla^2 \log \mathcal{L}$	X	X	X	X	X	X

where

$$\nabla Z_\phi = \sum_{\nu \in V_m(\phi)} \nabla Z_{m\nu\phi}$$

Functional elements shared between the log likelihood, its gradient, and its Hessian can be exploited to compute lower-order functions at virtually no cost. For example, computing the Hessian implicitly requires the computation of all the elements required to evaluate both the log likelihood and its gradient, and as such, each component only has to be evaluated once (Table 7).

2.2 Dynamic Programming Techniques

Both the R0 Bias model and *NRLB* require the evaluation of partition functions and their derivatives for all 4^l probes in the universe. Unfortunately, brute force evaluation of these sums would consume significant computational resources and make iterative optimization algorithms (Appendix B) prohibitively expensive to use. Alternative approaches involving sampling can drastically reduce computational time, but run the risk of introducing biases. However, dynamic programming approaches dramatically reduce the computational complexity of evaluating Z from exponential to linear in probe length by leveraging the structure of the models. As the selection rate κ_i in *NRLB* involves the R0 frequency as predicted by the R0 Bias model, the approaches for evaluating Z will build upon those developed for Z_0 .

Z_0 and its Derivative

Partition functions can be evaluated either in their ‘natural’ free energy space as an exponentiated sum of parameters or in the more awkward affinity space as a product of affinities:

$$Z_0 = \sum_i w_i = \sum_i e^{\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi}} = \sum_i \prod_{\phi \in \Phi_0} \alpha_\phi^{X_{i\phi}}$$

An example of how w_i can be represented as a product of terms is shown in Table 8 for the first eight sequences for a model where $l = 3$ and $k_0 = 2$.

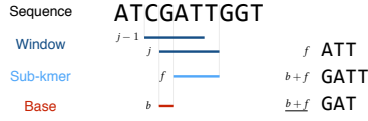


Figure 4: Sequence indices used in the partition function recursion relations.

Table 8: Explicit w_i Expansion

Sequence	w_i Expansion
AAA	$\alpha_{AA}\alpha_{AA}$
AAC	$\alpha_{AA}\alpha_{AC}$
AAG	$\alpha_{AA}\alpha_{AG}$
AAT	$\alpha_{AA}\alpha_{AT}$
ACA	$\alpha_{AC}\alpha_{CA}$
ACC	$\alpha_{AC}\alpha_{CC}$
ACG	$\alpha_{AC}\alpha_{CG}$
ACT	$\alpha_{AC}\alpha_{CT}$
\vdots	\vdots

Summing over these expansions highlights a factorization for Z_0 :

$$\begin{aligned}
 Z_0 &= \sum_i w_i = w_{AAA} + w_{CAA} + w_{GAA} + w_{TAA} + \dots \\
 &= \alpha_{AA}\alpha_{AA} + \alpha_{CA}\alpha_{AA} + \alpha_{GA}\alpha_{AA} + \alpha_{TA}\alpha_{AA} + \dots \\
 &= (\alpha_{AA} + \alpha_{CA} + \alpha_{GA} + \alpha_{TA})\alpha_{AA} + \dots
 \end{aligned}$$

This factorization provides insight into a recursive scheme for calculating Z_0 ; in the expansions above, the identity of the second feature is dependent only on the second base of the first feature:

$$\begin{aligned}
 \text{AAA: } & \alpha_{AA}\alpha_{AA} \\
 \text{ACA: } & \alpha_{AC}\alpha_{CA}
 \end{aligned}$$

As such, only ‘sub-kmers’ of length $k-1$ need to be tracked in the recursion. The following quantities and indices will be used to describe the recursion relations (Figure 4):

j index of the current ‘window’ offset within a probe sequence being evaluated; a window consists of k -bp starting at position j within the sequence. There are $l-k+1$ windows in any sequence on the forward strand, and more if the flanks are considered. For exam-

ple, tetramer ($k = 4$) windows in the sequence ATCGATTGGT are

j	kmer
1	ATCG
2	TCGA
3	CGAT
4	GATT
\vdots	\vdots

- f the sub-kmer currently in question. Sub-kmers are the set of kmers of length $k - 1$.
- b identity of the nucleotide being added. $b + f$ refers to the kmer ϕ obtained by concatenating b with f on the right. $\underline{b + f}$ refers to the sub-kmer one gets by considering the leftmost $k - 1$ bases of $f + b$. For example, let $f = \text{ATT}$ and $b = \text{G}$. Then $b + f = \text{GATT}$ and $\underline{b + f} = \text{GAT}$.
- $Z_0(f, j)$ the partial partition function at window j for sub-kmer f .
- $Z_0(f, j, \phi)$ the partial partition function at window j for sub-kmer f and feature ϕ .

The recursion relation for Z_0 is then

$$Z_0(f, j) = \sum_b Z_0(\underline{b + f}, j - 1) \cdot \alpha_{b+f} \quad \forall f \quad (28)$$

$$Z_0 = \sum_f Z_0(f, \text{final}) \quad (29)$$

Using the example in Figure 4, the expansion in (28) for $f = \text{ATT}$ is

$$\begin{aligned} Z_0(\text{ATT}, j) = & Z_0(\text{AAT}, j - 1) \cdot \alpha_{\text{AATT}} + Z_0(\text{CAT}, j - 1) \cdot \alpha_{\text{CATT}} + \\ & Z_0(\text{GAT}, j - 1) \cdot \alpha_{\text{GATT}} + Z_0(\text{TAT}, j - 1) \cdot \alpha_{\text{TATT}} \end{aligned}$$

When flanking sequences are not considered, the recursion can be initialized with

$$Z_0(f, 1) = \sum_b \alpha_{b+f} \quad \forall f$$

When considering flanking sequences, the initialization is even simpler:

$$Z_0(f, 0) = \begin{cases} 1, & f = \underline{\text{flank}} \\ 0, & \text{otherwise} \end{cases}$$

where flank refers to the $k - 1$ rightmost bases of the left flanking sequence.

$\nabla_\phi Z_0$ is the sum of w_i weighted by the presence of feature ϕ :

$$\nabla_\phi Z_0 = \sum_i X_{i\phi} w_i = \sum_i X_{i\phi} e^{\sum_{\phi \in \Phi_0} \beta_\phi X_{i\phi}} = \sum_i X_{i\phi} \prod_{\phi \in \Phi_0} \alpha_\phi^{X_{i\phi}}$$

The sum above only counts those sequences where $X_{i\phi} \neq 0$; consequently, any systematic recursion method must account for the existence of feature ϕ at any window j in the sequence. One approach isolates all sequences containing ϕ for a given j . For example, consider all sequences containing the kmer feature $\phi = \text{AA}$ for the previous model where $l = 3$ and $k_0 = 2$:

j	Sequences			
1	AAA	AAC	AAG	AAT
2	AAA	CAA	GAA	TAA

Summing over these sequences and factorizing gives

$$\nabla_{\text{AA}} Z_0 = \sum_i X_{i,\text{AA}} w_i = \alpha_{\text{AA}} (\alpha_{\text{AA}} + \alpha_{\text{AC}} + \alpha_{\text{AG}} + \alpha_{\text{AT}}) + (\alpha_{\text{AA}} + \alpha_{\text{CA}} + \alpha_{\text{GA}} + \alpha_{\text{TA}}) \alpha_{\text{AA}}$$

Given this sum, the gradient for a given ϕ can be viewed as a restricted partition function expansion. These restricted expansions can be computed with additional recursion relations designed to track specific features:

$$Z_0(f, j) = \sum_b Z_0(\underline{b+f}, j-1) \cdot \alpha_{b+f} \quad \forall f$$

$$Z_0(f, j, \phi) = \sum_b Z_0(\underline{b+f}, j-1, \phi) \cdot \alpha_{b+f} \quad \forall f \quad (30)$$

$$Z_0(f, j, b+f) = Z_0(\underline{b+f}, j-1) \cdot \alpha_{b+f} \quad \forall f, b \quad (31)$$

$$Z_0 = \sum_f Z(f, \text{final})$$

$$\nabla_\phi Z_0 = \sum_f Z_0(f, \text{final}, \phi) \quad (32)$$

The paired recursion relations (30) and (31) represent two simple operations: the first updates the total sum for feature ϕ at window $j-1$, or $Z_0(f, j-1, \phi)$, to its new value at window j ; this is similar to the recursion relation for computing Z_0 (28). The second relation updates the total sum $Z_0(f, j, \phi)$ with sequences that contain ϕ at j . As before, the relations need be evaluated for all f , with (31) requiring an additional evaluation over all bases b . Additionally, the order in which the recursions are performed is crucial; for every iteration j , (28) must be evaluated first, then (30), and (31) last. Initialization is straightforward when flanking sequences are not considered:

$$Z_0(f, 0) = 1, \quad Z_0(f, 0, \phi) = 0 \quad \forall f, \phi$$

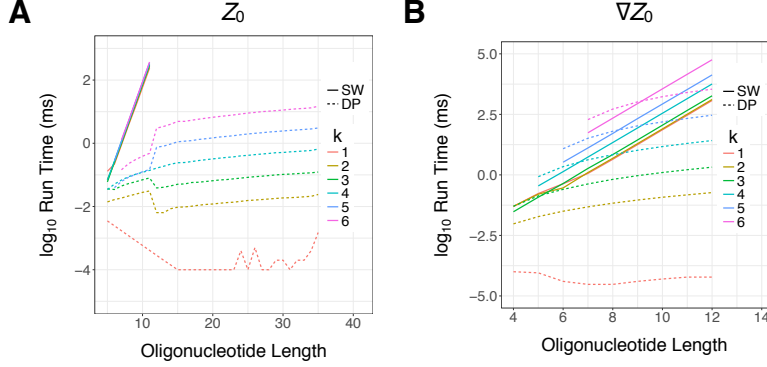


Figure 5: Average \log_{10} run times for Z_0 (A) and $\nabla_{\phi}Z_0$ (B) for various probe lengths and feature orders k_0 , using both brute-force ‘sliding-window’ sums (SW) and dynamic programming (DP) approaches. To conserve time, sliding window evaluations were not performed for probe lengths greater than 11. The dynamic programming evaluation for $k_0 = 1$ frequently took less than $1 \mu\text{s}$ per run, resulting in significant measurement error.

If flanking sequences are used, the previous initialization method can be used for $Z(f, 0)$. Figure 5 compares the run times for both brute-force and dynamic programming approaches for evaluating Z_0 and $\nabla_{\phi}Z_0$, demonstrating near linear time complexity for the dynamic programming approach.

Z_{mv} and its Derivatives

NRLB partition functions contain two models: the R0 Bias model and the biophysical model of the selection rate κ_i . Fortunately, as both models are amenable to factorization, *NRLB* recursion relations can be built upon the R0 Bias model methodology. However, the feature sets employed by *NRLB* pose unique challenges for recursive evaluation. The following section is largely concerned with addressing these issues.

We begin by noting that the partition functions (22), (23), and (25) do not contain cross-mode terms; therefore, without loss of generality, the mode index m can be dropped. The log likelihood, gradient, and Hessian partition functions can be written as

$$\begin{aligned}
 Z_v &= \sum_i \prod_{\phi \in \Phi_0} \alpha_{\phi}^{X_{i\phi}} \prod_{\phi' \in \Phi} \alpha_{\phi'}^{X_{iv\phi'}} \\
 \nabla Z_{v\phi} &= \sum_i X_{iv\phi} \prod_{\phi' \in \Phi_0} \alpha_{\phi'}^{X_{i\phi'}} \prod_{\phi'' \in \Phi} \alpha_{\phi''}^{X_{iv\phi''}} \\
 \nabla^2 Z_{v\phi\psi} &= \sum_i X_{iv\phi} X_{iv\psi} \prod_{\phi' \in \Phi_0} \alpha_{\phi'}^{X_{i\phi'}} \prod_{\phi'' \in \Phi} \alpha_{\phi''}^{X_{iv\phi''}}
 \end{aligned}$$

model with a 10 bp binding interface. Consequently, the recursion relations must use 5 bp windows j and 4 bp sub-kmers f . Then, for the random 16 bp sequence discussed earlier, the sequence at $j = 7$ is underlined below (the active region is in red, for $v = 6$):

GGACTCAGCTGGTTCC

Then, $X_{6,\phi}$ equals one for the nucleotide feature G at position 6 within the binding interface and the dinucleotide feature TG at position 6, and equals the value of the MGW for the pentamer AGCTG for the MGW feature at position 4. $X_{6,\phi}$ is zero for all other features. Note that the DNA shape feature window is offset by 2 bp within the binding interface.

$u_v(b + f, j)$ the update function representing the contribution of the selection model at current window j for sequence $b + f$:

$$u_v(b + f, j) = \alpha_{b+f} \prod_{\phi \in \Phi} \alpha_{\phi}^{X_{v,\phi}(b+f,j)} \quad (33)$$

where α_{b+f} is a contribution from an R0 model feature. $u_v = \alpha_{b+f}$ when not in any selection model active region.

$Z_v(f, j)$ the partial partition function at window j for sub-kmer f at view v .

$Z_v(f, j, \phi)$ the partial partition function at window j for sub-kmer f and feature ϕ at view v .

$Z_v(f, j, \phi, \psi)$ the partial partition function at window j for sub-kmer f and features ϕ and ψ at view v .

The recursion relations are

$$Z_v(f, j) = \sum_b Z_v(\underline{b+f}, j-1) \cdot u_v(b+f, j) \quad (34)$$

$$Z_v(f, j, \phi) = \sum_b Z_v(\underline{b+f}, j-1, \phi) \cdot u_v(b+f, j) \quad (35)$$

$$Z_v(f, j, \phi) = Z_v(\underline{b+f}, j-1) \cdot u_v(b+f, j) \cdot X_{v,\phi}(b+f, j) \quad (36)$$

$$Z_v(f, j, \phi, \psi) = \sum_b Z_v(\underline{b+f}, j-1, \phi, \psi) \cdot u_v(b+f, j) \quad (37)$$

$$Z_v(f, j, \phi, \psi) = Z_v(\underline{b+f}, j-1) \cdot u_v(b+f, j) \cdot X_{v,\phi}(b+f, j) \cdot X_{v,\psi}(b+f, j) \quad (38)$$

$$Z_v(f, j, \phi, \psi) = Z_v(\underline{b+f}, j-1, \phi) \cdot u_v(b+f, j) \cdot X_{v,\psi}(b+f, j) \quad (39)$$

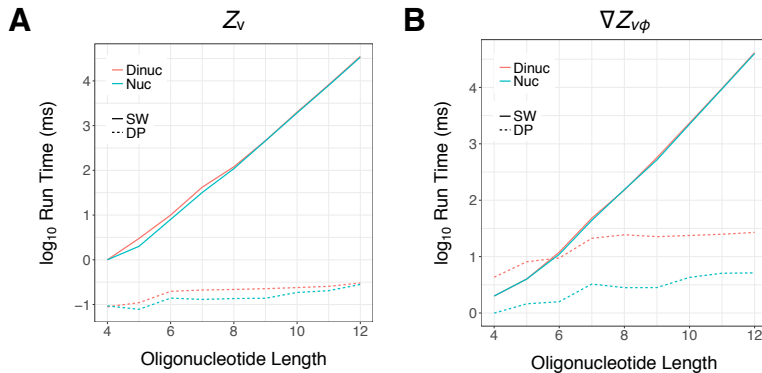


Figure 6: Average \log_{10} run times for Z_v (A) and $\nabla Z_{v\phi}$ (B) for various probe lengths using both brute-force ‘sliding-window’ sums (SW) and dynamic programming (DP) approaches. In both panels, $k = 4$, $f = 4$, and $k_0 = 3$.

The relations above need to be evaluated for all f and b (when applicable). The partition functions can then be found by summing

$$Z_v = \sum_f Z_v(f, \text{final}) \quad (40)$$

$$\nabla Z_{v\phi} = \sum_f Z_v(f, \text{final}, \phi) \quad (41)$$

$$\nabla^2 Z_{v\phi\psi} = \sum_f Z_v(f, \text{final}, \phi, \psi) \quad (42)$$

The previously described initialization schemes can be used for the *NRLB* recursion relations, however care must be taken when selecting the starting window j – the starting window is entirely dependent on the combination of features, views v and flanks f used. Figure 6 compares the run times for both brute-force and dynamic programming approaches for evaluating Z_v and $\nabla Z_{v\phi}$, demonstrating significant improvement in computational performance for the dynamic programming approach.

2.3 Invariances and Symmetries of the Likelihood

The value of *NRLB*’s log likelihood (16) does not always change when its parameters $\vec{\beta}$ change; in fact, there are permutations and transformations T of the parameters $\vec{\beta}$ that leave its value unchanged:

$$\log \mathcal{L}(\vec{\beta}) = \log \mathcal{L}(T(\vec{\beta})) \quad (43)$$

NRLB is said to be invariant under T if it satisfies the above equality. This section will discuss transformations $T(\vec{\beta}) \rightarrow \vec{\beta}^*$ satisfying this equality and how

they can be used to produce a ‘standardized’ representation of *NRLB* parameters.

Reverse-Complement Symmetry

In this basic symmetry, the transformation T_{RC} permutes β_ϕ of a specific mode in a binding model by taking their ‘reverse complement.’ For a sequence-based feature ϕ at position o , the reverse complement operation is

$$\beta_{\phi,o} = \beta_{\bar{\phi},k-o}^* \quad (44)$$

where $\bar{\phi}$ denotes the reverse complement of ϕ and k is the length of the binding interface. For all other features (such as DNA shape), the reverse complement operation is

$$\beta_{\phi,o} = \beta_{\phi,k-o}^* \quad (45)$$

This symmetry arises from the evaluation of views on both strands, resulting in every view having a paired reverse complement. As such, computing the likelihood with β or β^* is equivalent.

Nucleotide and Dinucleotide Invariances

At a given position o , data for nucleotide features must obey

$$\sum_b X_{b,o} = 1$$

where $b \in \{A, C, G, T\}$. We can define a transformation T_N where the parameters $\beta_{b,o}$ are uniformly shifted by δ_o :

$$\sum_b \underbrace{(\beta_{b,o} + \delta_o)}_{\beta_{b,o}^*} X_{b,o} = \delta_o + \sum_b \beta_{b,o} X_{b,o}$$

Applying T_N at a specific offset to a single mode model without nonspecific binding gives

$$\begin{aligned}
p_i &= \frac{p_{i,0} \sum_v e^{\sum_{\phi \in \Phi^*} \beta_{\phi}^* X_{iv\phi} + \sum_{\phi \notin \Phi} \beta_{\phi} X_{iv\phi}}}{\sum_i p_{i,0} \sum_v e^{\sum_{\phi \in \Phi^*} \beta_{\phi}^* X_{iv\phi} + \sum_{\phi \notin \Phi} \beta_{\phi} X_{iv\phi}}} \\
&= \frac{e^{\delta_o} \cdot p_{i,0} \sum_v e^{\sum_{\phi \in \Phi} \beta_{\phi} X_{iv\phi}}}{e^{\delta_o} \cdot \sum_i p_{i,0} \sum_v e^{\sum_{\phi \in \Phi} \beta_{\phi} X_{iv\phi}}} \\
&= \frac{p_{i,0} \sum_v e^{\sum_{\phi \in \Phi} \beta_{\phi} X_{iv\phi}}}{\sum_i p_{i,0} \sum_v e^{\sum_{\phi \in \Phi} \beta_{\phi} X_{iv\phi}}}
\end{aligned}$$

demonstrating the invariance. Here, Φ^* refers to the set of features transformed by T_N . T_N can be repeatedly applied at all offsets o within the mode; in such a case, there is a global shift given by

$$\delta = \sum_o \delta_o$$

When the model contains multiple binding modes and/or nonspecific binding, all modes must be rescaled by e^{δ} . For instance, consider an overall shift of δ for one mode in a model with two modes and nonspecific binding:

$$\begin{aligned}
p_i &= \frac{p_{i,0} \left[\sum_m \sum_{v \in V_m} e^{\sum_{\phi \in \Phi_m} \beta_{\phi} X_{imv\phi}} + e^{\beta_{NS}} \right]}{\sum_i p_{i,0} \left[\sum_m \sum_{v \in V_m} e^{\sum_{\phi \in \Phi_m} \beta_{\phi} X_{imv\phi}} + e^{\beta_{NS}} \right]} \\
&= \frac{p_{i,0} \left[\sum_{v \in V_1} e^{\sum_{\phi \in \Phi_1} \beta_{\phi}^* X_{i1v\phi}} + e^{\delta} \sum_{v \in V_2} e^{\sum_{\phi \in \Phi_2} \beta_{\phi} X_{i2v\phi}} + e^{\beta_{NS} + \delta} \right]}{\sum_i p_{i,0} \left[\sum_{v \in V_1} e^{\sum_{\phi \in \Phi_1} \beta_{\phi}^* X_{i1v\phi}} + e^{\delta} \sum_{v \in V_2} e^{\sum_{\phi \in \Phi_2} \beta_{\phi} X_{i2v\phi}} + e^{\beta_{NS} + \delta} \right]}
\end{aligned}$$

While the rescaling factor e^{δ} clearly shifts β_{NS} , there are many ways to absorb it into the parameters β_{ϕ} of a given mode. We will maintain the convention where only the nucleotide parameters at the first offset ($o = 1$) of every mode are shifted.

Next, consider dinucleotide features at position o . Let $B \in \{\text{A, C, G, T}\}$, and let $X_{B+b,o}$ and $\beta_{B+b,o}$ denote the data and the parameter, respectively, for the dinucleotide feature $B+b$ at offset o . $+$ is the concatenation operation defined earlier. If $X_{B,o} = 1$, then data for dinucleotide features must obey

$$\sum_b X_{B+b,o} = 1 \quad \text{and} \quad \sum_b X_{b+B,o-1} = 1$$

Given this, the sum $\sum_{\phi \in \Phi} \beta_{\phi} X_{iv\phi}$ for any mode is invariant under transformations T_D of the type

$$\begin{aligned} \beta_{B,o} X_{B,o} + \sum_b \beta_{B+b,o} X_{B+b,o} &= \underbrace{(\beta_{B,o} + \delta_o)}_{\beta_{B,o}^*} X_{B,o} + \sum_b \underbrace{(\beta_{B+b,o} - \delta_o)}_{\beta_{B+b,o}^*} X_{B+b,o} \\ \beta_{B,o} X_{B,o} + \sum_b \beta_{b+B,o-1} X_{b+B,o-1} &= \underbrace{(\beta_{B,o} + \delta_o)}_{\beta_{B,o}^*} X_{B,o} + \sum_b \underbrace{(\beta_{b+B,o-1} - \delta_o)}_{\beta_{b+B,o-1}^*} X_{b+B,o-1} \end{aligned}$$

For any given offset o , there are only 7 unique dinucleotide invariances. We can see this by representing T_N and T_D as vectors:

$$\vec{\beta}^* = \vec{\beta} + \delta \cdot \vec{T} \tag{46}$$

All invariant transformations can then be combined into a single transformation matrix \mathbf{T} . \mathbf{T} for a single dinucleotide offset o is

$$\mathbf{T} = \begin{matrix} & \vec{T}_N & \vec{T}_N & \vec{T}_D & \vec{T}_D & \vec{T}_D & \vec{T}_D & \vec{T}_D & \vec{T}_D & \vec{T}_D & \vec{T}_D \\ \begin{matrix} A_1 \\ C_1 \\ G_1 \\ T_1 \\ A_2 \\ C_2 \\ G_2 \\ T_2 \\ AA_1 \\ AC_1 \\ AG_1 \\ AT_1 \\ CA_1 \\ CC_1 \\ CG_1 \\ CT_1 \\ GA_1 \\ GC_1 \\ GG_1 \\ GT_1 \\ TA_1 \\ TC_1 \\ TG_1 \\ TT_1 \end{matrix} & \left[\begin{array}{cccccccccccc} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \end{matrix}$$

The rank of this matrix, which is equivalent to the number of unique invariances, is 9, implying that there are only 7 dinucleotide invariances. The vector representation of T_N and T_D used above will prove useful for defining a standardized representation of *NRLB* models.

Standardized Representations

The invariant transformations described above define directions that form the null space of the Hessian (18) and represent an over-parameterization of our model. Movement along these ‘null vectors’ does not contribute to a change in the log likelihood (16) - in other words, we can arbitrarily add any number of null vectors to the model parameters β without changing our predictions (46). We can remove this degeneracy by orthogonalizing β relative to the orthonormal basis computed by the QR Decomposition of \mathbf{T} . Removing this degeneracy is critical for the comparison of different models (Figure 7).

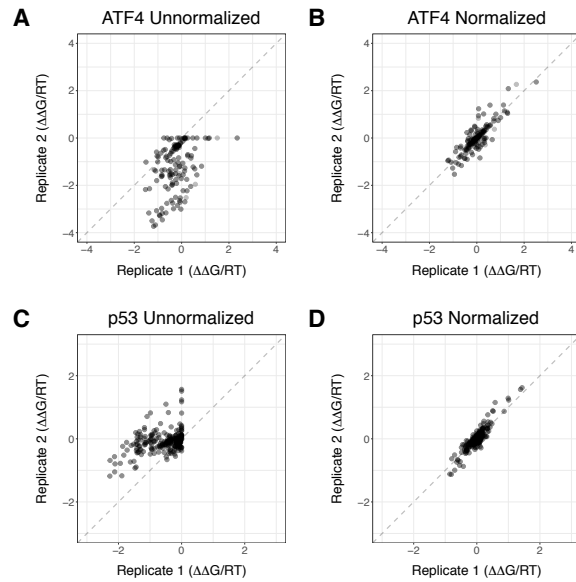


Figure 7: **(A, B)** Comparison of two ATF4 models trained on different datasets. **(A)** Unnormalized nucleotide and dinucleotide parameters show no agreement, suggesting that the two models represent alternative binding modes. **(B)** Normalized parameters show that the models are fairly similar and represent the same binding mode. **(C, D)** Comparison of two models trained on data from different p53 variants. **(C)** Unnormalized nucleotide and dinucleotide parameters show no agreement, suggesting that the two variants have different DNA binding specificities. **(D)** Normalization shows that the models trained on the two variants have nearly identical binding specificities.

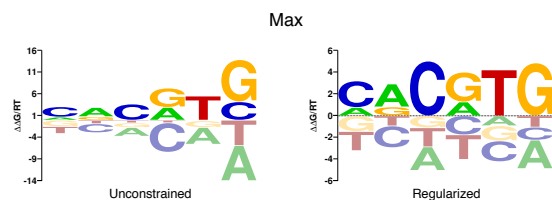


Figure 8: Energy logos for Max corresponding to an unconstrained ($\lambda = 0$) and a regularized ($\lambda = 1 \times 10^{-6}$) fit shown in Figure 9. The unconstrained model contains parameters with extremely large values (note scale), while the regularized model displays more rational parameter values.

2.4 Parameter Conditioning

For some SELEX-seq datasets, we discovered that *NRLB* consistently discovered models with extremely large parameter values, inconsistent with the known range of relative affinities of the transcription factor in question (Figure 8). This behavior is indicative of the model overfitting to data-specific biases [17]. Ideally, we should penalize these ‘run-away’ parameters from growing too large while ensuring that the predictions of the remaining parameters remain unchanged. Regularization is a commonly used statistical technique to achieve these aims [17]. Of the many penalty functions used as regularizers, the L_2 norm (also known as ridge regression) is the most appealing for us; it is easy to implement and is both smooth and convex, fitting perfectly within our existing optimization framework [[18]; Appendix B]. The modified likelihood is then

$$\mathcal{L}'(\vec{\beta} | \text{data}) = \mathcal{L}(\vec{\beta} | \text{data}) + n\lambda \sum_{\phi} \beta_{\phi}^2 \quad (47)$$

where the parameter λ is a per-read weight used to control the strength of the regularization term.

In order to understand the impact of the L_2 penalty on *NRLB* models, we compared β_{ϕ} and Hessian eigenvalues for different values of λ on three different datasets: the previously discussed dataset that produced models with run-away parameters (Figure 9A) and two where no such issues were found (Figure 9B, C). Our results suggest that adding a very weak L_2 norm - 1×10^{-6} per read - is sufficient to control parameter values without fundamentally altering the other parameter values, or even the local structure of the function.

From a Bayesian perspective, the L_2 penalty is equivalent to the addition of a Gaussian prior [17], and can appear to violate our modeling goals. However, it should be noted that the impact of the penalty on log likelihood is roughly 7 orders of magnitude smaller than that of the data (for a typical SELEX library design). As such, this ‘prior’ can be considered highly uninformative and be viewed more as a perturbation that aids in convergence and prevents the occurrence of unreasonable parameter values.

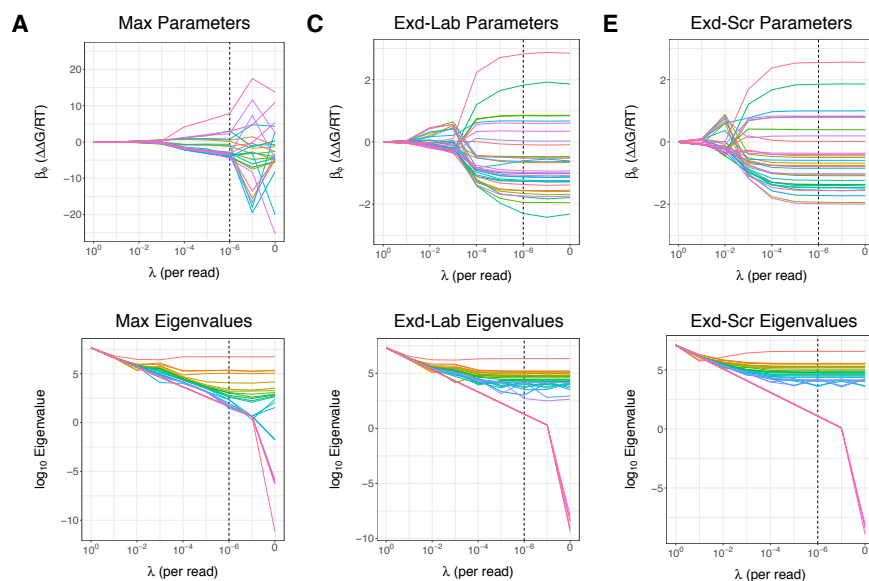


Figure 9: Parameter values and eigenvalues of the Hessian as a function of regularization strength λ for various fits. **(A)** Models for Max; unconstrained model ($\lambda = 0$) contains parameters with extremely large values. Regularization with the desired weight (black dashed line) does not significantly alter eigenvalue structure yet effectively controls parameter values. PSAMs for both cases shown in Figure 8. **(B)** Models for Exd-Labial; regularization at the desired weight (black dashed line) does not alter either the parameters or the eigenvalues in a significant way. **(C)** Models for Exd-Scr; once again, regularization at the desired weight does not alter the parameters or eigenvalues in a significant way.

2.5 Parameter Error

We can use the Hessian (18) to estimate the standard errors of the maximum likelihood estimates. We rely on the convergence of maximum likelihood estimates $\hat{\beta}_\phi$, in distribution, to the normal distribution:

$$\hat{\beta}_\phi \approx \text{Normal}(\beta_\phi, [\mathbf{J}(\vec{\beta})]_{\phi\phi}^{-1})$$

where $\mathbf{J}_{\phi\phi}^{-1}$ is (ϕ, ϕ) element of the inverse of the expected Fisher Information matrix. However, we will use the observed Fisher Information matrix \mathbf{J}' instead, as it is easier to compute and is preferred over the expected Fisher Information matrix [19]. \mathbf{J}' is given by

$$\mathbf{J}'(\hat{\beta}) = -\nabla^2 \log \mathcal{L}(\vec{\beta})|_{\vec{\beta}=\hat{\beta}} = -\mathbf{H}|_{\vec{\beta}=\hat{\beta}}$$

where \mathbf{H} is the Hessian. The standard error for $\hat{\beta}_\phi$ is given by

$$\text{Var}(\hat{\beta}_\phi) = \sqrt{[\mathbf{J}'(\hat{\beta})]_{\phi\phi}^{-1}} \quad (48)$$

Due to the overparameterization of *NRLB* and the existence of a null space, the inverse of its Hessian is not defined.² From our earlier discussion (Chapter 2.3), these null directions carry no useful information and can be ignored by computing the pseudoinverse. Alternatively, the Hessian can be inverted in the true parameter space via Jacobian transformations - this method was not pursued as it is less flexible. Unfortunately, the standard pseudoinverse fails when L_2 regularization and symmetrization are used in learning an *NRLB* model.

It is relatively straightforward to deal with symmetrization - if they exist, the inversion operation should take place in the reduced space. Handling the addition of the L_2 norm is more involved, as it breaks the invariance relations discussed in Chapter 2.3 and results in a full-rank Hessian. As the smallest diagonal entries are no longer zero, the error estimates are completely dominated by the highly uninformative contribution of the L_2 norm. Given that it is a weak perturbation, the null directions of the original likelihood are still irrelevant and should be removed. In fact, the value of λ was chosen to *avoid* altering the functional landscape near the solution. Expressing the inverse of the Hessian of the augmented likelihood, \mathbf{H}'^{-1} , in terms of \mathbf{H} informs a strategy for removing the impact of the L_2 norm on estimating parameter errors:

$$\begin{aligned} \mathbf{H}' &= \mathbf{H} + 2\lambda\mathbf{I} = \mathbf{V}\mathbf{D}\mathbf{V}^\top + 2\lambda\mathbf{I} = \mathbf{V}\mathbf{D}\mathbf{V}^\top + 2\lambda\mathbf{V}\mathbf{V}^\top\mathbf{I} = \mathbf{V}(\mathbf{D} + 2\lambda\mathbf{I})\mathbf{V}^\top \\ \mathbf{H}'^{-1} &= \mathbf{V}(\mathbf{D} + 2\lambda\mathbf{I})^{-1}\mathbf{V}^\top \end{aligned}$$

Here, \mathbf{V} is a unitary matrix that diagonalizes \mathbf{H} and \mathbf{D} is a diagonal matrix consisting of the eigenvalues of \mathbf{H} . This relation suggests that the original null vectors can be identified by subtracting 2λ from the eigenvalues of \mathbf{H}' . The modified pseudoinverse is outlined in Algorithm 1.

²For a square matrix \mathbf{H} , the inversion operation is not defined if null vectors exist, as they have zero eigenvalues.

Algorithm 1 Psuedoinverse with Symmetry and the L_2 Norm

```

function PSUEDOINVERSE (H)
   $\epsilon \leftarrow 1 \times 10^{-14}$                                  $\triangleright$  Define tolerance limit
   $[\mathbf{V}, \mathbf{D}] \leftarrow \text{eig}(\frac{1}{2}(\mathbf{H} + \mathbf{H}^\top))$        $\triangleright$  Eigendecomposition of H
   $\xi \leftarrow \text{diag}(\mathbf{D}) - 2\lambda$                          $\triangleright$  Remove  $L_2$  component
  for  $i = 1$  to  $\text{length}(\xi)$  do                           $\triangleright$  Set null vectors to 0
    if  $\xi_i < \epsilon \cdot \max(|\xi|)$  then
       $D_{ii} \leftarrow 0$ 
    end if
  end for
   $[\mathbf{V}, \mathbf{D}] \leftarrow \text{eig}(\mathbf{M}\mathbf{V}\mathbf{D}\mathbf{V}^\top\mathbf{M}^\top)$            $\triangleright$  Eigendecomposition in reduced space
   $\mathbf{D}' \leftarrow \mathbf{D}$ 
   $\xi \leftarrow \text{diag}(\mathbf{D})$ 
  while true do                                            $\triangleright$  Compute psuedoinverse
    for  $i = 1$  to  $\text{length}(\xi)$  do                           $\triangleright$  Invert the diagonal matrix D
      if  $\xi_i < \epsilon \cdot \max(|\xi|)$  then
         $D'_{ii} \leftarrow 0$ 
      else
         $D'_{ii} \leftarrow 1/D_{ii}$ 
      end if
    end for
     $\mathbf{H}^{-1} \leftarrow \mathbf{V}\mathbf{D}'\mathbf{V}^\top$ 
    if  $H_{ii}^{-1} > 0 \forall i$  then                                $\triangleright$  Test if pseudoinverse succeeded
      return  $\mathbf{M}^\top\mathbf{H}^{-1}\mathbf{M}$                                 $\triangleright$  Return inverse in full space
    else
       $\epsilon \leftarrow 10 \cdot \epsilon$                             $\triangleright$  Decrease tolerance otherwise
    end if
  end while
end function

```

Table 9: Parameter Settings for R0 Synthetic Data

l	k_0	KM
9	3, 4, 5	100 to 1,000 in steps of 100
10	3, 4, 5, 6	
11	4, 5, 6	2,000 to 10,000 in steps of 1,000

3 R0 Bias Model Validation

This chapter is concerned with demonstrating the ability of our models to accurately parameterize variation in SELEX and other experimental datasets. Synthetic data will be used to verify the ability of the R0 Bias model to reliably infer model parameters. Cross-platform validations and new experimental data will be used to show that *NRLB* models generalize well to other datasets and match or exceed the performance of other methods. Lastly, we will show that *NRLB* provides reliable models even under adverse conditions.

3.1 Synthetic Data

The ability of the R0 model to reliably infer parameters was characterized using synthetic data simulating the initial pool of SELEX probes, assuming it follows the distribution defined by (9) that does not go into the flanking region. The synthetic data was generated by drawing a fixed number of reads from this distribution using random parameter values β_ϕ that were first drawn from a normal distribution with $\mu = 0$ and $\sigma = 1$ and then shifted so that the largest value is 0. Numerous random parameter sets were generated for different l and k_0 values given in Table 9. Sequencing depth (or the total number of draws from the given distribution) was represented in terms of the ‘kmer multiplicity’ KM , or the expected k_0 -mer count of a uniformly random SELEX library with probe length l . KM is a natural parameterization of the sequencing depth, as the accuracy of inferred the parameters should increase with the number of observations of every k_0 -mer in the dataset. Sequencing depth is then

$$\# \text{ reads} = \frac{4^k}{l - k + 1} \cdot KM$$

For every random parameter set, numerous synthetic datasets were generated for different values of KM shown in Table 9; this scheme allows us to test the relationship between parameter accuracy and sequencing depth. Actual parameter values were compared to those inferred from models fit to synthetic data using coordinate search (Appendix B; Figure 10). Our results indicate that the R0 Bias model accurately recovers parameter values, especially for kmer multiplicities is near 10000. Practically, this result is promising, as it is easy to achieve such kmer multiplicities from real data - for example, it is fairly inexpensive to sequence the roughly 3.7 million reads required to achieve

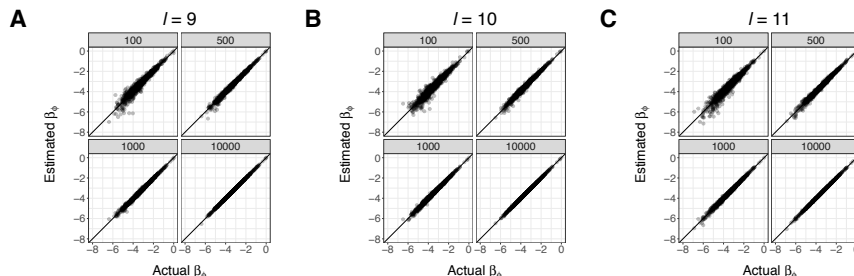


Figure 10: Selected comparisons of actual and inferred parameter values for the R0 Bias model using synthetic data constructed for different $l = 9$ (A), $l = 10$ (B), and $l = 11$ (C). In each panel, the same set of β_ϕ values are used to generate the synthetic datasets. $k_0 = 5$ in all panels.

$KM = 10,000$ when building a model with $k_0 = 6$ on a SELEX library design where $l = 16$.

In these experiments, the k_0 value used to generate the synthetic data was provided to the model. However, the true value of k_0 will not be known for experimental data *a priori*; rather, the cross-validation method described in Chapter 1.2 must be used. To verify that cross-validation can correctly identify the appropriate k_0 , paired test and training synthetic data with identical sequencing depth was generated for $l = 9$ and $k_0 = 1, 2, \text{ or } 3$ using the process described above. Every synthetic dataset was fit with models using k_0 values ranging from 1 through 6. Performance on the test datasets was measured using the log likelihood; the model exhibiting the highest cross-validated likelihood was selected. The method successfully identifies the correct k_0 value for these test cases (Figure 11), although it is easier to do so for smaller KM .

3.2 Real Data

Next, we wanted to understand if the fixed flanking regions introduce biases in the initial pool of reads. As we are attempting to understand the properties of real data, it would be inappropriate to use synthetic data to address this question. Instead, we used real experimental data in the `mplex` library from [1]; this dataset contains two identically prepared and multiplexed initial pools with and form natural test and training datasets (see Table 10 for more information). Using the L-BFGS minimizer (Appendix B), we fit models to the training data that either included or ignored the fixed flanking regions for k_0 ranging from 1 to 7. We evaluated model performance on the test data (Figure 12) and found that the fixed flanking regions bias predictions and noted similar behavior in other datasets. Consequently, we decided to only use models that consider the fixed flanking region.

Next, we wanted to assess and compare the ability of the Markov model from

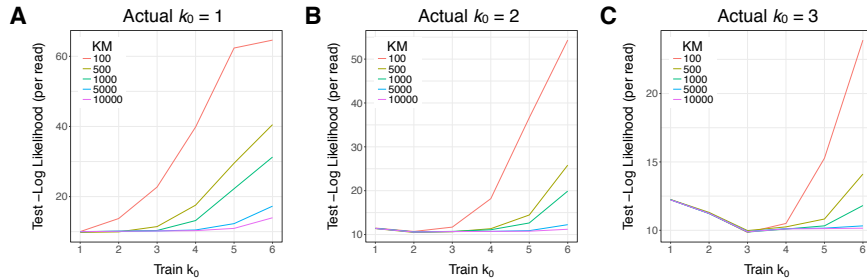


Figure 11: Plots of the test log likelihood using models trained on synthetic data with different k_0 values. Cross-validation selects the k_0 value that produces the lowest test log likelihood. For all cases, the lowest log likelihood values are achieved when the train k_0 equals the actual k_0 value. In each panel, the same set of β_ϕ values are used to generate the synthetic datasets. $l = 9$ in all panels.

Table 10: mplex Initial Pool Metadata

Type	Barcode	Total Reads	Length	Highest Count
Train	CCAGCTG	6,973,386	16	4
Test	CCACGTC	8,428,267	16	3

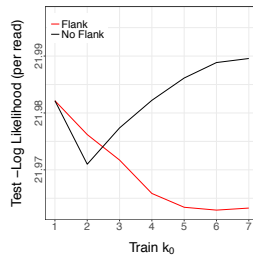


Figure 12: Test log likelihood of models that include (black line) or ignore (red line) the fixed flanking regions while training on data from [1].

[1] and the R0 Bias model to accurately parameterize biases in the initial pool. While the Markov model was able to accurately represent biases at the level of 8-mers [1], *NRLB* requires initial round models to represent the biases of all probes. To quantify this, we assumed that the observed probe counts followed a Poisson distribution and used either model to predict the frequency of all 4^l probes. Probes were then binned by expected frequency and their counts were aggregated. The observed counts were used to compute mean and variance in every bin (mean-variance method); in addition, the number of probes observed twice (n_2), once (n_1), or not at all (n_0) was recorded and used to compute the ratios n_1/n_0 and n_2/n_0 for each bin (ratio method). For data following a Poisson distribution with mean μ , the expected value of these ratios is given by

$$\frac{n_1}{n_0} = \frac{\text{Pois}(1, \mu)}{\text{Pois}(0, \mu)} = \mu \quad \text{and} \quad \frac{n_2}{n_0} = \frac{\text{Pois}(2, \mu)}{\text{Pois}(0, \mu)} = \frac{\mu^2}{2}$$

We used a previously described method [1] to build a 5th order Markov model on the training dataset discussed earlier (Table 10). Applying either the mean-variance or count ratio method to this model shows no agreement between the observed and expected values (Figure 13A). Surprisingly, the same analysis applied to a $k_0 = 6$ R0 Bias model trained on the same data shows near-perfect agreement between observed and expected frequencies for nearly 2.5 orders of magnitude (Figure 13B). The more stringent ratio method highlights mild over-dispersion at the extreme frequency ranges; this is to be expected, as fewer observed counts in that regime lead to noisier estimates. We observed similar trends with other datasets as well, firmly establishing the ability of our model to accurately quantify biases within the initial pool.

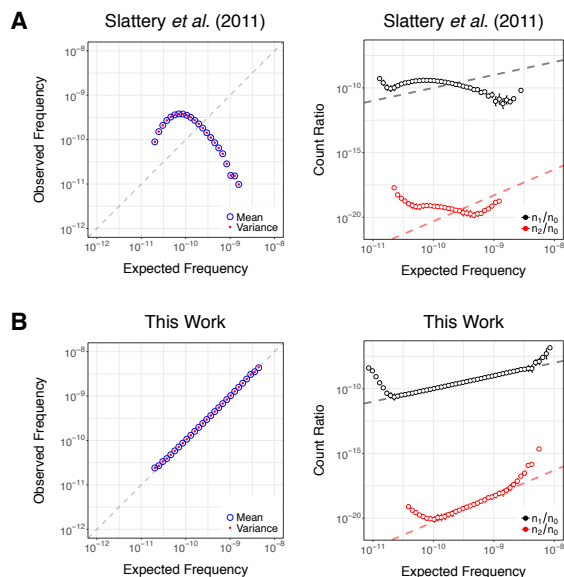


Figure 13: **(A)** Assessing the ability of Markov models constructed using the approach of [1] to accurately parametrize biases in R0 at the level of the entire probe. (Left) Probes are binned according to expected frequency; the mean (blue circles) and variance (red dots) of observed values deviate significantly from expectation (grey dashed line). (Right) A more sensitive analysis of the same Markov model using the count ratio method. Probes are binned according to expected frequency and the number of times each probe was observed; error bars are indicated by vertical lines. The observed ratios of these count values deviate significantly from expectation (dashed lines) assuming Poisson statistics. **(B)** (Left) Mean-variance analysis as in panel A for the improved bias model used in this work. Mean and variance are in near perfect agreement over the entire range. (Right) Count ratio analysis for the improved bias model. There is mild over-dispersion at extreme points.

A R0 Bias Model Feature Selection

The model presented in Eq. 9 uses the kmer composition of each probe to infer sequence biases in the initial R0 pool. We originally used kmer features ϕ consisting of all kmers of length 1 through k . For example, a bias model considering up to 3-mer features would learn on the nucleotide, dinucleotide, and trinucleotide kmer counts for all probes.

As a first test, we attempted to learn the model parameters used to generate synthetic data, evaluating Eq. 9 using coordinate search (Appendix B) and the numerical methods in Chapter 2.1. Unfortunately, we found that the MLE estimates were unable to converge on the right model parameters, even with a significant amount of data (the expected count of all probes > 1). We tried the following to identify the source of the failures:

Vary probe lengths: We tested the probe length l was tested for a variety of lengths (4, 6, 8, 9, and 10); this did not enable convergence to the appropriate model parameters.

Vary feature order: We varied the feature order (or maximum k) with which the synthetic data was generated and inferred on from 1 to 4, and found that order 1 (nucleotide only) features resulted in consistent convergence, regardless of probe length.

Seeding fits: Surprisingly, coordinate search was unable to converge to the right parameters even after seeding. However, this did not occur when nucleotide only features were considered.

From these results, it appeared that feature orders greater than 1 introduced tight correlations between kmer features. For example, every dinucleotide feature has nucleotide feature components; AA will necessarily contain contributions from A features, GC will contain contributions from G and C features, etc. As another example, Table 11 highlights the disproportionate contribution of A and CG when considering feature order = 3 for the sequence ACGA. As in the case with *NRLB* (Chapter 2.3, we believe this correlation results in a large null space and creates a continuum of optima. From these conclusions, we restricted the feature set to only k -mers.

Table 11: Kmer Feature Counts for ACGA

Feature	Counts
A	2
C	1
G	1
AC	1
CG	1
GA	1
ACG	1
CGA	1

B Minimization Methods

Given a statistical model with parameters and data, Maximum Likelihood Estimation (MLE) is used to find parameter values of the statistical model that maximize the likelihood of observing the data. Ideally, we would like to find the global maximum of the likelihood, or the point β^* where

$$\mathcal{L}(\beta^*) \geq \mathcal{L}(\beta) \quad \forall \beta \in \text{domain}$$

Finding the global maximum is guaranteed in cases where the objective function (in this case, the likelihood) is convex (Figure 14A). However, more often than not, one encounters convex objective functions without analytical solutions for the maxima or non-convex objective functions with multiple minima (Figure 14B). In most situations, finding the global optimum is further complicated by our lack of a global perspective on the likelihood, as it is often computationally expensive to evaluate \mathcal{L} and its derivatives. Thus, our knowledge of the objective function is confined to local values for a small set of points.

Thankfully, it is possible to use this limited information to explore the functional landscape in an intelligent manner. Starting with a best guess of the optimal set of parameters (also known as a seed), optimization algorithms exploit this localized information to iteratively refine this guess and ‘step’ closer to an optimum. As most statistical models place few constraints on their parameters, unconstrained optimization methods are often used for MLE. A variety of these methods exist, each utilizing different analytical methods and functional information. Some methods rely solely on function values, while others rely on first and higher-order derivatives to understand the curvature of the function. Methods leveraging curvature information trade-off the (generally) higher computational cost of computing derivatives for fewer, more optimal iterations towards the maximum. However, none of these methods can guarantee finding a global optimum in a non-convex setting.

The fitness of an optimization algorithm can be measured in its ability to accurately and efficiently identify minima for a given objective function. Unfortunately, there is no optimization method that is ‘uniformly superior’ for both

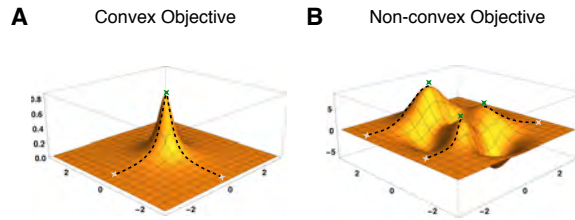


Figure 14: **(A)** For a convex objective function, the optimization algorithm will always converge to a unique and global maximum (green star), regardless of the starting point (or seed, grey star) and the path it takes (black dashed line). **(B)** The same does not hold true for a non-convex objective function, where the maximum found by an optimization algorithm is dependent on both the starting point and the path taken.

goals for all objective functions. An optimizer must be selected on both these metrics for the specific objective function at hand. We will briefly describe two optimization methods used in this work - a non-gradient approach called coordinate search and a gradient-based optimizer called Limited-Memory BFGS. Additionally, modifications made to the ‘standard’ algorithm and other analytical methods used to increase algorithm robustness will be discussed. Further information regarding optimization theory can be found in [20]. For the remainder of the discussion, we will follow convention and discuss the *minimization* of the negative of objective function $f(x)$, rather its maximization.

B.1 Optimization Algorithms

Coordinate Search

The coordinate search algorithm belongs to the direct search family of optimization methods. Coordinate search was one of the earliest (and simplest) iterative algorithms used for optimization; characterized as ‘slow but sure,’ it was used by Enrico Fermi and Nicholas Metropolis to fit pion-proton scattering data in the early 1950’s [21]. Its simplicity originates from the intuitive way it explores parameter space by evaluating the objective function. An outline of the algorithm is presented below, and a sample search trajectory is illustrated in Figure 15 for the Broyden tridiagonal function.

1. **Coordinate Search:** Start with step size δ . For each parameter i , see if stepping forward’ $x_i + \delta$ or backward’ $x_i - \delta$ improves the function value. If it does, keep the value of the new position. If it does not, move to the next parameter.
2. **Reduce Step Size:** If the searching process does not improve the function value for any parameter, then halve the step size and start again.

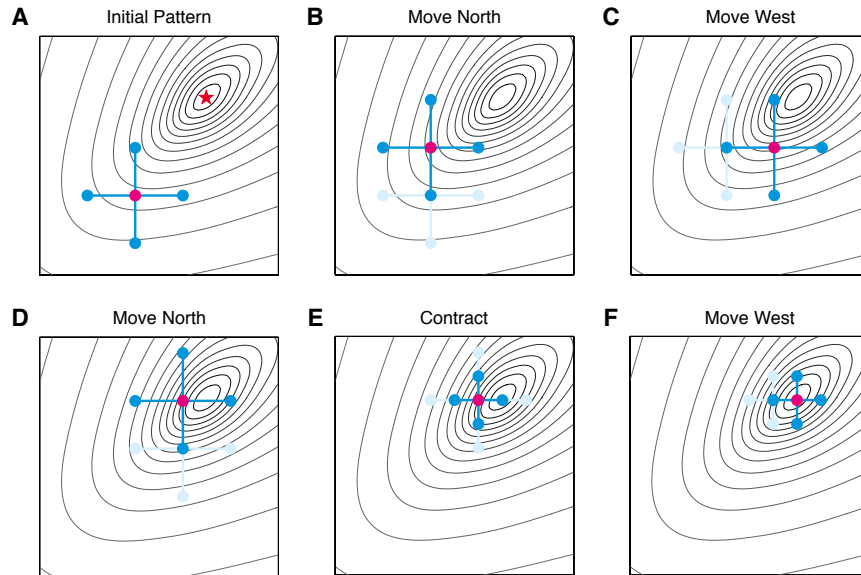


Figure 15: Five iterations of coordinate search when applied to the Broyden tridiagonal function starting from **A**. Figure adapted from [21].

3. **Declare Convergence:** The algorithm terminates when the step size goes below a preset threshold.

The implementation of coordinate search used in this work has modified this algorithm, allowing it to randomly select the order in which the parameters are varied and move in directions perpendicular to the null vectors of the objective function.

L-BFGS

Limited-Memory BFGS, or L-BFGS for short, is a quasi-newton method that uses gradients to find local minima for smooth functions. Broadly speaking, L-BFGS falls under the line search family of optimization methods. These methods transform a multi-dimensional unconstrained optimization problem into a series of sequential 1D optimization subproblems. Each 1D subproblem optimizes the objective function along a search direction p_k , selecting the next iterate x_{k+1} by finding a step size α_k that reduces the function value. Here, k refers to the current ‘step’ in the iterative process. While the ideal choice for α_k for each subproblem would be

$$\min_{\alpha > 0} f(x_k + \alpha p_k),$$

it is (in general) computationally expensive to find this value; instead, inexact line search methods are used.

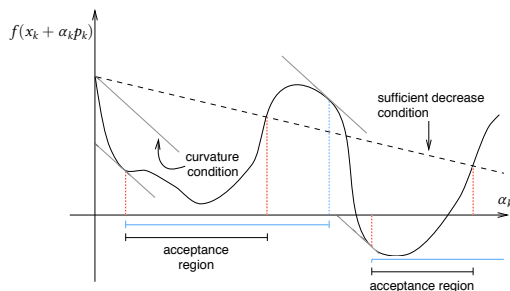


Figure 16: Step lengths α_k satisfying the Wolfe Conditions. Points below the black dashed line satisfy the sufficient decrease condition. Regions with slopes in between the grey lines satisfy the curvature condition and are highlighted by the blue bar. The acceptance regions satisfy both conditions. Figure adapted from [20].

Inexact line search methods find an α_k that satisfies the Wolfe conditions:

$$\begin{aligned} \text{Sufficient decrease} \quad & f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^\top p_k \\ \text{Minimum curvature} \quad & \nabla f(x_k + \alpha_k p_k)^\top p_k \geq c_2 \nabla f_k^\top p_k \end{aligned}$$

Intuitively, the first condition enforces a minimum decrease in the objective function proportional to the length of the step taken, while the second condition prevents the first condition from taking unacceptably small steps when the gradient is still highly negative (Figure 16). Two inexact line search implementations are used in this work - Algorithm 3.2 from [20] or the one described in [22].

For most line search methods, including the ones discussed above, p_k must be a descent direction:

$$p_k^\top \nabla f_k < 0$$

The two most commonly used descent directions are the steepest descent direction ($-\nabla f_k$), which relies on the gradient, and the Newton direction ($p_k = -\nabla^2 f_k^{-1} \nabla f_k$), which relies on the Hessian. While the Newton direction generally results in faster convergence, especially near the minimum, the high cost of computing the Hessian for large-scale multidimensional objective functions limits its use.

Quasi-Newton methods compute the descent direction $p_k = -B_k^{-1} \nabla f_k$ using an approximate Hessian B_k that mimics the true Hessian and uses gradients to compute cheap, low-rank updates. L-BFGS uses the rank-two update method given by

$$B_{k+1} = B_k + \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k} + \frac{y_k y_k^\top}{y_k^\top s_k}$$

where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$. The L-BFGS implementation used in this work uses Algorithms 9.1 and 9.2 from [20].

Convergence Performance

As mentioned earlier, the fitness of an optimization algorithm is measured by its ability to accurately and efficiently identify minima for a given objective function. To understand how the different coordinate search and L-BFGS minimizers perform on these metrics, we tested them on five functions commonly used to benchmark the performance of optimization algorithms [23]:

Beale's	$f(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$
Goldstein-Price	$f(x, y) = [1 + (x + y + 1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)] [30 + (2x - 3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)]$
Matyas	$f(x, y) = 0.26(x^2 + y^2) - 0.48xy$
Rosenbrock	$f(x, y) = 100(y - x^2)^2 + (x - 1)^2$
Sphere	$f(x, y) = x^2 + y^2$

We ran the algorithms on each function 10^4 times with a random seed within the x - y plane ($x, y \in [-1, 1]$). We recorded the number of steps taken and compared the computed minimum with the actual minimum to determine success. The results are presented in Table 12.

Convergence Tests

A point x^* is a strict local minimizer of $-f(x)$ if the following conditions are met [20]:

1. x^* is a stationary point: $-\nabla f(\beta^*) = 0$
2. The Hessian $-\nabla^2 f$ is continuous in an open neighborhood of x^*
3. The Hessian $-\nabla^2 f|_{x=x^*}$ is positive definite (eigenvalues $\lambda_i > 0 \ \forall i$)

For the functions discussed in this work, the Hessian is continuous (proof not given). Therefore, we can guarantee that x^* is a true minimum by ensuring that the eigenvalues of the Hessian are positive.

Unfortunately, L-BFGS routinely discovered *NRLB* models where the Hessian eigenvalues had mixed sign, a hallmark of saddle points. This is expected, as L-BFGS and other Newton and Quasi-Newton methods are known converge at saddle points [24]. To address this issue, we designed a heuristic scheme to test minima and escape from saddle points. The scheme is based on the observation that eigenvectors with negative eigenvalues point towards the nearest (and more optimal) stationary point (Figure 17):

1. **Compute Eigenvalues:** Diagonalize the Hessian in the symmetrized space, if applicable.

Table 12: Convergence Performance of Various Minimizers

	Coordinate Search	Coordinate Search + Random Axis	L-BFGS	L-BFGS + MT Search
Beale's				
Successes	98.5 %	85.5 %	90.4 %	90.3 %
Steps	194.8	196.6	13.0	12.7
Function Calls	848.1	855.5	14.9	14.8
Goldstein-Price				
Successes	52.3 %	60.7 %	33.3 %	34.9 %
Iterations	166.5	93.9	13.0	12.8
Function Calls	735.1	444.7	16.8	17.2
Matyas				
Successes	100.0 %	100.0 %	100.0 %	100.0 %
Iterations	28.6	34.1	3.9	3.9
Function Calls	183.5	205.2	5.5	5.5
Rosenbrock				
Successes	0.2 %	0.2 %	100.0 %	100.0 %
Iterations	19.6	27.0	25.2	25
Function Calls	147.4	176.8	30.5	30.6
Sphere				
Successes	100.0 %	100.0 %	100.0 %	100.0 %
Iterations	12.8	12.8	2.0	2.0
Function Calls	120.1	120.1	3.2	3.2

Performance metrics for coordinate search (the standard version and one where the parameter order was randomized) and L-BFGS (using either the [20] or [22] line search algorithms). For every test function, all algorithms were evaluated on 10^4 random start points. Displayed values are averages across all runs; successes refers to the number of functions where the true minimum was found.

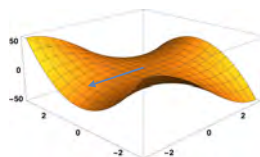


Figure 17: The monkey saddle function near the origin, with the negative eigenvector direction at $(0.1, 0.1)$ highlighted (blue arrow).

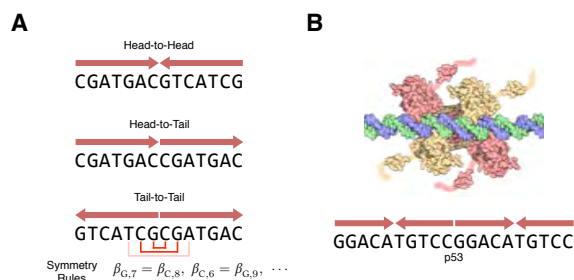


Figure 18: **(A)** Various configurations of the monomer subunits in homodimer complexes. Symmetry rules establish a relationship between the parameters at different offsets o within the binding interface to enforce a known subunit configuration. **(B)** The geometric configuration of the tumor suppressor p53 homotetrameric complex. Image from [28].

2. **Find Negative Eigenvalues:** Negative eigenvalues are those with a value less than -10^{-12} of the largest positive eigenvalue; this cutoff was selected from experience to ignore both precision errors and the null vectors in the space (Chapter 2.3).
3. **Escape Saddle Point:** If negative eigenvalues exist, the following process starts from the most negative eigenvalue:
 - (a) Perform a 1D line search in the eigenvector direction
 - (b) Accept first point that satisfies the convergence criteria used in L-BFGS
 - (c) Set this point as the new starting point and repeat with the next most negative eigenvalue
4. **Restart L-BFGS:** Restart minimization using the last point from the process above

B.2 Jacobian Based Symmetries

Certain classes of TFs form polymeric complexes composed of several repeating units in various spatial configurations. For example, homodimeric complexes can be arranged in head-to-head, tail-to-tail, or other unique combinations [25]; specific examples include basic-leucine zipper domains (bZIPs), which form homodimers with head-to-head symmetry [26], and the tumor suppressor protein p53, which forms tetramers with both head-to-head and tail-to-tail symmetry [Figure 18; [27]]. For this work, it would be useful to enforce known symmetries of TF complexes on *NRLB* models.

A symmetry configuration enforces an equality relationship between two or more parameters and can be represented by a set of symmetry rules (Figure

18). Given that TF complexes can bind in a variety of combinations, a general method for enforcing such rules while inferring models is required. The most flexible approach constrains the optimizer rather than the model, forcing the optimization method to move in the symmetrized space. While this approach may require more computational time, it can be easily implemented using elementary matrix math.

The symmetry rules discussed above can be implemented in the form of a coordinate transformation (or Jacobian) matrix \mathbf{M} , which defines a linear map from the full parameter space \mathbf{F} to the ‘reduced’ or symmetrized parameter space \mathbf{R} . For our symmetry rules, the Jacobian has a simple form: every row in \mathbf{M} corresponds to a particular symmetry rule. All entries in the row are 0, except those corresponding to the parameters mentioned in the rule, which are 1. As an example, consider a model with six parameters in the full space $\{x_1^{\mathbf{F}}, x_2^{\mathbf{F}}, x_3^{\mathbf{F}}, x_4^{\mathbf{F}}, x_5^{\mathbf{F}}, x_6^{\mathbf{F}}\}$. These parameters obey the following symmetry rules:

$$\begin{aligned}x_1^{\mathbf{R}} &= x_1^{\mathbf{F}} = x_6^{\mathbf{F}} \\x_2^{\mathbf{R}} &= x_2^{\mathbf{F}} = x_4^{\mathbf{F}} = x_5^{\mathbf{F}} \\x_3^{\mathbf{R}} &= x_3^{\mathbf{F}}\end{aligned}$$

where $\{x_1^{\mathbf{R}}, x_2^{\mathbf{R}}, x_3^{\mathbf{R}}\}$ represents the reduced space. Then we get

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

\mathbf{M} can be used to transform any quantity to the reduced space, *other* than the parameter vector (see Table 13). Only a single set of parameter values needs to be extracted when reducing the parameter vector; for this, we use a secondary matrix \mathbf{M}' . \mathbf{M}' is the same as \mathbf{M} , except for every row, all parameters other than the first nonzero parameter are set to zero. For the previous example, we get

$$\mathbf{M}' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

\mathbf{M} and \mathbf{M}' can be used by the minimizer to convert the parameter vector, gradient, and Hessian from the full space to the reduced space using the operations in Table 13.

Table 13: Symmetry Transformations

Quantity	Full Space	Reduced Space
Position	x^F	$\mathbf{M}'x^F$
Position	$\mathbf{M}'^T x^R$	x^R
Gradient	G^F	$\mathbf{M}G^F$
Gradient	$\mathbf{M}^T G^R$	G^R
Hessian	\mathbf{H}^F	$\mathbf{M}\mathbf{H}^F\mathbf{M}'^T$
Hessian	$\mathbf{M}'^T\mathbf{H}^R\mathbf{M}$	\mathbf{H}^R

References

- [1] Matthew Slattery, Todd Riley, et al. Cofactor binding evokes latent differences in DNA binding specificity between hox proteins. *Cell*, 147(6):1270–82, 2011.
- [2] Barrett C Foat, Alexandre V Morozov, et al. Statistical mechanical modeling of genome-wide transcription factor occupancy data by MatrixREDUCE. *Bioinformatics*, 22(14):e141–9, 2006.
- [3] Yue Zhao, David Granas, et al. Inferring binding energies from selected binding sites. *PLoS Comput. Biol.*, 5(12):1–8, 2009.
- [4] Remo Rohs, Xiangshu Jin, et al. Origins of specificity in protein-DNA recognition. *Annu. Rev. Biochem.*, 79:233–69, 2009.
- [5] Gary D Stormo, Thomas D Schneider, et al. Use of the Perceptron algorithm to distinguish translational initiation sites in e. coli. *Nucleic Acids Res.*, 10(9):2997–3011, 1982.
- [6] Timothy L Bailey, Charles Elkan, et al. Fitting a mixture model by expectation maximization to discover motifs in bipolymers. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 2:28–36, 1994.
- [7] GD Stormo. DNA binding sites: representation and discovery. *Bioinformatics*, 16(1):16–23, 2000.
- [8] GD Stormo, TD Schneider, et al. Quantitative analysis of the relationship between nucleotide sequence and functional activity. *Nucleic Acids Res.*, 14(16):6661–6679, 1986.
- [9] TR Riley, A Lazarovici, et al. Building accurate sequence-to-affinity models from high-throughput in vitro protein-dna binding data using featurereduce. *elife*, 4:e06397, 2015.

- [10] Yue Zhao, Shuxiang Ruan, et al. Improved models for transcription factor binding site identification using nonindependent interactions. *Genetics*, 191(3):781–790, 2012.
- [11] Raluca Gordân, Ning Shen, et al. Genomic regions flanking e-box binding sites influence dna binding specificity of bhlh transcription factors through dna shape. *Cell Reports*, 3(4):1093–1104, 2013.
- [12] Fantine Mordelet, John Horton, et al. Stability selection for regression-based models of transcription factor–dna binding specificity. *Bioinformatics*, 29(13):i117–i125, 2013.
- [13] Remo Rohs, Sean M West, et al. The role of dna shape in protein–dna recognition. *Nature*, 461(7268):1248–1253, 2009.
- [14] Rohit Joshi, Jonathan M Passner, et al. Functional specificity of a hox protein mediated by the recognition of minor groove structure. *Cell*, 131(3):530–543, 2007.
- [15] L Yang, Y Orenstein, et al. Transcription factor family-specific dna shape readout revealed by quantitative specificity models. *Mol. Syst. Biol.*, 13(2):910, 2017.
- [16] Tianyin Zhou, Lin Yang, et al. Dnashape: a method for the high-throughput prediction of dna structural features on a genomic scale. *Nucleic Acids Res.*, 41:W56–62, 2013.
- [17] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.
- [18] Andrew Y Ng. Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.
- [19] Bradley Efron and David V. Hinkley. Assessing the accuracy of the maximum likelihood estimator: Observed versus expected fisher information. *Biometrika*, 65(3):457–482, 1978.
- [20] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer, 2006.
- [21] Tamara G Kolda, Robert Michael Lewis, et al. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45(3):385–482, 2003.
- [22] Jorge J Moré and David J Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM T. Math. Software*, 20(3):286–307, 1994.
- [23] Momin Jamil and Xin-She Yang. A literature survey of benchmark functions for global optimisation problems. *Int. J. Math. Mod. Num. Opt.*, 4(2):150–194, 2013.

- [24] Yann N. Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NIPS'14, pages 2933–2941, Cambridge, MA, USA, 2014. MIT Press.
- [25] Arttu Jolma, Jian Yan, et al. Dna-binding specificities of human transcription factors. *Cell*, 152(1):327–339, 2013.
- [26] Charles Vinson, Max Myakishev, et al. Classification of human b-zip proteins based on dimerization properties. *Mol. Cell. Biol.*, 22(18):6321–6335, 2002.
- [27] Kevin G McLure and Patrick WK Lee. How p53 binds dna as a tetramer. *EMBO J.*, 17(12):3342–3350, 1998.
- [28] David Goodsell. p53 tumor suppressor, Jul 2002. <http://pdb101.rcsb.org/motm/31> [Accessed: May 1, 2017].