FLIGHTED: Inferring Fitness Landscapes from Noisy High-Throughput Experimental Data

Vikram Sundar
Computational and Systems Biology Program
Massachusetts Institute of Technology
Cambridge, MA 02139
vsundar@mit.edu

Boqiang Tu
Department of Biological Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139

Lindsey Guan
Computational and Systems Biology Program
Massachusetts Institute of Technology
Cambridge, MA 02139

Kevin Esvelt
Media Lab
Massachusetts Institute of Technology
Cambridge, MA 02139
esvelt@media.mit.edu

Abstract

Machine learning (ML) for protein design frequently requires large datasets of protein fitness generated by high-throughput experiments, and many ML models use these datasets for training, fine-tuning, and benchmarking. However, these approaches do not account for underlying experimental noise, potentially making their conclusions inaccurate. In this work, we present FLIGHTED (Fitness Landscape Inference Generated by High-Throughput Experimental Data), a Bayesian method for generating fitness landscapes with calibrated errors from noisy high-throughput experimental data. We apply FLIGHTED to datasets generated by single-step enrichment-based selection assays such as Fluorescence-Activated Cell Sorting (FACS) and phage display and to data from a novel high-throughput assay DHARMA (direct high-throughput activity recording and measurement assay) that ties fitness to base editing activity. Our results suggest that de-noising single-step selection data generates well-calibrated predictions that are sufficient to change which models perform best in benchmarking studies. Applying FLIGHTED to DHARMA provides more accurate fitness measurements with better calibrated errors; FLIGHTED-DHARMA can be used to generate large protein fitness datasets with up to 10^6 variants. FLIGHTED can be used on any high-throughput assay and makes it easy for ML scientists to account for experimental noise when modeling protein fitness.

1 Introduction

Machine learning (ML) approaches have been remarkably successful in a variety of protein design problems [9, 8]. Many of these approaches rely on data generated by high-throughput experiments; this data is used as training or fine-tuning data for a large ML model which optimizes the desired protein function [3, 2, 26, 9]. Fitness measurements from high-throughput experiments are also more broadly used for benchmarking protein language models and *in silico* retrospective model performance evaluations [5, 20, 19, 14, 15].

However, high-throughput experiments are inherently noisy; for example, experiments like deep mutation scanning (DMS) or phage display that rely on a single selection step have substantial noise in the measured enrichment ratio [7, 21, 4]. Despite warnings in the literature, most ML protein

Machine Learning for Structural Biology Workshop, NeurIPS 2023.

modeling efforts completely ignore any experimental noise present in large fitness datasets. Using noisy datasets for training, benchmarking, and evaluating models can lead to inaccurate conclusions.

In this paper, we present FLIGHTED (Fitness Landscape Inference Generated by High-Throughput Experimental Data), a new approach to generating reliable fitness landscapes with calibrated errors from noisy high-throughput experiments. Previous work in this area has been limited to single-step selection experiments [4, 21] or limited in the complexity of ML model [7]. FLIGHTED is applicable to any high-throughput experiment and an arbitrarily complex ML model.

Key Contributions We present three key contributions:

1. FLIGHTED, a Bayesian method to generate fitness landscapes with calibrated errors from noisy high-throughput experimental data, which can be generalized to any high-throughput experiment.
2. FLIGHTED-Selection, a model for single-step selection experiments (including phage display and deep mutational scans) that produces robust, calibrated errors and can be used to correct datasets for benchmarking.
3. A new high-throughput experimental assay, DHARMA, and denoising model FLIGHTED-DHARMA that can be used to generate fitness datasets with millions of data points.

2 Methods

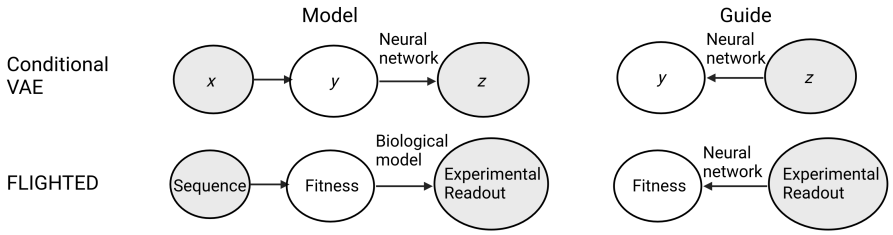


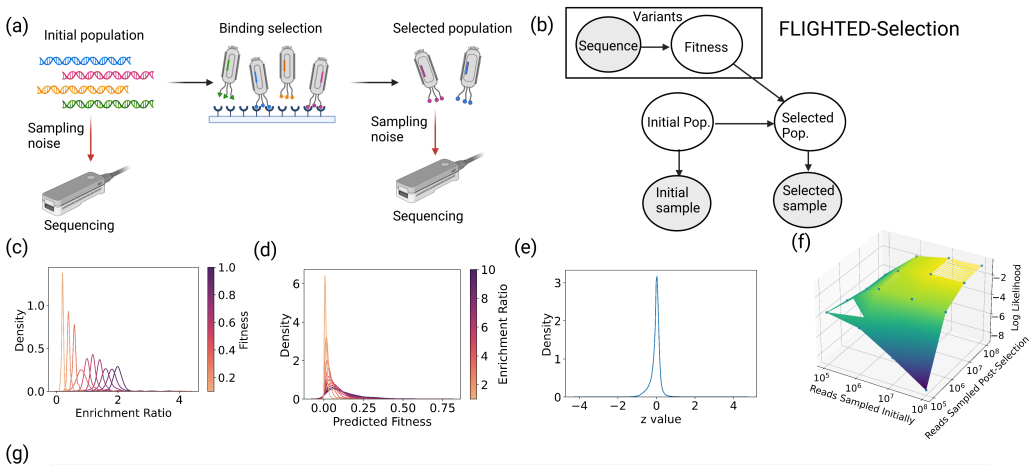
Figure 1: A Comparison between FLIGHTED and a Conditional Variational Auto-Encoder. This provided the motivation for selection of the model and guide for FLIGHTED.

FLIGHTED was motivated by analogy to a conditional variational auto-encoder (CVAE), as shown in Figure 1. In general, a high-throughput experiment may be modeled as shown in the bottom left of Figure 1. This probabilistic graphical model is identical to that of a CVAE [23], where an output z is predicted from an input x through a latent variable y. We can use stochastic variational inference for learning like the CVAE. Stochastic variational inference has two main advantages. It allows for model parameters to be fit once to a calibration dataset and then used to estimate a posterior given any subsequent experimental measurements. Further, the models built can have arbitrary complexity (including neural networks), since the posterior does not need to be solved for exactly.

Instead of an arbitrary neural network, we use biological knowledge to develop a specific probabilistic graphical model of how to generate an experimental readout from fitness. Our guide, or variational distribution, must predict fitness from the experimental readout; we use a neural network since we do not have specific biological knowledge here.

Given these building blocks, we proceed as follows:

1. Use biological knowledge to build a probabilistic graphical model for a given high-throughput experiment.
2. Pick a neural network architecture for the guide to predict fitness from the experimental readout. Simpler architectures tend to perform the best.
3. Generate a calibration dataset via simulation or experiment that includes both experimental readouts and ground-truth fitness values.
4. Train the model on solely the experimental readouts, and evaluate model accuracy and calibration using the calibration dataset.



(g)

Task / Model	One vs Rest	One vs Rest FLIGHTED	Two vs Rest	Two vs Rest FLIGHTED	Three vs Rest	Three vs Rest FLIGHTED	Low vs High	Low vs High FLIGHTED
Linear	1.464443	11.892308	1.513193	10.037789	1.516534	7.975007	4.620592	7.286427
CNN	2.234338	14.902342	1.401910	3.689020	0.516611	1.501666	3.114972	0.632784
TAPE	1.497365	11.440956	1.972100	6.831995	0.964711	3.706415	3.598206	2.592706
ESM-1b	1.490587	11.293457	1.398277	5.648212	0.951439	3.205659	3.584803	2.484083
ESM-1v	1.522624	14.740891	1.567272	6.825393	0.903832	3.628704	3.298765	1.922280
ESM-2	1.877468	12.154076	1.335592	5.038312	0.883998	2.845415	3.459866	2.317814
TAPE (CNN)	1.902995	15.777972	1.213981	4.343023	0.461875	1.347543	3.127995	0.786387
ESM-1b (CNN)	1.957240	15.977012	1.172148	5.244392	0.430597	1.378595	3.283653	1.078013
ESM-1v (CNN)	1.889501	12.838265	1.205914	6.933992	0.452065	1.130676	3.270220	0.737013
ESM-2 (CNN)	1.842407	15.139809	1.356091	6.974940	0.478068	1.157305	3.514309	1.167338

Figure 2: FLIGHTED-Selection Model Performance. (a) Single-step selection assays always include sampling noise when reads are sampled for sequencing. (b) The probabilistic graphical model used for FLIGHTED-Selection. (c) Simulations indicate that the enrichment ratio has considerable noise in a single-step selection experiment due to sampling. (d) Typical FLIGHTED-Selection predictions for a given enrichment ratio. (e) FLIGHTED-Selection model predictions are well-calibrated. (f) FLIGHTED-Selection model performance is robust to changes in number of reads selected pre- and post-selection if there are more reads than variants (20^4 here). (g) Mean squared error of downstream protein models on the GB1 dataset [25] following standard benchmark splits [5] with and without FLIGHTED-Selection. The top-performing model changes, suggesting that FLIGHTED-Selection affects the conclusions of these benchmarking studies.

The resulting model is trained in a fully unsupervised fashion without any ground-truth fitness data; the calibration dataset is only used for hyperparameter tuning and to ensure model accuracy. Learning from only experimental readouts is possible because the biological knowledge used to build the model constrains the space of possible models.

3 Results

3.1 Single-Step Selection

Single-step enrichment-based selection assays are experimental assays in which a library of variants is produced, a selection step occurs to select fitter variants, and the remaining selected population is measured; see Figure 2a [11, 22]. These include mRNA display, phage display, and many DMS studies, and comprise a substantial portion of fitness landscapes used for ML [25, 5]. Fitness is usually measured as the enrichment ratio, or the ratio of variant sequences sampled post- to pre-selection [11]. However, the enrichment ratio is an inherently noisy measurement due to sampling noise, the noise associated with sampling the reads to be sequenced from a larger library [11]. Sampling noise is unavoidable in a single-step selection assay and the source of noise we choose to model [11].

To understand the impacts of sampling noise and generate calibration datasets for downstream training, we ran simulations of a single-step selection assay on a randomly generated fitness landscape with

20^4 variants. Figure 2c plots the probability distributions of the enrichment ratio for a given fitness, showing there is considerable noise. Therefore, enrichment ratio is not a reliable measure of fitness.

The probabilistic graphical model for FLIGHTED-Selection is shown in Figure 2b; exact mathematical statements can be found in the appendix. The guide model uses the logit of the enrichment ratio as the mean and predicts the variance using linear regression. FLIGHTED-Selection is trained on a calibration dataset of simulations on a randomly generated landscape. We use simulations as the training data since we need to know the ground-truth fitness to accurately evaluate FLIGHTED-Selection's performance; there is no easy way to measure ground-truth fitnesses for given real-world single-step selection experiments. Typical model predictions are shown in Figure 2d for given enrichment ratios; for more active variants, single-step selection experiments produce less reliable fitness measurements.

To evaluate model performance, we focused on calibration of the variance, since the mean was not predicted. We generate a separate random fitness landscape to eliminate data leakage, simulate single-step selection assays, and compute the z-value of the true fitness compared to the model prediction. The resulting distribution is shown in Figure 2e and looks very similar to a normal distribution, suggesting that FLIGHTED-Selection is well-calibrated. The mean log likelihood of our predictions is -0.97, very close to that expected for a normal distribution.

Next, we evaluated the robustness of FLIGHTED-Selection to various experimental conditions. The model is robust to most parameters (see Figure 2f and Supplementary Figures S3 and S4). The parameters that matter the most are the number of reads drawn pre- and post-selection. If the number of reads sampled is fewer than the number of variants (20^4), the model is no longer well-calibrated.

Our results suggest that ML methods currently benchmarked on single-step selection datasets need to be re-benchmarked. Unlike previous work [4, 7], we do not believe a comparison between models trained on the original noisy data and the corrected data is appropriate; this is an apples-to-oranges comparison since the test set has changed. As an example, we looked at the popular GB1 dataset as processed by the FLIP benchmark [25, 5], using FLIGHTED-Selection to add error bars to each datapoint in the dataset. A comparison of the 4 GB1 benchmark tasks with and without FLIGHTED is shown in Figure 2g, for models proposed by FLIP. On 3 out of the 4 tasks, the top-performing model changes between the original and corrected datasets. Our results demonstrate that protein fitness datasets from single-step selection experiments need to be corrected with FLIGHTED-Selection to account for sampling noise.

3.2 DHARMA

To mitigate some of the flaws in single-step selection assays, we turn to DHARMA (direct high-throughput activity recording and measurement assay), a recently developed high-throughput protein fitness assay [24]. DHARMA measures fitness by linking protein fitness to transcription of a base editor; this base editor is targeted to a contiguous DNA segment ("canvas") where it randomly causes C$\rightarrow$T edits, as shown in Figure 3a. Higher fitness leads to higher base editor transcription and more C$\rightarrow$T edits [24].

DHARMA can be run in high-throughput on up to 10^6 variants at once, since the readout is measured by cheap long-read sequencing. It can measure any protein function that can be linked to transcription, which includes protease activity, gene editing, and protein and DNA binding [13]. However, transcription is an inherently stochastic process [17] so DHARMA output is noisy; Figure 3e demonstrates that repeated DHARMA experiments on the same variant can result in differing C$\rightarrow$T edit counts. Therefore denoising via an ML model is essential for reliable fitness measurements.

The probabilistic graphical model for FLIGHTED-DHARMA is shown in Figure 3b; exact mathematical statements can be found in the appendix. The guide predicts both the mean and variance of fitness given a single DHARMA read. We generated a calibration dataset by running a DHARMA experiment on a 3-site library of T7 polymerase. Ground-truth fitness measurements were produced by Fluorescence-Activated Cell Sorting (FACS) on a subset of 119 variants, in which T7 polymerase drives the expression of GFP instead of base editor. Error was minimized in these FACS measurements by using a large number of cells per variant. To eliminate data leakage, FLIGHTED-DHARMA was not trained on any DHARMA read from any variant for which FACS was measured. Concretely, in Figure 3f we show the predicted number of C$\rightarrow$T edits with error given a particular fitness.

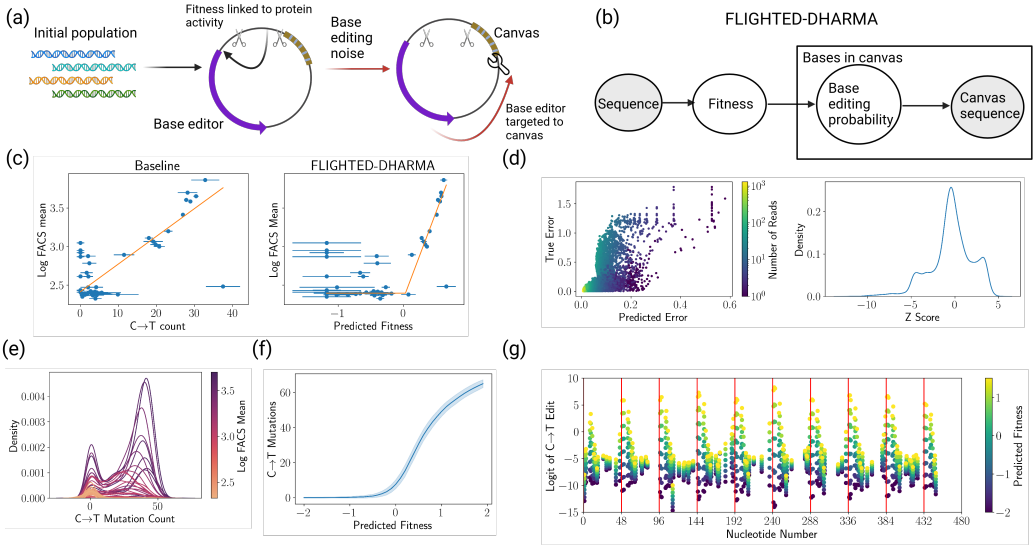


Figure 3: FLIGHTED-DHARMA Model Performance. (a) DHARMA links fitness to edit rates of a DNA segment (canvas) via transcriptional control of a base editor, an inherently noisy process. (b) The probabilistic graphical model for FLIGHTED-DHARMA. (c) FLIGHTED-DHARMA improves performance of fitness predictions when compared to a baseline model that predicts number of C→T mutations. (d) FLIGHTED-DHARMA is reasonably well-calibrated on even small subsets of DHARMA reads. (e) Noise in DHARMA as a function of true fitness. (f) FLIGHTED-DHARMA's prediction of number of C→T mutations as a function of fitness. (g) Canvas nucleotides most likely to be edited according to FLIGHTED-DHARMA are found at guide RNA binding sites, as expected.

Fitnesses predicted by FLIGHTED-DHARMA can be related to the FACS readout by an arbitrary nondecreasing function. We used a validation set to fit this to a piecewise linear function. The initial flat section of the piecewise linear functions corresponds to low-activity variants where background fluorescence dominates the FACS measurement. We then evaluated model performance on the remaining test set, as shown in Figure 3c. FLIGHTED-DHARMA was compared to a baseline model that predicted the mean and variance of C→T mutation count. We found that FLIGHTED-DHARMA's MSE was 0.72, an improvement over the baseline MSE of 0.78.

We also evaluated the calibration of our predicted variances. In Figure 3d, we selected random subsets of DHARMA reads and measured the true and predicted error of FLIGHTED-DHARMA. We then computed z-scores to evaluate model calibration. The mean log likelihood was -3.93, suggesting that FLIGHTED-DHARMA is slightly overconfident but decently well-calibrated. Baseline model performance was substantially worse with a mean log likelihood of -8.00 (see Supplementary Figure S7). Since our calibration tests included small subsets of DHARMA reads, we can be confident that FLIGHTED-DHARMA will produce reasonable errors even when given few DHARMA reads. This aids experimentalists by informing them whether a given number of DHARMA reads is sufficient for a fitness measurement. Unlike single-step selection, predicted errors in a DHARMA fitness measurements decrease as fitness increase, suggesting that DHARMA may sometimes be more appropriate for measuring the fitness of highly active variants.

Finally, since FLIGHTED uses a biological model of DHARMA, we can examine the parameters of that biological model directly to see whether they make sense. In Figure 3g, we compute the logit of the C→T edit probability at a given canvas residue as a function of fitness. The probability of a C→T edit increases and is more fitness-dependent every 48 residues. The base editor is targeted at these locations, so we expect the probability of a C→T edit to increase and be more dependent on base editor activity, as we see. This is an example of how FLIGHTED leads to an interpretable model.

4 Discussion

FLIGHTED has consistently generated fitness landscapes with calibrated errors for two very different high-throughput experiments. Since it is Bayesian, it can be easily extended further. We are currently working on an extension to multi-step selection experiments like Phage-Assisted Continuous Evolution (PACE) [6, 13] and extensions to other high-throughput experiments should be straightforward. FLIGHTED also makes combining data from different high-throughput experiments possible.

Our results on FLIGHTED-Selection suggest that common ML benchmarks like FLIP [5] or ProteinGym [15] need to be corrected to account for noise inherent in single-step selection. Correcting these benchmarks may change downstream evaluations of protein models. Further, single-step selection experiments may not be well-suited for distinguishing between highly active variants.

FLIGHTED-DHARMA enables the use of DHARMA as a data generation method for machine learning. The noisy nature of biological circuits presents challenges in effectively using DHARMA to quantify activities. We provide calibrated error estimates that inform practitioners when they have enough DHARMA reads to make a reliable fitness estimate. DHARMA can be used to cheaply generate large datasets of up to 10^6 variants, and in some contexts may be more accurate that single-step selection experiments on distinguishing between highly active variants. This is an example of how FLIGHTED can select which experiment is used to generate the most accurate fitness landscape data for downstream ML training.

4.1 Data and Code Availability

All code used is available at this Github repository, including code to set up and use FLIGHTED-selection and FLIGHTED-DHARMA on new datasets. All data used to generate figures in this paper is available at this Zenodo repository. Any further information (like hyperparameter scans for generating the FLIGHTED models) can be obtained by contacting the authors.

Acknowledgments and Disclosure of Funding

V.S. acknowledges funding from the Fannie and John Hertz Foundation. B.T. acknowledges funding from the National Science Foundation Graduate Research Fellowship under Grant No. 2141064. L.G. acknowledges funding from the National Institutes of Health under Grant. No. T32GM087237. We are grateful to gifts from Open Philanthropy Project and the Aphorism Foundation (to K.M.E.).

References

[1] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman. Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*, 20:1–6, 2019.

[2] S. Biswas, G. Khimulya, E. C. Alley, K. M. Esvelt, and G. M. Church. Low-N protein engineering with data-efficient deep learning. *Nature Methods 2021 18:4*, 18(4):389–396, Apr. 2021. Publisher: Nature Publishing Group ISBN: 4159202101100.

[3] D. H. Bryant, A. Bashir, S. Sinai, N. K. Jain, P. J. Ogden, P. F. Riley, G. M. Church, L. J. Colwell, and E. D. Kelsic. Deep diversification of an AAV capsid protein by machine learning. *Nature Biotechnology*, 39(6):691–696, June 2021.

[4] A. Busia and J. Listgarten. Model-based differential sequencing analysis, Apr. 2023.

[5] C. Dallago, J. Mou, K. E. Johnston, B. J. Wittmann, N. Bhattacharya, S. Goldman, A. Madani, and K. K. Yang. FLIP: Benchmark tasks in fitness landscape inference for proteins, Jan. 2022. Section: New Results Type: article.

[6] K. M. Esvelt, J. C. Carlson, and D. R. Liu. A system for the continuous directed evolution of biomolecules. *Nature*, 472(7344):499–503, 2011.

[7] J. Fernandez-de Cossio-Diaz, G. Uguzzoni, and A. Pagnani. Unsupervised Inference of Protein Fitness Landscape from Deep Mutational Scan. *Molecular Biology and Evolution*, 38(1):318–328, Jan. 2021. Publisher: Oxford University Press.

[8] V. Frappier and A. E. Keating. Data-driven computational protein design. *Current Opinion in Structural Biology*, 69:63–69, Aug. 2021. Publisher: Elsevier Ltd.

[9] K. E. Johnston, C. Fannjiang, B. J. Wittmann, B. L. Hie, K. K. Yang, and Z. Wu. Machine Learning for Protein Engineering, May 2023. arXiv:2305.16634 [q-bio].

[10] E. Jones, T. Oliphant, P. Peterson, and Others. SciPy: Open source scientific tools for Python, 2001.

[11] W. Mandecki, J. Y.-C. Chen, and N. Grihalde. A Mathematical Model for Biopanning (Affinity Selection) Using Peptide Libraries on Filamentous Phage. *Journal of Theoretical Biology*, 176(4):523–530, Oct. 1995.

[12] J. Meier, R. Rao, R. Verkuil, J. Liu, T. Sercu, and A. Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Proceedings of the 35th Conference on Neural Information Processing Systems*, 2021.

[13] S. M. Miller, T. Wang, and D. R. Liu. Phage-assisted continuous and non-continuous evolution. *Nature Protocols 2020 15:12*, 15(12):4101–4127, Nov. 2020. Publisher: Nature Publishing Group.

[14] E. Nijkamp, J. Ruffolo, E. N. Weinstein, N. Naik, and A. Madani. ProGen2: Exploring the Boundaries of Protein Language Models, June 2022. arXiv:2206.13517 [cs, q-bio].

[15] P. Notin, M. Dias, J. Frazer, J. Marchena-Hurtado, A. Gomez, D. S. Marks, and Y. Gal. Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. arXiv, May 2022. Number: arXiv:2205.13760 arXiv:2205.13760 [cs].

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. 2019.

[17] A. Raj and A. v. Oudenaarden. Nature, Nurture, or Chance: Stochastic Gene Expression and Its Consequences. *Cell*, 135(2):216–226, Oct. 2008. Publisher: Elsevier.

[18] R. Rao, N. Bhattacharya, N. Thomas, Y. Duan, X. Chen, J. Canny, P. Abbeel, and Y. S. Song. Evaluating protein transfer learning with TAPE. In *Advances in Neural Information Processing Systems*, 2019. arXiv: 1906.08230 ISSN: 1049-5258.

[19] R. Rao, J. Liu, R. Verkuil, J. Meier, J. F. Canny, P. Abbeel, T. Sercu, and A. Rives. MSA Transformer, Aug. 2021. Section: New Results Type: article.

[20] A. Rives, S. Goyal, J. Meier, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):622803, Apr. 2021. Publisher: bioRxiv.

[21] A. F. Rubin, H. Gelman, N. Lucas, S. M. Bajjalieh, A. T. Papenfuss, T. P. Speed, and D. M. Fowler. A statistical framework for analyzing deep mutational scanning data. *Genome Biology*, 18(1):150, Aug. 2017.

[22] M. Russel, H. B. Lowman, and T. Clackson. Introduction to phage biology and phage display. In *Phage Display: A Practical Approach*, volume 266 of *Practical Approach Series*, page 26. Oxford University Press, Oxford, 2004.

[23] K. Sohn, H. Lee, and X. Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[24] B. Tu and K. M. Esvelt. High-throughput molecular recording can determine the identity and biological activity of sequences within single cells, Apr. 2022. Section: New Results Type: article.

[25] N. C. Wu, L. Dai, C. A. Olson, J. O. Lloyd-Smith, and R. Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 5(JULY), July 2016. Publisher: eLife Sciences Publications Ltd.

[26] Z. Wu, S. B. Jennifer Kan, R. D. Lewis, B. J. Wittmann, and F. H. Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences of the United States of America*, 116(18):8852–8858, Apr. 2019. arXiv: 1902.07231 Publisher: National Academy of Sciences.

A Detailed Methods

All models were implemented using the probabilistic programming package Pyro [1] alongside PyTorch [16]. Use of Pyro makes model development substantially easier due to rapid iteration of model architectures.

A.1 Single-Step Selection

Simulation and Dataset Generation Single-step selection experiments were simulated on a landscape of $N_{\text{var}} = 20^4$ variants, with the ith variant having a fitness or selection probability p_i uniformly distributed between 0 and 1, i.e. $p_i \sim \mathcal{U}([0, 1])$. The total initial population $N_{\text{init}} = 10^{11}$. Each variant i had an initial population drawn from the Dirichlet distribution $N_{\text{init},i} \sim N_{\text{init}}\text{Dir}(\alpha), \alpha = (1, \ldots, 1)$. An initial pre-selection sample $N_{\text{sampled, init},i}$ was drawn according to a multinomial distribution $\text{Mult}(N_{\text{init},i})$, with $N_{\text{sampled, init}} = 10^8$ samples being drawn.

In the selection step, the ith variant had a post-selection population drawn from the binomial distribution, i.e. $N_{\text{final},i} \sim B(N_{\text{init},i}, p_i)$. The post-selection sample $N_{\text{sampled, final},i}$ was similarly drawn according to a multinomial distribution $\text{Mult}(N_{\text{final},i})$ with $N_{\text{sampled, final}} = 10^8$ samples being drawn. We ran 100 simulations on a single landscape to generate training data and to assess the noise levels in the enrichment ratio as shown in Figure 2c.

The parameters of this simulation were selected to be similar to those used in generating the GB1 dataset [25]. Simulations with smaller numbers of reads drawn both pre- and post-selection were used to assess robustness. Simulations were also run with varying Dirichlet parameters and using a beta distribution for the selection probability instead of a uniform distribution; these did not significantly affect the noise found in the enrichment ratio.

The FLIGHTED-Selection Model The model is an implementation of the probabilistic graphical model shown in Figure 2b in Pyro. The sequence-to-fitness function is simply a dictionary lookup, with the fitness of each variant i defined as a parameter to be optimized. Fitness means m_i and variances σ_i are predicted by the sequence-to-fitness function/dictionary, and the variance is transformed by a softplus function F to ensure it is positive. The fitness of each variant is then sampled from a normal distribution $\mathcal{N}(m_i, F(\sigma_i))$ and transformed by a sigmoid function to be between 0 and 1. To avoid issues near the boundaries, fitnesses are clamped to be between 0.001 and 0.999. The initial population of each variant $N_{\text{init},i}$ is a per-data-point parameter in Pyro. We then sample from the multinomial and binomial distributions as described in the simulation section to implement the desired probabilistic graphical model.

The guide predicts the fitness mean for a given experiment as logit $\left(\frac{\text{ENR}}{100}\right)$, where ENR is the enrichment ratio. The scale factor of 100 ensures that the transformed enrichment ratio is between 0 and 1 before being fed into the logit function; it can be changed arbitrarily with no impact on model performance after retraining. The variance is predicted by a linear model with 2 inputs: $\frac{N_{\text{sampled, init},i}}{N_{\text{sampled, init}}}$ and $\frac{N_{\text{sampled, final},i}}{N_{\text{sampled, final}}}$. The predicted variance is also transformed by a softplus function to ensure that it is positive.

Each data point for the FLIGHTED-Selection model consists of an experiment or simulation, not the outcome of a single variant within an experiment, because the number of reads of a given variant sampled pre- and post-selection depends on the population of other variants in the sample. To combine predictions from the guide model for different experiments in a single batch, we simply multiply the relevant Gaussians and sample the fitness from the resulting combined Gaussian distribution.

We used a learning rate of 10^{-2}, a learning rate on the landscape model (the sequence-to-fitness function) of 10^{-1}, a batch size of 10, 150 epochs, 10 particles in the ELBO loss function in Pyro, and a plateau learning rate scheduler with a patience of 4 epochs. Other hyperparameters can be found in the released code. The model was trained on 80 of the 100 simulations, validated on another 10 (randomly split), and tested on the final 10. To minimize data leakage, final test results were computed on a randomly generated set of 100 simulations on a completely different random fitness landscape.

GB1 Landscape Benchmarking To compute the corrected GB1 landscape with FLIGHTED-Selection, we took the guide and ran inference on the released GB1 landscape data, which provided both pre- and post-selection read counts in addition to the enrichment ratio [25]. We omitted all data points that were omitted in the original study due to not being observed. We then followed the published data splits provided by FLIP [5] to generate datasets both with and without FLIGHTED-Selection for the GB1 problem. Models trained on datasets with corrections from FLIGHTED-Selection are trained with a weighted mean-squared-error (MSE) loss, weighted by the inverse variance. Weighted MSE is used to account for the variance, assuming that the likelihood of the data is the same as that of a normal distribution with the provided variance. Performance results reported in Figure 2g are regular MSEs for the datasets without corrections and weighted MSEs for datasets with corrections.

The models tested largely follow the ones proposed in FLIP with a few modifications, but we specify all the details here for clarity [5]. The linear regression model (labeled Linear) takes one-hot embeddings of the full sequence and runs them through 1 linear layer. It uses an Adam optimizer with a learning rate of 10^{-2}, a batch size of 256, and a weight decay of 1. The CNN model (labeled CNN) takes one-hot embeddings of the full sequence and has 1 convolutional layer with 1024 channels and filter size 5, and same padding. It then has a 1D batch normalization layer and a ReLU activation, followed by an embedding neural network consisting of a linear layer to 2048 dimensions and a ReLU activation. Then there is a max-pooling layer over residues, a dropout layer with probability 0.2, and a final linear layer for the output. The CNN is trained with a batch size of 256 and an Adam optimizer with a learning rate of 10^{-3} and weight decay of 0 for the convolutional layer, a learning rate of $5 * 10^{-5}$ and weight decay of 0.05 for the embedding layer, and a learning rate of $5 * 10^{-6}$ and weight decay of 0.05 for the output layer. Unlike the original FLIP paper, we did not use early stopping and trained all models for a full 500 epochs, using validation set performance to select the optimal model.

The models labeled TAPE, ESM-1b, ESM-1v, and ESM-2 all use mean embeddings across the entire sequence that are fed into a feedforward neural network to compute the output [20, 12, 18]. The output feedforward neural network has 2 layers with a hidden dimension equal to the embedding dimension of the protein language model and ReLU activation. All models are trained with an Adam optimizer with batch size 256 and learning rate 10^{-3}. The TAPE model used was the transformer, the ESM-1v model used was version 1, and the ESM-2 model used was the 3 billion parameter version (due to memory issues with the larger model).

The models labeled TAPE (CNN), ESM-1b (CNN), ESM-1v (CNN), and ESM-2 (CNN) use residue-level embeddings that are fed into a CNN, similar to the baseline CNN described above. The intermediate dimension output by the embedding neural network is set to be twice the dimension of the output of the protein language model. All other parameters remained the same.

A.2 DHARMA

T7 Polymerase Dataset Generation We engineered a biological circuit to associate the enzymatic characteristics of T7 RNA polymerase variants with mutations accumulating on a designated DNA sequence, known as the canvas. T7 RNA polymerase is an enzyme responsible for transcribing DNA into RNA, and we constructed a library of variants with different recognition and activity profiles.

The transcription of the base editor, an enzyme that can induce mutations, is controlled by a T3 promoter. This promoter is selectively recognized by a subset of the T7 RNA polymerase variants. To moderate the expression levels of the T7 polymerase library and prevent rapid saturation of the canvas—which would compromise the collection of meaningful activity data—we used both a weak constitutive promoter and a weak ribosomal binding site.

We incorporated this biological circuit into cells already containing plasmids that express the single guide RNA (sgRNA), through a technique called electroporation. The sgRNA serves to guide the base editor to the specific canvas sequence where mutations are to be introduced. After electroporation, the cells were grown continuously in a bioreactor. Following this growth period, the region of the T7 polymerase library and the canvas with mutations was selectively amplified as contiguous fragments of DNA. The amplified material was then subjected to long-read Nanopore sequencing for detailed analysis of the mutations and activities of different T7 RNA polymerase variants.

To facilitate data analysis, we implemented a data processing pipeline. Its core function is to identify, tabulate, and assign mutations present in each sequencing read to the corresponding library member. This assignment is based on internal barcodes represented as degenerate codons in the T7 RNA polymerase sequence. The pipeline accepts raw sequence reads in standard genomic formats and performs length-based filtering and optional sequence trimming. An algorithm incorporating local sequence alignment is used for barcode recognition and classification during the demultiplexing of reads. Additionally, each read is aligned to the reference sequence of the canvas to identify the location of each C→T mutation, which is then stored in a matrix for downstream ML model training.

The FLIGHTED-DHARMA Model The model is an implementation of the probabilistic graphical model shown in Figure 3b in Pyro. The sequence-to-fitness function is simply a dictionary lookup, with the fitness of each variant i defined as a parameter to be optimized. Fitness means m_i and variances σ_i are predicted by the sequence-to-fitness function/dictionary, and the variance is transformed by a softplus function F to ensure it is positive. The fitness of each variant is then sampled from a normal distribution $\mathcal{N}(m_i, F(\sigma_i))$ and fitnesses are clamped at -2.

For each position i in the canvas, we set a learnable parameter r_i for a baseline rate of generic mutations (which accounts for most sequencing error from the long-read sequencing). For positions that are cytosines, we set a learnable parameter m_i for the fitness-dependent slope and b_i for the intercept. Then for all cytosines, given a fitness f of the variant, the logit of the C→T edit is set to $m_i + b_i f$ and the logit of any other mutation or deletion is r_i. For non-cytosine residues, we simply have a logit of any mutation set to r_i. We sample from the one-hot categorical distribution for each residue independently to get the output DHARMA read, i.e. from $\text{Cat}((1, m_i + b_i f, r_i))$ for cytosine residues and $\text{Cat}((1, -\infty, r_i))$ for non-cytosine residues.

The guide predicts fitness (with variance) from a single DHARMA read. We used a feedforward network with 2 layers with a hidden dimension of 10 and a ReLU activation between the two layers. To simplify the learning problem, only cytosine residues were fed into the guide and each position was featurized as either a C→T edit or a non-C→T edit (including both no mutation or other mutations or deletions). Variances were transformed under the softplus function. Fitnesses of variants with multiple reads can be predicted by multiplying the appropriate predicted Gaussians of the individual read.

The ELBO loss used 1 ELBO particle. The landscape model (sequence-to-fitness function) learning rate was 10^{-2}, the learning rate elsewhere was 10^{-4}, and the batch size was 2. We used a plateau learning rate scheduler with a patience of 1 epoch and a factor of 0.1, stepped based on the training loss (not validation loss). The model was trained for 25 epochs.

To minimize data leakage, all reads corresponding to sequences in the fluorescence dataset were eliminated from training. 10% of the remaining reads were held out as a validation set to pick the optimal epoch for the model.

Many of the hyperparameters were carefully tuned using a grid search. For tuning, we wanted to use the MSE with the FACS data, so we needed to fit the predicted fitnesses to the FACS data. We did not use Spearman correlation so we could measure true error (as opposed to rank-order error) and so we could account for situations where the function relating the FACS data to predicted fitness was not increasing (as happened here). For this, we split out a 50% of the FACS data to use as a fitness regression/validation set and evaluated the mean of the logarithm of all FACS samples. Each model was given the option of fitting using either a piecewise linear function or a linear regression. The linear regression was fit normally. The piecewise linear function corresponded to the functional form

$$f(x) = \begin{cases} a & x \leq x_c \\ m(x - x_c) + a & x > x_c. \end{cases}$$

This ensured that for low fitnesses, the predicted fluorescence was constant, as would be expected for background fluorescence. We then used orthogonal distance regression, as implemented in `scipy`, to fit the piecewise linear function accounting for errors in both predicted fitness and the FACS data. [10] Between the linear regression and piecewise linear fit, the function with the lower MSE on the validation data was selected.

This validation set MSE was also used for selecting hyperparameters, leaving the remaining 50% of the FACS data as a test set used solely for evaluating model performance as shown in Figure 3c. Hyperparameters tuned included the batch size, learning rate, learning rate scheduler step, and

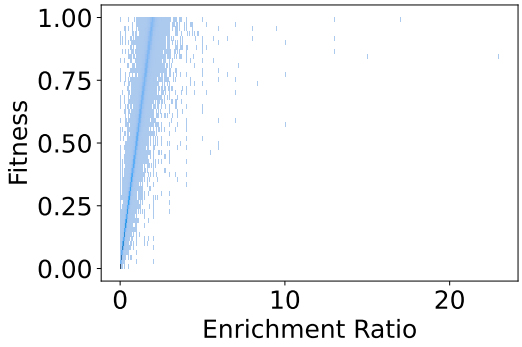


Figure S1: Fitness vs. Enrichment Ratio for Single-Step Selection Simulation. Another perspective on the noise in Figure 2c, showing significant noise in the enrichment ratio.

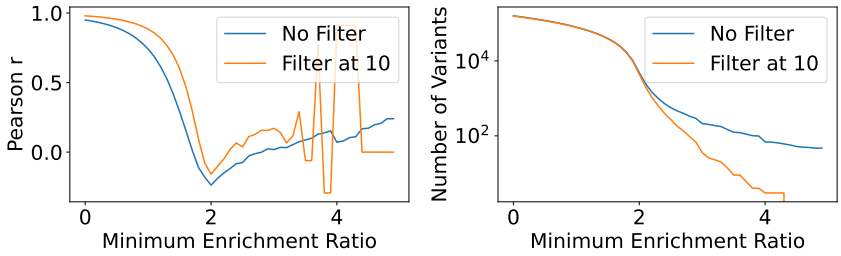


Figure S2: Noise in Single-Step Selection Simulations Increases for Higher Enrichment Ratio. In the left figure, the x-axis is the minimum enrichment ratio and the y-axis is the Pearson r between enrichment ratio and fitness for all data points above that minimum. The drastic drop-off in Pearson r as minimum enrichment ratio increases shows that high-activity variants essentially provide 0 information from just their enrichment ratio. The right figure shows that this is a significant number of variants. We also tried filtering at a minimum of 10 reads, which improves results but still shows significant noise.

number of layers in the guide model. We also evaluated the optimal number of hours to grow the cells in the bioreactor, as an example of an experimental parameter than can be tuned using FLIGHTED.

Model Evaluation Model performance was measured with the 50% held-out test set of the FACS data, as described above, using the fitnesses predicted by the guide model in FLIGHTED-DHARMA and the fitness-to-fluorescence function fit on the validation set. The results of that performance evaluation are shown in Figure 3c.

We then evaluated calibration of the model. To increase the number of data points (since we only had 192 variants measured through FACS), we subsampled subsets of reads for each variant. Specifically, for all test set variants, we sampled 10 subsets each of sizes ranging from 1 to the maximum possible number of reads. We then predicted the fitness of each variant using the given subset of reads with the guide model. We computed the true fitness using the FACS data, eliminating any data points where the true fitness was below the baseline of the piecewise linear function. We then computed the z-score by comparing this predicted and true fitness. This generates the plots shown in Figure 3d.

B Supplementary Results

B.1 FLIGHTED-Selection

Noise in Single-Step Selection Simulations We provide a few more perspectives on the level of noise found in single-step selection experiments based on our simulations. First, Figure S1 shows the enrichment ratio and fitness of all variants in a single-step selection simulation. This is another perspective on the data shown in Figure 2c.

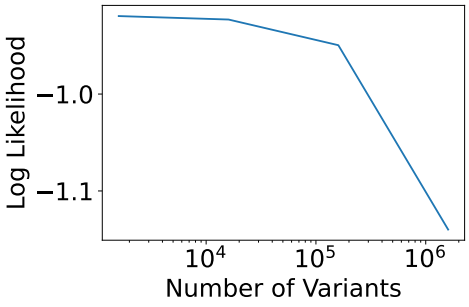


Figure S3: Robustness of FLIGHTED-Selection as a Function of Number of Variants. As the number of variants increases, the log likelihood of FLIGHTED-Selection becomes slightly worse, but not substantially so.

To directly show the extent of the impact on downstream ML, we measure the Pearson correlation between fitness and enrichment ratio. In Figure S2, we do so for subsets of the data with increasing minimum enrichment ratio cutoffs. The Pearson r drops off dramatically as the minimum enrichment ratio increases, showing that noise in the enrichment ratio for high-activity variants is so significant that the data itself is essentially meaningless. The right half of Figure S2 shows that this is still a significant number of variants, at least 1000.

Many papers including the GB1 landscape [25] filter their data by using a minimum number of reads to reduce noise. To evaluate the effect of this, we set a minimum number of 10 reads and repeated the above analysis. We still found considerable noise in the enrichment ratio as shown in Figure S2 (orange line) indicating that setting a minimum is not sufficient to reduce noise in the enrichment ratio.

FLIGHTED-Selection Model Performance We examine the robustness of FLIGHTED-Selection as a function of various parameters of the data generation process. In Figure 2f we showed the log likelihood as a function of number of reads sampled pre- and post-selection and demonstrated that the model was very reliable as long as the number of reads sampled was greater than the number of variants (at 20^4). Next, we fix the number of reads sampled pre- and post-selection and evaluate robustness as a function of number of variants. In Figure S3, we see that the log likelihood of FLIGHTED-Selection does not get considerably worse as a function of the number of variants.

We also experimented with varying various parameters of the random distributions in the simulation to see whether that affected FLIGHTED-Selection model performance. We replaced the uniform distribution used to draw the selection probabilities with a beta distribution, and varied the argument of the beta distribution. This represents fitnesses that peak at lower or higher values as opposed to being uniform between 0 and 1. We also replaced the argument of the Dirichlet distribution used for drawing the initial population with $(\alpha, \ldots, \alpha)$, allowing for biases in the initial population towards one variant. In Figure S4, we see that these factors generally have very little effect on robustness. A Dirichlet distribution with parameter $(0.1, \ldots, 0.1)$ does show some drop in log likelihood, corresponding to a case where one variant is a dramatically larger proportion of the population. So in situations where one variant has dominated the population, FLIGHTED-Selection should not be used.

B.2 FLIGHTED-DHARMA

FLIGHTED-DHARMA Model Performance To better contextualize the fitness-to-FACS fit, we show the fits on the validation set for both FLIGHTED-DHARMA and the baseline model in Figure S5. Generally, FLIGHTED-DHARMA's predicted fitnesses were fit using a piecewise linear function, while depending on the timepoint, the baseline model used either a linear or piecewise linear function.

We next examine the calibration of FLIGHTED-DHARMA as compared to the C→T baseline model. First, we directly examine calibration on the test data, using all reads available for each datapoint in the test set. In Figure S7, we see the true and predicted errors from FLIGHTED-DHARMA and the baseline model on the test set data, as well as the histogram of z scores for both models. The baseline

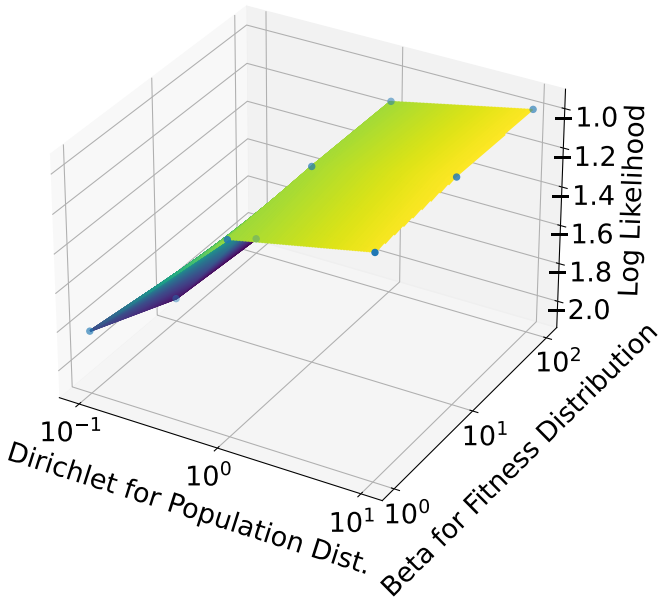


Figure S4: Robustness of FLIGHTED-Selection as a Function of Simulation Parameters. The beta distribution is used to determine the distribution of fitnesses and the Dirichlet distribution is used to determine the distribution of initial populations. We see that these parameters in general have minimal effect on FLIGHTED-Selection log likelihood, but with very low Dirichlet parameters there is a decrease in performance.

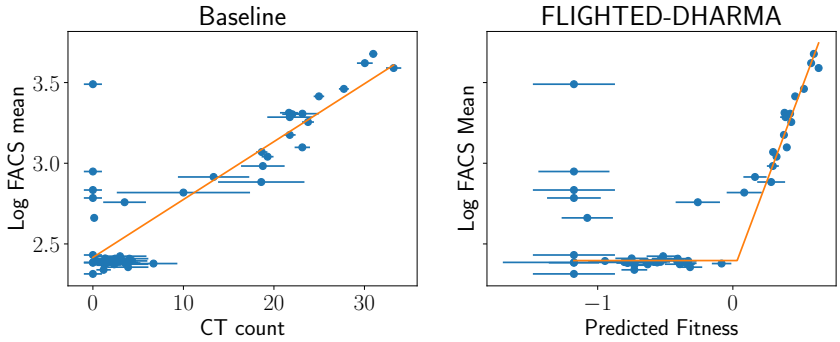


Figure S5: Accuracy on Validation set for (left) Baseline and (right) FLIGHTED-DHARMA. This shows the fit between the fitnesses predicted by each model and the log FACS mean, as done by either a piecewise linear function or a linear function.

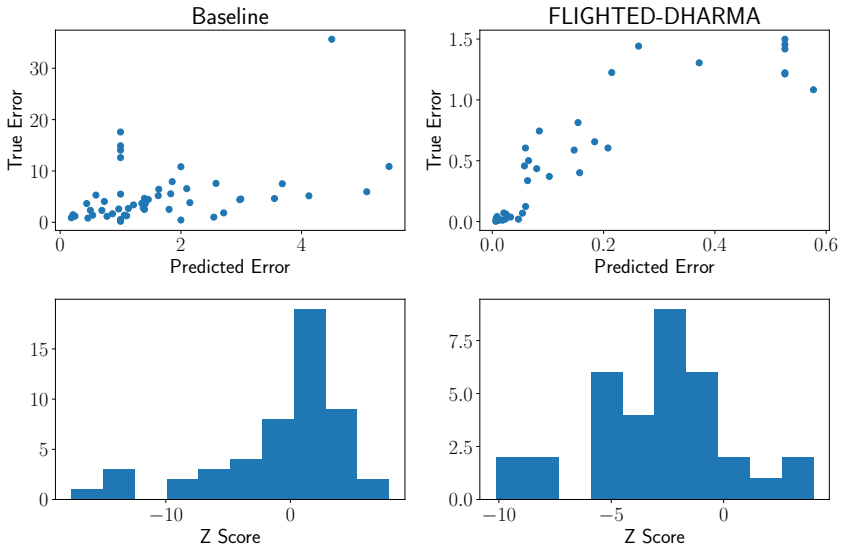


Figure S6: Calibration of FLIGHTED-DHARMA as Compared to Baseline. The baseline model's true errors are not related to the predicted error, while FLIGHTED-DHARMA has higher true error when predicted error is higher. As such, FLIGHTED-DHARMA is considerably better calibrated with many fewer z-score outliers.

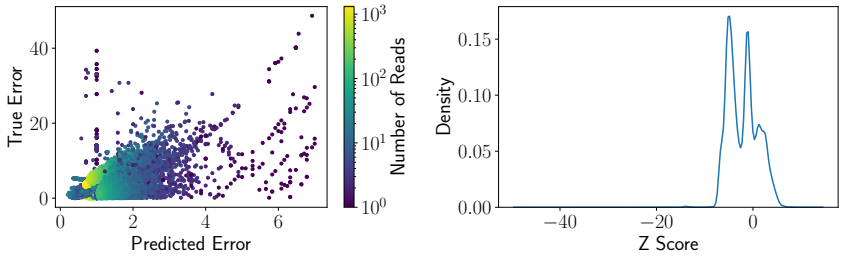


Figure S7: Calibration of Baseline C→T Model. The baseline model's calibration is considerably worse compared to FLIGHTED-DHARMA's in Figure 3d.

model predicted errors are largely unrelated to true error, while the FLIGHTED-DHARMA model's predicted errors increase as true error increases. The log likelihood of the baseline model on this dataset was -19.8 while the log likelihood of FLIGHTED-DHARMA was -10.0, so FLIGHTED-DHARMA shows considerable improvement.

This is a relatively small dataset since we did not subsample subsets of reads as we did in Figure 3d. The subsampling process gives us a better understanding of how FLIGHTED-DHARMA behaves with very small subsets of reads. In Figure S7, we have the calibration of the baseline model as compared with that of FLIGHTED-DHARMA in Figure 3d. The calibration is considerably worse, with the z-score distribution looking very non-normal and a much lower log likelihood of -8.00 as compared to FLIGHTED-DHARMA's log likelihood at -3.93.