Diagnostic design with machine learning model-based optimization

Hayden C. Metsky[1,2] §, Nicole L. Welch[1,3], Nicholas J. Haradhvala[1,4], Laurie Rumker[1,5], Yibin B. Zhang[1,6], Priya P. Pillai[1], David K. Yang[1], Cheri M. Ackerman[1], Juliane Weller[1], Paul C. Blainey[1,7], Cameron Myhrvold[1]∗, Michael Mitzenmacher[1,8]∗, Pardis C. Sabeti[1,6,9,10]∗

[1]Broad Institute of MIT and Harvard, Cambridge, MA, USA. [2]Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA. [3]Virology Program, Division of Medical Sciences, Harvard Medical School, Boston, MA, USA. [4]Biophysics Program, Harvard Medical School, Boston, MA, USA. [5]Bioinformatics and Integrative Genomics Program, Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA. [6]Department of Organismic and Evolutionary Biology, Harvard University, Cambridge, MA, USA. [7]Department of Biological Engineering, MIT, Cambridge, MA, USA. [8]School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. [9]Department of Immunology and Infectious Diseases, Harvard T.H. Chan School of Public Health, Harvard University, Boston, MA, USA. [10]Howard Hughes Medical Institute, Chevy Chase, MD, USA.

∗ Jointly supervised this work.

§ Correspondence to H.C.M. (hmetsky@broadinstitute.org).

Abstract

Diagnostics, particularly for rapidly evolving viruses, can greatly benefit from principled machine learning-based design. Concentrating here on CRISPR-based viral diagnostics, we screen ∼19,000 guide-target pairs and devise a deep neural network to predict a design's diagnostic signal; we then leverage this model to generate assays that are optimal in expectation over a virus's genomic variation. We introduce ADAPT, a system for rapid and fully-automated design using our approach. We design CRISPR-based diagnostics for the 1,933 vertebrate-infecting viral species, and experimentally show ADAPT's designs are sensitive and specific to the lineage-level against extensive viral variation and exhibit lower limits of detection than designs from standard techniques. Model-based optimization is primed to advance CRISPR-based viral diagnostics and many genome-informed technologies for detecting and responding to pathogens.

Main text

Viral diagnostics and surveillance underpin infectious disease efforts. In recent years technological developments, such as CRISPR-based tools, have enhanced viral detection[1–7]. Yet there has been a dearth of progress in using computational design to enhance diagnostics and surveillance. This is surprising in light of recent advances in machine learning and optimization algorithms, the explosion of viral genomic data, and the importance of diagnostics, highlighted by the COVID-19 pandemic. Indeed, designing viral assays from genomic data is still done largely by hand, without well-defined objectives and with a great deal of trial and error.

Harnessing predictive models, to allow for model-based optimization, promises many practical benefits for diagnostics. It could design assays that are more sensitive than existing ones at low viral concentrations and across viral variation. It would enable a large resource of proactively-developed assays that are broadly effective against known viruses and even many novel ones. And it would allow for rapid design of new assays and efficient redesign to reflect new viral variants over time. Here, we demonstrate these capabilities by developing and experimentally validating a design approach that combines a deep learning model with combinatorial optimization across vast viral genomic data. As an application we concentrate on CRISPR-based viral diagnostics, though the approach can be applied to other pathogens and other diagnostic technologies, such as RT-qPCR, and sequence-based therapy and vaccine design.

There are three key design challenges for detecting viruses. One is viral variation, which afflicts diagnostics and surveillance in practice. Influenza A virus (FLUAV) RT-qPCR tests often have false-negative rates over 10%[8–10] (nearly 100% on some circulating strains) owing to extensive variation, and the issue besets diagnostics for many other viruses[11–15]. Prior analytical work has focused on qPCR-based diagnostics[16–26]. Generally, the design process treats variation as a marginal consideration by wholly ignoring it or designing assays within conserved genomic regions from a single viral genome sequence. Even the latter technique is inadequate because conserved regions are rarely free of variation, and targeting them may not give the optimal sensitivity and antagonizes specificity between related viruses. Our approach, in contrast to prior approaches, directly integrates viral variation into an objective function. We use combinatorial optimization to generate designs that maximize detection in expectation over a virus's full genomic variation.

The second design challenge is predicting detection. Existing analytical methods, such as for qPCR, make binary decisions—an assay detects a viral genome or does not—according to thermodynamic principles and rudimentary heuristics. Yet binary decisions are unrealistic—some assays have much better sensitivity or specificity than others—so they still impel experiments to find the best and might miss the optimum. And thermodynamic criteria and basic heuristics are not generalizable across technologies. In contrast to prior techniques, our approach uses a machine-learned model and extensive experimental data to quantitatively evaluate viral detection. Focused on CRISPR-based diagnostics, we screened 19,209 guide-target pairs, forming the largest dataset for diagnostics. Our deep learning model predicts the extent of enzymatic activity in a detection reaction, which corresponds to the sensitivity of a diagnostic readout. The model learns the effect of viral variation on a diagnostic because our dataset emphasizes imperfect homology. Hence, in our approach, a deep learning model guides the optimization process across viral variation and throughout the genome.

The third design challenge is scalability. The number of viral genomes is growing exponentially[27,28], encompassing well-known viruses and new strains that may cause epidemics (Supplementary Fig. 1).

This growth complicates the design of diagnostic and surveillance tools, which is based on those genomes. Moreover, viral evolution can necessitate periodic updating as variants emerge. Well-designed FLUAV subtyping assays will lose sensitivity over time, especially profoundly following a pandemic (Fig. 1a and Supplementary Fig. 2). And in the case of COVID-19, as in other outbreaks, mutations quickly accumulated on top of the early genome sequences (Fig. 1b) and some have created failures in widely-used SARS-CoV-2 diagnostics[29,30]; it is likely that SARS-CoV-2 assays will be redesigned to accommodate variation. Yet designing an assay requires curating viral genome data, which in practice is laborious, a bottleneck, and unsustainable. To overcome this obstacle, we also introduce ADAPT, which implements our approach, is fast, and automatically integrates the latest viral genomes from public databases into the design process. ADAPT is available at https://github.com/broadinstitute/adapt, and is written flexibly to support applications beyond CRISPR-based viral diagnostics.

We applied our machine learning-based optimization approach to design maximally active, species-specific CRISPR diagnostics for the 1,933 viral species known to infect vertebrates. We experimentally show their potential on SARS-CoV-2 and other viruses. Having already performed over 19,000 tests for modeling, we also validate our approach against target variation far more extensively than is typical: we evaluate 69 designs, each against as many as 15 targets to encompass one virus's variation (290 different targets in total). The results demonstrate that our approach provides designs with comprehensive detection across genomic variation and lineage-level specificity. Our designs also outperform assays designed using standard techniques that are based on conventional CRISPR diagnostic rules and sequence conservation.

Designing optimally active probe sets

First, we develop an algorithm to design diagnostic probe sequences (for example, CRISPR guides) that optimally detect known variation. The result is broadly applicable to many other sequence-based design tasks. We rely on all known sequences within a genomic region, represented by S, and a function that quantifies detection activity between a particular probe and a targeted sequence such that higher activities are more desirable. Later, we will describe how to identify optimal regions and devise an activity function, for CRISPR-based diagnostics, using a machine-learned model. We initially construct a ground set of possible probes by finding representative subsequences in S using locality-sensitive hashing[31]. Our goal is to find the set P of probes, a subset of the ground set, that maximizes the expected activity between P and S (Fig. 1c). In this objective function, the expectation is over the sequences in S and the probability for each sequence reflects a prior on applying the probes toward that sequence; it could weigh highly sequences that are recent or from a particular geographic region, though in practice we usually weigh them uniformly. Larger numbers of probes might require more detection reactions or interfere if multiplexed in a single reaction, and thus we impose a penalty in our objective function and a hard constraint on the number of probes (details in Supplementary Note 1a).

Having formulated an objective function, we then developed an approach to solve it using combinatorial optimization. Our objective function is non-monotone submodular and, under activity constraints on probes allowed in the ground set, is non-negative. Submodularity indicates that, as the number of probes increases, the function reports diminishing improvements in performance, as expected. We apply a fast randomized combinatorial algorithm[32] for maximizing a non-negative and non-monotone submodular function under a cardinality constraint, which provides a probe set whose objective value is within a factor $1/e$ of the optimal. Supplementary Note 1a contains the proofs and algorithmic details. We also evaluated the canonical greedy algorithm[33] for constrained

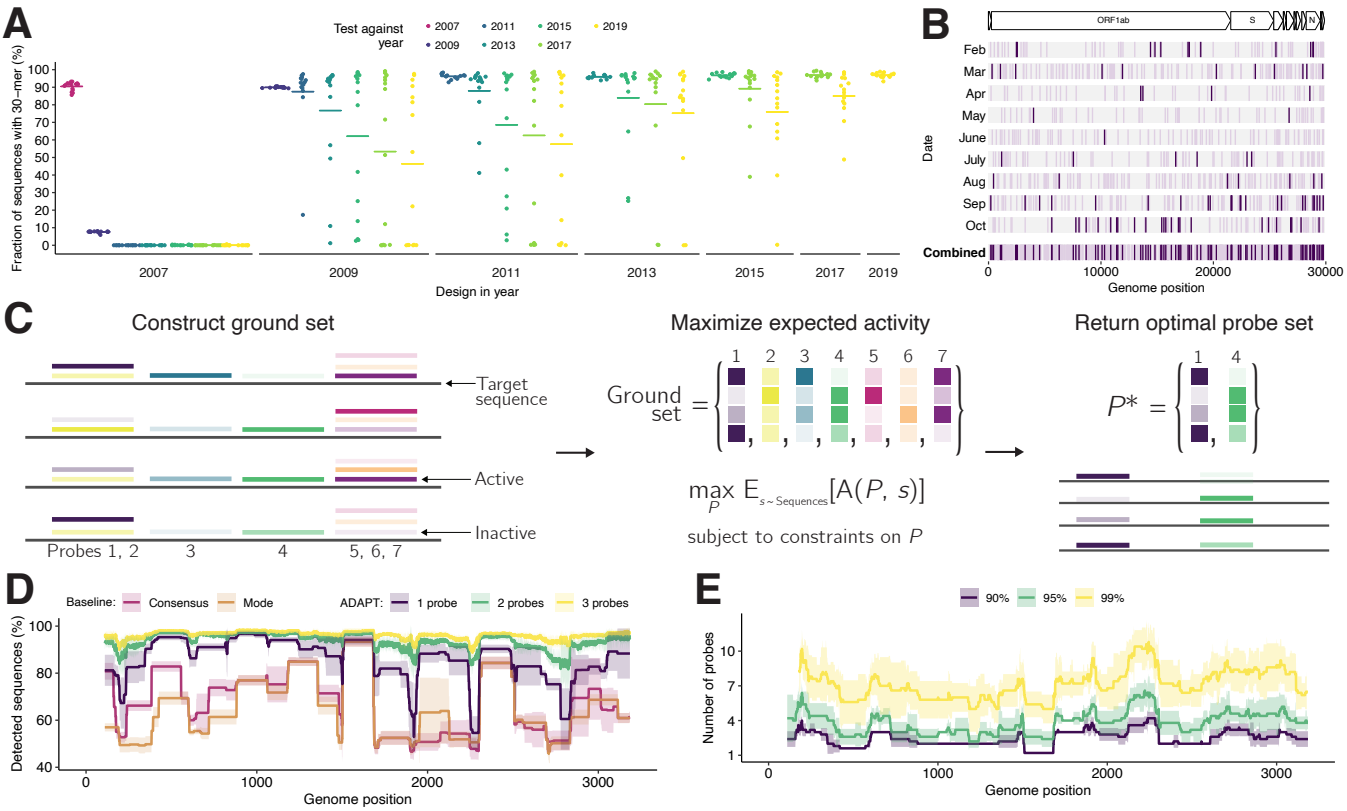


Figure 1 — Maximizing detection across genomic variation. **(a)** Diagnostic performance for influenza A virus subtyping may degrade over time, even considering conserved sites. At each year, we select the 15 most conserved 30-mers from recent sequences for segment 6 (N) for all N1 subtypes; each point is a 30-mer. Plotted value is the fraction of sequences in subsequent years (colored) that contain the 30-mer; bars are the mean. To aid visualization, only odd years are shown. 2007 N1 30-mers are absent following 2007 owing to shift during the 2009 H1N1 pandemic. **(b)** Variants along the SARS-CoV-2 genome emerging over time. Bottom row ('Combined') shows all 853 variants, among 180,258 genomes, that crossed 0.1% or 1% frequency between February 1 and November 1, 2020—i.e., (i) at $<$0.1% frequency in genomes collected before February 1 and 0.1–1% frequency by November 1 (light purple); or (ii) at $<$1% frequency before February 1 and at $\geq$1% frequency by November 1 (dark). Other rows show the month in which each variant crosses the frequency threshold. **(c)** Our approach for designing maximally active probe sets. We (1) find representative subsequences in a genomic region, which are possible probes (colored); (2) calculate an activity (shaded) between each probe and each target sequence s, forming the ground set of probes; (3) find a probe set P, a subset of the ground set, maximizing the expected activity $\mathrm{A}(P, s)$ between P and s, subject to soft and hard constraints on P. **(d)** Fraction of Lassa virus (LASV; segment S) genomes detected, with different design strategies in a 200 nt sliding window, using a model in which 30 nt probes detect a target if they are within 1 mismatch, counting G-U pairs at matches. 'Consensus', probe-length consensus subsequence that detects the most number of genomes; 'Mode', most common probe-length subsequence. Our approach (ADAPT), with hard constraints of 1–3 probes, maximizes activity. **(e)** ADAPT with a dual objective: minimal number of guides to detect $>$90%, $>$95%, and $>$99% of LASV genomes using the model in (d). In (d) and (e), shaded regions are 95% pointwise confidence bands across randomly sampled input genomes.

submodular maximization; it returns similar results in practice (Supplementary Fig. 3), though does not offer provable guarantees in our case because it requires monotonicity.

We benchmarked our approach's comprehensiveness across sequence variation. To make the benchmarking interpretable, we chose a detection activity function that equals 1 if a probe is within 1 mismatch of a target (detected) and 0 otherwise (not detected); this function has the property that expected activity is equivalent to the fraction of genomes detected. Two simple strategies for constructing probes, using the consensus or most common sequence in a region, fail to capture diversity for Lassa virus and other highly diverse viruses (Fig. 1d and Supplementary Fig. 4a).

Our strategy, to maximize expected activity, yields designs with greater comprehensiveness than the two simple strategies on these same viruses. They detect more variation—even when constrained to a single probe—across the genome, and the amount detected increases as we permit more probes (Fig. 1d and Supplementary Fig. 4a). This result is expected because our approach explicitly maximizes detection over the variable sequences. Using a related objective function that minimizes the number of probes subject to constraints on comprehensiveness (Supplementary Note 1b), we find it is possible to reach near-complete comprehensiveness across the genome with few probes (Fig. 1e and Supplementary Fig. 4b). On species with less extensive diversity, such as Zika virus, simple strategies perform well (Supplementary Fig. 4a), suggesting that our more involved approach is not always necessary. Nevertheless, having options to comprehensively target many regions of a genome enables us to integrate genuine activity predictions and enforce viral taxon-specificity, both of which will constrain the designs.

Predicting CRISPR-based diagnostic signal with deep learning

In practice, to maximize our objective we will need a realistic activity function: for this function, we use a machine-learned model that predicts the resulting activity of a probe-target combination. We developed such a model informed by detection reaction kinetics. We focus on CRISPR-Cas13a technologies[1,2], in which Cas13a enzymes use probes (here, guide RNAs) to detect a target sequence and then exhibit collateral RNase activity that cleaves fluorescent reporters, leading to a readout. Prior studies have established design principles for guide activity—such as the importance of a protospacer flanking site (PFS), RNA secondary structure, and a mismatch-sensitive "seed" region[34–36]—but have not modeled the activity of Cas13a or other CRISPR enzymes for detection applications.

To enable a model, we first designed a library of 19,209 unique LwaCas13a guide-target pairs (Fig. 2a and Supplementary Fig. 5a). The library has a sequence composition representative of viral genomes (Methods), an average of 2.9 mismatches between each guide and target (Supplementary Fig. 5b), and a representation of the different PFS alleles (Supplementary Fig. 5c). We model reporter presence during each pair's reaction with an exponential decay and use the negative decay to model fluorescence over time (Fig. 2a; Methods). The fluorescence growth rate is a function of the target concentration and enzymatic efficiency of a guide-target-Cas13a complex, so we hold the former constant to evaluate the latter (Supplementary Fig. 6a,b); we define activity as the logarithm of this growth rate.

To perform the detection reactions that test our library, we used CARMEN-Cas13[37]. CARMEN is a massively multiplexed droplet-based platform; its scalability enables our extensive dataset and our large-scale experiments that validate our machine learning-based design approach. By measuring fluorescence roughly every 20 minutes, we calculate the activity for each guide-target pair in the library. The data include, on average, about 15 replicate activity measurements per pair (Supplementary Fig. 7a–d) and exhibit activity differences, spanning several orders of magnitude in fluorescence growth rate, between different guide sequences and across the different target variants each guide detects (Supplementary Fig. 7e–g). Our measurements are consistent with expected covariates of activity, including the PFS and number of mismatches.

Using our dataset, we developed a machine-learned model to predict CRISPR-Cas13a activity of a guide-target pair. We use a two-step hurdle model: classifying a pair as inactive or active, and then regressing activity for active pairs (Fig. 2b and Supplementary Fig. 6b; 86.8% of the full dataset is labeled active). For classifying guide-target pairs, we performed nested cross-validation

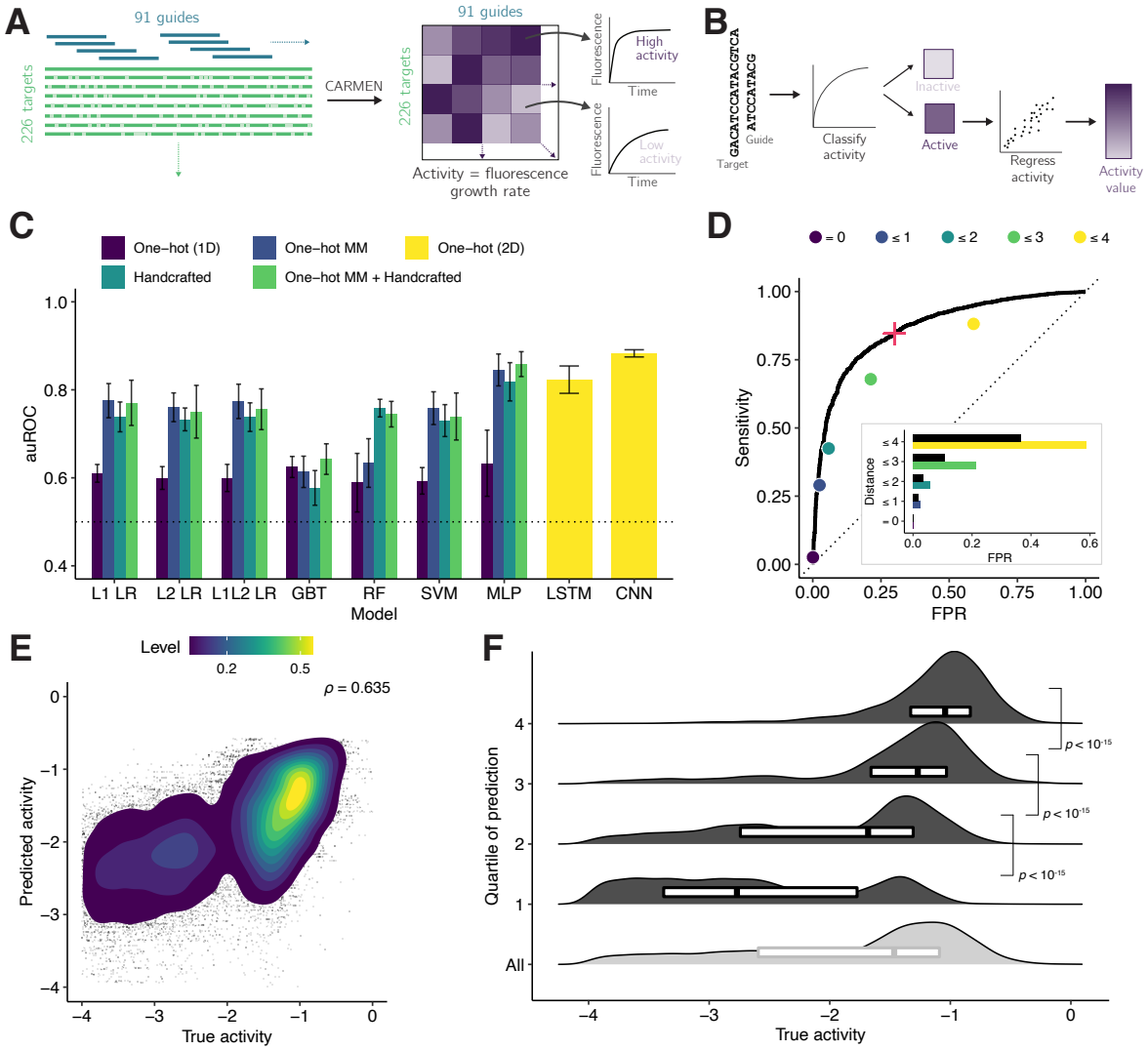


Figure 2 — Predicting CRISPR-Cas13a detection activity. (a) The library consists of a 865 nt long wildtype target sequence and 91 guide RNAs complementary to it, along with 225 unique targets containing mismatches and varying PFS alleles relative to the wildtype. We measure fluorescence every ∼20 minutes for each pair and use the growth rate to quantify activity. The dataset contains 19,209 unique guide-target pairs. **(b)** We predict activity for a guide-target pair in two parts, which requires training a classifier on all pairs and a regression model on the active pairs. **(c)** Results of model selection, for classification, with nested cross-validation. For each model and input type (color) on five outer folds, we performed a five-fold cross-validated hyperparameter search. Plotted value is the mean auROC of the five models, and error bar indicates the 95% confidence interval. L1 LR and L2 LR, logistic regression; L1L2 LR, elastic net; GBT, gradient-boosted classification tree; RF, random forest; SVM, support vector machine; MLP, multilayer perceptron; LSTM, long short-term memory recurrent neural network; CNN, neural network with parallel convolutional filters and a locally-connected layer. One-hot (1D), one-hot encoding of target and guide sequence without explicit pairing; One-hot MM, one-hot encoding of target sequence and of mismatches in guides relative to the target; Handcrafted, curated features of hypothesized importance (details in Methods); One-hot (2D), one-hot encoding of target and guide sequence with encoded guide-target pairing. Supplementary Fig. 8 shows auPR and regression models. **(d)** ROC curve of CNN, which is used in ADAPT, classifying pairs as inactive or active on a held-out test set. Points indicate sensitivity and FPR for a baseline classifier: choosing a guide-target pair to be active if it has a non-G PFS and the guide-target Hamming distance is less than a specified threshold (color). Inset plot shows comparison of FPR between CNN (black) and baseline classifiers at equivalent sensitivity. Red '+' indicates decision threshold in ADAPT. **(e)** Regression results of CNN for predicting activity values from active guide-target pairs in the test set. Color, point density. ρ, Spearman correlation. Not shown: points with predicted activity < -4 (0.23% of all). **(f)** Same data as (e). Each row contains one quartile of pairs according to predicted activity (top row is predicted most active), with the bottom row showing all active pairs. Smoothed density estimates and interquartile ranges show the distribution of true activity for the pairs from each quartile. *P*-values are computed from Mann-Whitney *U* tests (one-sided).

to evaluate our hyperparameter search and to compare nine models potentially suited to this task using different inputs, including one-hot encodings and handcrafted features. We found that a deep convolutional neural network (CNN), trained from nucleotide sequence, outperforms the other models (Fig. 2c and Supplementary Fig. 8a). For regression on active pairs, we also found that a CNN outperforms other models, albeit with a less noticeable improvement over the simpler models than for classification (Supplementary Fig. 8b,c). CNNs have also performed well when regressing Cas12a guide RNA editing activity on matching target sequences[38], likely because the convolutional layers detect motifs in the sequences; here, such layers may also detect patterns of mismatches. In all model training and testing, we accounted for measurement error (Methods).

We proceeded with further evaluating CNN models to use for designing CRISPR-based diagnostics. During model selection, our space of CNN models allows for multiple parallel convolutional and locally connected filters of different widths (Supplementary Fig. 9). Using a locally connected layer significantly improves model performance (Supplementary Fig. 10), which we hypothesize is because it helps the model learn spatial dependencies in the data—for example, a mismatch-sensitive seed region—that may be missed by convolutional layers and difficult for fully connected layers to ascertain.

We evaluated our CNN models' performance on a held-out test set of CRISPR-Cas13a activity (Methods) and against methods that use standard Cas13a design rules. Our classification CNN performs well on the held-out data (auROC = 0.866; auPR = 0.972 with 85.6% of the test set being true positive; Fig. 2d and Supplementary Fig. 11a). When the guide and target are not identical, it yields a lower false-positive rate and higher precision than a simple method classifying activity according to the PFS and guide-target divergence (Fig. 2d and Supplementary Fig. 11b). Our regression CNN also predicts the activity of active guide-target pairs well ($\rho = 0.635$; Fig. 2e and Supplementary Fig. 12a) and accurately separates pairs into quartiles based on predicted activity (Fig. 2f and Supplementary Fig. 12b). Simple features, such as the PFS and number of mismatches (Supplementary Fig. 13), extracted by the CNN models could explain much of the models' performance, yet they retain predictive ability when evaluated individually on different choices of PFS (Supplementary Fig. 14a,b and 15a–d) and mismatch thresholds (Supplementary Fig. 14c,d and 15e–g). A learning curve shows that additional data similar to our current dataset would not improve model performance (Supplementary Fig. 16), though this does not preclude the possibility of additional distinct guide sequences improving performance.

To use these models as part of our machine learning-based design, we chose a decision threshold for the classifier that yields a precision of 0.975 (Methods; Fig. 2d and Supplementary Fig. 11a) and then defined a piecewise activity function that is 0 if a guide-target pair is classified as inactive, and the regressed activity if active.

Beyond modeling CRISPR diagnostic activity, we also examined our dataset to understand LwaCas13a preferences, starting with the PFS. Prior studies have seen weaker detection activity with a `G` PFS for LwaCas13a[1] and characterized the PFS preference, if any, for other orthologues[34,36] and in other settings[35]. We also observe reduced activity with a `G` PFS and, extending to another position, we find that `GA`, `GC`, and `GG` provide higher activity than `GT` (Supplementary Fig. 17a), suggesting 3′ `HV` may be a more stringent preference. Mechanistically, the reduced activity of a `G` PFS results from base pairing with the crRNA's complementary `C` in the direct repeat sequence immediately 5′ of the spacer, which prevents separation of the guide-target duplex[39]; the direct repeat's subsequent base is `A`, so the additional complementarity of `GT` may explain this further reduction.

The breadth of our data also clarifies effects of mismatches (Supplementary Fig. 17b) and the mismatch-sensitive seed region that had previously been identified, using a limited number of guide-target pairs, for LbuCas13a[36] and LshCas13a[34]. In particular, we find that the worst-performing guide-target pairs are relatively likely to contain mismatches in positions 6–11 of the guide RNA spacer (Supplementary Fig. 17c), concordant with the known region. We also find nearly full tolerance for multiple mismatches on the 3′ end of the spacer (Supplementary Fig. 17d), a property that our predictive model learns and leverages when positioning CRISPR guides in variable regions. Coefficients from the regularized linear models included in our model comparison are consistent with our findings and also reveal position-specific allele mismatches that affect activity (Supplementary Fig. 13). A thorough modeling of defined features, like that performed for Cas13d[40], could highlight additional principles for Cas13a guide design.

Enforcing specificity among viral taxa

In many diagnostic applications, assays must distinguish between viral species or strains that are genetically similar. When solving our combinatorial optimization problem, we enforce taxon-specificity by constraining the ground set of probes to only ones deemed specific (Supplementary Note 2a). Determining whether a probe is specific to a viral taxon faces two computational challenges: it should tolerate multiple mismatches between a probe and potential off-targets and, when the probes and targets are RNA, account for G-U wobble base pairs. Accommodating the latter is critical: we found that ignoring G-U pairing when designing viral species-specific probes can result in missing nearly all off-target hits and deciding many probes to be specific to a viral species when they likely are not (Supplementary Fig. 18). Similar challenges arise in other RNA applications, such as the design of small interfering RNA, but most prior approaches simply ignore the G-U problem and, if they do address it, they use a solution inadequate for viral diagnostic goals (details in Supplementary Note 2b).

We experimented with two algorithms, which address these challenges, to decide a probe's specificity to a viral taxon. We first tested a probabilistic near-neighbor query algorithm that has a tunable reporting probability for identifying off-target hits (Supplementary Note 2c). When amplified across many designs, the technique misses off-target hits and permits non-specific guides; a sufficiently high reporting probability would make this outcome unlikely, but we found it makes queries too slow to be practical. Thus, we instead developed a data structure and query algorithm that are fully sensitive to high divergence and G-U wobble base pairing. The data structure indexes k-mers from all input taxa, in which the k-mers are split across many small tries according to a hash function (Supplementary Fig. 19 and Supplementary Note 2d). The approach is exact—it provably identifies all off-target hits—and therefore, in theory, guarantees high specificity of our designs. When designing species-specific viral diagnostics, the approach runs 10–100 times faster than a simple data structure with the same capability (Supplementary Fig. 20).

Designing broadly-effective diagnostics at scale

We built ADAPT, which implements our machine learning-based optimization approach and performs a fully end-to-end design of viral diagnostics (Fig. 3a). ADAPT performs a search across a viral genome to identify regions to target, scored according to their amplification potential (including presence of conserved primers) and the activity of an optimal probe set within the region. Its genome-wide search follows the branch and bound paradigm and identifies the best N diverse design options; users preset N, for which smaller values speed the search. Diverse design options enable

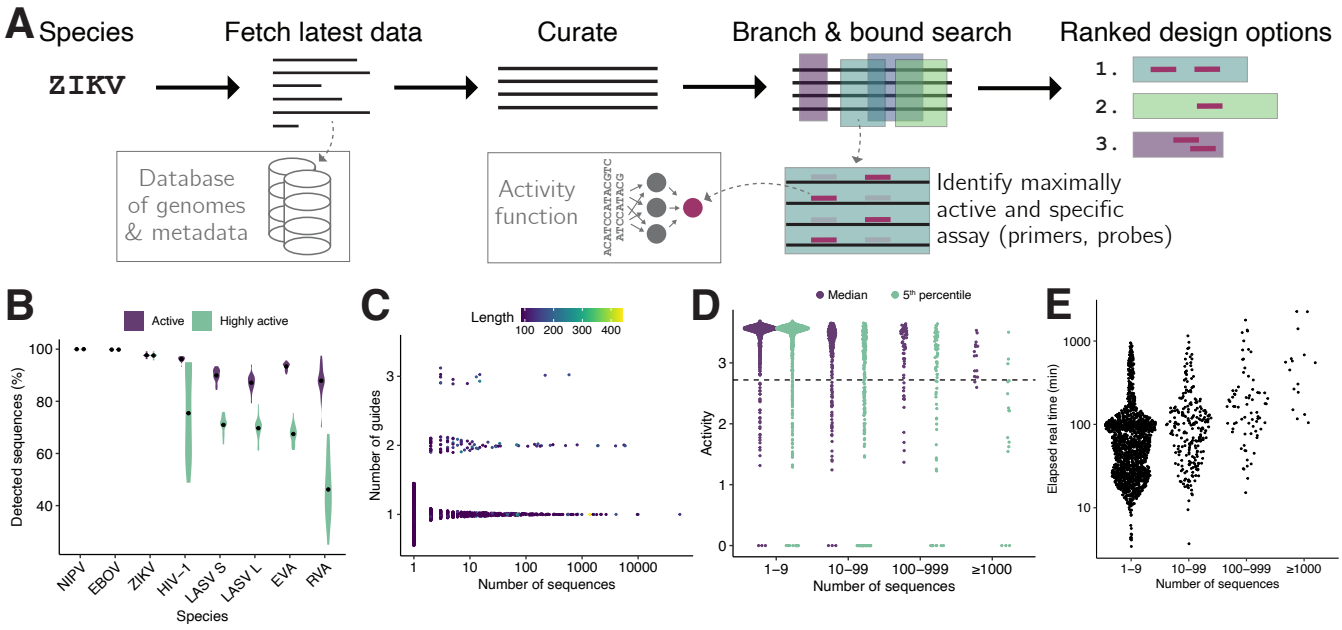


Figure 3 — End-to-end design with ADAPT. (a) Sketch of ADAPT's steps. ADAPT accepts a list of taxonomy identifiers and fetches and curates their sequences from NCBI's viral genome databases. It performs a branch and bound search to find genomic regions—each is an amplicon with primers—that contain a maximally active probe set, while enforcing taxon-specificity. ADAPT outputs the top design options, ranked by an objective function. **(b)** Cross-validated evaluation of detection. For each species, we ran ADAPT on 80% of available genomes and estimated performance, averaged over the top 5 design options, on the remaining 20%. Distributions are across 20 random splits and dots indicate mean. Purple, fraction of genomes detected by primers and for which Cas13a guides are classified as active. Green, same except Cas13a guides also have regressed activity in the top 25% of our dataset. NIPV, Nipah virus; EBOV, Zaire ebolavirus; ZIKV, Zika virus; LASV S/L, Lassa virus segment S/L; EVA, Enterovirus A; RVA, Rhinovirus A. **(c)** Number of Cas13a guides in the top-ranked design option for each species in the vertebrate-infecting virus designs. Color is the length of the targeted region (amplicon). **(d)** Activity of each guide set with two summary statistics: median and the 5$^{\text{th}}$-percentile taken across each species's sequences. For the latter, value a indicates that 95% of sequences are detected with activity $\geq a$. Dashed line indicates the "high activity" threshold from (b). Sequences at 0 activity are classified as not being detected. Activities shown here are shifted up by 4 compared to the model output from Fig. 2. **(e)** End-to-end elapsed real time running ADAPT. In (c–e), each point is a vertebrate-infecting viral species.

assays that target multiple distinct regions of a genome. ADAPT memoizes repeated computations during its search, which decreases its runtime by 99.71–99.96% for species we tested (Supplementary Fig. 21). Details are in Supplementary Note 3a. ADAPT automatically downloads and curates all near-complete or complete genomes from NCBI databases[28] for a specified virus taxonomy, as well as for building the index that enforces specificity (Supplementary Note 3b). Fully-automated sequence curation greatly simplifies the design process, especially for viruses with a large and growing number of public genome sequences.

For some viruses there are few available genome sequences, especially early in an outbreak, and therefore little informative data. We developed a proactive scheme that uses the GTR nucleotide substitution model[41] to find relatively likely combinations of genome substitutions in the region a probe detects, allowing us to estimate a probability that a probe's activity will degrade over time (Supplementary Note 4 and Supplementary Fig. 22a). Applied to SARS-CoV-2, we found that, for some Cas13a designs, there is a low probability ($\sim$10%) their predicted activity will degrade over a 5 year period (Supplementary Fig. 22b,c). The decay results from mutations at mismatch-sensitive sites or at sites in the context of the Cas13a protospacers, which would impact some designs more than others. This information may be helpful in risk-averse situations, but it has only a minor effect overall on ADAPT's design option rankings (Supplementary Fig. 22d).

We computationally evaluated designs output by ADAPT, accounting together for their primers and probes, on seven RNA viruses spanning a range of genomic diversity. We used our CRISPR-Cas13a activity function, so the probes are Cas13a guides. Algorithmic randomness and the particular distribution of known sequences may introduce variability; we found that ADAPT's designs are rarely exactly the same across runs (Supplementary Fig. 23a,b), but that they do often target similar genomic regions (Supplementary Fig. 23c,d). Cross-validation confirms the designs' generalization: designs are predicted to be active in detecting >85% of held-out genomes for all seven species and have, in all but one species, "high activity" (defined as top 25% of measurements in our dataset) in detecting the majority of held-out genomes (Fig. 3b). The results show that ADAPT's outputs are robust across different viruses.

We then applied ADAPT to design species-specific detection assays, including primers and Cas13a guides, for the 1,933 viral species known to infect vertebrates. The designs have short amplicons (generally <200 nt) and use 3 or fewer guides for all species, including ones with thousands of genome sequences (Fig. 3c and Supplementary Fig. 24a). Thus, the assays are practical. For 95% of species, the guides detect the majority of known genomes with high predicted activity (Fig. 3d; for 88% of species, they detect >95% with high activity). If we were to instead optimize a reformulation of our maximization problem—minimize the number of guides subject to detecting >98% of genomes with high activity—we would also obtain few guides for most species although 40 species would require more than 3 guides (Supplementary Fig. 24b) and, in one extreme case, as many as 73 (Enterovirus B).

Along with detecting known viruses, our assays—designed to comprehensively detect species-level diversity—could detect novel viruses that are nested within known species, as is commonly the case. We simulated the design of assays in 2018 for detecting the SARS-related coronavirus species and then evaluated their detection of SARS-CoV-2, which is a part of the species but did not emerge until a year later. ADAPT's second-highest–ranked assay is predicted to detect SARS-CoV-2 well and two other designs in the top five are predicted to exhibit low activity for SARS-CoV-2 (Supplementary Fig. 25a,b). This result is facilitated by bat SARS-like viral genomes that were available in 2018 and had homology to SARS-CoV-2. Nevertheless, heavy sampling biases hinder the efficacy of ADAPT's designs: in 2018, SARS-CoV-1 was overrepresented in the species (85%) relative to bat SARS-like viruses owing to SARS outbreak sequencing. After down-weighing ADAPT's consideration to SARS-CoV-1 (Methods), four of ADAPT's five highest-ranked 2018 assays are predicted to detect SARS-CoV-2 well (Supplementary Fig. 25c,d). Such broadly effective assays promise to benefit surveillance and diagnostics by constituting a proactively developed toolkit.

We also examined the computational requirements of designing species-specific assays for 1,933 viral species. End-to-end design completed in under 2 hours for 80% of species, under 24 hours for all but 3 species (human cytomegalovirus, SARS-related coronavirus, and FLUAV), and under 38 hours for all; runtime depended in part on the number of genome sequences (Fig. 3e) and it required about 1 to 100 GB of memory per species (Supplementary Fig. 24c). After curating available sequences, ADAPT considers all or almost all genome sequences for most species (Supplementary Fig. 24d–f), including the ones with >1,000 sequences; it does retain fewer than half of sequences for 38 species, which might be appropriate but prompts further investigation. Enforcing species-specificity imposes a computational burden, adding to the runtime and memory usage (Supplementary Fig. 26a,b) while decreasing the solution's activity and objective value because of the constraints, as expected (Supplementary Fig. 26c,d). Overall, the low runtime implies it is possible to rapidly produce new, optimal designs that reflect evolving genomic variation or to detect a novel virus.

Experimental evaluation of ADAPT's designs

We experimentally benchmarked the diagnostic utility of our approach. We first focused on the US CDC's SARS-CoV-2 qPCR assay amplicons, a target both of qPCR and CRISPR-based diagnostic assays[42]. As baselines in the N1 amplicon, we selected a Cas13a guide at the site of the qPCR probe and 10 random guides within the amplicon, all having an active (non-G) PFS; selecting guides according to the PFS is the canonical design strategy, and thus the distribution of their activity is a benchmark. The guide designed by our approach exhibits higher fluorescence at low target concentrations than all 11 of the baseline guides (Fig. 4a). Its fluorescence also grows at a faster rate than all 11 (Fig. 4b and Supplementary Fig. 27a). We observed similar results repeating the comparison in the N2 amplicon (Supplementary Fig. 27b,c). These findings indicate that our machine learning-based design permits better sensitivity, against a known target sequence, than the canonical approach focused on the PFS.

Next, we validated the comprehensiveness and specificity offered by our approach by considering multiple taxa comprising the SARS-related coronavirus (SARS-related CoV) species (Fig. 4c). We tested Cas13a-based designs against representative targets within each taxon, which we identified systematically according to sequence composition (Methods). Our testing directly measures the fluorescent signal yielded by the Cas13a guides at varying target concentrations. We started with precise targeting of SARS-CoV-2. Using ADAPT, we generated assays for detecting SARS-CoV-2, with lineage-level specificity, that should not detect any known bat or pangolin SARS-like coronaviruses, including the RaTG13 genome (96% identity to SARS-CoV-2[43]), as well as SARS-CoV-1 and other coronaviruses (Fig. 4c). All three of our approach's best design options (ranked according to predicted activity) detect SARS-CoV-2 with complete specificity—we observe no fluorescent signal for the closely related lineages (Fig. 4d).

We then broadened the targeted space within SARS-related CoV. We designed assays for the SARS-CoV-2–related lineage[44], which consists of SARS-CoV-2 and its closely related bat and pangolin CoVs. Our approach's three top-ranked designs sensitively and specifically detect all representative targets within SARS-CoV-2–related (Fig. 4e). Unlike the SARS-CoV-2 designs, we observed low off-target SARS-CoV-1 signal with the SARS-CoV-2–related designs because their added comprehensiveness antagonizes specificity; however, this is unlikely to affect diagnostic results that use an adequate signal threshold for detection. We also designed species-specific assays for the full SARS-related CoV species, and ADAPT's top-ranked designs detect all representative targets within the species without any signal for three other betacoronaviruses (Fig. 4f).

Overall, all of our top-ranked designs—generated fully automatically and without any human input or experimental refinement—perform as desired across the SARS-related CoV taxa. For each taxon, ADAPT also generated additional design options (25 total) and we generally observe the desired activity; designs with poorer sensitivity or specificity are in the lower half of rankings according to predicted activity (Supplementary Fig. 28a–c). Four of the designs use two Cas13a guides and, in these cases, our measurements show that our combinatorial optimization algorithm selects them to detect distinct lineage groupings in order to maximize their collective sensitivity for the taxon (Supplementary Fig. 29).

We also evaluated limits of detection across extensive genomic variation. We focused on enteroviruses, which are estimated to cause millions of symptomatic infections yearly and frequent outbreaks[45]. They have over 100 types and likely many undiscovered types that cause human disease[46]. Testing increasingly relies on pan-enterovirus qPCR by targeting a highly conserved region, which has clinical value but limited surveillance utility[47]; an assay that provides more resolution

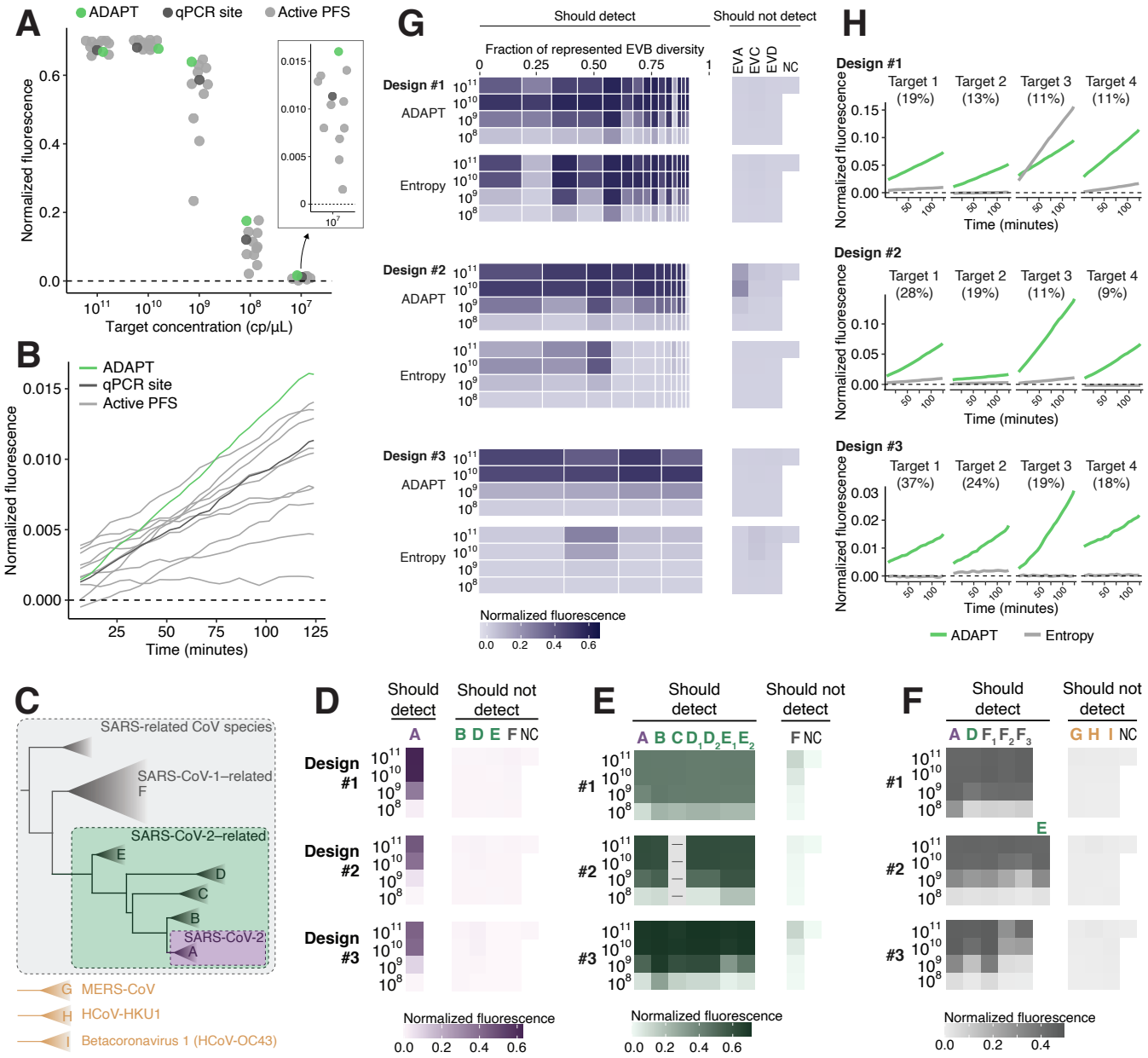


Figure 4 — Experimental evaluation of designs. **(a)** Fluorescence at varying target concentrations in the US CDC's SARS-CoV-2 N1 qPCR amplicon. Compared Cas13a guides are from our approach using ADAPT (green); a guide with an active (non-G) PFS at the site of the qPCR probe (dark gray); and 10 randomly selected guides with an active PFS (light gray). All were constrained to the amplicon. **(b)** Fluorescence over time for guides from (a) at target concentration of 10^7 cp/μL. (a) shows final time point (124 minutes). **(c)** Phylogeny within SARS-related CoV species based on ref. 43, and 3 related betacoronaviruses. **(d)** Fluorescence for ADAPT's top-ranked SARS-CoV-2 designs in detecting representative targets of the clades in (c). Rankings reflect ADAPT's predicted performance. Left label indicates target concentration (cp/μL). NC, no template control. In (d–f), colors refer to (c). **(e)** Same as (d), but for SARS-CoV-2–related taxon. Empty column with dashes indicates sequence in the design's amplicon has high ambiguity and was not tested. Clades D and E sometimes have two representative targets in these amplicons; when two target labels were tested with one representative sequence, the same value is plotted for each. **(f)** Same as (d), but for SARS-related CoV species. Clade F_1 is SARS-CoV-1, and F_2 and F_3 are related bat SARS-like CoVs. E requires a separate representative target in only one amplicon. G, H, and I are defined in (c). **(g)** Fluorescence for ADAPT's top-ranked EVB designs in detecting EVB and representative targets for Enterovirus A/C/D (EVA/C/D). Each band is an EVB target having width proportional to the fraction of EVB genomic diversity represented, within the amplicon of ADAPT's design. Under each ADAPT design is a baseline guide ("Entropy") from the site in the amplicon with an active PFS and minimal entropy. **(h)** Fluorescence over time at target concentration of 10^8 cp/μL, for the 4 EVB targets representing the largest fraction of EVB genomic diversity. (g) shows middle time point (59 minutes). In all panels, fluorescence is reference-normalized and background-subtracted (Methods).

than pan-enterovirus qPCR or serology, albeit less than sequencing, would aid surveillance.

We used ADAPT to design species-specific assays for Enterovirus B (EVB), which is widespread[48] and exceptionally diverse, with 63 known types[46]. We found all three of ADAPT's top-ranked designs detect the spectrum of genomic diversity with specificity for EVB, as desired (Fig. 4g). To benchmark the efficacy of ADAPT's approach, we targeted conserved sites—a standard strategy for managing sequence diversity—by designing a guide within each amplicon at the site with minimal Shannon entropy and an active PFS (Methods). The entropy-based strategy fails to detect many targets representative of EVB's genomic diversity (Fig. 4g). By contrast, ADAPT's machine learning-based optimization provides a higher fluorescent signal in nearly all representative targets, enabling a lower limit of detection in about half of them (Fig. 4h and Supplementary Fig. 30a–c). In many design options we tested beyond the top three, the entropy-based strategy is more sensitive than ADAPT's approach; however, in these cases the entropy-based strategy lacks species-specificity and ADAPT's designs are ranked lower according to their predicted activity (Supplementary Fig. 31). Though ADAPT's designs incorporate 1–3 guides and the entropy-based strategy uses one, we tested multiple entropy-based guides in five designs and found they exhibit similar activity at low target concentrations (Supplementary Fig. 30d–f and 32). Our results indicate that machine learning-based optimization designs diagnostics that sensitively detect vast genomic diversity, including with improved sensitivity over a standard conservation-based strategy.

Discussion

We combined a deep neural network with combinatorial optimization to design diagnostics. We applied our approach to CRISPR-based viral diagnostics, generating the largest dataset on diagnostic activity, and learned a model that quantitatively predicts diagnostic readout. Critically, our approach directly integrates viral variation into an objective function to generate designs that are optimally active across viral variants. We implemented our approach by building ADAPT, which runs rapidly at scale, across thousands of viruses, and automatically integrates public viral genome data into the design process.

We applied our machine learning-based design approach and performed a rigorous experimental validation across extensive target variation, testing 69 CRISPR-based diagnostic designs against a total of 290 different targets. We demonstrated that ADAPT's designs (i) exhibit significantly higher fluorescence at low target concentrations than designs based on standard design rules for SARS-CoV-2; (ii) are sensitive and specific down to the lineage-level across multiple closely-related taxa; and (iii) delineate a diverse species with a lower limit of detection, across the spectrum of its genomic diversity, than a state-of-the-art design strategy focused on sequence conservation.

In January, 2020, days after SARS-CoV-2 was first sequenced, we applied an early version of ADAPT to design a CRISPR-based SARS-CoV-2 diagnostic assay, and shortly after assays for 66 other viruses relevant to the COVID-19 pandemic[49]. Though designed from only the 20 genomes available at the time[50], we predict the SARS-CoV-2 assay to detect 99.8% of the ∼180,000 genomes available in mid-November, 2020. Primitive versions of ADAPT were also applied to design CRISPR-based assays for detecting 169 human viruses and influenza subtyping[37], and CRISPR-based diagnostics for Lassa virus[51].

We envision running ADAPT regularly for thousands of viruses. This will provide a resource of broadly-effective assays in advance of an outbreak; assays for many strains could be proactively validated. Continually running ADAPT will also provide assay designs that account for viral variation over time and best reflect the latest known variation.

Several components of our methods can be applied to other problems. These include our CRISPR-Cas13a neural network and off-target querying algorithm, which could benefit other tasks that involve designs for CRISPR-Cas13a or short RNA-RNA binding where high specificity is critical.

More broadly, our machine learning-based optimization framework is broadly applicable to detecting and responding to viral threats. Though we trained a deep learning model on a CRISPR-Cas13a dataset, one could train similar models for other diagnostic technologies and use them through ADAPT; indeed, ADAPT flexibly accommodates other machine-learned models. A notable example is RT-qPCR, which remains widely-used for nucleic acid diagnostics; COVID-19 qPCR assays exhibit high variability in their reported sensitivities[52] and many target regions that have acquired mutations[29,30,53], which suggests room for design improvement. ADAPT could design highly specific qPCR assays, including primers and probes, with maximal amplification efficiency over genomic variation, both for broad diagnostics and lineage typing. The design would be accomplished most effectively with a learned model for qPCR, but would also be possible using a model proxy based on existing conventionally-considered hybridization criteria.

Beyond viral diagnostics, other efforts require designing maximally active sequences from viral genome data. For example, genomic variation impacts the efficacy of sequence-based siRNA[54] and antibody[55] therapeutics. CRISPR-based antiviral development also requires deep consideration of sequence diversity, guide activity, and specificity[56]. These uses fit directly into ADAPT's framework. ADAPT's framework could also benefit sequence-based vaccine selection[57,58]. Comparative genomic analyses demonstrate a considerable potential to improve vaccine antigen candidates for pathogens with high strain diversity, and comparative-genomics-informed design provides putative vaccine antigens with greater coverage than existing ones[59]. With appropriate models, our model-based optimization approach could further improve upon comparative-genomics-informed design by rapidly designing antigens that yield maximal predicted antibody titers and T cell responses over strain diversity.

Our approach, together with the introduction of ADAPT, improves the development speed and efficacy of viral diagnostics, and has the potential to do so for sequence-based therapies and vaccines.

Acknowledgements

We thank Yaron Singer, Remy Tuyeras, and Daniel Kassler for fruitful discussions and pointers. This project was funded by DARPA grant D18AC00006, HHMI, and the AWS Diagnostic Development Initiative. N.J.H. was funded by the Landry Cancer Biology Consortium Fellowship and NIH/NIGMS grant T32 GM008313. M.M. was funded by NSF grants CCF-1535795 and CCF-1563710.

Author contributions

H.C.M. initiated and led the study, and developed the algorithms and models in ADAPT, with advice from C.M., M.M., and P.C.S. H.C.M. and P.P.P. implemented ADAPT. N.J.H. designed the Cas13a guide-target library, tested it experimentally with C.M.A. and C.M., and analyzed its data with H.C.M. N.L.W., H.C.M., Y.B.Z., and P.P.P designed experiments evaluating ADAPT's designs and analyzed results. N.L.W. performed those experiments with insight from P.C.B. and J.W. L.R. and D.K.Y. contributed to computational methods development and analyses. H.C.M. wrote the paper with input from all authors.

Competing interests

H.C.M., N.J.H., C.M., and P.C.S. are co-inventors on a patent application filed by the Broad Institute related to work in this manuscript. N.J.H. is a consultant to Constellation Pharmaceuticals. P.C.B. is a consultant to and equity holder in 10X Genomics, GALT, Celsius Therapeutics, and Next Generation Diagnostics. P.C.S. is a co-founder of and consultant to Sherlock Biosciences and a Board Member of Danaher Corporation, and holds equity in the companies.

Methods

ADAPT

Supplementary Notes describe ADAPT's algorithms, data structures, and implementation details. Supplementary Note 1 defines and solves objective functions. Supplementary Note 2 describes how ADAPT enforces specificity. Supplementary Note 3 describes how ADAPT searches for genomic regions to target and links with sequence databases.

Introductory analyses

To illustrate viral database growth, we charted the growth in the number of viral genomes and their unique 31-mers over time (Supplementary Fig. 1). We first curated a list of viral species from NCBI[60] known to infect humans (November 2019). For each, we took all NCBI genome neighbors[28] (influenza sequences from the Influenza Virus resource[61]), which represent near-complete or complete genomes. To assign a date for each, we used the GenBank entry creation date rather than sample collection date for several reasons, including that this date more directly represents our focus in the analysis (when the sequence becomes present in the database) and that every entry on GenBank contains a value for this field. To control for some viruses having multiple segments (and thus sequences), we only used counts for one segment for each species, namely the segment that has the most number of sequences.

We used influenza A virus subtyping as an example to demonstrate the effect of evolution on diagnostics (Fig. 1a and Supplementary Fig. 2). We selected the most conserved k-mers—representing probe or guide sequences—from the sequences available at different years. Here, for simplicity, we ignored all other constraints, such as detection activity and specificity (the latter of which is critical for subtyping), which would further degrade the temporal performance of the selected k-mers. In particular, for each design year Y, we selected the 15 non-overlapping 30-mers found in the largest number of sequences taken from the two most recent years ($Y-1$ and Y). We then measured the fraction of sequences in subsequent test years (Y, $Y+1$, ...) that exactly contain each of these k-mers. We performed the design strategy over 10 resamplings of the sequences and use the mean fraction. We repeated this 4 times: for segment 4 (HA) sequences of H1 and H3 subtypes, and segment 6 (NA) sequences of N1 and N2 subtypes.

To visualize mutations accumulating on a genome during the course of an outbreak (Fig. 1b), we used complete SARS-CoV-2 genomes from GISAID[50]. We called variants in all genomes, through November 1, 2020, against the reference genome '`hCoV-19/Wuhan/IVDC-HB-01/2019`' (GISAID accession '`EPI_ISL_402119`'). For every date d between February 1, 2020 and November 1, 2020, spaced apart by one month, at every position we calculated the fraction of all genomes collected up to d that have a variant against the reference. We called all variants present between 0.1% and 1% frequency on some d as "low" frequency and variants at $\geq 1\%$ frequency on some d as "high" frequency. We ignored all variants present at $\geq 1\%$ frequency on the initial d (ancestral) or that were both low frequency on the initial d and stayed low frequency by the final d—i..e, we kept the variants that transitioned to low or high frequency by the final d. We show the d when the variant first becomes called as low (light purple) or high (dark purple) frequency. If a variant transitions both to low and then to high frequency by the final d, we only show it for the d when it becomes high frequency.

Cas13a library design and testing

We designed a collection of CRISPR-Cas13a crRNA guides and target molecules to evaluate guide-target activity, focusing on assessing likely-active guide-target pairs. First, we designed a target (the *wildtype* target) that is 865 nt long (design details below). We then created 94 guides (namely, the 28 nt spacers) tiling this target (Fig. 2a and Supplementary Fig. 5a). The tiling scheme is such that there are 30 nt blocks with 4 guides overlapping, in which the starts of the 3 guides, from the start of the most 5′ guide, are 4 nt, 13 nt, and 23 nt. Of the 94 guides, 87 are considered to be experimental, 3 to be negative controls, and 4 to be positive controls. We created 229 unique target sequences: 1 of them is the wildtype (effectively, a positive control), 225 are experimental (mismatches and varying PFS alleles against the guides), and 3 are negative controls. All guides exactly match the wildtype target and should detect this, except the 3 negative control guides, which are not intended to detect any targets except one of the 3 negative control targets each. The 4 positive control guides target 4 30-nt regions with a perfectly complementary sequence and non-`G` PFS that are held constant across all targets, with the exception of the 3 negative control targets. Across the experimental targets, the mismatches profile varying choices of positions and alleles against the guide. For the experimental targets, we introduced single mismatches evenly spaced every 30 nt along the experimental region such that every guide targeting this region has either a single mismatch or an altered +1 or +2 PFS; we created a total of 45 such targets to probe all 3 possible base mismatches and 15/30 of the possible phasings. In the remainder of the experimental targets, we generated targets with 2, 3, or 4 mismatches per 30 nt block with respect to the guide RNA in phase with the block. Mismatch positions were randomly chosen to uniformly sample (or, when possible, exhaustively enumerate) average mismatch spacing and average mismatch distance to the center of the spacer. The 87 experimental guides may detect up to 226 unique target sequences (the wildtype and 225 experimental targets), providing 19,662 experimental guide-target pairs.

To construct the wildtype target sequence, we aimed to produce a composition spanning viral genomic sequence diversity. In particular, first we took a dataset of genomes from human-infecting viral species[62], constructed a vector of dinucleotide frequencies for each species, and performed principal component analysis of the species from these vectors. For each 30 nt block of the wildtype target, we selected a point from the space of the first 3 principal components (uniformly at random), reconstructed a corresponding vector of dinucleotide frequencies (i.e., transformed the point back to the original space), and then iteratively selected every next nucleotide in the block according to the distribution of dinucleotides. In positions that would serve as a PFS site for a guide, we disallowed `G`, and proportionately adjusted upwards the probability of choosing a `G` in non-PFS positions to maintain the total dinucleotide frequency in accordance with the randomly selected distribution (mismatches in experimental targets can still introduce a `G` PFS).

We synthesized the targets as DNA, in vitro transcribed them to RNA, and synthesized the crRNAs as RNA. On all crRNAs, we used the same direct repeat (`GAUUUAGACUACCCCAAAAACGAAGGGGACUAAAAC`). To determine a reasonable concentration for measuring fluorescence over time points, we tested 8 concentrations of 2 targets and 2 guides in a pilot experiment (Supplementary Fig. 6a) and proceeded with 6.25×10^9 cp/µL. We tested the library using CARMEN, a droplet-based Cas13a system; we followed the methodology described in ref. 37, which also contains the protocol. Briefly, a guide-target pair is enclosed in a droplet, together with the Cas13a enzyme, that may result in a detection reaction and thus fluorescence. We took an image of each location on each chip roughly every 20 minutes to measure this fluorescence. To alleviate the presence of microdroplets in this experiment (i.e., an irregular pairing of target and guide; about 1/3 of the droplets), we trained and applied a convolutional neural network on hand-labeled

data to identify and remove these.

Quantifying activity

In our Cas13a detection experiments, a fluorescent reporter is cleaved over time and its cleavage follows first-order kinetics:

$$\frac{d[R]}{dt} = -\frac{k_{cat}}{K_M}[E][R]$$
$$\implies [R] = [R]_0 e^{-\frac{k_{cat}}{K_M}[E]t}$$

where $[R]$ is the concentration of the not-yet-cleaved reporter, $[E]$ is the concentration of the Cas13a guide-target complex, $\frac{k_{cat}}{K_M}$ is the catalytic efficiency of the particular guide-target complex, and t is time. The fluorescence measurements that we make, y, are proportional to the quantity of cleaved reporter at some time point:

$$y \propto [R]_0 - [R].$$

Therefore, for each guide-target complex we fit a curve of the form

$$y = C(1 - e^{-kt}) + B.$$

Here, C and B represent the saturation point and background fluorescence, respectively. k represents the rate at which the reporter is cleaved, and it is proportional to the catalytic efficiency of the particular guide-target complex:

$$k = \frac{k_{cat}}{K_M}[E].$$

This relationship is validated by the linear relationship between k and $[E]$ (Supplementary Fig. 6a) when we vary the concentration of target (the limiting component of the complex). In producing our dataset, we held $[E]$ constant. We used $\log_{10}(k)$ as our measurement of guide activity (Fig. 2 and Supplementary Fig. 6a,b). Intuitively, each step-increase in $\log_{10}(k)$ corresponds to a fold-decrease in the half-life of the reporter in the reaction.

Our experimental data incorporates multiple droplets for each guide-target pair (Supplementary Fig. 7a). Each droplet represents one technical replicate of a particular guide-target pair. Thus, we have fluorescence values for each replicate at different time points, and in practice we compute the activity $\log_{10}(k)$ for each replicate.

We curated the data to obtain a final dataset. Namely, we discarded data from two guides that showed no activity between them and any targets, owing to low concentrations in their synthesis. We also did not use data from positive or negative control guides, nor from the negative control target. Our final dataset contains 19,209 unique guide-target pairs (Supplementary Fig. 5b,c), counting 20 nt of sequence context around each protospacer in the target (18,253 unique pairs when not counting context).

Most guide-target pairs show activity (Supplementary Fig. 17d), as expected. At small values of k on a limited time scale (t up to ∼120 minutes), we do not observe reporter activation (Supplementary Fig. 6b). Moreover, the curve becomes approximately linear (first order Maclaurin expansion: $y \approx Ckt + B$). At such values of k, we cannot estimate both C and k together; intuitively, this is because there is too little detectable signal. Therefore, there is a cutoff at which we can estimate k; we labeled activities at $\log(k) > -4$ as active, and the others as inactive. This phenomenon also implies that at smaller values of k, including ones we label as active, activity estimates might be less reliable.

Predicting detection activity

Measurement error

To account for measurement error, we sampled, with replacement, 10 technical replicate measurements of activity for each guide-target pair (Supplementary Fig. 7a). We used this strategy to ensure that, although there are differing numbers of replicates per guide-target pair, each pair would be represented in the dataset with the same number of replicates. There are $19{,}209 \times 10 = 192{,}090$ points in total in our dataset that we use for training and testing. When plotting regression results with guide-target pairs (Fig. 2e and Supplementary Fig. 15, 12a), we use each point to represent a replicate of a guide-target pair; replicates of the same pair yield the same predicted activity but different true activities owing to measurement error, and thus appear on a horizontal line when the vertical axis is predicted activity.

Model and input descriptions

We approached prediction using a two-step hurdle model, reasoning that (i) separate processes govern whether a guide-target pair is active compared to the level of its activity; and (ii) we could better predict the activity of active pairs if we exclude the inactive pairs from a regression. We developed a classifier to decide whether a pair is inactive or active, and a regression model to predict the activity of only active pairs.

We explored multiple models for classification (Fig. 2c and Supplementary Fig. 8a), each with a space of hyperparameters:

- L1 logistic regression: regularization strength (logarithmic in $[10^{-4}, 10^4]$)
- L2 logistic regression: regularization strength (logarithmic in $[10^{-4}, 10^4]$)
- L1+L2 logistic regression (elastic net): regularization strength (logarithmic in $[10^{-4}, 10^4]$), L1/L2 mixing ratio ($1.0 - 2^x + 2^{-5}$ for x uniform in $[-5, 0]$)
- Gradient-boosted trees (GBT): learning rate (logarithmic in $[10^{-2}, 1]$), number of trees (logarithmic in $[1, 2^8]$, integral), minimum number of samples for splitting a node (logarithmic in $[2, 2^3]$, integral), minimum number of samples at a leaf node (logarithmic in $[1, 2^2]$, integral), maximum depth of a tree (logarithmic in $[2, 2^3]$, integral), number of features to consider when splitting a node (for n features, chosen uniform among considering all, $0.1n$, $\sqrt{n}$, and $\log_2 n$)
- Random forest (RF): number of trees (logarithmic in $[1, 2^8]$, integral), minimum number of samples for splitting a node (logarithmic in $[2, 2^3]$, integral), minimum number of samples at a leaf node (logarithmic in $[1, 2^2]$, integral), maximum depth of a tree (chosen uniformly among not restricting the depth or restricting the depth to a value picked logarithmically from $[2, 2^4]$ and made integral), number of features to consider when splitting a node (for n features, chosen uniform among considering all, $0.1n$, $\sqrt{n}$, and $\log_2 n$)
- Support vector machine (SVM; linear): regularization strength (logarithmic in $[10^{-8}, 10^8]$), penalty type (chosen uniformly among L1 and L2)
- Multilayer perceptron (MLP): number of layers excluding the output layer (uniform in $[1, 3]$), dimensionality of each layer excluding the output layer (each chosen uniformly in $[4, 127]$), dropout rate in front of each layer (uniform in $[0, 0.5]$), activation function (chosen uniformly among ReLU and ELU), batch size always 16
- Long short-term memory recurrent neural network (LSTM): dimensionality of the output vector (logarithmic in $[2, 2^8]$, integral), whether to be bidirectional (chosen uniformly among unidirectional and bidirectional), dropout rate in front of the final layer (uniform in $[0, 0.5]$), whether to perform an embedding of the one-hot encoded nucleotides and the dimensionality

if so (chosen with 1/3 chance to not perform an embedding, and with 2/3 chance to perform an embedding with dimensionality chosen uniformly in $[1, 8]$), batch size is always 16

- Convolutional neural network (CNN): number of parallel convolutional filters and their widths (chosen uniformly among not having a convolutional layer, 1 filter of width 1, 1 filter of width 2, 1 filter of width 3, 1 filter of width 4, 2 filters of widths $\{1, 2\}$, 3 filters of widths $\{1, 2, 3\}$, and 4 filters of widths $\{1, 2, 3, 4\}$), convolutional dimension (uniform in $[10, 249]$), pooling layer width (uniform in $[1, 3]$), pooling layer computation (chosen uniformly among maximum, average, and both), number of parallel locally connected layers and their widths (chosen uniformly among not having a locally connected layer, 1 filter of width 1, 1 filter of width 2, and 2 filters of widths $\{1, 2\}$), locally connected filter dimension (uniform in $[1, 4]$), number of fully connected layers and their dimensions (chosen uniformly among 1 layer with dimension uniform in $[25, 74]$ and 2 layers each with dimension uniform in $[25, 74]$), whether to perform batch normalization in between the convolutional and pooling layers (uniform among yes and no), activation function (chosen uniformly among ReLU and ELU), dropout rate in front of the fully connected layers (uniform in $[0, 0.5]$), L2 regularization coefficient (lognormal with mean $\mu = -13, \sigma = 4$), batch size (uniform in $[32, 255]$), learning rate (logarithmic in $[10^{-6}, 10^{-1}]$)

Similarly, for regression we explored multiple models (Supplementary Fig. 8b,c), each with a space of hyperparameters:

- L1 linear regression: regularization strength (logarithmic in $[10^{-8}, 10^{8}]$)
- L2 linear regression: regularization strength (logarithmic in $[10^{-8}, 10^{8}]$)
- L1+L2 linear regression (elastic net): regularization strength (logarithmic in $[10^{-8}, 10^{8}]$), L1/L2 mixing ratio ($1.0 - 2^{x} + 2^{-5}$ for x uniform in $[-5, 0]$)
- Gradient-boosted trees (GBT): same hyperparameter space as for classification
- Random forest (RF): same hyperparameter space as for classification
- Multilayer perceptron (MLP): same hyperparameter space as for classification
- Long short-term memory recurrent neural network (LSTM): same hyperparameter space as for classification
- Convolutional neural network (CNN): same hyperparameter space as for classification

Model selection and evaluation describes the search process.

When training and testing the models, we used 28 nt guide and target sequence, and include 10 nt of context in the target sequence on each side of the protospacer. We tested the following different inputs:

- 'One-hot (1D)': vector containing 4 bits to encode the nucleotide at each target position and 4 bits similarly for each guide position; with a 28 nt guide and 10 nt of context in the target around the protospacer, there are $(10 + 28 + 10 + 28) \times 4 = 304$ bits
- 'One-hot MM': similar to 'One-hot (1D)' except explicitly encoding mismatches between the guide and target—i.e., vector containing 4 bits to encode the nucleotide at each target position and 4 bits, at each guide position, encoding whether there is a mismatch (if not, all 0) and, if so, the guide allele; same length as 'One-hot (1D)'
- 'Handcrafted': features are count of each nucleotide in the guide, count of each dinucleotide in the guide, GC count in the guide, total number of mismatches between the guide and target sequence, and a one-hot encoding of the 2-nt PFS (coupling the 2 nucleotides); the number

of features are $4 + 16 + 1 + 1 + 16 = 38$

- 'One-hot MM + Handcrafted': concatenation of features from 'One-hot MM' and 'Handcrafted', except removing from 'One-hot MM' the bits encoding the 2-nt PFS because these are included in 'Handcrafted'

We used these inputs for all models except the LSTM and CNN. For these two models, which can capture and extract spatial relationships in the input, we used an alternative input (labeled 'One-hot (2D)' in figures). Here, the input dimensionality is $(48, 8)$ and consists of a concatenated one-hot encoding of the target and guide sequence. Namely, each element x_i ($i \in \{1 \dots 48\}$) is a vector $[x_{i,t}, x_{i,g}]$. Target context corresponds to $i \in \{1 \dots 10\}$ (5′ end) and $i \in \{39 \dots 48\}$ (3′ end); for these i, $x_{i,t}$ is a one-hot encoding of the target sequence and $x_{i,g}$ is all 0. The guide binds to the target at $i \in \{11 \dots 38\}$ and, for these i, $x_{i,t}$ is a one-hot encoding of the target sequence protospacer at position $i - 10$ of where the guide is designed to bind, while $x_{i,g}$ is a one-hot encoding of the guide at position $i - 10$.

We evaluated all models, except the MLP, LSTM, and CNN, in scikit-learn 0.22[63]. We implemented and evaluated the MLP, LSTM, and CNN models in TensorFlow 2.1.0[64].

For the MLP, LSTM, and CNN models, we used binary cross-entropy as the loss function for classification and mean squared error for regression. For these 3 models, we used the Adam optimizer[65] and performed early stopping during training (maximum of 1,000 epochs) with a held-out portion of the training data. Additionally, for the CNN we regularized the weights (L2). When training all classification models, we weighted the active and inactive classes equally.

Data splits and test set

In evaluating our models, we must determine folds of the data and pick a held-out test set. One challenge is that, in our design, guides overlap according the position against which they were designed along the wildtype target. Although effects on activity might be position-dependent within the guide, this overlap can cause guides to have similar sequence composition or to be in regions of the target sequence with similar structure. To remove this possibility of leakage between a data split, after making a split of X into X_{train} and X_{test}, we remove all guide-target pairs from X_{test} for which the guide has any overlap, in target sequence they are designed to detect, with a guide in X_{train}. We perform this strategy during all cross-validated analyses. We also use it to choose a test set that we hold out from all analyses and use only for evaluating the final CNNs. This test set consists of 30% of all guides (counted before removing overlaps between X_{train} and X_{test}), taken from the far 3′ end of the target.

Model selection and evaluation

We performed nested cross-validation to select models—both for classification and regression—and evaluate our selection of them (Fig. 2c and Supplementary Fig. 8). We used 5 outer folds of the data. For each outer fold, we searched for hyperparameters using a cross-validated (5 inner folds) random search over the space defined in Model and input descriptions; we scored using the mean auROC (classification) or Spearman correlation (regression) over the inner folds. In each random search, we used 100 hyperparameter choices for all models, except for the LSTM and CNN models (50), which we found slower to train.

The CNN models outperformed others in the above analysis, so we selected a final CNN model for classification and another for regression. For each of classification and regression, we performed a random search across 5 folds of the data using 200 random samples. We selected the model with the

highest auROC (classification) or Spearman correlation (regression) averaged over the folds. Our evaluations of these two models used the held-out test set.

Incorporating into ADAPT

We integrated the CNN models into ADAPT. First, we set the decision threshold on the classifier's output to be 0.577467. Precision matters greatly in our context because we would like confidence that designs determined to be active are indeed active. We chose the threshold, via cross-validation, to achieve a desired precision of 0.975. In particular, we took 5 folds of our data (excluding test data) and, for each fold, we calculated the threshold that achieves a precision of 0.975 on the validation data. Our decision threshold is the mean across the folds.

We then defined a piecewise function, incorporating the classification and regression models, as:

$$d(p,s) = \begin{cases} 0 & \text{if } C(p,s) < t \\ \max(0, r + R(p,s)) & \text{else} \end{cases}$$

where $d(p,s)$ is the predicted detection activity between a probe p and target sequence s (s includes 10 nt of context). $C(p,s)$ is the output of the classifier, t is the classification decision threshold, and $R(p,s)$ is the output of the regression model. r is a shift that we add to regression outputs to ensure $d(p,s)$ is non-negative; though a nice property, it is not strictly needed as long as we constrain the ground set as described in Supplementary Note 1a. The choice of r should depend on the range of activity values in the dataset; here, $r = 4$.

ADAPT analyses

Comparing algorithms for submodular maximization

To compare the canonical greedy algorithm for constrained monotone submodular maximization[33] with the fast randomized combinatorial algorithm[32] (Supplementary Fig. 3), we ran ADAPT 5 times under each choice of parameter settings and species and, for each run, considered the mean final objective value taken across the best 5 design options. We used the arguments '`-pm 3 -pp 0.9 --primer-gc-content-bounds 0.3 0.7 --max-primers-at-site 10 -gl 28 --max-target-len 250`' with our Cas13a activity model. We used the default objective function in ADAPT: $4 + A - 0.5P - 0.25L$, where A is the expected activity of the guide set, P is the number of primers, and L is the target length.

Benchmarking comprehensiveness

To benchmark comprehensiveness (Fig. 1d,e and Supplementary Fig. 4), we ran ADAPT with three approaches. In all approaches, we decided that a probe detects a target sequence if and only if they are within 1 mismatch, counting G-U wobble pairs as matches; used a sliding window of 200 nt and a probe length of 30 nt; and used, as input, all NCBI genome neighbors[28] for each species. In the first approach, we used ADAPT's `design_naively.py` program to select a single probe within each window via two strategies: (1) the consensus probe, computed at every site within the window, that detects the most number of sequences ('consensus'); and (2) the most common probe sequence, determined at every site within the window, that detects the most number of sequences ('mode'). In the second approach, we maximized expected activity using ADAPT across the target sequences with different numbers of probes (hard constraints) using a penalty strength of 0 (i.e., no soft constraint). Here, we defined the activity to be binary: 1 for detection, and 0 otherwise; this has the property that expected activity is equivalent to the fraction of sequences detected. In the third

approach, we use the objective function in ADAPT that minimizes the number of probes subject to constraints on the fraction of sequences detected (specified via '`-gp`'; 0.9, 0.95, and 0.99).

Evaluating dispersion and generalization

We evaluated the dispersion, owing to randomness and sampling, in ADAPT's designs (Supplementary Fig. 23). In all cases, we used all NCBI genome neighbors[28] for each species and used the following arguments with ADAPT: '`--obj maximize-activity --soft-guide-constraint 1 --hard-guide-constraint 5 --penalty-strength 0.25 -gl 28 -pl 30 -pm 3 -pp 0.98 --primer-gc-content-bounds 0.35 0.65 --max-primers-at-site 10 --max-target-length 500 --obj-fn-weights 0.50 0.25`', with a cluster threshold such that there is only 1 cluster, and used our Cas13a activity model. We ran ADAPT in two ways: 20 times without changing the input (output differences are owing to randomness) and 20 times with resampled input (output differences are owing both to randomness and sampling of the input sequences). Then, we measured dispersion by treating the 5 highest-ranked design options from each run as a set and computing pairwise Jaccard similarities across the 20 runs. This computation requires us to evaluate overlap between two sets: in one comparison, we consider a design option x to be in another set if x is present exactly in that other set (same primers and probes) and, in the other comparison, we consider a design option x to be in another set if that other set has some design option with both endpoints within 40 nt of x's endpoints.

To evaluate the generalization of ADAPT's designs (Fig. 3b), we performed cross-validation via repeated random subsampling. For each species, we took all NCBI genome neighbors[28] and, 20 times, randomly selected 80% of them to use as input for design and the remaining 20% to test against. For each split, we used the same arguments with ADAPT as when evaluating dispersion: '`--obj maximize-activity --soft-guide-constraint 1 --hard-guide-constraint 5 --penalty-strength 0.25 -gl 28 -pl 30 -pm 3 -pp 0.98 --primer-gc-content-bounds 0.35 0.65 --max-primers-at-site 10 --max-target-length 500 --obj-fn-weights 0.50 0.25`', with a cluster threshold such that there is only 1 cluster, and used our Cas13a activity model. When computing the fraction of sequences in the test set that are detected, we required the sequence to be detected by a primer on the 5′ and 3′ ends of a region (within 3 mismatches) and a probe (here, guide) to detect the region; we used the `analyze_coverage.py` program in ADAPT for this computation. We labeled detection of a sequence as "active" if a guide in the guide set is decided by our Cas13a classification model to be active against the target. We labeled the detection as "highly active" if a guide in the guide set is both decided to be active by the Cas13a classification model and its predicted activity, according to the Cas13a regression model, is ≥ 2.7198637 (4 added to the output of the model, -1.2801363). This threshold corresponds to the top 25% of predicted values on the subset of our held-out test set that is classified as active.

Benchmarking trie-based specificity queries

We benchmarked the approach described in Supplementary Note 2d against a single, large trie (Supplementary Fig. 20). For this, we sampled 1.28% of all 28-mers from 570 viral species ($\sim$78.7 million 28-mers in total), and built data structures indexing these. We then randomly selected 100 species (here, counting each segment of a segmented genome as a separate species), and queried 100 randomly selected 28-mers from each of these for hits against the other 569 species. We performed this for varying choices of mismatches. We used the same approach to generate results in Supplementary Fig. 18, there comparing queries with and without tolerance of G-U base pairing.

Benchmarking runtime improvement with memoization

We benchmarked the effect on runtime of memoizing repeated computations (Supplementary Fig. 21), as described in Supplementary Note 3a. We used all genome neighbors from NCBI's viral genomes resource[28] as input for each of the three species tested. To run ADAPT while memoizing computations, we used the arguments: '`--obj maximize-activity --soft-guide-constraint 1 --hard-guide-constraint 5 --penalty-strength 0.25 --maximization-algorithm random-greedy -pm 3 -pp 0.9 --primer-gc-content-bounds 0.3 0.7 --max-primers-at-site 10 -gl 28 --max-target-len 250 --best-n-targets 10 --id-m 4 --id-frac 0.01 --id-method shard`'. We also used our Cas13a predictive model and enforced specificity against all other species within each species's family. To perform runs without memoizing computations, we did the same except added the argument '`--do-not-memoize-guide-computations`', which skips all memoization steps during ADAPT's search (except for calls to the predictive model).

Design of broadly-effective SARS-related coronavirus assays in 2018 and their evaluation

To evaluate the efficacy of species-level assays on a novel virus (Supplementary Fig. 25), we focused on SARS-related coronavirus. We simulated the 2018 design of assays for detecting the SARS-related CoV species, roughly a year before the initial detection of SARS-CoV-2. In particular, we used as input all genome neighbors from NCBI's viral genomes resource[28] for SARS-related CoV that were released on or before December 31, 2018 (there are 311 genomes). For ADAPT's designs, we used the same parameters used for the vertebrate-infecting viral species designs (Designs across vertebrate-infecting species), except tolerating up to 1 mismatch between primer and target sequences; the specificity criteria were also the same as in those designs.

In 2018, SARS-related CoV was biased toward SARS-CoV-1 genomes (owing to SARS outbreak sequencing) relative to viruses sampled from animals. To alleviate this overrepresentation, we also produced designs using ADAPT in which the input downsampled SARS-CoV-1 to a single genome (Supplementary Fig. 25c,d). We used the RefSeq, GenBank accession AY274119, as that genome.

To determine the performance of these designs on SARS-CoV-2, we used the 184,197 complete genomes (low-quality removed) available on GISAID[50] as of November 12, 2020. For an assay to be predicted to detect a target sequence (Supplementary Fig. 25b,d), we require that (i) primers on both ends are within 3 mismatches of the target sequence; and (ii) a guide in the guide set is classified by our Cas13a predictive model as active. We used this criteria both for evaluating detection of SARS-CoV-2 and of the design's input.

Designs across vertebrate-infecting species

We found all viral species in NCBI's viral genomes resource[28] that have a vertebrate as a host, as of April, 2020. These are species ratified by the International Committee on Taxonomy of Viruses (ICTV)[66]. We added to this list others that may have been incorrectly labeled, as well as influenza viruses, which are separate from the resource. There were 1,933 species in total and we used ADAPT to design primers and Cas13a guides to detect them. As input, we used all genome neighbors from NCBI's viral genomes resource[28] (influenza database for influenza species[61]). We ran ADAPT in May–June, 2020, and thus the input incorporates sequences available through those dates.

We constrained primers to have a length and GC content that are recommended for use with RPA[67],

and thus are suitable for use with SHERLOCK[1] detection. We enforced specificity at the species-level within each family. That is, we required that the guides for each species not have off-target hits to sequence from any other species in its same family. Restricting our specificity queries to one family at a time reduces ADAPT's memory usage and runtime.

We used the following arguments when running ADAPT to maximize expected activity:

- Initial clustering: clustered with a maximum distance of 30% (`--cluster-threshold 0.3`)
- Primers and amplicons: primer length of 30, primers must have GC content between 35% and 65%, at most 10 primers at a site[1], up to 3 mismatches between primers and target sequence for hybridization, primers must hybridize to $\geq$ 98% of sequences, and length of a targeted genome region (amplicon) must be $\leq$ 250 nt (`-pl 30 --primer-gc-content-bounds 0.35 0.65 --max-primers-at-site 10 -pm 3 -pp 0.98 --max-target-length 250`)
- Guides: Cas13a guide length of 28 nt, together with our Cas13a predictive model (`-gl 28 --predict-activity-model-path models/classify/model-51373185 models/regress/model-f8b6fd5d`)
- Guide activity objective: soft constraint of 1 guide, hard constraint of 5 guides, guide penalty (λ) of 0.25, using the randomized greedy algorithm (`--obj maximize-activity --soft-guide-constraint 1 --hard-guide-constraint 5 --penalty-strength 0.25 --maximization-algorithm random-greedy`)
- Specificity: query up to 4 mismatches counting G-U pairs as matches, calling a guide non-specific if it hits $\geq$ 1% of sequences in another taxon (`--id-method shard --id-m 4 --id-frac 0.01`)
- Objective function and search: weights $\lambda_A = 0.5$ and $\lambda_L = 0.25$ in the objective function (defined in Supplementary Note 3a) and finding the best 20 design options (`--obj-fn-weights 0.5 0.25 --best-n-targets 20`)

We made some species-specific adjustments. For influenza A and dengue viruses, two especially diverse species, we decreased the number of tolerated primer mismatches to 2 and allowed at most 5 primers at a site (`-pm 2 --max-primers-at-site 5`); while these further constrain the design, they decrease runtime. For Norwalk virus and Rhinovirus C, we relaxed the number of primers at a site and the maximum region length to identify designs (`--max-primers-at-site 20 --max-target-length 500`). For Cervid alphaherpesvirus 2, which has a short genome, we changed the GC-content bounds on primers to be 20%–80% (`--primer-gc-content-bounds 0.2 0.8`) to allow more potential amplicons. For 42 species, we relaxed specificity constraints to identify designs (list and details in code).

Of the 1,933 species, 7 could not produce a design while maximizing activity and enforcing specificity, even with species-specific adjustments. They are: Bat mastadenovirus, Bovine associated cyclovirus 1, Chiropteran bocaparvovirus 4, Cyclovirus PKgoat21/PAK/2009, Finkel-Biskis-Jinkins murine sarcoma virus, Panine gammaherpesvirus 1, and Squirrel fibroma virus. Each of these 7 species has just one genome sequence and ADAPT could not identify a guide set satisfying specificity constraints; it is possible they are misclassified or have very high genetic similarity to other species. When showing results for this objective, we report on 1,926 species.

In addition to using the above settings, which maximizes activity and enforces specificity, we ran

[1] Although high, this is only an upper bound and is meant to restrict the search space and thus restrict runtime.

ADAPT with 3 other approaches: We minimized the number of guides while enforcing specificity, requiring that guides be predicted to be highly active (as defined in Evaluating dispersion and generalization) in detecting 98% of sequences. We also ran the objectives to maximize activity and minimize guides without enforcing specificity. 67 of the 1,933 species did not yield a design when minimizing the number of guides and enforcing specificity, owing to the constraints with this objective: ADAPT could not identify a guide set that is predicted to be highly active and achieves the desired coverage and specificity.

For species with segmented genomes, we ran ADAPT and produced designs separately for each segment. We then selected the segment whose highest-ranked design option has the best objective value (if multiple clusters, according to the largest cluster). We expect the selected segment to generally be the most conserved one.

In all analyses showing results of the designs (e.g., number of guides, guide activity, and target length), we used the highest-ranked design option output by ADAPT. For the species with more than one cluster, we report the mean across clusters from the highest-ranked design option in each cluster.

Designs for evaluating sensitivity and specificity

ADAPT design parameters

To generate designs with ADAPT for experimental testing, we used the following arguments unless otherwise noted:

- Initial clustering: force a single cluster (`--cluster-threshold 1.0`)
- Primers and amplicons: primer length of 30, primers must have GC content between 35% and 65%, at most 5 primers at a site, up to 1 mismatch between primers and target sequence for hybridization, primers must hybridize to $\geq$ 98% of sequences, and length of a targeted genome region (amplicon) must be $\leq$ 250 nt (`-pl 30 --primer-gc-content-bounds 0.35 0.65 --max-primers-at-site 5 -pm 1 -pp 0.98 --max-target-length 250`)
- Guides: Cas13a guide length of 28 nt, together with our Cas13a predictive model (`-gl 28 --predict-activity-model-path models/classify/model-51373185 models/regress/model-f8b6fd5d`)
- Guide activity objective: soft constraint of 1 guide, hard constraint of 5 guides, guide penalty (λ) of 0.25, using the randomized greedy algorithm (`--obj maximize-activity --soft-guide-constraint 1 --hard-guide-constraint 5 --penalty-strength 0.25 --maximization-algorithm random-greedy`)
- Specificity: query up to 4 mismatches counting G-U pairs as matches, calling a guide non-specific if it hits $\geq$ 1% of sequences in another taxon (`--id-method shard --id-m 4 --id-frac 0.01`)
- Objective function and search: weights $\lambda_A = 0.5$ and $\lambda_L = 0.25$ in the objective function (defined in Supplementary Note 3a) (`--obj-fn-weights 0.5 0.25`)

For SARS-CoV-2 input sequences, we used the 9,054 complete genomes available on GISAID[50] as of April 28, 2020. We also used genomes from GISAID for pangolin SARS-like CoV input sequences (isolates from Guangxi, China and Guandong, China). For all other input sequences—SARS-like CoV isolates RaTG13, ZC45, and ZXC21; other SARS-like CoVs; SARS-CoV-1 (also referred to as SARS-CoV); and other Coronaviridae species—we used all genome neighbors from NCBI from each

species[28].

Generating test target sequences

Experimentally testing design options output by ADAPT also requires generating representative target sequences. We found representative sequences for a design option, using a collection of genomes spanning diversity of a taxon, as follows: (1) We extracted the amplicon (according to provided positions, e.g., from primer sequences), while extending outward to achieve a minimum length (usually 500 nt). (2) We removed sequences that are too short, e.g., owing to gaps in the alignment. (3) We computed pairwise Mash distances[68] and performed hierarchical clustering (average linkage) to achieve a desired number of clusters or a maximum inter-cluster distance. (4) To avoid outliers, we greedily selected (in order of descending size) clusters that include a desired total fraction of sequences, a particular number of targets, or ones representing particular taxa (specifics below). (5) We computed the medoid of each cluster—i.e., the sequence with minimal total distance to all other sequences in the cluster. (6) We used the medoids of each clusters as representative target sequences. The `pick_test_targets.py` program in ADAPT implements the procedure and we used this program.

Baseline distribution of activity

We established a baseline distribution of activity using Cas13a guides, to detect SARS-CoV-2, selected from the genomic regions targeted by the United States Centers for Disease Control and Prevention (US CDC) RT-qPCR assays[69]. In particular, we picked 10 random 28-mers from within the US CDC N1 amplicon that would have a non-G PFS and used these as Cas13a guides, according to the '`hCoV-19/Wuhan/IVDC-HB-01/2019`' genome[50]. We also chose another Cas13a guide at the site of the TaqMan probe. We did the same from the US CDC N2 amplicon. In addition, in the N1 and N2 amplicons, we used ADAPT to design a single guide with maximal activity (ignoring specificity) from within the amplicon. This provides 24 guides in total.

Experimental designs with ADAPT

To evaluate the activity and lineage-level specificity of SARS-CoV-2 designs, we used ADAPT to produce 10 design options for detecting SARS-CoV-2. We increased the specificity in ADAPT to call a guide non-specific if it hits any sequence outside SARS-CoV-2 and also use the greedy maximization to obtain more intuitive outputs because, in this case, we expect only a single Cas13a guide for each design option (`--id-frac 0 --maximization-algorithm greedy`). We enforced specificity to not detect any sequences outside of SARS-CoV-2 from the SARS-related CoV species (including related bat and pangolin coronavirus isolates) and also to not detect sequences from the other 43 species in the Coronaviridae family. Owing to experimental constraints, we tested the highest-ranked 5. We generated targets for each design option against which to test, using the ones representative of SARS-CoV-2; pangolin SARS-like CoVs (isolates from Guangxi, China); bat SARS-like CoV isolates ZC45 and RaTG13; and SARS-CoV-1.

To further evaluate activity and subspecies-comprehensiveness, we used ADAPT to produce 10 design options for detecting the SARS-CoV-2–related taxon. In referring to SARS-CoV-2–related, we use the definition given in Fig. 1b of ref. 44; it encompasses SARS-CoV-2 and several related bat and pangolin SARS-like coronaviruses. To correct for sampling biases, we used 10 sampled SARS-CoV-2 genomes as input so that they make up roughly half of sequences in the SARS-CoV-2–related taxon. We used the same adjusted arguments in ADAPT as used for the SARS-CoV-2 designs (`--id-frac 0 --maximization-algorithm greedy`). We enforced specificity to not detect

any sequences outside of SARS-CoV-2–related from the SARS-related CoV species (including other bat SARS-like coronaviruses) and also to not detect sequences from the other 43 species in the Coronaviridae family. For each design option, we generated targets, and used the ones representative of SARS-CoV-2; pangolin SARS-like CoVs (isolates from Guangxi, China and Guangdong, China); bat SARS-like CoV isolates ZC45, ZXC21, and RaTG13; and SARS-CoV-1. For this experiment, the SARS-CoV-1 target allows us to evaluate specificity, while the others allow us to evaluate activity and subspecies-comprehensiveness.

We used ADAPT to produce 10 design options to detect the SARS-related CoV species, and we used these to evaluate activity, species-comprehensiveness, and specificity. To correct for sampling biases, we used 300 sampled SARS-CoV-2 genomes as input so that they make up roughly half of sequences in the species. We enforced specificity to not detect sequences from the other 43 species in the Coronaviridae family. For each design option, we generated representative targets that encompass SARS-CoV-2, SARS-CoV-1, bat SARS-like CoVs, pangolin SARS-like CoVs, MERS-CoV, Human coronavirus OC43, and Human coronavirus HKU1. There were 8 or 9 representative targets in total for each design option.

To evaluate species-comprehensiveness, we focused on Enterovirus B and used ADAPT to produce 10 design options. Owing to its extensive diversity, we made several adjustments to arguments, which help to increase the space of potential design options (`--primer-gc-content-bounds 0.30 0.70 -pm 4 -pp 0.80 --max-primers-at-site 10 --id-frac 0.10 --penalty-strength 0.15`). We enforced specificity to not detect the 18 other species in the Enterovirus genus. For each design option, we generated representative targets from clusters that encompass at least 90% of all sequences. There were between 1 and 15 targets for each design option (the precise number depends heavily on the location of the design option amplicon in the genome). We additionally tested specificity within the Enterovirus genus by generating a single representative target for each of Enterovirus A, Enterovirus C, and Enterovirus D.

To benchmark ADAPT's designs for Enterovirus B, we created baseline Cas13a guides using an entropy-based approach that identifies conserved sites. For each of ADAPT's design options, we considered the amplicon it targets. Then, we computed the information-theoretic (Shannon) entropy, over alleles, at every site in the amplicon. (We counted an ambiguous base fractionally and a gap as a "base".) We define the average entropy of a 28 nt site to be the mean entropy across its 28 positions. The approach finds the site in the amplicon that has the minimal average entropy and an active (non-`G`) PFS in GenBank accession MK800120. Our entropy-based baseline guide is the sequence from GenBank accession MK800120 at this site. We performed this process in the amplicon from each of ADAPT's designs to generate and test one baseline guide; for 5 of the 10 designs, we generated and tested two baseline guides, where the second was from the site with the second lowest entropy and an active PFS. The approach is implemented in ADAPT's `design_naively.py` program.

We built a positive control into each target. In particular, we added the sequence `5'-CACTATAGGGGCTCTAGCGACTTCTTTAAATAGTGGCTTAAAATAAC-3'` to the 5′ end of each target and included in our tests of every target a guide with protospacer sequence `5'-GCTCTAGCGACTTCTTTAAATAGTGGC-3'`.

Experiments evaluating sensitivity and specificity

Experimental procedure

We largely followed the CARMEN-Cas13 platform[37] for experimentally validating ADAPT's designs, with some key differences. DNA targets were ordered from Integrated DNA Technologies and in vitro transcribed using the HiScribe T7 High Yield RNA Synthesis Kit (New England Biolabs). Transcriptions were performed according to the manufacturer's recommendations with a reaction volume of 20 µL that was incubated overnight at 37 °C. The transcribed RNA products were purified using RNAClean XP beads (Beckman Coulter) and quantified using NanoDrop One (Thermo Scientific). The RNA was serially diluted from 10^{11} to 10^4 cp/µL and used as input into the detection reaction. crRNAs were synthesized by Integrated DNA Technologies, resuspended in nuclease-free water, and diluted to 1 µM for input into the detection reaction. The Cas13 detection reactions were made into two separate mixes for loading onto a 192.24 Dynamic Array IFC for Gene Expression (Fluidigm). The assay mix contained 42.5 nM LwaCas13a, 42.5 nM crRNA, 2× Assay Loading Reagent (Fluidigm), and nuclease-free water. The sample mix contained 1 µL RNAse Inhibitor (New England Biolabs), 1× ROX Reference Dye (Invitrogen), 1× GE Sample Loading Reagent (Fluidigm), 1.95 nM quenched synthetic fluorescent RNA reporter (`FAM/rUrUrUrUrUrUrU/3IABkFQ/`, Integrated DNA Technologies), 9 nM $MgCl_2$ in a nuclease assay buffer (40 mM Tris-HCl, 1 mM DTT pH 7.5). Syringe, Actuation Fluid, Pressure Fluid (Fluidigm), and 4 µL of assay and sample mixtures were loaded into their respective locations on a 192.24 IFC according to the manufacturer's instructions. The IFC was loaded onto the IFC Controller RX (Fluidigm) where the 'Load Mix' script was run. After proper IFC loading, images over a two-hour period were collected using a custom protocol on Fluidigm's Biomark HD.

Displaying experimental results

We plotted reference-normalized background-subtracted fluorescence for guide-target pairs. For a guide-target pair (at some time point t and target concentration), we first computed the reference-normalized value as

$$\text{median}\left(\frac{P_t - P_0}{R_t - R_0}\right)$$

where P_t is the guide signal (FAM) at the time point, P_0 is its background measurement before the reaction, R_t is the reference signal (ROX) at the time point, R_0 is its background measurement, and the median is taken across Fluidigm's replicates. We performed the same calculation for the no template (water) control of the guide, providing a background fluorescence value for the guide at t (when there were multiple technical replicates of such controls, we took the mean value across them). The reference-normalized background-subtracted fluorescence for a guide-target pair is the difference between these two values. Note that, by definition, the no template control ('NC' in figures) has value of 0. In heatmaps showing fluorescence at a fixed time point, we used the middle time point (59 minutes). In kinetic curves that show fluorescence over time (for example, Fig. 4b), we smoothed the value by taking the rolling mean within a window of 2 time points.

When displaying the top-ranked design options from ADAPT (for example, in Fig. 4d–h), we ordered them according to the predicted activity of the Cas13a guides in expectation across the input genomes. ADAPT's ranking incorporates additional factors (Supplementary Note 3a) that reflect amplification potential, and we used ADAPT's objective function to identify the top N design options to test. But we ordered them according to only predicted fluorescent activity because our experimental testing did not involve amplification. When plotting fluorescence for a design that uses more than one guide, we plot the maximum fluorescence across the guides (computed

separately at each target, target concentration, and measurement time point). This is analogous to ADAPT's model for measuring a probe set's activity (Supplementary Note 1a), in which its activity in detecting a target sequence equals that of the best probe in the set for detecting that sequence.

Data availability

Data is available in several repositories:

- The CRISPR-Cas13a library and activity dataset is available at: https://github.com/broadinstitute/adapt-seq-design/tree/82db28/data
- Serialized trained models are available at: https://github.com/broadinstitute/adapt-seq-design/tree/82db28/models/cas13
- Experimentally tested designs are available at: https://github.com/broadinstitute/adapt-designs/tree/783ac9

Code availability

Code is available in several repositories:

- ADAPT is freely available under the MIT license at: https://github.com/broadinstitute/adapt
- Code to replicate the predictive modeling and analyses is available at: https://github.com/broadinstitute/adapt-seq-design
- Code to replicate the designs across the vertebrate-infecting viral species is available at: https://github.com/broadinstitute/adapt-designs-continuous
- Code to replicate the other analyses in this paper is available at: https://github.com/broadinstitute/adapt-analysis

Supplementary Note 1

This note describes two formulations for objective functions implemented in ADAPT. Unless otherwise noted, for designs and analyses in the paper, we use the formulation in Design formulation #1: maximizing expected activity.

1a Design formulation #1: maximizing expected activity

Objective

Let S be an alignment of sequences from species t in a genomic region. We wish to find a set P of probes that maximizes detection activity over these sequences. This objective bears some similarity to the problem of designing PCR primers that cover a maximum number of sequences[16], though solutions to that problem binarize decisions about detection rather than accommodate continuous predictions. As described in Methods, we have a model to predict a measurement of detection activity between one probe p and one sequence $s \in S$, which we represent by $d(p, s)$. (In Fig. 1c, we use $A(P, s)$ to represent this quantity.) While more than one probe in the set P may be able to detect s, we use the best probe against s to measure P's detection activity for s; that is, the predicted detection activity is

$$d(P, s) = \max_{p \in P} d(p, s).$$

One way to consider why taking the maximum is reasonable is that, in practice, we could apply each $p \in P$ in parallel reactions to a sample even if only one p works well for the particular target in that sample. We also define $d(P, s)$ when P is the empty set to be the lowest value in the range of $d(p, s)$, indicating no detection (in practice, 0).

We represent the predicted detection activity by P, against all sequences in S, with the function $F(P)$. $F(P)$ is the expected value of $d(P, s)$ taken over all the $s \in S$; the weights w_s for each s can reflect a prior probability on applying the detection in practice to targeting a sequence like s (e.g., based on the genome's date or geographic location). Thus, we define

$$F(P) = \sum_{s \in S} w_s \cdot d(P, s).$$

We currently set a uniform prior over targeting the genomes in S, with the effect being that all w_s are equal.

We must introduce penalties for the number of probes. Striving for a small number of probes is important because (a) if used in separate reactions, this adds time and labor; (b) if multiplexed in one reaction, there is generally competition in binding to a target, and having more in a reaction may reduce the resulting detection signal; and (c) they require time and money to synthesize, and to experimentally validate. For this penalty, we use a soft constraint m_p and hard constraint $\overline{m_p}$ on the number of probes, with $\overline{m_p} \geq m_p$. We wish to solve

$$\max_P \{F(P) - \lambda \cdot \max(0, |P| - m_p) : |P| \leq \overline{m_p}\}$$

where $\lambda > 0$ serves as a weight on the penalty. λ reflects a tolerance for higher $F(P)$ at the expense of more probes.

Submodularity of the objective

Let $\widetilde{F}(P) = F(P) - \lambda \cdot \max(0, |P| - m_p)$. We want to prove that $\widetilde{F}(P)$ is submodular.

We start by showing first that $d(P, s)$ is submodular. For ease of notation, we drop s, referring to $d(P, s)$ as $d(P)$ and $d(p, s)$ as $d(p)$. Consider probe sets A and B, with $A \subseteq B$, and some possible probe $x \notin B$. Note that

$$\begin{aligned} d(A \cup \{x\}) &= \max_{p \in A \cup \{x\}} d(p) \\ &= \max(\max_{p \in A} d(p), d(x)) \\ &= \max(d(A), d(x)) \end{aligned}$$

and therefore $d(A \cup \{x\}) - d(A) \geq 0$. Likewise, $d(B \cup \{x\}) = \max(d(B), d(x))$, and $d(B \cup \{x\}) - d(B) \geq 0$. Also, since $A \subseteq B$, we have

$$\begin{aligned} d(B) &= \max_{p \in B} d(p) \\ &= \max(\max_{p \in A} d(p), \max_{p \in B \setminus A} d(p)) \\ &= \max(d(A), \max_{p \in B \setminus A} d(p)) \\ &\geq d(A). \end{aligned}$$

We now consider two cases:

- Assume $d(B) \geq d(x)$. Then, $d(B \cup \{x\}) - d(B) = \max(d(B), d(x)) - d(B) = 0$. Therefore, $d(A \cup \{x\}) - d(A) \geq d(B \cup \{x\}) - d(B)$.
- Assume $d(B) < d(x)$. It follows from $d(B) \geq d(A)$ that $d(x) > d(A)$ and that $d(x) - d(A) \geq d(x) - d(B)$. Since $\max(d(A), d(x)) = d(x)$ and $\max(d(B), d(x)) = d(x)$, we have $\max(d(A), d(x)) - d(A) \geq \max(d(B), d(x)) - d(B)$. Therefore, in this case as well, $d(A \cup \{x\}) - d(A) \geq d(B \cup \{x\}) - d(B)$.

Hence, $d(P, s)$ is submodular. Since $F(P)$ is a non-negative linear combination of $d(P, s)$, it follows that $F(P)$ is submodular.

Using the above result, we show that $\widetilde{F}(P)$ is submodular. Again, consider probe sets A and B, with $A \subseteq B$, and some possible probe $x \notin B$. We want to show that $\widetilde{F}(A \cup \{x\}) - \widetilde{F}(A) \geq \widetilde{F}(B \cup \{x\}) - \widetilde{F}(B)$. We have that

$$\begin{aligned} \widetilde{F}(A \cup \{x\}) - \widetilde{F}(A) &= F(A \cup \{x\}) - \lambda \cdot \max(0, |A| + 1 - m_p) - F(A) + \lambda \cdot \max(0, |A| - m_p) \\ &= F(A \cup \{x\}) - F(A) - \lambda(\max(0, |A| + 1 - m_p) - \max(0, |A| - m_p)) \\ &\geq F(B \cup \{x\}) - F(B) - \lambda(\max(0, |A| + 1 - m_p) - \max(0, |A| - m_p)) \qquad (\star) \end{aligned}$$

where the last step follows from submodularity of $F(P)$. We now consider two cases:

- Assume $|A| \geq m_p$. Since $A \subseteq B$, $|B| \geq m_p$. Continuing from ($\star$), we have

$$\begin{aligned} \widetilde{F}(A \cup \{x\}) - \widetilde{F}(A) &\geq F(B \cup \{x\}) - F(B) - \lambda(|A| + 1 - m_p - (|A| - m_p)) \\ &= F(B \cup \{x\}) - F(B) - \lambda \\ &= F(B \cup \{x\}) - F(B) - \lambda(|B| + 1 - m_p - (|B| - m_p)) \\ &= F(B \cup \{x\}) - \lambda(|B| + 1 - m_p) - [F(B) - \lambda(|B| - m_p)] \\ &= F(B \cup \{x\}) - \lambda \cdot \max(0, |B \cup \{x\}| - m_p) - [F(B) - \lambda \cdot \max(0, |B| - m_p)] \\ &= \widetilde{F}(B \cup \{x\}) - \widetilde{F}(B). \end{aligned}$$

- Assume $|A| < m_p$. Continuing from ($\star$) in this case, we now have

$$\begin{aligned}\widetilde{F}(A \cup \{x\}) - \widetilde{F}(A) &\geq F(B \cup \{x\}) - F(B) \\ &\geq F(B \cup \{x\}) - F(B) - \lambda[\max(0, |B| + 1 - m_p) - \max(0, |B| - m_p)] \\ &= F(B \cup \{x\}) - \lambda \cdot \max(0, |B| + 1 - m_p) - [F(B) - \lambda \cdot \max(0, |B| - m_p)] \\ &= F(B \cup \{x\}) - \lambda \cdot \max(0, |B \cup \{x\}| - m_p) - [F(B) - \lambda \cdot \max(0, |B| - m_p)] \\ &= \widetilde{F}(B \cup \{x\}) - \widetilde{F}(B).\end{aligned}$$

Hence, $\widetilde{F}(P)$ is submodular.

Note that $\widetilde{F}(P)$ is non-monotone owing to the penalty term.

Non-negativity of the objective

Now we show how to ensure that $\widetilde{F}(P)$ is non-negative. Let P only contain probes p such that $F(\{p\}) \geq \lambda \cdot (\overline{m_p} - m_p)$. Since F is monotonically increasing, $F(P) \geq \lambda \cdot (\overline{m_p} - m_p)$. Thus,

$$\begin{aligned}\widetilde{F}(P) &= F(P) - \lambda \cdot \max(0, |P| - m_p) \\ &\geq \lambda \cdot (\overline{m_p} - m_p) - \lambda \cdot \max(0, |P| - m_p).\end{aligned}$$

If $|P| \leq m_p$, then

$$\begin{aligned}\widetilde{F}(P) &\geq \lambda \cdot (\overline{m_p} - m_p) - 0 \\ &\geq 0\end{aligned}$$

where the last inequality follows from $\overline{m_p} \geq m_p$. If $|P| > m_p$, then

$$\begin{aligned}\widetilde{F}(P) &\geq \lambda \cdot (\overline{m_p} - m_p) - \lambda \cdot (|P| - m_p) \\ &= \lambda \cdot (\overline{m_p} - m_p - |P| + m_p) \\ &= \lambda \cdot (\overline{m_p} - |P|) \\ &\geq 0\end{aligned}$$

where the last inequality follows from $\overline{m_p} \geq |P|$. If P is the empty set, $\widetilde{F}(P) = 0$ according to our definition of $d(P, s)$. Therefore, $\widetilde{F}(P) \geq 0$ always. Let our ground set Q be the set of probes from which we select P—i.e., $P \subseteq Q$. To enforce non-negativity, we restrict Q to only contain probes p such that $F(\{p\}) \geq \lambda \cdot (\overline{m_p} - m_p)$. In other words, every probe has to be sufficiently good. In practice, given our activity function, $\lambda \in [0.1, 0.5]$ is a reasonable choice and the constraint on $F(\{p\})$ is generally met; for example, with $\lambda = 0.25$, $\overline{m_p} = 5$, and $m_p = 1$, then we require $F(\{p\}) \geq 1$.

Solving for P

Recall we want to solve

$$\max_P \left\{\widetilde{F}(P) : |P| \leq \overline{m_p}\right\}$$

where $\widetilde{F}(P) = F(P) - \lambda \cdot \max(0, |P| - m_p)$. We need to maximize a non-negative and non-monotone submodular function subject to a cardinality constraint. The classical discrete greedy algorithm[33] may provide poor results, and would not offer theoretical guarantees, because it assumes a monotone function. We apply the recently-developed discrete randomized greedy algorithms in ref. 32, namely

Algorithm 1 Construct set of probes P to maximize $\widetilde{F}(P)$ subject to hard constraint.

Input

T	alignment of sequences extracted from a window of S, from taxon t
l_p	probe length
$\widetilde{F}$	function of probe set, including soft constraint
$\overline{m_p}$	hard constraint on number of probes

Output

P	collection of probes

1 **function** DETERMINE-PROBE-SET(T, l_p, $\widetilde{F}$, $\overline{m_p}$)
2 $C \leftarrow$ Clusters of l_p-mers at and across each position of T
3 $Q \leftarrow$ Representative (consensus) of each cluster in C ▷ ground set
4 $Q \leftarrow Q \setminus \{l_p\text{-mers in } Q \text{ not meeting non-negativity constraint}\}$
5 $Q \leftarrow Q \setminus \{l_p\text{-mers in } Q \text{ not specific to } t\}$ ▷ enforce specificity
6 $Q \leftarrow Q \cup \{2 \cdot \overline{m_p}$ "dummy" elements that contribute 0 to $\widetilde{F}\}$
7
8 $P \leftarrow \{\}$
9 **for** $j \leftarrow 1$ to $\overline{m_p}$ **do**
10 $M_j \leftarrow \overline{m_p}$ elements from $Q \setminus P$ that maximize $\sum_{u \in M_j} \left(\widetilde{F}(P \cup \{u\}) - \widetilde{F}(P)\right)$
11 $p^* \leftarrow$ Element from M_j chosen uniformly at random
12 $P \leftarrow P \cup \{p^*\}$
13 $P \leftarrow P \setminus \{$"dummy" elements$\}$
14 **return** P

Algorithm 1, which provides a $1/e$-approximation for non-monotone functions. (Algorithm 5, which provides a better approximation ratio, is likely to not be much better in our case because the constraint $\overline{m_p}$ is small compared to the size of the ground set.)

Based on the work in ref. 32, the function DETERMINE-PROBE-SET (Algorithm 1) shows how we compute P to detect a particular genomic window of an alignment S. We use locality-sensitive hashing to rapidly cluster potential probe sequences[2] throughout the window, and their representatives form the ground set Q of probes (line 3). Then, we require that probes in Q be specific to the taxon to which S belongs, using the methods in Supplementary Note 2d). We add to the ground set "dummy" elements that provide a marginal contribution of 0 to any set input to $\widetilde{F}$ (line 6), as required by an assumption of the algorithm (Reduction 1 in ref. 32). Then, we greedily choose $\leq \overline{m_p}$ probes, at each iteration selecting one randomly from a set of not-yet-chosen probes that maximize marginal contributions to $\widetilde{F}$ (lines 10–11).

The runtime to design probes is practical in the typical case. Here we ignore the runtime of evaluating specificity (line 5), which is given in Supplementary Note 2d. Let L be the window length and n be the number of sequences. There are $O(nL)$ probes in the ground set in the worst-case, and they take $O(nL)$ time to construct (line 3). Finding the $\overline{m_p}$ elements that maximize marginal contributions (line 10) takes $O(nL)$ time, and we do this $O(\overline{m_p})$ times. Thus, the runtime in the worst-case is $O(nL\overline{m_p})$. In a typical case, the number of clusters at a position in the window

[2]In particular, by sampling nucleotides—i.e., concatenating locality-sensitive hash functions drawn from a Hamming distance family—and taking the consensus of sequences within each cluster, where clusters are defined by their hash values.

is a small constant ($\ll n$) owing to sequence homology in the alignment; thus, the size of the ground set is $O(L)$, although it still takes $O(nL)$ time to construct. Now, finding the $\overline{m_p}$ elements that maximize marginal contributions takes $O(L)$ time, and we do this $O(\overline{m_p})$ times. So the runtime in a typical case is $O(nL + L\overline{m_p})$. Note that, in general, $\overline{m_p} \ll L$ and $\overline{m_p} \ll n$.

1b Design formulation #2: minimizing the number of probes

Objective

As in the above objective, let S be an alignment of sequences from species t in a genomic region and let $d(p, s)$ be a predicted detection activity between one probe p and one sequence $s \in S$. We wish to find a set P of probes with minimal $|P|$ that satisfies constraints on detection activity across these sequences. In particular, we introduce a fixed detection activity m_d and say that p is highly active in detecting s if $d(p, s) \geq m_d$. To define whether P detects a sequence with high activity, let

$$d(P, s) = \begin{cases} 1 & \text{if } |\{p : p \in P,\ d(p, s) \geq m_d\}| \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

We additionally introduce a lower bound f_S on the minimal fraction of sequences in S that must be detected with high activity. Then, we wish solve

$$\min_P \{|P| : |\{s : s \in S,\ d(P, s) = 1\}|/|S| \geq f_S\}\,.$$

That is, we want to find the smallest probe set that detects, with high predicted activity, at least a fraction f_S of all sequences.

Solving for P

To approximate the optimal P, ADAPT follows the canonical greedy solution to the set cover problem[70,71] in which the universe consists of the sequences in S and each possible probe covers a subset of sequences in S. Similar approaches have been used for PCR primer selection[18,21,26,72,73]; in contrast to prior approaches, rather than starting with a collection of candidate probes (i.e., the sets), we construct them on-the-fly.

Iteratively, we approximate a probe that covers the most number of sequences that still need to be covered. Here, a probe p *covers* a sequence s if $d(p, s) \geq m_d$. Find-Optimal-Probe, shown in Algorithm 2, implements a heuristic. Briefly, at each position Find-Optimal-Probe rapidly clusters l_p-mers in the input sequences (l_p is the probe length) by sampling nucleotides—i.e., concatenating locality-sensitive hash functions drawn from a Hamming distance family—and uses each of these clusters to propose a probe. It iterates through the clusters in decreasing order of score, stopping early (line 11) if it is unlikely that remaining clusters will provide a probe that achieves more coverage than the current best. This procedure relies on two subroutines, Score-Cluster and Num-Detect, that are described below.

Using this procedure, it is straightforward to construct a set of probes in the window (region) that achieve the desired coverage by repeatedly calling Find-Optimal-Probe. This is shown by Determine-Probe-Set, in Algorithm 3. In other words, the output probes collectively detect, with high activity, the sequences in the region.

This approach, with on-the-fly construction of probes, is similar to a reduction to an instance of the set cover problem, the solution to which is essentially the best achievable approximation[74,75]. In such a reduction, each set would represent one of the 4^{l_p} possible probes, consisting of the sequences

Algorithm 2 Construct probe p^* with highest coverage.

Input

M	sequences in S to cover, from taxon t
l_p	probe length

Output

p^*	probe in window

```
function FIND-OPTIMAL-PROBE(M, l_p)
    Initialize p*
    for each length l_p sub-window w in M do
        clusts ← Cluster all l_p-mers of M in w
        clusts ← Sort clusts, descending, according to SCORE-CLUSTER
        repeat
            p ← Representative (consensus) of l_p-mers in next best cluster in clusts
            if p is specific to taxon t then                          ▷ enforce specificity
                if NUM-DETECT(p, M) > NUM-DETECT(p*, M) then
                    p* ← p
        until early stopping criterion is met
    return p*
```

Algorithm 3 Construct minimal collection of probes in window that collectively achieve desired detection coverage.

Input

T	alignment of sequences extracted from a window of S, from taxon t
l_p	probe length
f_S	fraction of sequences in T to detect

Output

C	collection of probes

```
function DETERMINE-PROBE-SET(T, l_p, f_S)
    C ← {}
    while |{s : s ∈ T, d(C, s) = 1}|/|T| < f_S do
        M ← Sequences s ∈ T such that d(C, s) = 0
        p* ← FIND-OPTIMAL-PROBE(M, l_p)
        C ← C ∪ {p*}
    return C
```

that it would detect with high activity. Then, each iteration would identify the probe that detects, with high activity, the most not-yet-covered sequences. Here, rather than starting with such a large space, we use a heuristic to approximate the probe at each iteration.

The runtime to design probes in a window is poor in the worst-case but practical in the typical case. Let n be the number of sequences in the alignment and L be length of the window. In the worst-case, we choose n different probes in the window. Each choice requires iterating over $O(L)$ positions, and at each one we iterate through $O(n)$ clusters, taking $O(n)$ time to evaluate the probe proposed by each cluster with NUM-DETECT. Thus, this is $O(n^3L)$ time. In a typical case, there is a small constant number of clusters owing to sequence homology across the alignment, and the

number of probes needed to achieve the constraint is also a small constant. Selecting each probe requires iterating over $O(L)$ positions, and at each one we consider $O(1)$ clusters, taking $O(n)$ time again to evaluate the probe proposed. So the runtime is $O(nL)$ with these assumptions.

Scoring clusters and detection across sequences

Sequences from S can be *grouped* according to metadata such that each group receives a particular desired coverage (f_{S_g}). For example, in ADAPT's implementation they can be grouped according to year (each group contains sequences from one year), with a desired coverage that decays for each year going back in time, so that ADAPT weighs more recent sequences more heavily in the design.

There are two subroutines in Algorithm 2 that we consider here: scoring a cluster and computing the number of sequences detected by a probe. These must account for groupings. First, on line 5 of FIND-OPTIMAL-PROBE, the function SCORE-CLUSTER($clust$) computes the number of sequences $clust \in clusts$ contains that are needed to achieve the desired coverage across all the groups. That is, it calculates

$$\sum_{x \in X} \min(n_x, |\widetilde{clust} \cap M_x|)$$

where X is the collection of sequence groups, n_x is the number of sequences from group x that must still be covered to achieve x's desired coverage, $\widetilde{clust}$ gives the sequences of M from which the l_p-mers in $clust$ originated, and M_x consists of the sequences in M that are in group x. In essence, it computes a contribution of each cluster toward achieving the needed coverage of each group, summed over the groups. Similarly, on line 9 of FIND-OPTIMAL-PROBE, the function NUM-DETECT(p, M) is the detection coverage provided by probe p across the groups. In particular, its value is

$$\sum_{x \in X} \min(n_x, |B \cap M_x|)$$

where B is the set of sequences in M that p covers—i.e., $B = \{s : s \in M,\ d(p, s) \geq m_d\}$.

These subroutines are intuitive in the case where sequences are not grouped. Equivalently, consider a single group x_0. Here, SCORE-CLUSTER($clust$) is $\min(n_{x_0}, |\widetilde{clust} \cap M_{x_0}|)$. Since $\widetilde{clust} \subseteq M_{x_0} = M$, this is $\min(n_{x_0}, |\widetilde{clust}|)$. Thus, the score is simply the size of the cluster (larger clusters are preferred), or n_{x_0} for clusters large enough so as to provide more than sufficient coverage. Similarly, NUM-DETECT(p, M) is $\min(n_{x_0}, |B \cap M_{x_0}|)$. Because $B \subseteq M_{x_0} = M$, this is $\min(n_{x_0}, |B|)$. So NUM-DETECT is effectively the number of sequences covered by p that must still be covered to achieve the coverage constraint.

Furthermore, if sequences are grouped, note that line 3 of Algorithm 3 instead iterates until achieving the desired coverage for each group.

A recent paper[76] on submodular optimization looks at a similar problem; it refers to the groupings in this problem as ground sets and provides an approximation ratio given by the greedy algorithm.

Supplementary Note 2

This note describes an overview of the challenge of evaluating specificity and two formulations, implemented in ADAPT, for doing so. Unless otherwise noted, for designs and analyses in the paper, we use the formulation in Exact trie-based search for probe near neighbors.

2a Overview

In applications where differentially identifying a taxonomy is important, ADAPT ensures that the probes it constructs are specific to the taxonomy they are designed to detect. In general, the probes directly perform detection; thus, their specificity is ADAPT's focus, rather than other aspects of a design, such as primers.

The framework for this is as follows. Initially, ADAPT constructs an index of probes across all input taxonomies, which includes the taxonomies and particular sequences containing each probe. This index could also include background sequence to avoid, such as the human transcriptome, although we generally do not include non-viral background sequence. Then, when designing a probe for a taxonomy t_i with genomes S_i, ADAPT queries this index to determine its specificity against all sequences from any S_j for $j \neq i$—namely, to find hits within a specified number of mismatches of the query. The results inform whether the probe might detect some fraction of sequence diversity in t_j. Typically, ADAPT deems a probe to be non-specific if the query yields hits in at least 1% of the sequences from another taxon. ADAPT performs this query while constructing the ground set, as described in Supplementary Note 1.

This problem is computationally challenging. When querying, we generally wish to tolerate a high divergence within a relatively short query to be conservative in finding potential non-specific hits—e.g., up to ∼5 mismatches within 28 nt. Also, G-U wobble base pairing (described below) generalizes the usual alphabet of matching nucleotides. Together, these challenges mean that popular existing approaches, including seed/MEM techniques, are not fully adequate for performing queries.

2b G-U wobble base pairing

Some detection applications (e.g., CRISPR-Cas13) rely on RNA-RNA binding. That is, the probe we design is synthesized as RNA and the target is RNA as well. RNA-RNA base pairing allows for more pairing possibilities than with DNA-DNA. In particular, `G` may bind with `U`, forming a *G-U wobble base pair*. It has similar thermodynamic stability to the usual Watson-Crick base pairs[77]. Its effect on an enzymatic process may differ from other base pairs, but in some of ADAPT's applications it is comparable to Watson-Crick base pairs.

In ADAPT, we wish to treat G-U base pairs as matching when querying for a probe's specificity. For simplicity, here we will use `T` instead of `U` (the RNA nucleobase `U` replaces the DNA nucleobase `T`), and thus we consider G-T base pairing. In particular, we consider a base $g[i]$ in a probe to match a base $s[i]$ in a target sequence if either (a) $g[i] = s[i]$, (b) $g[i] =$ `A` and $s[i] =$ `G`, or (c) $g[i] =$ `C` and $s[i] =$ `T`[3]. Note that activity models in ADAPT that are trained for a particular detection technology could prune the query results if the effect is different in some application.

Tolerating G-U base pairing considerably complicates the problem for several reasons. The addition of G-U base pairing raises the probability of a matching hit between a 28-mer and an arbitrary

[3]We synthesize the reverse complement of g and use that for detection, so these rules correspond to permitting G-T base pairing.

target, thereby expanding the space of potential query results. It also means the Hamming distance between a query and valid hit (considered in the same frame) can often exceed 50% and be as high as 100%. Supplementary Fig. 18 illustrates the challenge in practice on viral genome data.

A similar challenge arises in determining off-target effects when designing small interfering RNA (siRNA)[78,79]. It is common to ignore the problem (e.g., using BLAST to query for off-targets)[80–83]. Other approaches do address it. One is to treat G-U pairs like a mismatch, albeit not as heavily penalized as a Watson-Crick mismatch[84]; however, with this approach, searching for candidate hits may fail to find valid hits if the Hamming distance between the query and hit is sufficiently high owing to G-U pairs. Another approach uses the seed-and-extend technique where the seed is in a well-defined "seed region" that requires an exact match, tolerating G-U pairs in the seed[85]; although applicable to siRNA, a seed-based approach may fail to generalize if there is no seed region, if it is too short, or if it is not consistent or is tolerant of mismatches. For some RNA interference applications, G-U pairs may be detrimental to the activity of an enzyme complex[86], and therefore it may not be necessary to fully account for it when determining specificity. None of these approaches are fully satisfying in ADAPT.

To approach the challenge of G-U wobble base pairing, at several points in the algorithms below we use a transformed sequence (Supplementary Fig. 19a). We transform a probe g into g' by changing `A` to `G` and changing `C` to `T`; in g', the only bases are `G` and `T`. Likewise, we do this for a target sequence s. This is useful because any G-T matching between s and the complement of g is not reflected by different letters between g' and s'—i.e., if the reverse complement of g (what we synthesize) matches with s up to G-U base pairing, then g' and s' are equal strings.

2c Probabilistic search for probe near neighbors

To permit queries for specificity, we first experimented with performing an approximate near neighbor lookup similar to the description in ref. 87 for points under the Hamming distance. Here, we wish to find probes that are $\leq m$ mismatches from a query.

The approach precomputes a data structure $H = \{H_1, H_2, \ldots, H_L\}$ where each H_i is a hash table that has a corresponding locality-sensitive hash function h_i, which samples b positions of a probe. The h_is bear similarity to the concept of spaced seeds[88]. It chooses L to achieve a desired reporting probability r:

$$L = \lceil \log_{1-P^b}(1-r) \rceil,$$

where $P^b = (1 - m/k)^b$ is a lower bound on the probability of collision (for a single h_i) for nearby probes of length k. In ADAPT, we have used $r = 0.95$ and $b = 22$. For all probes g across all sequences in all taxa t_j, each $H_i[h_i(g')]$ stores $\{(g, j)\}$ where j is an identifier of a taxon from which g arises and g' is g in the two-letter alphabet described above. Additionally, the data structure holds a hash table G where $G[(g, j)]$ stores identifiers of the sequences in j that contain g. From these data structures, queries are straightforward. For a probe q to query, the query algorithm looks up q' in each H_i and check if q detects (is within m mismatches) each resulting g. For the ones that it does detect, G provides the fraction of sequences in each taxon containing g and therefore provides the fraction of sequences in each taxon that q detects. The algorithm deems q specific iff this fraction is sufficiently small. Note that, when designing probes for a taxon t_j, it is straightforward to mask j from each H_i; this is important for query runtime because most near neighbors would be from j.

This approach would be suitable if we were to not have to consider G-U base pairing, but this

consideration makes it too slow for many applications. To accommodate G-U base pairs, it stores two-letter transformed probes (g') and likewise queries transformed probes (q'). The dimensionality reduction enables finding hits within $\leq m$ mismatches of a query q, sensitive to G-U base pairs, but it also means that most results in each $H_i[h_i(q')]$ are far from q. As a result, the algorithm spends most of its time validating each of these results by comparing it to q. A higher choice of b can counteract this issue, but results in higher L and thus requires more memory. Also, the approach is probabilistic and may fail to detect non-specificity; while the reporting probability might be high per-taxon, if we use ADAPT to design across many taxa it becomes more likely to output a non-specific assay. Thus, below, we develop an alternative approach that is more tailored to the particular challenges we face.

2d Exact trie-based search for probe near neighbors

Here we describe a data structure and query algorithm that permits fully accurate queries for non-specific hits of a probe. Unlike the probabilistic approach above, this will always detect non-specificity if present, and we show it is fast compared to a baseline. Having one trie containing all the indexed probes would satisfy the goal of being fully accurate because we could branch, during a query, for mismatches and G-U base pairs; however, the extensive branching involved means that query time would depend on the size of the trie and may be slow. To alleviate this, we place (or *shard*) the probes across many smaller tries.

Briefly, the data structure stores an index of all probes across the input sequences from all taxa. Let k be the probe length (e.g., 28). The data structure splits each probe into p partitions (without loss of generality, assume p divides k). Each partition maps to a $\frac{k}{p}$-bit *signature* such that any two matching strings map to the same signature, tolerating G-U base pairing; each bit corresponds to a letter from the two-letter alphabet described in G-U wobble base pairing. There are $p \cdot 2^{k/p}$ tries in total, each associated with a signature and a partition, and every probe is inserted into p tries according to the signatures of its p partitions.

To query a probe q, the algorithm relies on the pigeonhole principle: tolerating up to m mismatches across all of q, there will be at least one partition with $\leq \lfloor m/p \rfloor$ mismatches against each valid hit. For each partition of q, the query algorithm produces all combinations of signatures within $\lfloor m/p \rfloor$ mismatches—there are $\sum_{i=0}^{\lfloor m/p \rfloor} \binom{k/p}{i}$ of them—and looks up q in the tries with these signatures for the partition. During each lookup, it branches to accommodate G-U base pairing and up to m mismatches. Note that the bit signature is sensitive to G-U base pairing—i.e., two positions have the same bit if they might be a match, including owing to G-U pairing—so the algorithm finds all hits, even if the query and hit strings diverge due to G-U pairing.

Supplementary Fig. 19 provides a visual depiction of building the data structure and performing queries, and Algorithms 4 and 5 provide pseudocode.

A loose bound on the runtime of a query is

$$O\left(p \cdot \frac{n}{2^{k/p}} \cdot \sum_{i=0}^{\lfloor m/p \rfloor} \binom{k/p}{i}\right)$$

where n be the total number of probes indexed in the data structure. The query algorithm performs a search for p partitions of a query q. For each partition, it considers $\sum_{i=0}^{\lfloor m/p \rfloor} \binom{k/p}{i}$ tries, one for each combination of $\lfloor m/p \rfloor$ bit flips. The size of each trie is a loose upper bound on the query time within it; assuming uniform sharding, the size of each is $O(\frac{n}{2^{k/p}})$. Multiplying the size of each

Algorithm 4 Build data structure of tries to support specificity queries.

Input

$\{S\}$	collection of sequences across taxonomies
k	probe length
p	number of partitions

Output

$\mathcal{T}$	space of tries indexing probes

function BUILD-TRIES($\{S\}$, k, p)
  Initialize $\mathcal{T}$ ▷ contains $p \cdot 2^{k/p}$ tries, one per pair of partition and bit vector
  for each taxonomy t_i **do**
    $S_i \leftarrow$ Sequences for t_i
    for each probe k-mer (probe) g in S_i **do**
      for $r = 1$ **to** p **do**
        $g_r \leftarrow$ Partition r of g
        $g'_r \leftarrow$ Hash of g_r: `A` $\rightarrow 0$, `G` $\rightarrow 0$, `C` $\rightarrow 1$, `T` $\rightarrow 1$ ▷ bit vector
        $T \leftarrow$ Trie in $\mathcal{T}$ corresponding to partition r and bit vector g'_r
        Insert g into T ▷ include t_i and sequence identifier in leaf node
  return $\mathcal{T}$

Algorithm 5 Query tries to find non-specific hits.

Input

q	probe to query for specificity to taxon t_i
m	number of mismatches to tolerate (counting G-U pairs as matches)
p	number of partitions

Requires: $\mathcal{T}$ from BUILD-TRIES
Requires: taxon t_i is masked from $\mathcal{T}$

Output

G	taxon and sequence identifiers of non-specific hits

function QUERY(q, m, p)
  $G \leftarrow \{\}$
  for $r = 1$ **to** p **do**
    $q_r \leftarrow$ Partition r of q
    $q'_r \leftarrow$ Hash of q_r: `A` $\rightarrow 0$, `G` $\rightarrow 0$, `C` $\rightarrow 1$, `T` $\rightarrow 1$ ▷ bit vector
    for each variant $(q'_r)'$ of q'_r with $\leq \lfloor m/p \rfloor$ flipped bits **do**
      $T \leftarrow$ Trie in $\mathcal{T}$ corresponding to partition r and bit vector $(q'_r)'$
      $g \leftarrow$ Query results for q in T, branching always for G-U
        pairing and for up to m mismatches
      $G \leftarrow G \cup \{g\}$
  return G

trie by the number of them considered during a query provides the stated runtime. Adjusting p, a small constant, allows us to tune the runtime: higher choices reduce the number of bit flips, and thus the number of tries to search, but yield larger tries, and thus requires more time searching within each of them. The runtime does not scale well with our choice of m, but this is generally a small constant (up to $\sim$5). The term providing the worst-case query time within each trie, $O(\frac{n}{2^{k/p}})$, is likely to be a considerable overestimate in practice because queries usually do not need to fully explore a trie.

Because the data structure stores each probe in p separate tries, the required memory is $O(np)$. Although this scales reasonably with n, it involves large constant factors and is memory-intensive in practice; one future direction is to compress the tries.

Supplementary Note 3

This note describes how we link methods, from Supplementary Notes 1 and 2, in ADAPT to form an end-to-end system for designing assays. In particular, this involves searching across genomic regions and connecting with publicly available genome databases.

3a Branch and bound search for genomic regions

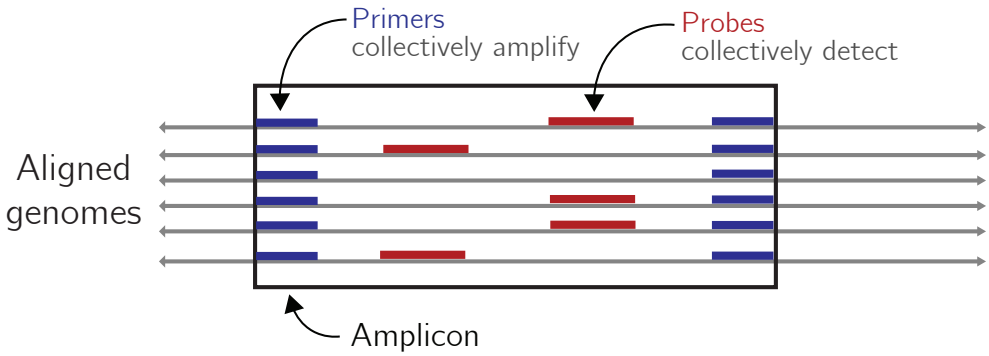
In many nucleic acid applications, we must amplify a genomic region to obtain enough material for detection. The probes in a probe set P ought to be within a genomic region of the alignment S that is bound by conserved sequence so that we can design primers to amplify the region (Supplementary Note Fig. 1); as with probes, we want to penalize the number of primers because they can interfere with each other or require multiple reactions. Similarly, we wish to penalize the length of the region because longer regions are less efficient to amplify; penalizing the logarithm of length approximates the length-dependence of amplification efficiency. We first walk through the search using the objective that maximizes expected activity (Supplementary Note 1a). For this, we now perform a search for a genomic region R that encompasses P and solve

$$\max_{P,R} \left\{ \widetilde{F}(P) - \lambda_A |R_A| - \lambda_L \log(R_L) : |P| \leq \overline{m_p} \right\}$$

where $\widetilde{F}(P)$ and $\overline{m_p}$ are defined in Supplementary Note 1a, R_A gives the set of primers bounding the region, R_L gives the nucleotide length of the region, and λ_A and λ_L give weights on the penalties[4]. Note that λ_A and λ_L can optionally be set to 0, removing the requirement that a region be bound by conserved sequence and represent an amplicon.

To solve this, we use an algorithm in which we search over options for R and prune unnecessary ones (Supplementary Note Fig. 2). Rather than finding a single maximum, we wish to compute the highest N solutions—i.e., N regions, each containing a probe set—to the objective. This is important so that multiple design options can be tested and compared experimentally; it also provides the option for an assay to target multiple regions in a genome. Note also that the range of

[4]We typically add a fixed constant (4) to the objective values before reporting them to the user, which we find makes the values more interpretable to users because it makes them more likely to be non-negative. This shift has no impact on the design options or their rankings.



Supplementary Note Figure 1 — Searching for genomic regions. ADAPT searches for a region of the genome, bound by conserved sequence to use for primers, that contains probes that can collectively detect the region. The requirement that a region be bound by conserved sequence and represent an amplicon is optional.

$\widetilde{F}$ has an upper bound, which we call $\widetilde{F}_{\text{hi}}$, calculated from $F(P)$'s highest value (predicted activities are bounded) and $|P| = 1$.

We maintain a min heap h of the N designs with the highest value of the objective. First, at every position of the alignment S, we approximate a minimal set of primers that achieves a desired coverage over the genomes; we use Algorithm 3 in Supplementary Note 1, except parameterized for primers[5]. Then, we search over pairs of positions in the alignment, considering the regions that would be amplified by primers at each pair. Although the number of such regions is quadratic in the alignment length, we can effectively prune regions based on $|R_A|$ and R_L. We calculate, with these values, the objective value using $\widetilde{F}_{\text{hi}}$ in place of $\widetilde{F}(P)$; this value provides an upper bound on the solution. If this value falls below that of the minimum in h, the region cannot be in the top N and thus we do not need to compute P. For regions that could be in the best N, we compute the probe set P with the maximal $\widetilde{F}(P)$ as described in Supplementary Note 1a (Solving for P). If the objective value for the design given by (R, P) is greater than the minimum in h, we pop from h and push the design to it. This search is guaranteed to identify the top N regions according to the objective, up to our approximation of $\widetilde{F}(P)$.

This search follows the branch and bound paradigm in which the candidate solutions (R, P) make up a 3-level tree, excluding the root. The levels represent the (1) $5'$ primers, (2) $3'$ primers, and (3) probe set P. We can prune the choice of $3'$ primers based on the length of the amplicon they would form. Exploring the final level in particular—determining P—is the slow step. Since we can easily construct an upper bound on the candidate solution for nodes in the final level, which we compare to the minimum in h, we can discard nodes and thus avoid having to compute P for many candidate solutions.

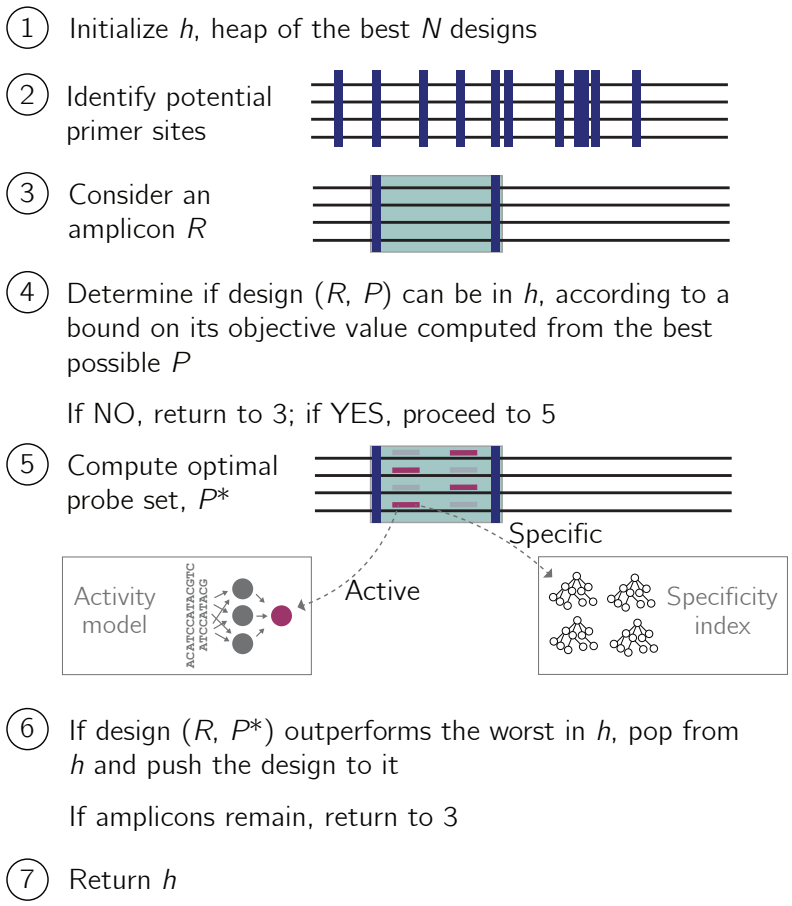
It is also important that the design options are diverse, i.e., reflect meaningfully different regions rather than being simple shifts of one another. To account for this, we implement the following: if a design to push to h has a region overlapping that of an existing design in h, it must replace that existing design (and only does so if the new one has a higher objective value).

Additionally, during our search many of the computations—particularly when computing probe sets—would be performed repeatedly from the same input, owing to overlap between different regions across the search. As a result, we memoize results of these probe set computations according to genome position. A branch and bound implementation, as described above, might start with all the $5'$ primers (first level) and then select the best $3'$ primers (second level), before advancing to computing probe sets. However, this could force each successive probe set computation to jump to a different region in the genome, as defined by the primer pairs: we would not be able to efficiently cleanup memoizations for these computations and memory would grow throughout the search. To avoid this issue, we scan linearly along the genome and, each time we advance the $5'$ primer position, we determine probe set memoizations that we no longer need to store. Memoizing these computations provides a considerable improvement in runtime (Supplementary Fig. 21).

The above description applies to maximizing expected activity, but it is straightforward to adjust the strategy when minimizing the number of probes (Supplementary Note 1b). In this case, we change our objective to solve for

$$\min_{P,R} \{|P| + \lambda_A |R_A| + \lambda_L \log(R_L)\}$$

[5]The parameters include primer length, number of tolerated mismatches, bounds on GC content, and the fraction of genomes that must be covered.



Supplementary Note Figure 2 — Branch and bound search for genomic regions. Sketch of the search for genomic regions and optimal probe sets within them. Supplementary Note 1 describes the algorithms in Step 5. Step 5 makes use of the predictive activity model and the data structure (Supplementary Note 2) for evaluating specificity.

where we also impose the constraint on coverage described in Supplementary Note 1b. The search now stores a max heap h of the designs with the smallest values of the objective. For pruning, we compute a lower bound on the candidate solution by letting $|P| = 1$, and compare this bound to the maximum in h.

The search is embarrassingly parallel. One future direction is to parallelize the search across genomic regions, in which we perform it separately for contiguous parts of the genome and then merge the resulting heaps. In practice, the primary challenge is likely to be handling shared memory, in particular for the large index used to enforce specificity.

3b Fetching and curating sequences to target

ADAPT accepts a collection of taxonomies provided by a user: $\{t_1, t_2, \ldots\}$. It can either design for one t_i or for all t_i, in either case ensuring designs are specific accounting for all t_j where $j \neq i$. Each t_i generally represents a species, but can also be a subspecies taxon[6]. In NCBI's databases, each taxonomy has a unique identifier[60] and ADAPT accepts these identifiers. ADAPT then downloads

[6]One technicality: many species have segmented genomes. For these, ADAPT also needs the label of the segment. ADAPT effectively treats each segment as a separate taxonomy—i.e., for species that are segmented, the t_is are actually pairs of taxonomy ID and segment.

all near-complete and complete genomes for each t_i from NCBI's genome neighbors database, but uses its Influenza Virus Resource database[61] for influenza viruses. It also fetches metadata for these genomes (e.g., date of sample collection), which some downstream design tasks process.

We must then prepare these genomes for design. Briefly, for each t_i we curate the genomes by aligning each one to one or more reference sequences[7] for t_i and remove genomes that align very poorly to all references, as measured by several heuristics: by default, we remove a genome that has $< 50\%$ identity to all references or that have $< 60\%$ identity to all references after collapsing consecutive gaps to a single gap. This process prunes genomes that are misclassified, have genes in an atypical sense, or are highly divergent for some other reason. Then, we cluster the genomes for t_i with an alignment-free approach by computing a MinHash signature for each genome, rapidly estimating pairwise distances from these signatures (namely, the Mash distance[68]), and performing hierarchical clustering using the distance matrix. The default maximum inter-cluster distance (approximate average nucleotide dissimilarity) for clustering is 20%. In general, we obtain a single cluster for a species (Supplementary Fig. 24f). This provides another curation mechanism, because it can discard clusters that are too small (by default, just one sequence). Finally, ADAPT aligns the genomes within each cluster using MAFFT[89]. This yields a collection of alignments, where each is for a cluster of genomes from taxon t_i.

Many of these computations—such as curation, clustering, and alignment—are slow yet repeated on successive runs of ADAPT. ADAPT memoizes results of the above computations, to disk, to reuse on future runs when the input permits it. This memoization improves runtime for routine use of ADAPT.

[7]The "reference" sequences are determined by NCBI, but can also be provided by the user. They are manually curated, high-quality genomes and encompass major strains.

Supplementary Note 4

This note describes our approach to evaluate probe activity over time by simulating relatively likely substitutions. Results are in Supplementary Fig. 22.

4a Motivation

Nucleic acid diagnostics are susceptible to degraded performance as viral genomes accumulate substitutions. Extensive genomic data can inform where to design probes, for example, by identifying regions with less variability or regions where a probe (e.g., Cas13a guide) can maintain high activity across variation. However, finding such sites is difficult or impossible when there is little genomic data, as is the case early in an outbreak of a novel virus or for an understudied virus.

One option is to use a substitution model to simulate likely types of substitutions in the genome, and then to predict activity against these simulated sequences. Parameters of the model can be transferred from related viruses. The approach would account for the possibility that some potential target regions are more likely to accrue mutations that degrade activity than other regions. For example, consider a region rich in `T` nucleotides—complementary probes have `A`—and a virus with a high relative rate of `T` to `C` transitions. Also, consider a case where `A-C` probe-target mismatches harm activity, particularly in a specific location of the probe. Such substitutions in the genome would induce those mismatches; simulating these substitutions could inform ADAPT to avoid the regions or to position probes to avoid this type of mismatch.

4b Background on substitution model

We use the general time-reversible (GTR) model, which is based on a continuous-time Markov process that accounts for relative rates of substitutions. Ref. 90 contains further information, and this subsection summarizes the model. The parameters of this model are the equilibrium base frequencies, $(\pi_A, \pi_C, \pi_G, \pi_T)$ and the rate parameters, $r_{AC}, r_{AG}, r_{AT}, r_{CG}, r_{CT}, r_{GT}$. Note that $r_{ij} = r_{ji}$. For convenience, denote the above parameters as a, b, c, d, e, f respectively.

The GTR model includes a rate matrix $Q = \{q_{ij}\}$, giving the instantaneous rate at which a base $i \in \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$ changes to $j \in \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$ (i.e., the probability after an infinitesimal time):

$$Q = \begin{pmatrix} \cdot & a\pi_C & b\pi_G & c\pi_T \\ a\pi_A & \cdot & d\pi_G & e\pi_T \\ b\pi_A & d\pi_C & \cdot & f\pi_T \\ c\pi_A & e\pi_C & f\pi_G & \cdot \end{pmatrix}$$

The diagonal entries are set so that each row sums to 0 and it is typical to normalize Q so that the average rate is 1 (i.e., $-\sum_{i=1}^{4} \pi_i Q_{ii} = 1$).

The GTR model specifies a transition probability matrix P, computed numerically via matrix exponentiation:

$$P(t) = e^{Q \cdot \mu \cdot t} = \begin{pmatrix} p_{\mathrm{AA}}(t) & p_{\mathrm{AC}}(t) & p_{\mathrm{AG}}(t) & p_{\mathrm{AT}}(t) \\ p_{\mathrm{CA}}(t) & p_{\mathrm{CC}}(t) & p_{\mathrm{CG}}(t) & p_{\mathrm{CT}}(t) \\ p_{\mathrm{GA}}(t) & p_{\mathrm{GC}}(t) & p_{\mathrm{GG}}(t) & p_{\mathrm{GT}}(t) \\ p_{\mathrm{TA}}(t) & p_{\mathrm{TC}}(t) & p_{\mathrm{TG}}(t) & p_{\mathrm{TT}}(t) \end{pmatrix}$$

where $p_{ij}(t)$ is a probability that i transitions to j after elapsed time t. The overall substitution rate is μ and $\mu \cdot t$ has units of expected number of substitutions per site. We use P, below, to simulate substitutions.

In analyses in Supplementary Fig. 22, we used $\mu = 10^{-3}$ substitutions/site/year and $t = 5$ years. We also empirically computed base frequencies from the SARS-CoV-2 genome used for those analyses and used the following relative rates (normalized to r_{GT}), which we estimated: $r_{AC} = 1.3$, $r_{AG} = 5.4$, $r_{AT} = 1.8$, $r_{CG} = 0.8$, $r_{CT} = 9.5$, and $r_{GT} = 1.0$.

More sophisticated models—such as ones that accommodate rate variation among sites—may perform better for this task, but we have not experimented with them.

4c Evaluating probes against simulated sequences

Let s be the sequence of a virus at a given time, and S_t be a discrete random variable representing the sequence of the virus after time t has elapsed. The above probability matrix P allows us to compute $\Pr(S_t = \overline{s}|s, t, P)$, the likelihood of sequence $\overline{s}$ given the current sequence s. We use this model to construct a distribution of potential sequences after time t, and assess our probe's activity against these sequences using our predictive model (namely, for Cas13a guides). For each pair of a probe g and original target sequence s_i, we can obtain a sampling of activities,

$$\{d(g, s_{i,1}), \ldots, d(g, s_{i,n})\},$$

where each $s_{i,j}$ is one of n simulated sequences and $d(g, s_{i,j})$ is a predicted detection activity.

Algorithm 6 Estimate probe activity against simulated sequences.

Input

$\{s_i\}$	collection of target sequences for taxon, at and around site where g binds
g	probe sequence
d	activity prediction function
P	transition probability matrix
n	number of sequences to simulate for each original target sequence s_i

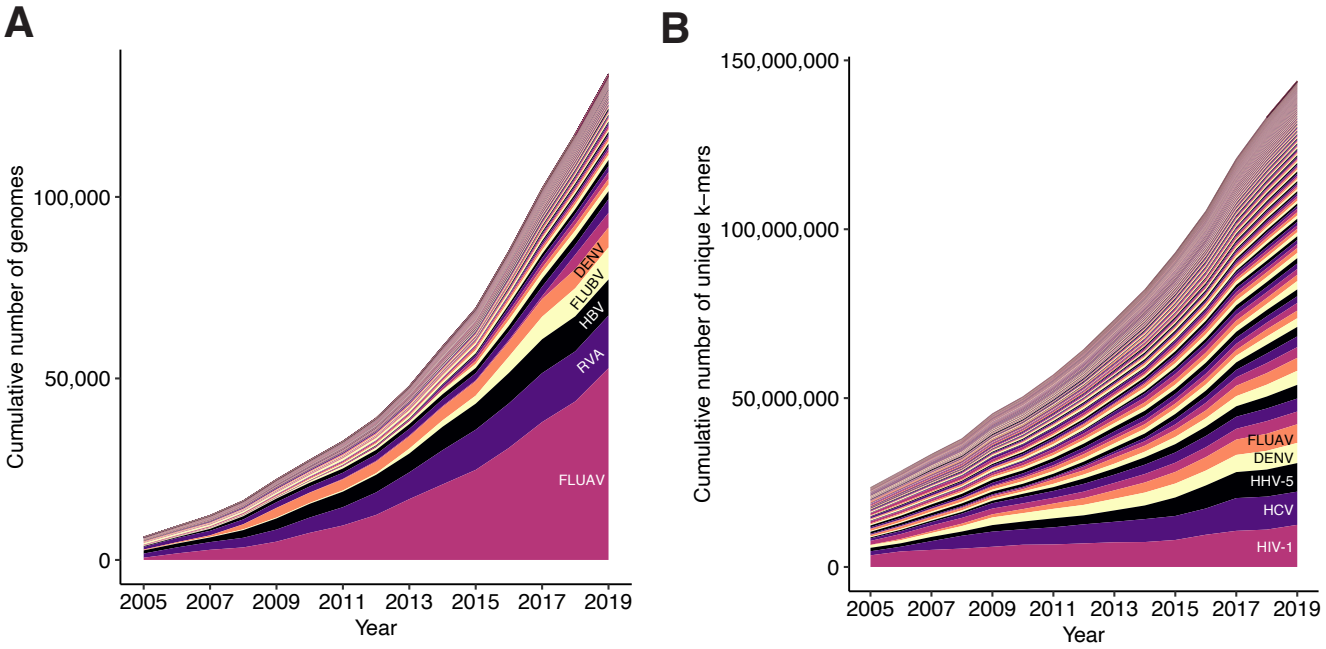
Output

mean activity across target genomes

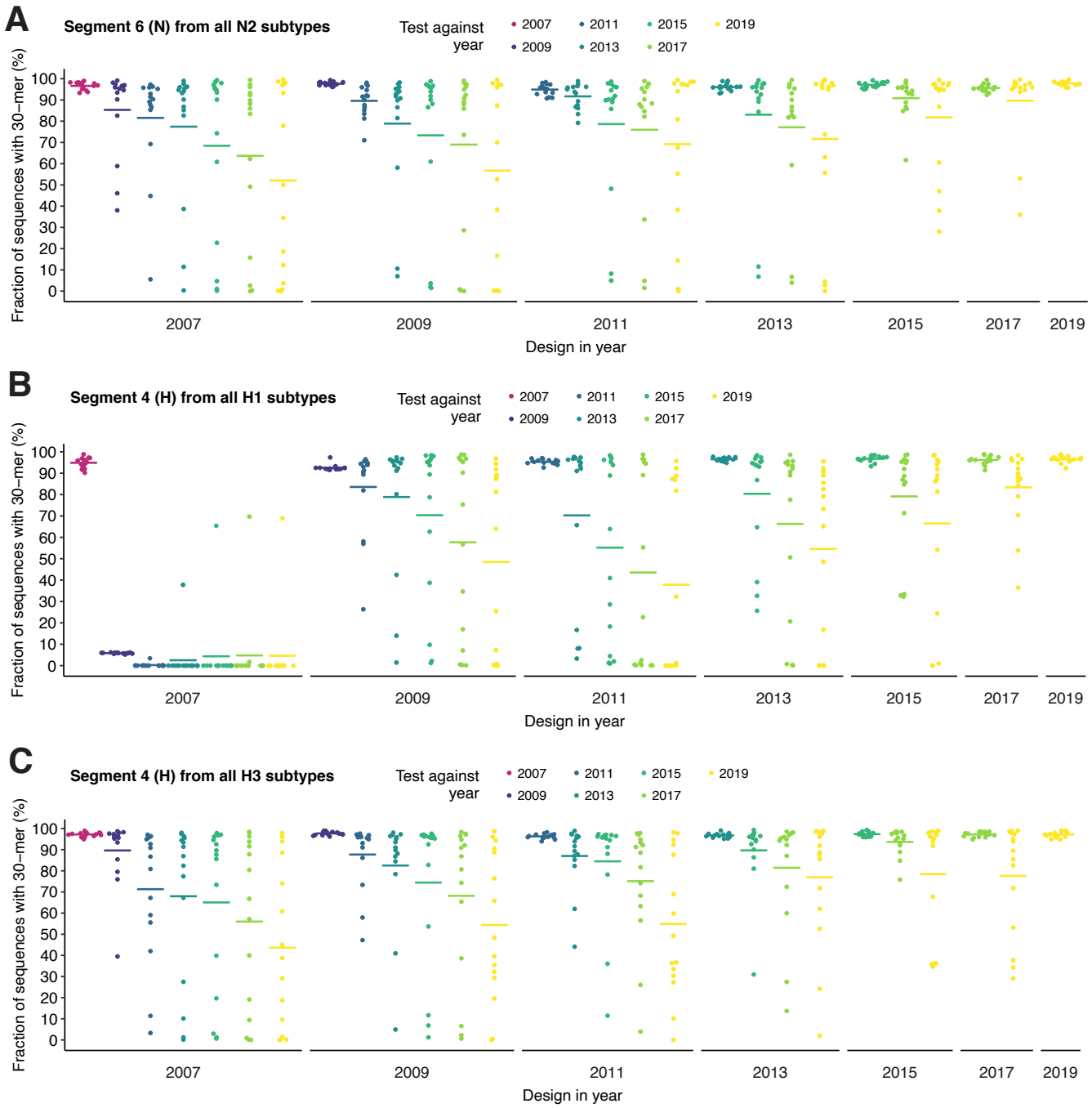
```
function EVALUATE-PROBE-AGAINST-SIMULATED-SEQUENCES({s_i}, g, d, P, n)
    C ← {}
    for each target sequence s_i in {s_i} do
        A_i ← {}
        for j ← 1 to n do
            s_{i,j} ← Simulate(s_i, P)          ▷ sample substitutions against s_i
            A_i ← A_i ∪ {d(g, s_{i,j})}          ▷ predict detection activity
        C ← C ∪ { Bottom-5th-Percentile(A_i) }
    return mean(C)
```

We are often interested (Supplementary Fig. 22d) in whether a probe maintains activity against most of the potential substitutions—that is, we may want to be "risk-averse" and avoid a situation, even if unlikely, where we observe degraded activity owing to possible substitutions. For this goal, we use the bottom 5$^{\text{th}}$ percentile of the different $d(g, s_{i,j})$ to summarize the activity against simulated sequences originating with each s_i. And then we summarize these across the different s_i by taking the mean. Algorithm 6 shows pseudocode.

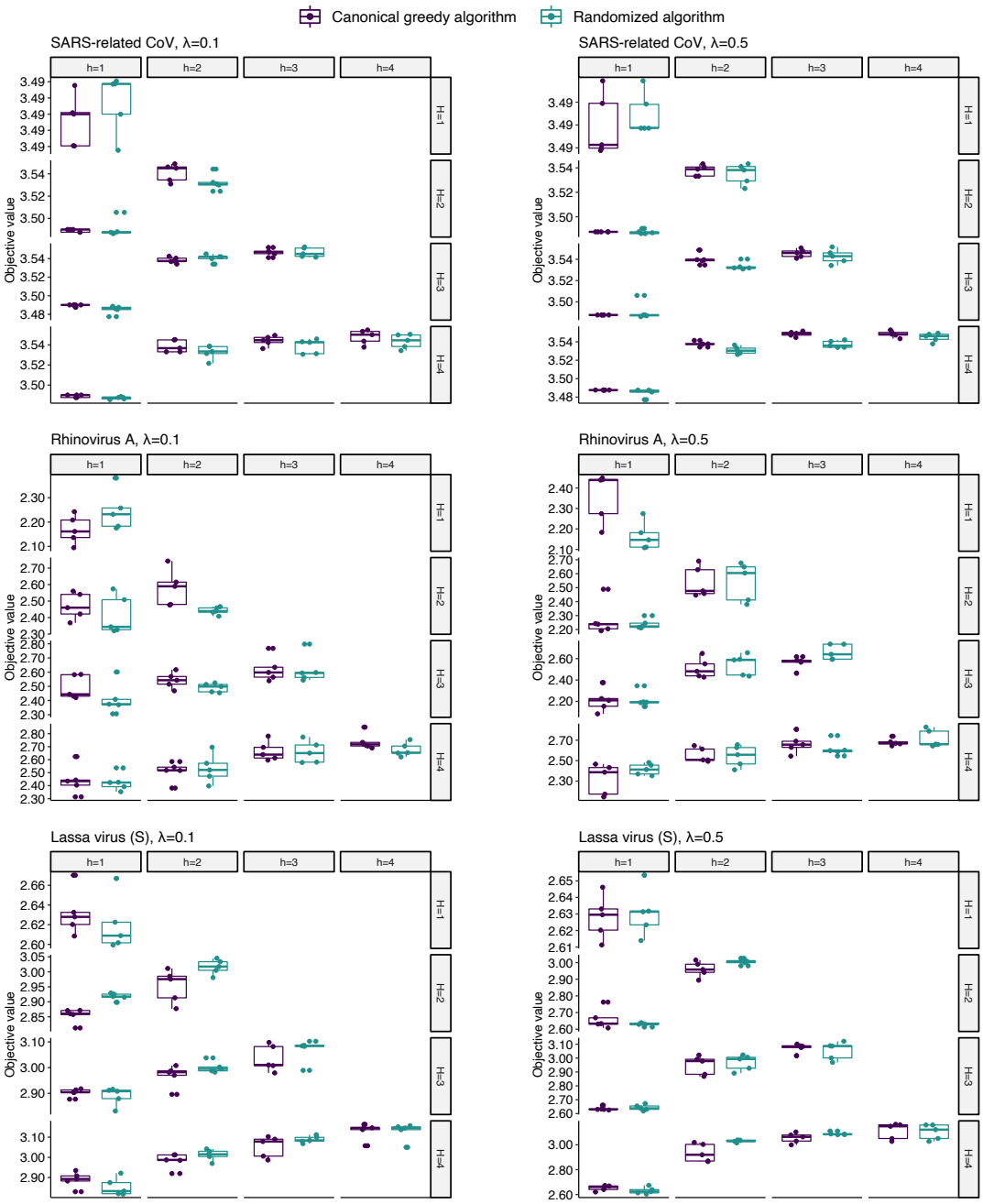
Supplementary Figures



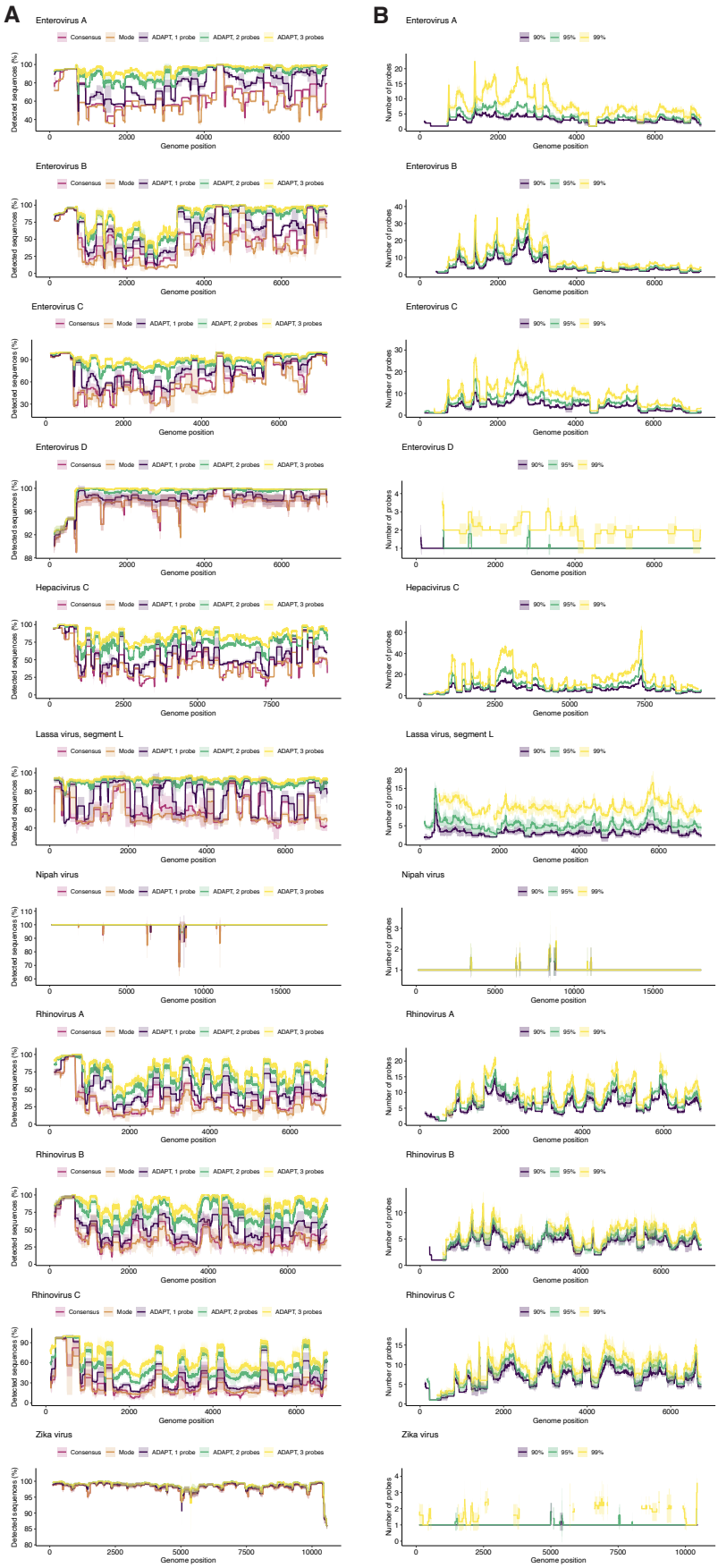
Supplementary Figure 1 — Growing number of viral genomes and diversity. Growth of data over time for 573 viral species known to infect humans. Each species is a color. **(a)** Cumulative number of genome sequences, counted from NCBI[28] viral genome neighbors and influenza databases, for each species that were available up to each year. For genomes with multiple segments, this counts only the number of sequences of the segment that has the most sequences. 5 species with the most number of genomes are labeled. FLUAV, influenza A virus; RVA, rotavirus A; HBV, hepatitis B virus; FLUBV, influenza B virus; DENV, dengue virus. **(b)** Number of unique 31-mers for the genomes in (a), a simple measure of diversity. HIV-1, human immunodeficiency virus 1; HCV, hepatitis C virus; HHV-5, human betaherpesvirus 5. In both panels, year indicates the year of the entry creation date in the database.



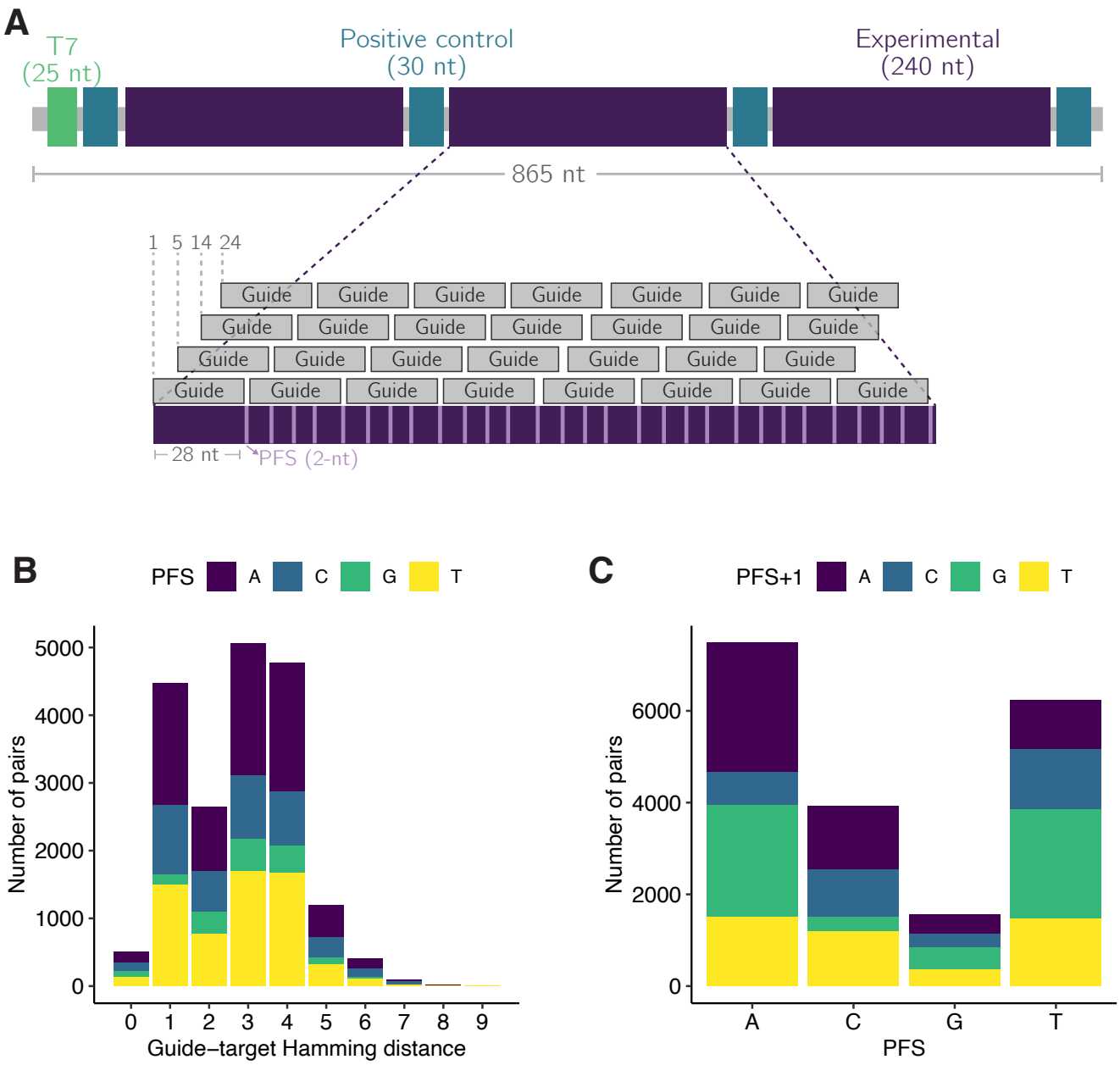
Supplementary Figure 2 — Comprehensiveness of conserved influenza A virus 30-mers over time. Even when considering the most conserved sequences, diagnostic performance of probes can degrade over time owing to genomic changes. At each year, we select the 15 most conserved non-overlapping 30-mers according to recent sequence data up to that year—a simple model for designing diagnostic probes at different years, without any consideration to other constraints such as specificity or activity. Each point represents a 30-mer from the year in which it was designed. We then measure the fraction of all sequences in subsequent years (colored) that contain each 30-mer—a simple test of comprehensiveness. Bars indicate the mean fraction of sequences containing the 15 30-mers at each combination of design and test year. To aid visualization, only odd years are shown. **(a)** Segment 6 (N) sequences from all N2 subtypes. **(b)** Segment 4 (H) sequences from all H1 subtypes. **(c)** Segment 4 (H) sequences from all H3 subtypes. Fig. 1a shows segment 6 (N) sequences from N1 subtypes.



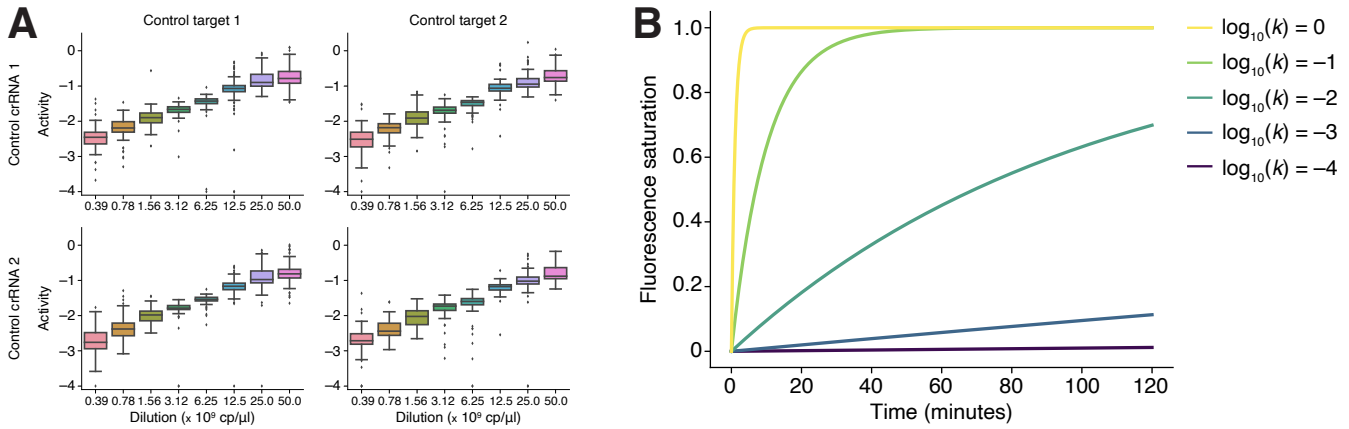
Supplementary Figure 3 — Comparison of algorithms for submodular maximization. Objective values of the optimal solutions identified by two algorithms for submodular maximization: the canonical greedy algorithm for monotone functions[33] and a randomized algorithm with provable guarantees on non-monotone functions[32]. The function here is non-monotone, but neither algorithm clearly outperforms the other. Three viral species are shown. Each was evaluated for two choices of the weight on the soft constraint/penalty, indicated by λ, as well as different choices of the soft cardinality constraint (h) and hard constraint (H) with $h \leq H$. Supplementary Note 1 contains a definition of the objective function, including the penalty weight and cardinality constraints. Each point indicates the result of one of 5 runs; differences account for randomness both in the randomized greedy algorithm and in constructing the ground set.



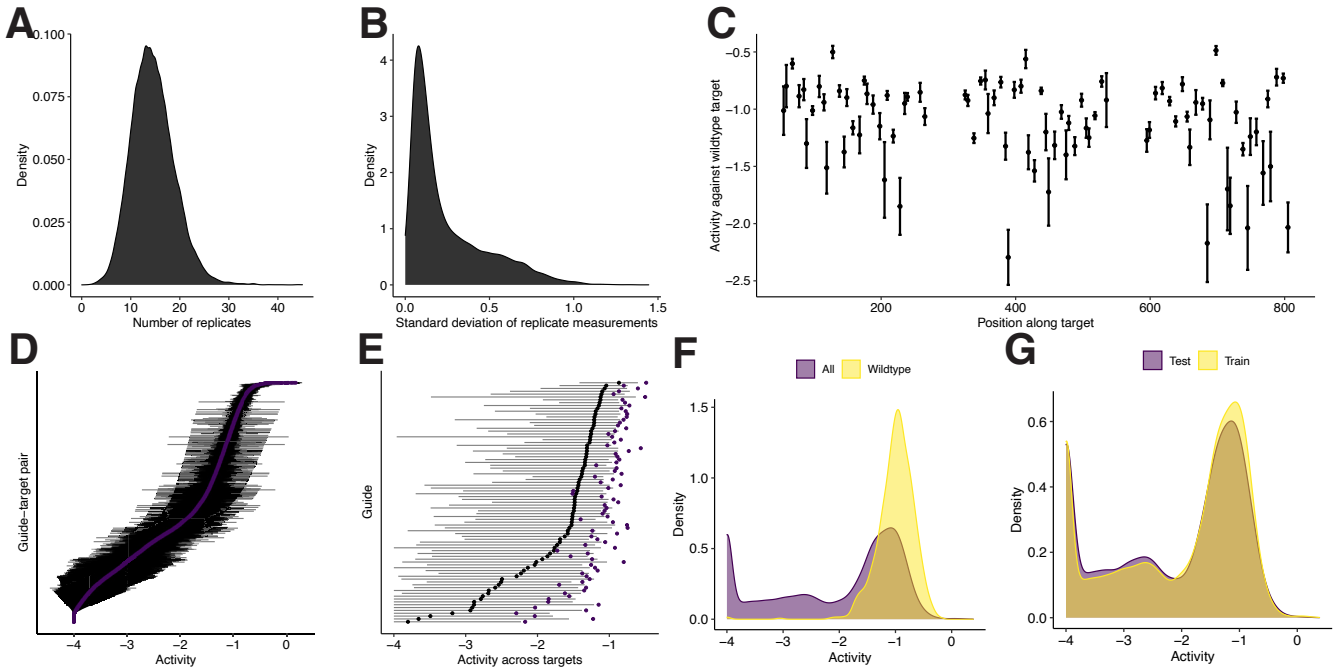
Supplementary Figure 4 — Comprehensiveness of probe design. (a) Same as Fig. 1d but with additional viruses. **(b)** Same as Fig. 1e but with additional viruses. Gaps at a site are present when it is not possible to construct a probe set that reaches the desired coverage, owing to gaps or missing data.



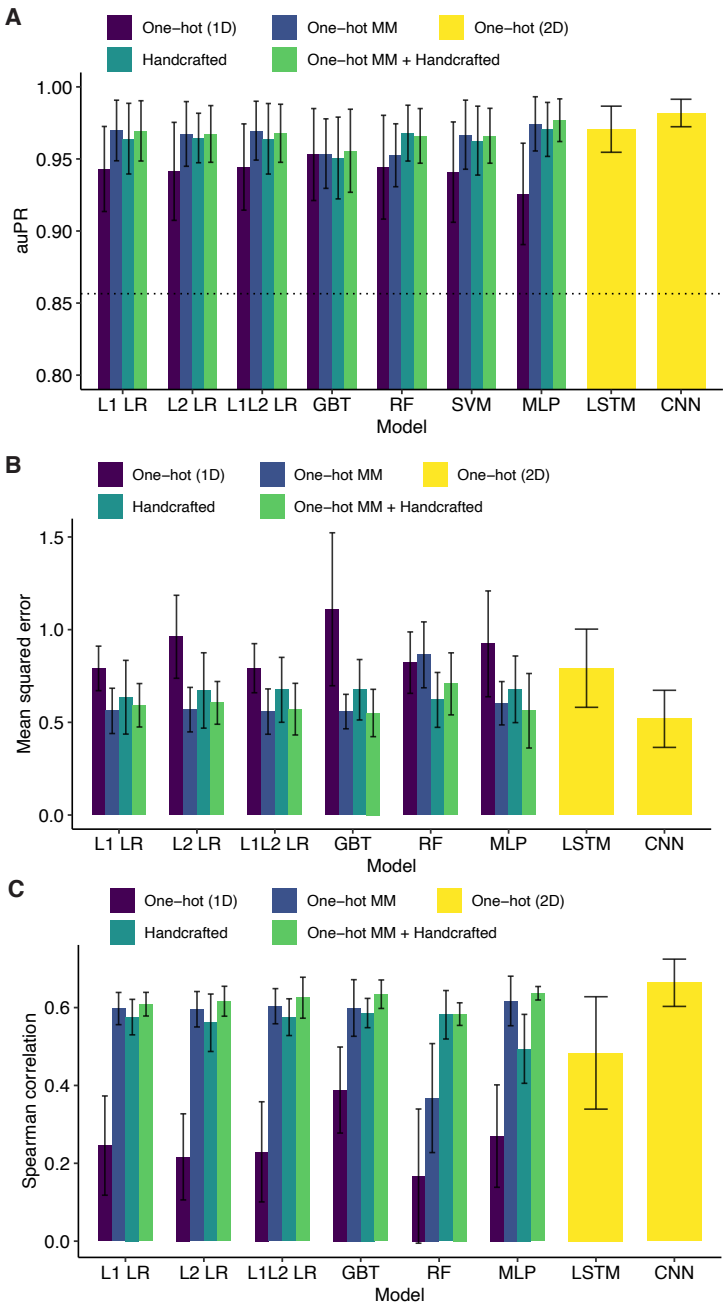
Supplementary Figure 5 — Guide-target library design. **(a)** Top, depiction of the wildtype target. The wildtype contains a T7 promoter on the 5′ end for transcription, four positive control regions, and three experimental regions. Each positive control region contains a unique guide that matches perfectly all targets, except negative control targets (not shown). Bottom, zoom of one experimental region from the wildtype target. Guide sequence comes from tiling along this region—29 guides per experimental region. There are also negative control guides (not shown) that only match the negative control targets. Other targets contain mismatches relative to the wildtype target, and thus contain mismatches relative to the guide sequences. **(b)** Distribution of the Hamming distance between guide and target across guide-target pairs. Color represents the number of pairs with each protospacer flanking site (PFS) at each Hamming distance. The 19,209 unique guide-target pairs included in our final, curated dataset (Methods) are shown. **(c)** Same as (b), but the distribution of PFS across the guide-target pairs. Color represents the number of pairs with each nucleotide immediately following the PFS (3′ end of protospacer).



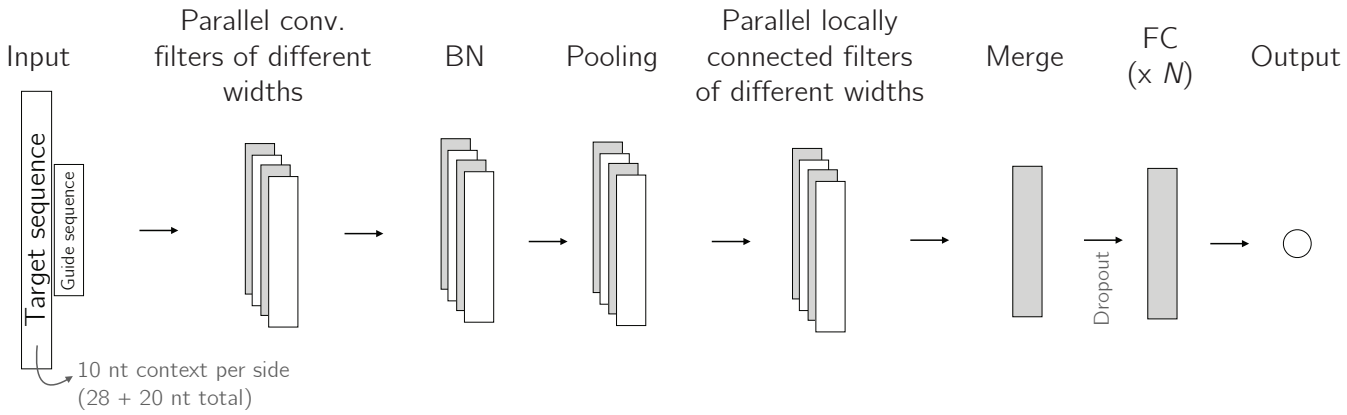
Supplementary Figure 6 — Assessing activity through CRISPR-Cas13a reaction kinetics. (a) Cas13 activity for a series of target concentrations, using two control targets and guides from our Cas13 library. We model fluorescence for each guide-target pair over time (Fig. 2a; Methods), fitting a curve of the form $C(1 - e^{-kt}) + B$ where t is time and e^{-kt} represents remaining reporter presence over time. We take $\log_{10}(k)$ to be the measure of Cas13 activity. Boxes show quartiles and whiskers extend, past the low/high quartiles, up to 1.5 times the interquartile range. **(b)** Theoretical fluorescence saturation over time—namely, the term $1-e^{-kt}$—for five activity values. Over the time scale of our experiment (t up to $\sim$120 minutes), we cannot observe reporter activation when k is small, motivating the use of an activity cutoff. Therefore, we label guide-target pairs with $\log_{10}(k) \leq -4$ as inactive and those with $\log_{10}(k) > -4$ as active.



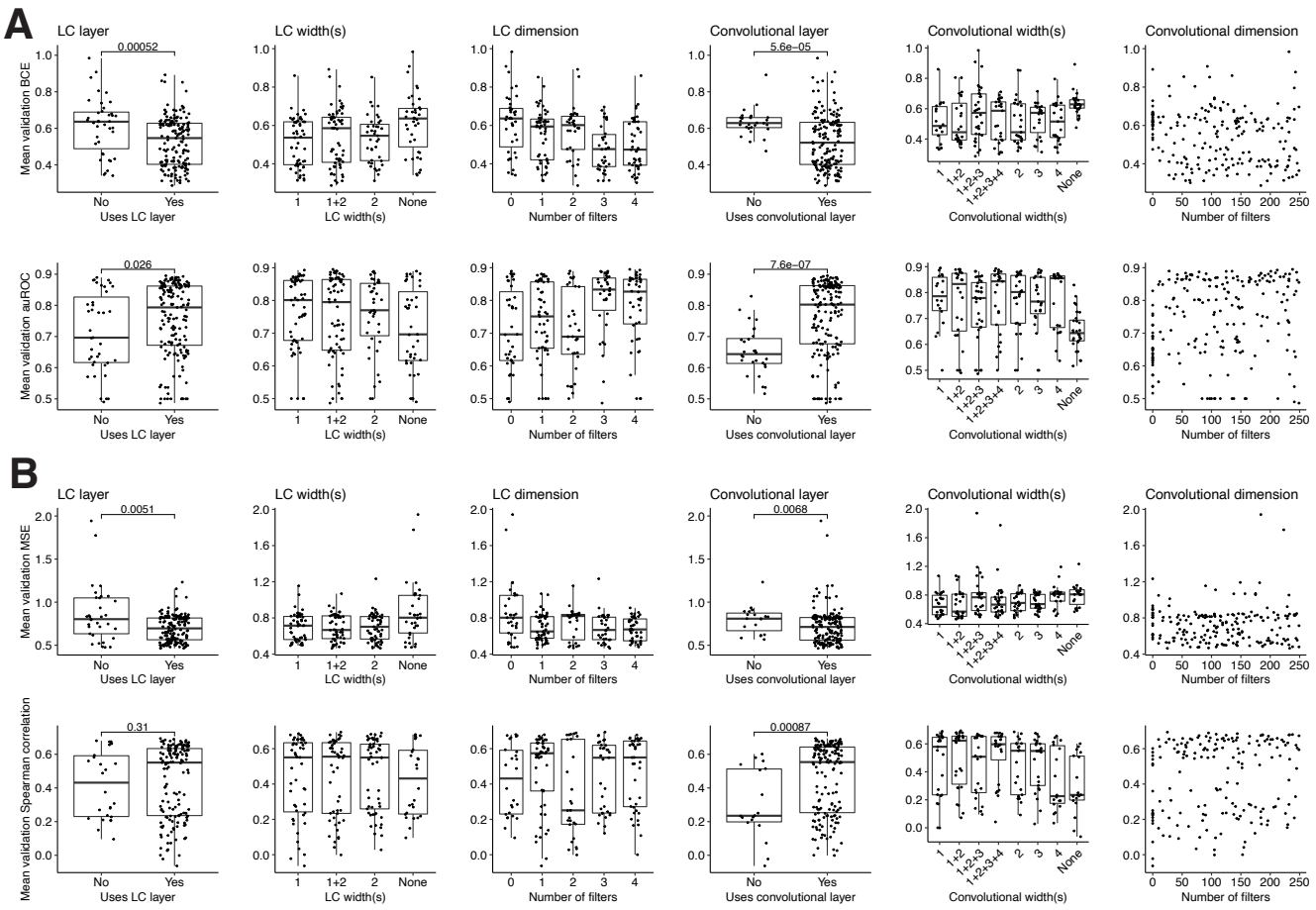
Supplementary Figure 7 — Dataset of CRISPR-Cas13a guide-target pairs. All panels show the 18,508 unique guide-target pairs in the dataset used for training and testing. Activity is defined in Methods. **(a)** Distribution of number of replicate activity measurements for each pair. **(b)** Distribution of standard deviation across replicate activity measurements for each pair. **(c)** Activity of each guide against the wildtype target (matching exactly), shown by their position along the target. Dot indicates the mean activity across the wildtype targets, shown with a 95% confidence interval. **(d)** Variation in activity across guide-target pairs and among replicate measurements. Each row represents a guide-target pair. Purple dot indicates the mean activity across replicate measurements; pairs are sorted vertically by this value. Bars indicate the 95% confidence interval for the mean. **(e)** Variation in activity between guides and across targets for each guide. Each row represents a guide. Black dot indicates the median activity across all targets and bars span the 20^{th} and 80^{th} percentiles of activity across all targets. Purple dot indicates the mean activity across the wildtype targets (matching the guide exactly). **(f)** Distribution of activity across all guide-target pairs and only pairs with the wildtype target. **(g)** Distribution of activity across all guide-target pairs in the training data and the pairs in the test data (the two sets do not overlap along the target or contain the same guides; Methods). In (d–g), there are 10 resampled replicate activity measurements for each guide-target pair.



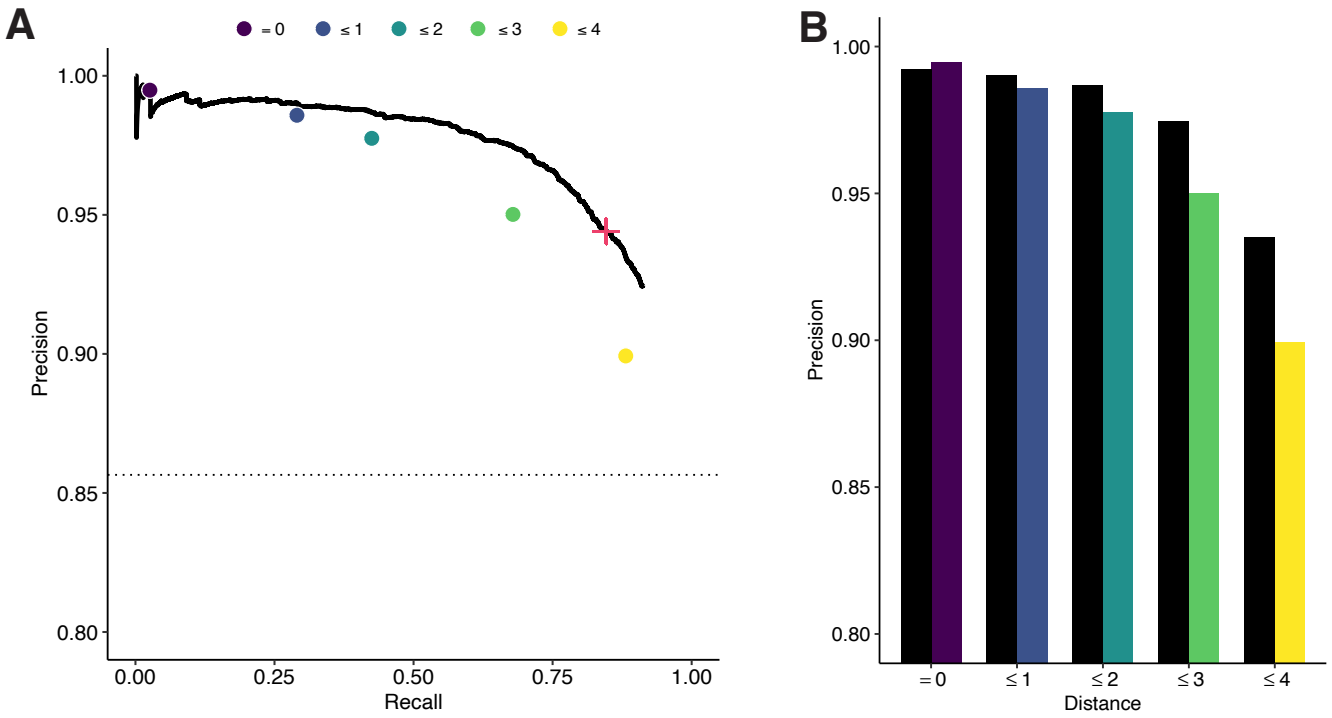
Supplementary Figure 8 — Nested cross-validation for classification and regression. For each model and input type (color) on each of five outer folds, we performed a five-fold cross-validated hyperparameter search. The plotted value is the mean of a statistic across the five outer folds, and the error bar indicates the 95% confidence interval. **(a)** Area under precision-recall curve (auPR) for different classification models. auROC is in Fig. 2c. L1 LR and L2 LR, logistic regression; L1L2 LR, elastic net; GBT, gradient-boosted classification tree; RF, random forest; SVM, support vector machine; MLP, multilayer perceptron; LSTM, long short-term memory recurrent neural network; CNN, convolutional neural network including parallel convolution filters of different widths and a locally-connected layer. One-hot (1D) is one-hot encoding of both target and guide sequence without explicit pairing; One-hot MM is one-hot encoding of target sequence and of mismatches in guides relative to the target; Handcrafted is curated features of hypothesized importance (Methods); One-hot (2D) is one-hot encoding of both target and guide sequence with encoded guide-target pairing. Dashed line is precision of random classifier (equivalently, the fraction of guide-target pairs that are active). **(b)** Mean squared error (MSE) for different regression models (lower is better). L1 and L2 LR, regularized linear regression; L1L2 LR, elastic net; GBT, gradient-boosted regression tree; RF, MLP, LSTM, and CNN are as in (a) except constructed for regression. Input types are as in (a). **(c)** Same as (b) but the statistic is Spearman correlation.



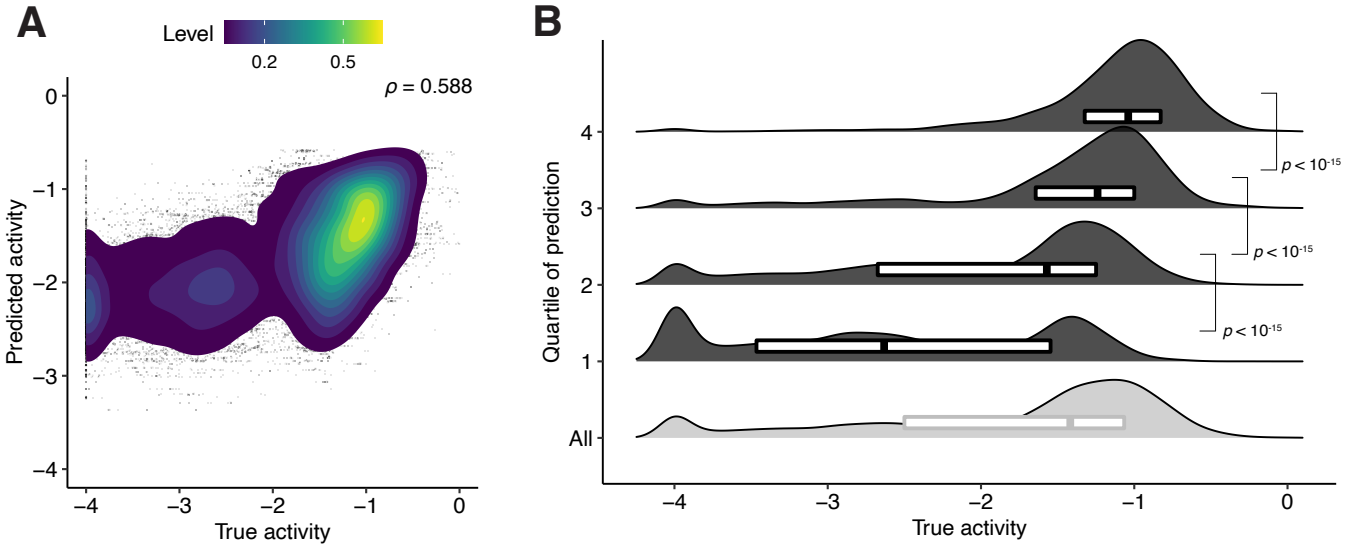
Supplementary Figure 9 — Architecture of convolutional neural network for guide-target activity prediction. Convolutional neural network (CNN) architecture for classifying and regressing activity; hyperparameter search and training is separate for each task. The inputs are one-hot encoded for the target and guides sequences (8 channels in total). There are multiple convolutional filters of different widths processing the input in parallel, as well as multiple locally connected filters of different widths; outputs of these different filters are concatenated in the merge layer. Pooling includes maximum, average, and both. 'BN' is batch normalization and 'FC' is fully connected. There are N fully connected layers. The dropout layers are in front of each fully connected layer.



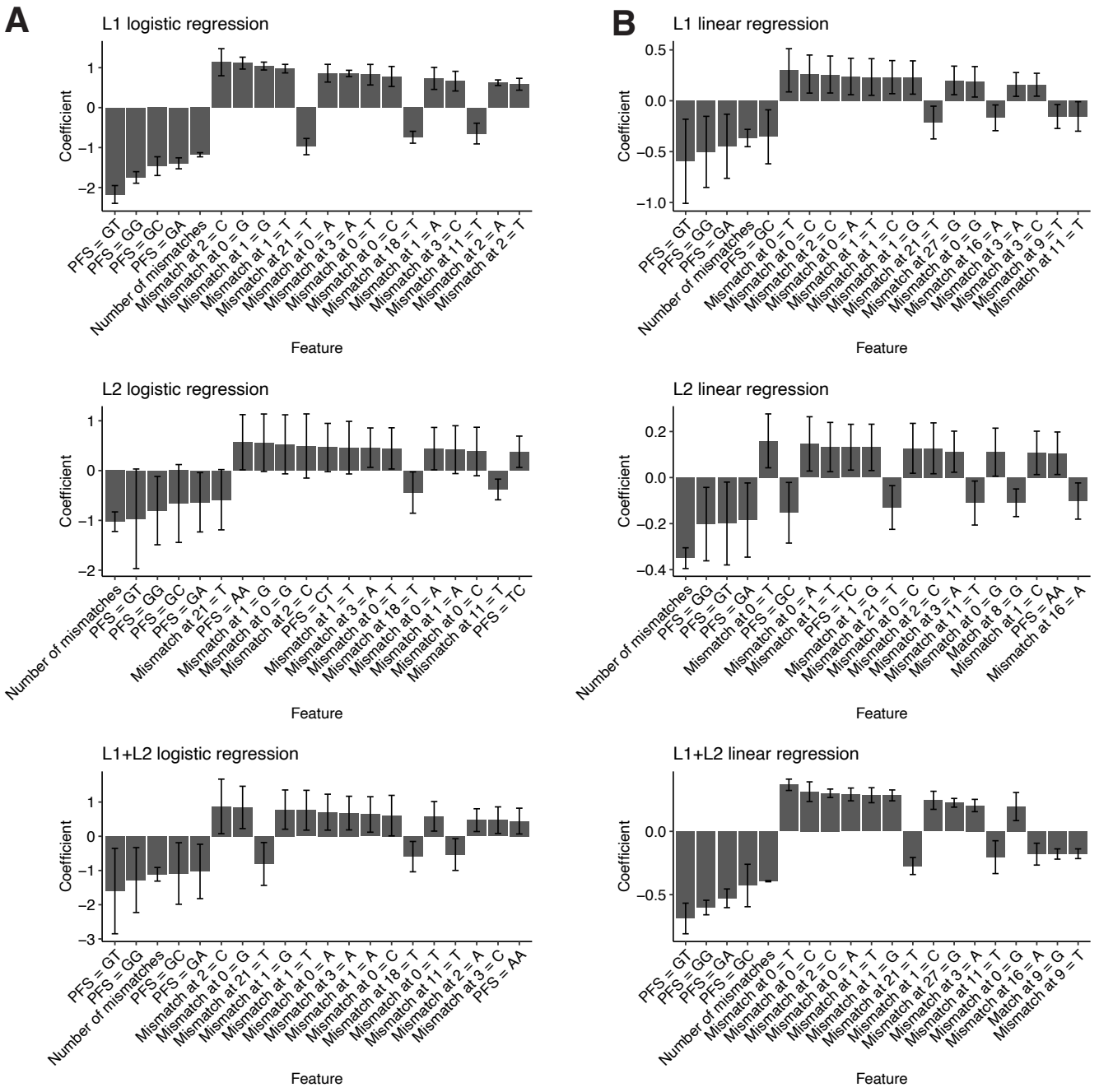
Supplementary Figure 10 — Hyperparameter search for convolutional neural networks. We used a random search over the hyperparameter space (200 draws) to select each convolutional neural network (CNN) model. Each plot corresponds to a hyperparameter and shows choices of that hyperparameter; see Methods for all hyperparameters. The evaluations are cross-validated: each dot indicates the mean of a metric, computed across five folds, for a draw of hyperparameters. 'LC', locally connnected. The '+' in LC and convolutional widths separates different widths of parallel filters; 'None' indicates that the model does not use an LC or convolutional layer. *P*-values are computed from Mann-Whitney *U* tests (one-sided). **(a)** Results of hyperparameter search for classification. BCE, binary cross-entropy. **(b)** Results of hyperparameter search for regression. MSE, mean squared error.



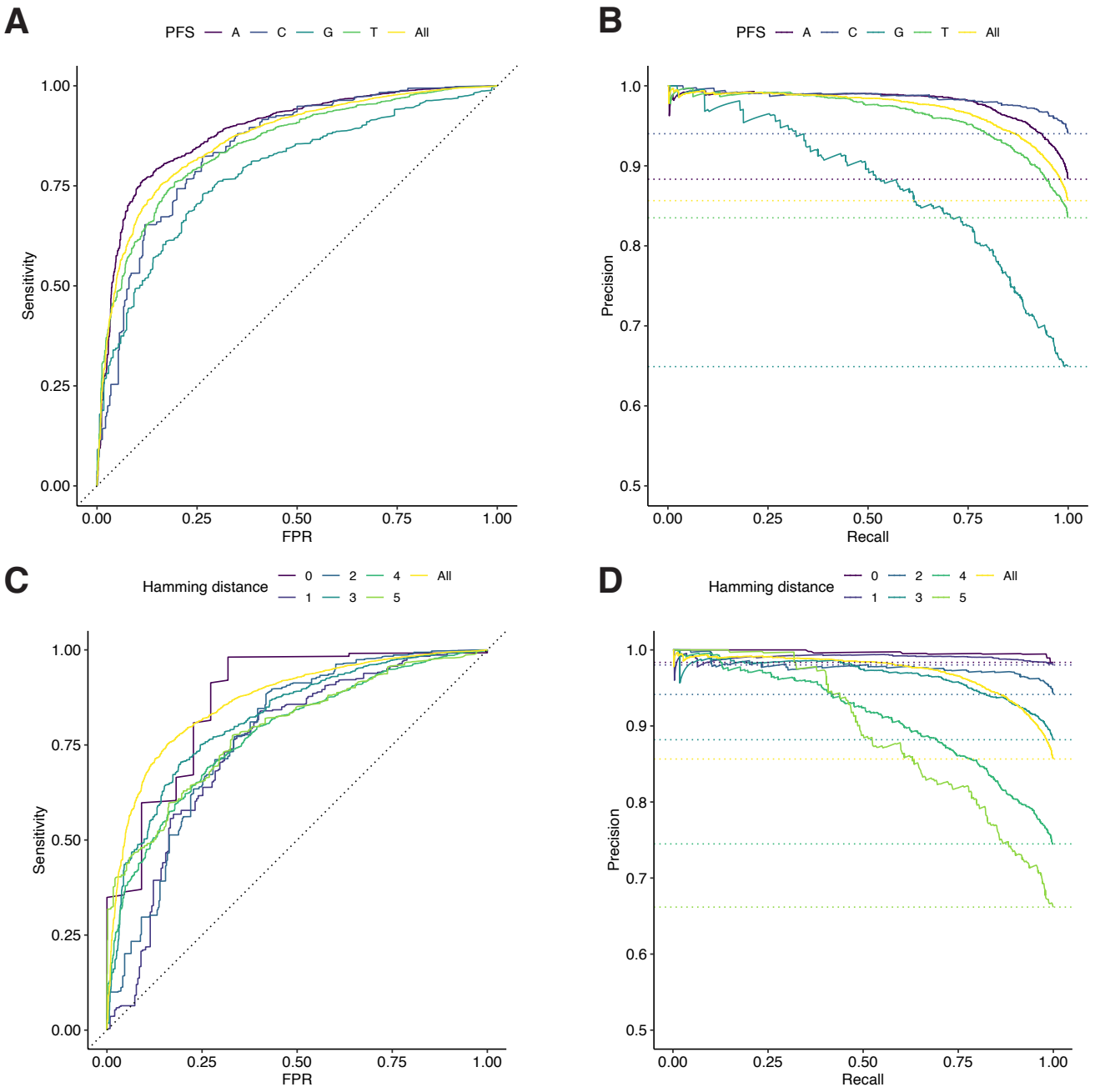
Supplementary Figure 11 — Precision-recall curve of classifier. **(a)** Precision-recall (PR) curve of CNN model, which is used in ADAPT, classifying pairs as inactive or active on a held-out test set. ROC curve is in Fig. 2d. Points indicate precision and recall for a baseline classifier: choosing a guide-target pair to be active if and only if it has an active (non-G) PFS and the Hamming distance between the guide and target is less than the specified threshold (color). Red '+' indicates the decision threshold in ADAPT. Dashed line is precision of random classifier (equivalently, the fraction of guide-target pairs that are active). **(b)** Comparison of precision between CNN (black) and baseline classifiers (color as in (a)) at equivalent recall.



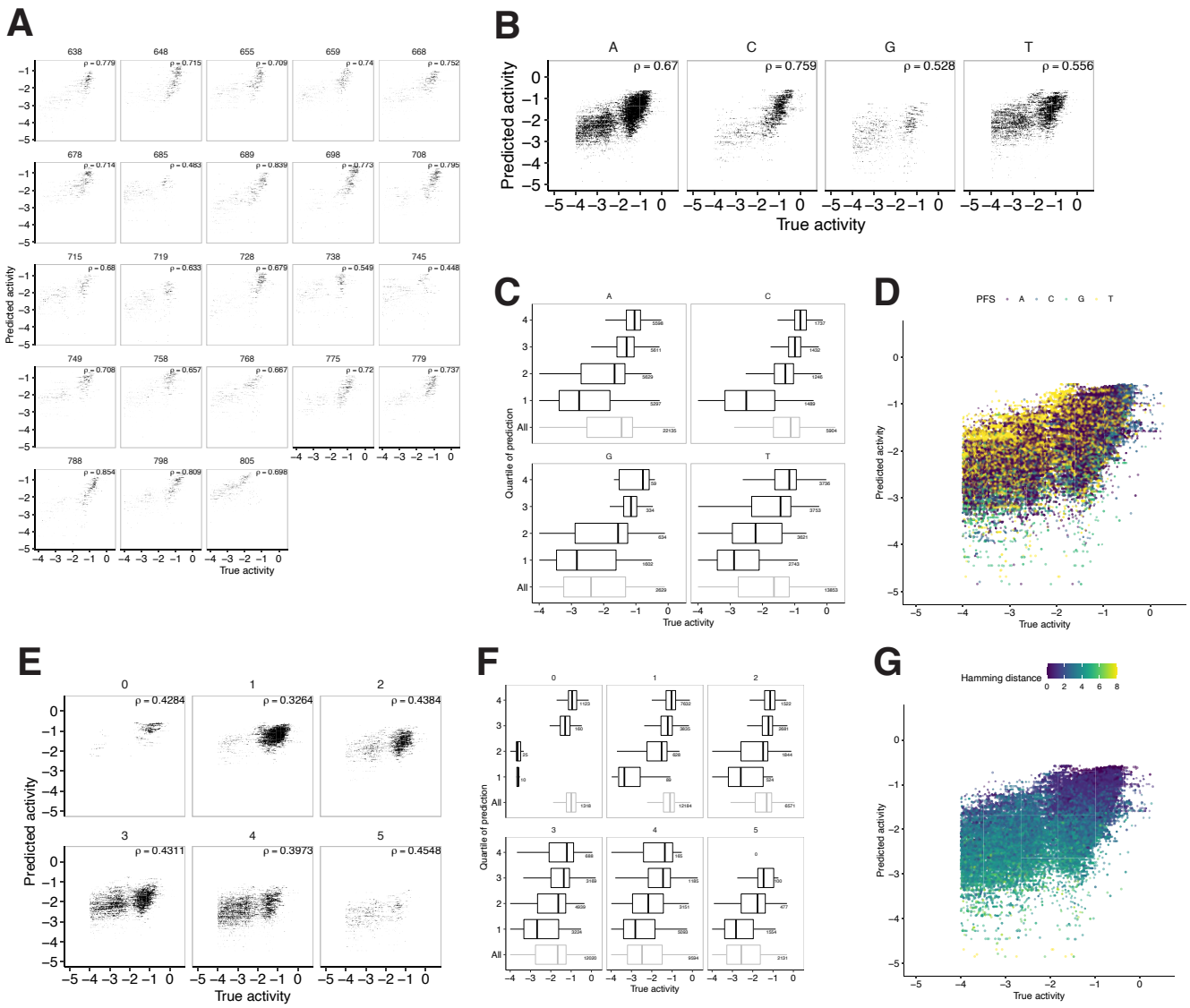
Supplementary Figure 12 — Regression results on guide-target pairs classified as active. **(a)** Same as Fig. 2e, except on guide-target pairs that are *classified* to be active by the CNN reported in Fig. 2d (other data show regression results on pairs that are true active). Color, point density. ρ, Spearman correlation. **(b)** Same as Fig. 2f, except on guide-target pairs that are classified to be active. Each row contains one quartile based on their predicted activity (top row is predicted most active), with the bottom row showing all pairs classified to be active. Smoothed density estimates and interquartile ranges show the distribution of true activity for the pairs from each quartile. *P*-values are computed from Mann-Whitney *U* tests (one-sided).



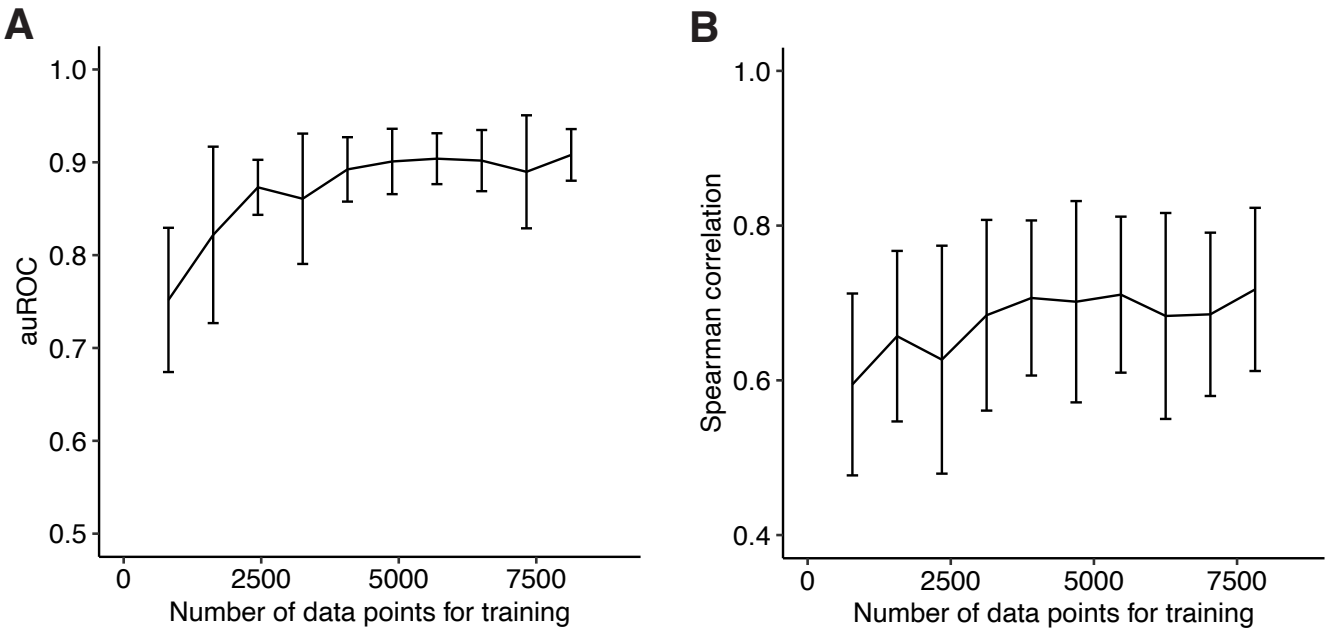
Supplementary Figure 13 — Importance of features in linear models. (a) Feature coefficients in linear models for classification. Plotted value is the mean of the coefficient across training on five folds, and error bar is the 95% confidence interval. Input type for all models is 'One-hot MM + Handcrafted' as defined in Fig. 2c and Supplementary Fig. 8. Coefficients are sorted by absolute value and the top 20 are shown. 'L1+L2' is elastic net. PFS, protospacer flanking site (3′ end) with two positions. All positions are defined based on the protospacer target sequence and are 0-based. Mismatch alleles are in the guide spacer sequence and are mismatches relative to the target. **(b)** Same as (a) but for regression models. Note that positive coefficients for mismatch features do not necessarily imply that mismatches improve activity compared to a matching guide-target pair.



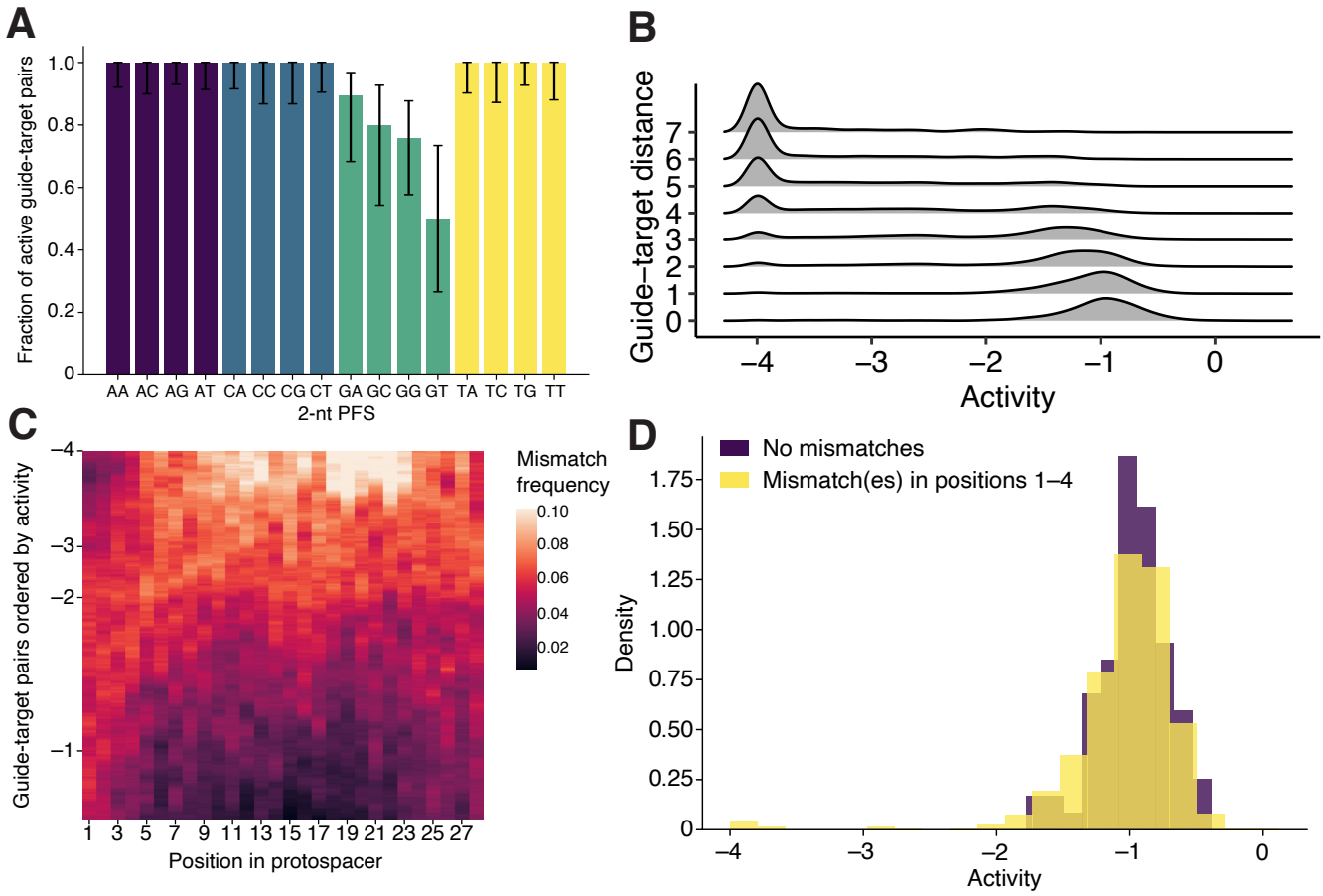
Supplementary Figure 14 — Classification performance on subsets of test data. Evaluations of classification on different subsets of the held-out test data corresponding to variables that may considerably affect activity. Here, the model is the same for all evaluations and only tested (not trained) on different subsets. **(a)** ROC curves computed from guide-target pairs with the different protospacer flanking site (PFS) nucleotides. **(b)** Precision-recall (PR) curves computed from pairs with the different PFS nucleotides. Dashed lines are precision of random classifiers for each PFS (equivalently, the fraction of guide-target pairs that are active with each PFS). **(c)** ROC curves computed from pairs with different Hamming distances between guide and target. **(d)** PR curves computed from pairs with different Hamming distances between guide and target. Dashed lines are precision of random classifiers for each choice of Hamming distance (equivalently, the fraction of guide-target pairs that are active at each Hamming distance). In all panels, yellow curve is for all test data.



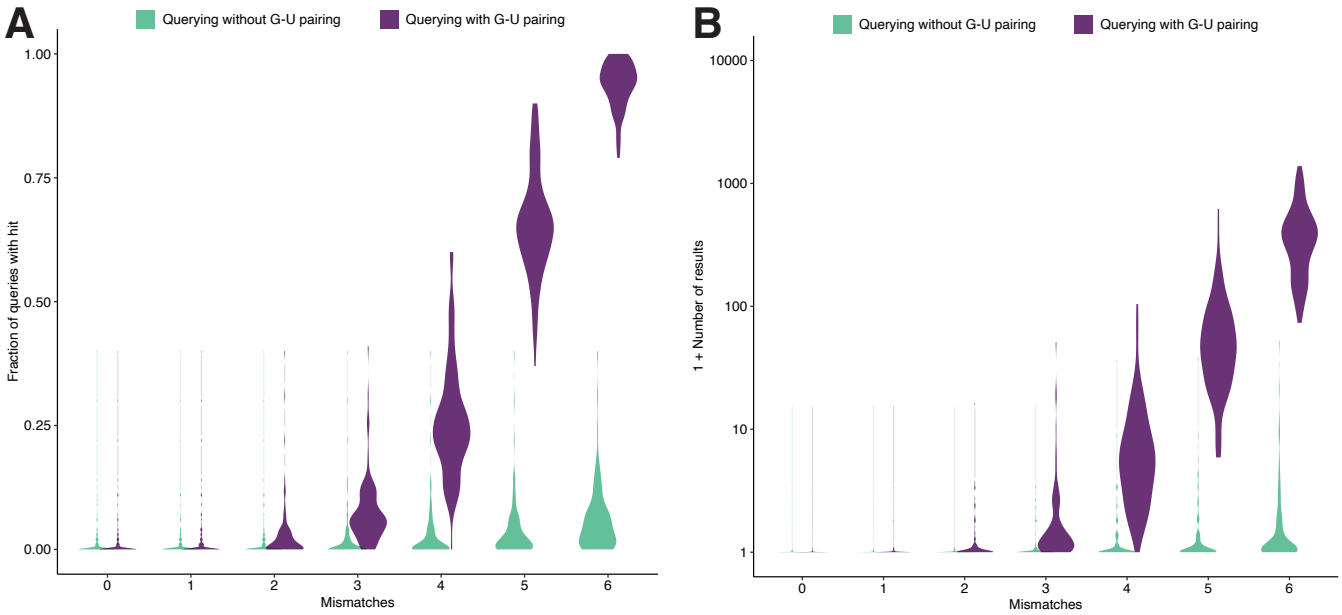
Supplementary Figure 15 — Regression performance on subsets of test data. Evaluations of regression on different subsets of active guide-target pairs in the held-out test data, where the subsets correspond to variables that may considerably affect activity. Here, the model is the same for all evaluations and only tested (not trained) on different subsets. **(a)** Each plot corresponds to one guide, with the points shown representing guide-target pairs across the different targets the guide detects. Number above each plot is the position of the guide along the wildtype target. **(b)** Pairs separated by the different protospacer flanking site (PFS) nucleotides, indicated above each plot. Each point is a guide-target pair. **(c)** Pairs separated by the different PFS nucleotides. Each row contains one quartile based on their predicted activity (top row is predicted most active), with the bottom row showing all active pairs with the PFS. Numbers indicate the number of pairs in each quartile; the quartile for each pair is based on its predicted activity across all PFS nucleotides, not only the PFS for each plot. **(d)** Guide-target pairs colored by PFS. Same data as in Fig. 2e. **(e)** Same as (b), except separated by different Hamming distances between guide and target. **(f)** Same as (c), except separated by Hamming distance. **(g)** Same as (d), except colored by Hamming distance. In (a), (b), and (e), ρ is Spearman correlation. In (c) and (f), boxes show quartiles and whiskers extend, past the low/high quartiles, up to 1.5 times the interquartile range (outliers not shown).



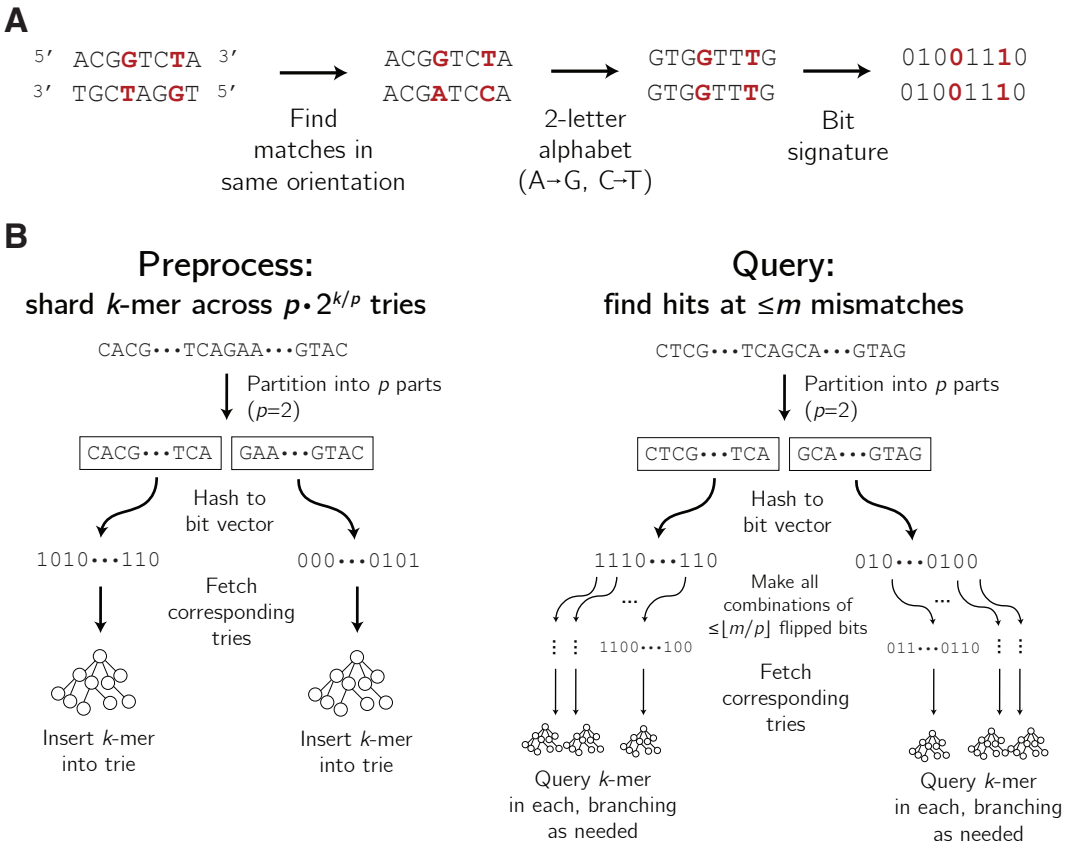
Supplementary Figure 16 — Learning curves. Learning curves for the convolutional neural networks used in ADAPT. At each number of input training data points, we perform nested cross-validation to select models: on each of five outer folds, we perform a five-fold cross-validated hyperparameter search to select a model. Line indicates the mean of a statistic on the validation data across the five selected models and error bars give a 95% confidence interval. **(a)** Learning curve selecting models for classification. **(b)** Learning curve selecting models for regression.



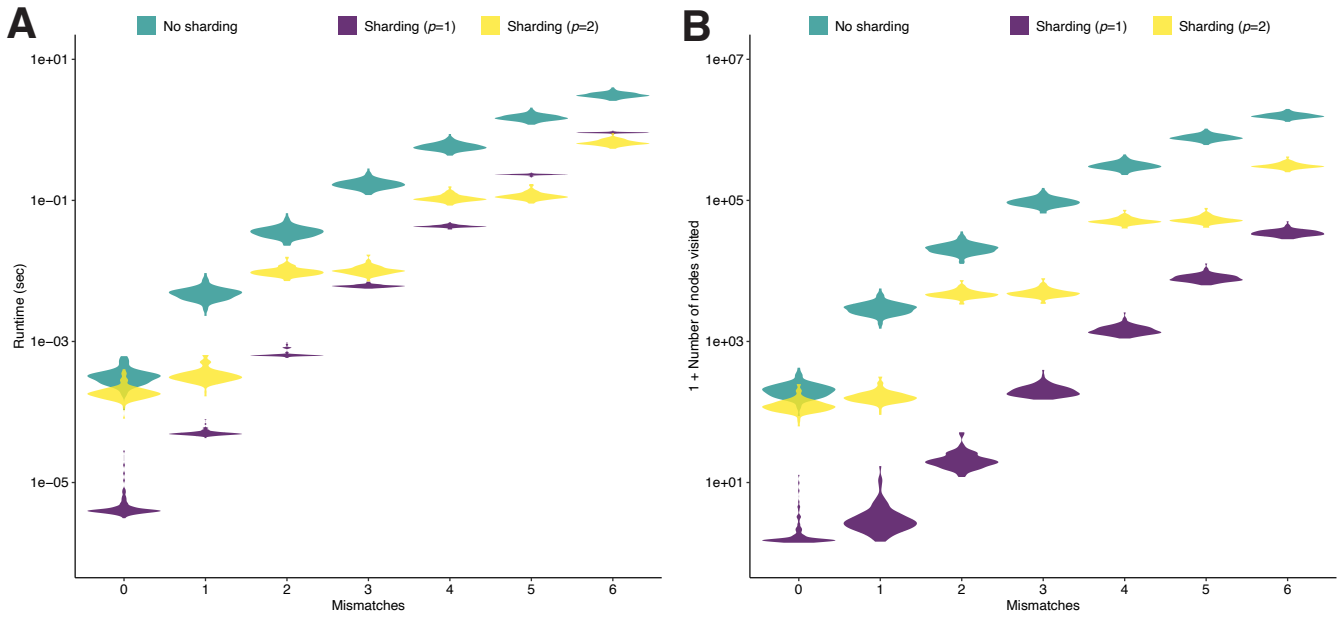
Supplementary Figure 17 — CRISPR-Cas13a guide-target activity. **(a)** Fraction of guide-target pairs that are active for each 2-nt protospacer flanking site allele (PFS; i.e., the canonical Cas13a PFS together with the nucleotide adjacent on the 3′ side of the protospacer). This analysis considers only matching guide-target pairs (i.e., no mismatches) and determines a pair to be active if the median $\log(k)$ value across replicates is > -2. Error bars represent 95% exact binomial confidence intervals. **(b)** Density of activity for different numbers of mismatches between guides and targets. Here, the number of mismatches is equivalent to Hamming distance. **(c)** Profile of mismatches among guide-target pairs with similar activity. Each row in the heatmap represents a guide-target pair, ordered by activity, with those having the least activity on top; values on the left indicate activity. For each row y, we consider the 1,000 guide-target pairs with activity closest to the pair represented by y. Then, at each position x in the protospacer, we consider all mismatches at x across our dataset and calculate the fraction of them to which the 1,000 guide-target pairs, centered at y, contribute. We plot this fraction; higher values at a row indicate a preponderance of mismatches among the guide-target pairs with the activity represented by that row. **(d)** Density of guide-target pairs that have no mismatches (purple) compared to those that have at least one mismatch in the first four positions of the protospacer and no mismatch elsewhere (yellow). As in (b), here a G-U pair is counted as a mismatch.



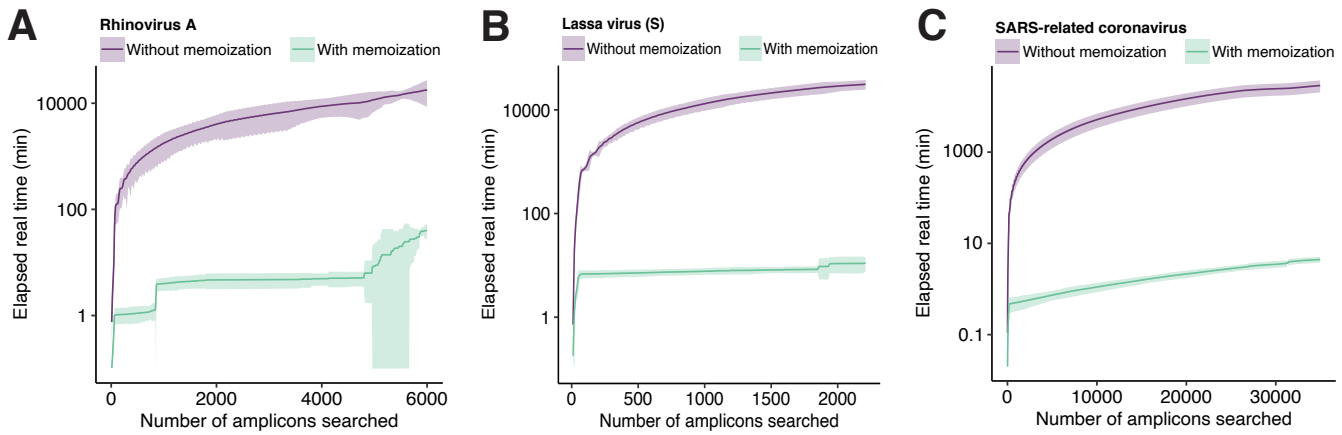
Supplementary Figure 18 — Potential hits with tolerance of G-U base pairing. Being tolerant of G-U base pairing increases the potential for non-specific hits of a *k*-mer. We built an index of ∼1 million 28-mers from 570 human-associated viral species. For each of 100 randomly selected species, we queried 28-mers for hits against the other 569 species (details in Methods). We performed this for each choice of *m* mismatches, counting a non-specific hit as one within *m* mismatches of the query, both being sensitive to G-U base pairing (purple; counting it as a match) and not being sensitive to it (green; counting it as a mismatch). Violin plots show the distribution, across the selected species, of the mean of the measured value taken over the queries for each species. **(a)** Fraction of queries that yield a non-specific hit. The measured value for a query is 0 (no hit) or 1 ($\geq$ 1 hit), so the mean represents the fraction of queries with a hit. **(b)** Number of non-specific hits per query.



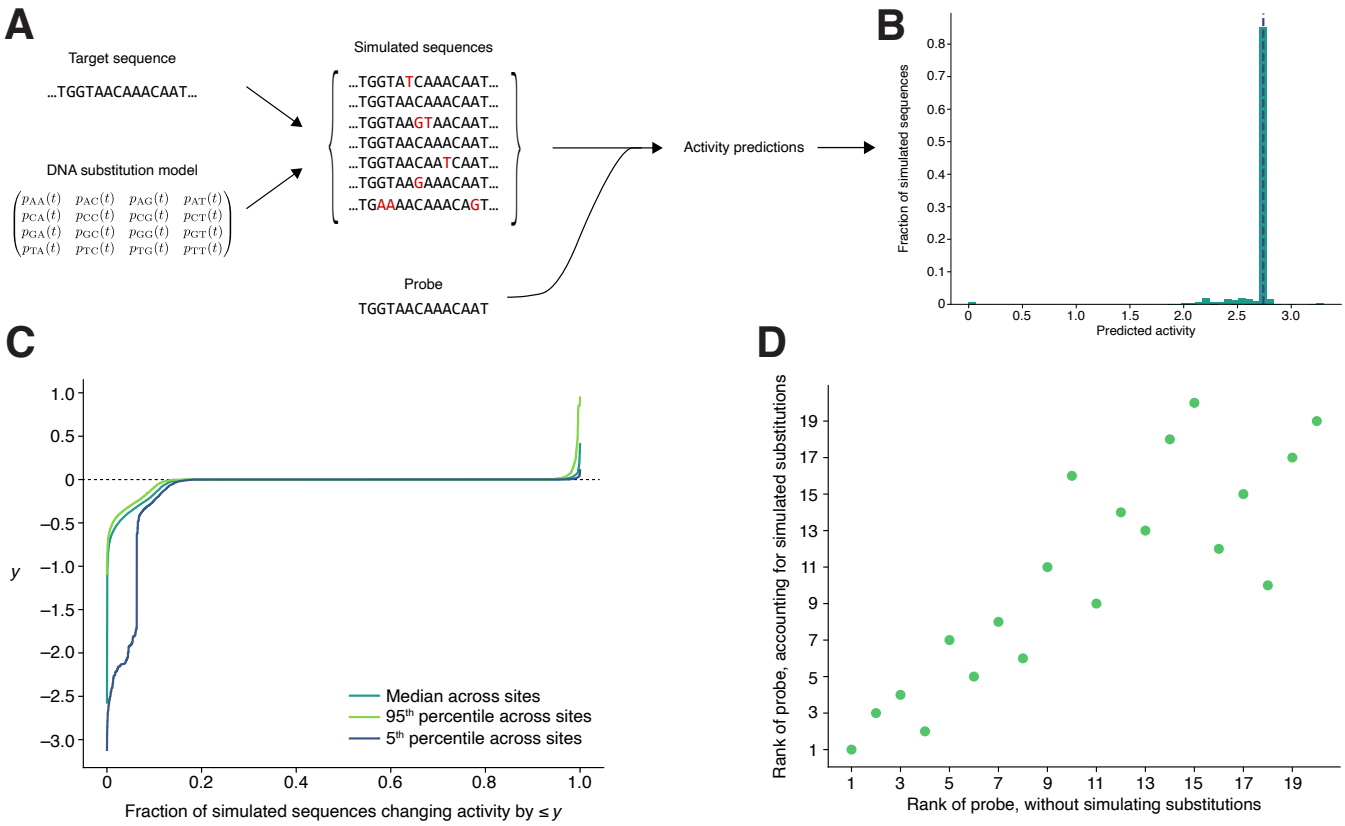
Supplementary Figure 19 — Sharding k-mers across tries for specificity queries. (a) Constructing a bit signature after transforming a string to a two-letter alphabet, described in Supplementary Note 2b. Two strings that match up to G-U base pairing (shown here as G-T) have the same bit signature. **(b)** Left: Inserting a k-mer into the data structure of tries. Each k-mer is inserted into p tries, and there are $p \cdot 2^{k/p}$ tries in total. Right: Querying a k-mer for near neighbors (within m mismatches, sensitive to G-U base pairing as a match).



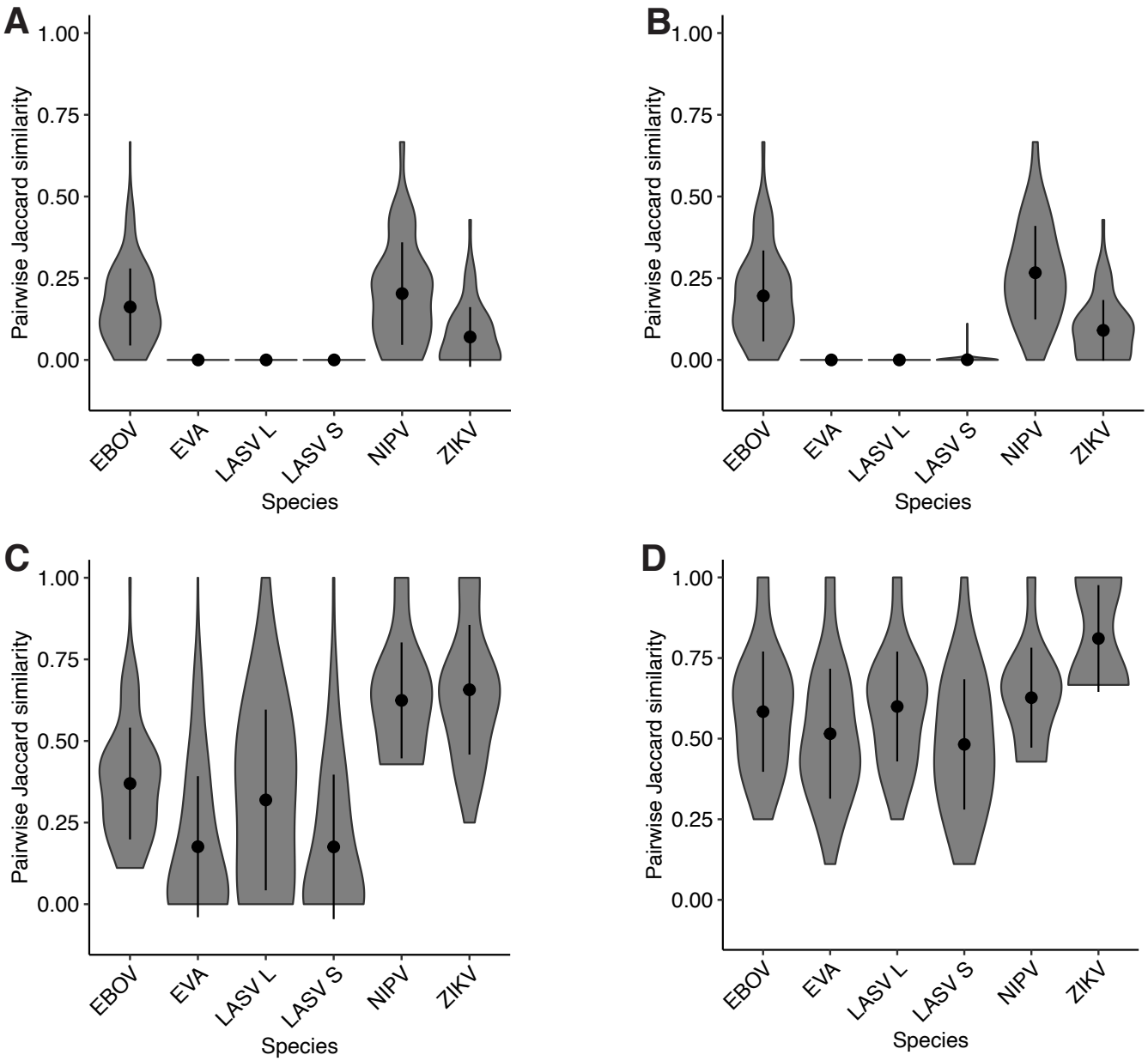
Supplementary Figure 20 — Benchmarking of specificity queries. **(a)** The runtime of querying using an index of ∼1 million 28-mers across 570 human-associated viral species. For each of 100 randomly selected species, we queried 28-mers for hits against the other 569 species. Violin plots show the distribution, across the selected species, of the mean runtime for each query. Green shows results on a single, large trie of 28-mers; purple ($p = 1$) and yellow ($p = 2$) show results on the approach described in Supplementary Note 2d, with two choices of the partition number p. **(b)** Same as (a), but showing total number of nodes visited across the trie(s). The decrease in this value using our approach suggests that parallelizing the approach—by searching within multiple tries in parallel—may provide a further speedup.



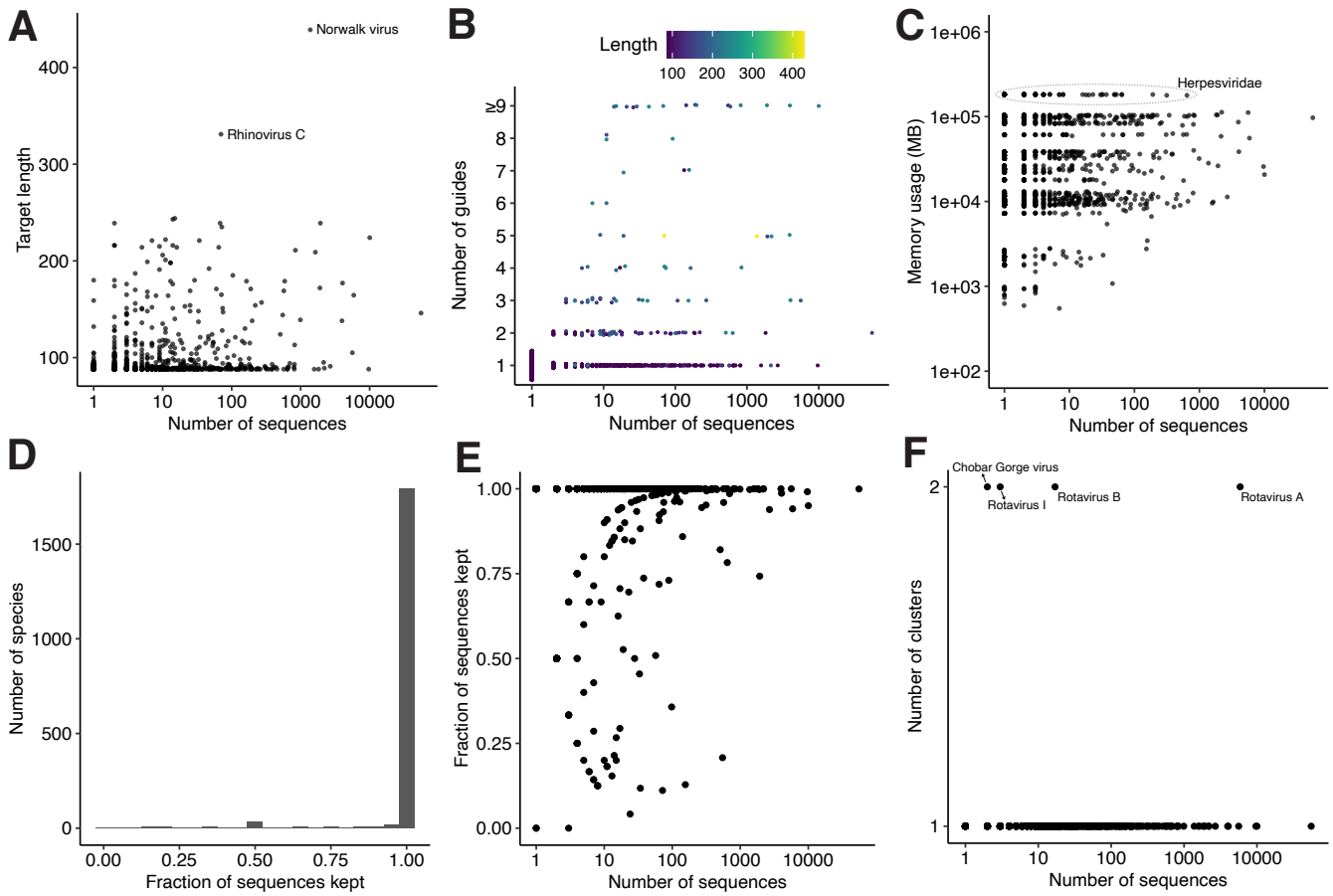
Supplementary Figure 21 — Runtime improvement provided by memoization. Runtime of ADAPT's search with and without memoizing computations, for three species. We plot the cumulative elapsed real time (minutes) at each successive window (amplicon) that ADAPT considers during its search. Shaded regions indicate a 95% confidence interval calculated across 3 runs (same input) and line is the mean. **(a)** Rhinovirus A. The lower end of the confidence interval is cutoff at 0.1. Memoization provides a 99.71% reduction in runtime (mean). **(b)** Lassa virus, segment S. Memoization provides a 99.75% reduction in runtime (mean). **(c)** SARS-related coronavirus. Memoization provides a 99.96% reduction in runtime (mean). For (b) and (c), the search without memoization was ended before its completion; thus, for these, the reduction is a lower bound assuming a continued faster growth of the runtime without memoization.



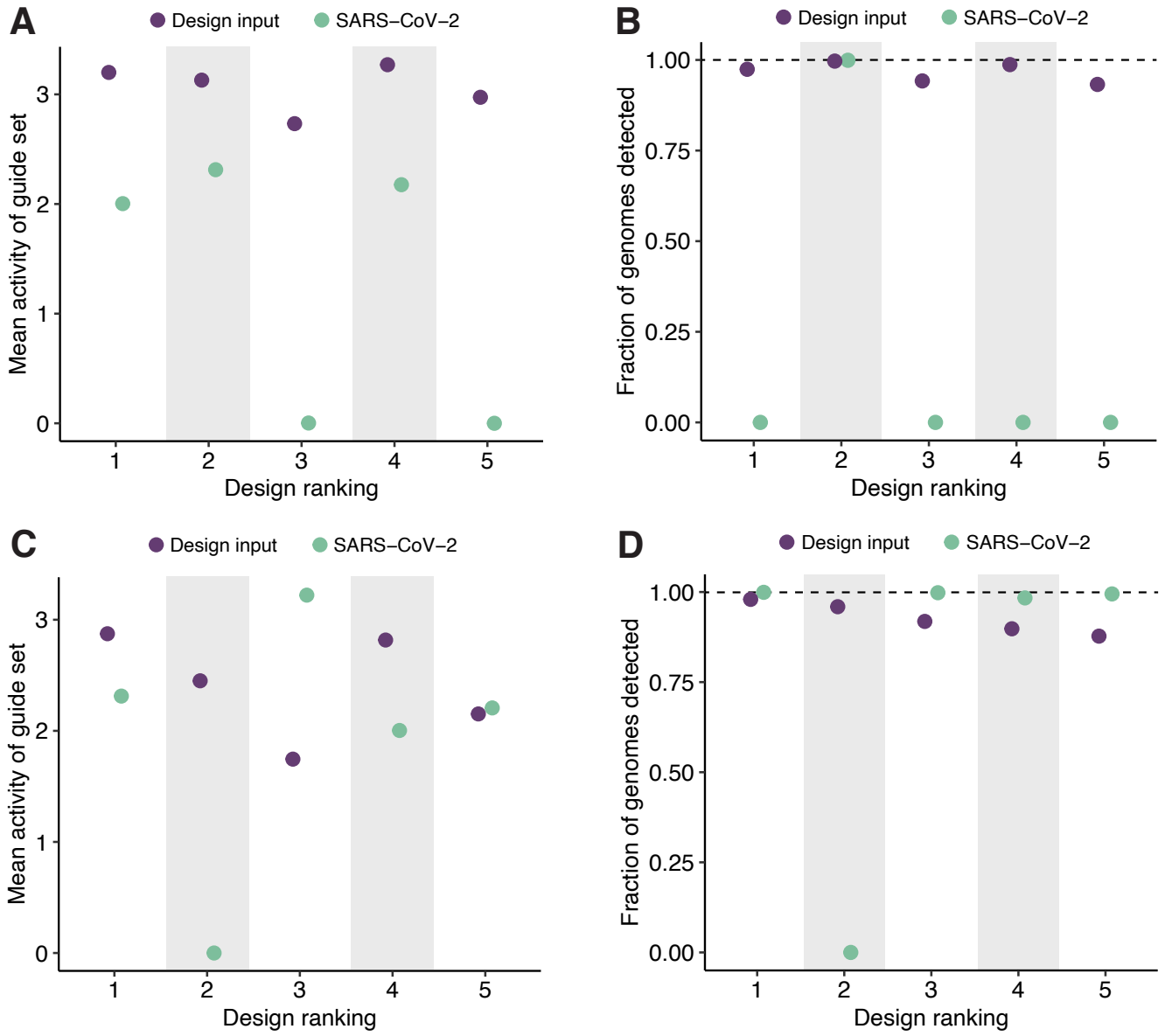
Supplementary Figure 22 — Estimating probe activity under simulated substitutions. **(a)** Sketch of proactive scheme to estimate probe performance, after a period of time, by simulating relatively likely nucleotide substitutions. Starting with a target sequence and a GTR substitution model, we sample from a distribution of substitutions made to the original target sequence. In analyses that follow, we sample after 5 years. Against each simulated sequence, we predict the detection activity of a given probe using our Cas13a predictive model; across the simulated sequences, these predictions provide a distribution of activity under potential substitutions. **(b)** Distribution of predicted detection activity across 10,000 simulated sequences that originate from a target sequence at one site in the SARS-CoV-2 genome. The probe is complementary to the original target sequence, except we randomly introduce a single mismatch. Most simulated sequences do not introduce any substitutions (i.e., they are identical to the original); the peak in the histogram (and vertical dashed line) represents these ones. Other than these simulated sequences, most simulated substitutions degrade the activity of the probe (left of the dashed line). Some enhance its activity (right of the dashed line), for example, by reversing the existing mismatch. **(c)** Inverse CDF of the change in predicted detection activity after simulating substitutions, summarized across 1,000 random sites in the SARS-CoV-2 genome; (b) showed one such site. At each of these 1,000 sites, we simulate 10,000 target sequences according to our substitution model and construct a distribution of the change in the probe's predicted detection activity compared to its activity in detecting the original sequence. As in (b), at each site the probe is complementary to the original target sequence, except with one random mismatch. Plotted is the median change taken across sites, as well as the 95$^{\text{th}}$ and 5$^{\text{th}}$ percentiles. The faster the curve rises to 0, the less likely there is to be degraded activity. That the 5$^{\text{th}}$ percentile curve shows a sharp drop for low values ($\sim$<0.1) on the horizontal axis indicates that some sites may experience a pronounced degradation in detection activity over time, but that even for these sites it is unlikely ($\sim$10% chance). **(d)** Effect of simulating substitutions on the ordering of ADAPT's designs. We begin with the top 20 design options output by ADAPT for targeting SARS-CoV-2 genomes and, for this analysis, consider only the probe (Cas13a guide) from each design option. Each point represents one of the 20 probes. We rank the probes according to their mean predicted detection activity across the genomes; this ranking is on the horizontal axis. Then, for each genome, we simulate 10,000 sequences according to our substitution model (at the site where a probe binds) and compute the 5$^{\text{th}}$ percentile of the predicted detection activities between the probe and these simulated sequences. We rank the probes accounting for simulated substitutions (vertical axis) according to the mean of this 5$^{\text{th}}$ percentile value taken across the genomes. In this analysis, we use only 500 randomly sampled genomes from the set of genomes used to design the 20 probes with ADAPT, in order to reduce runtime.



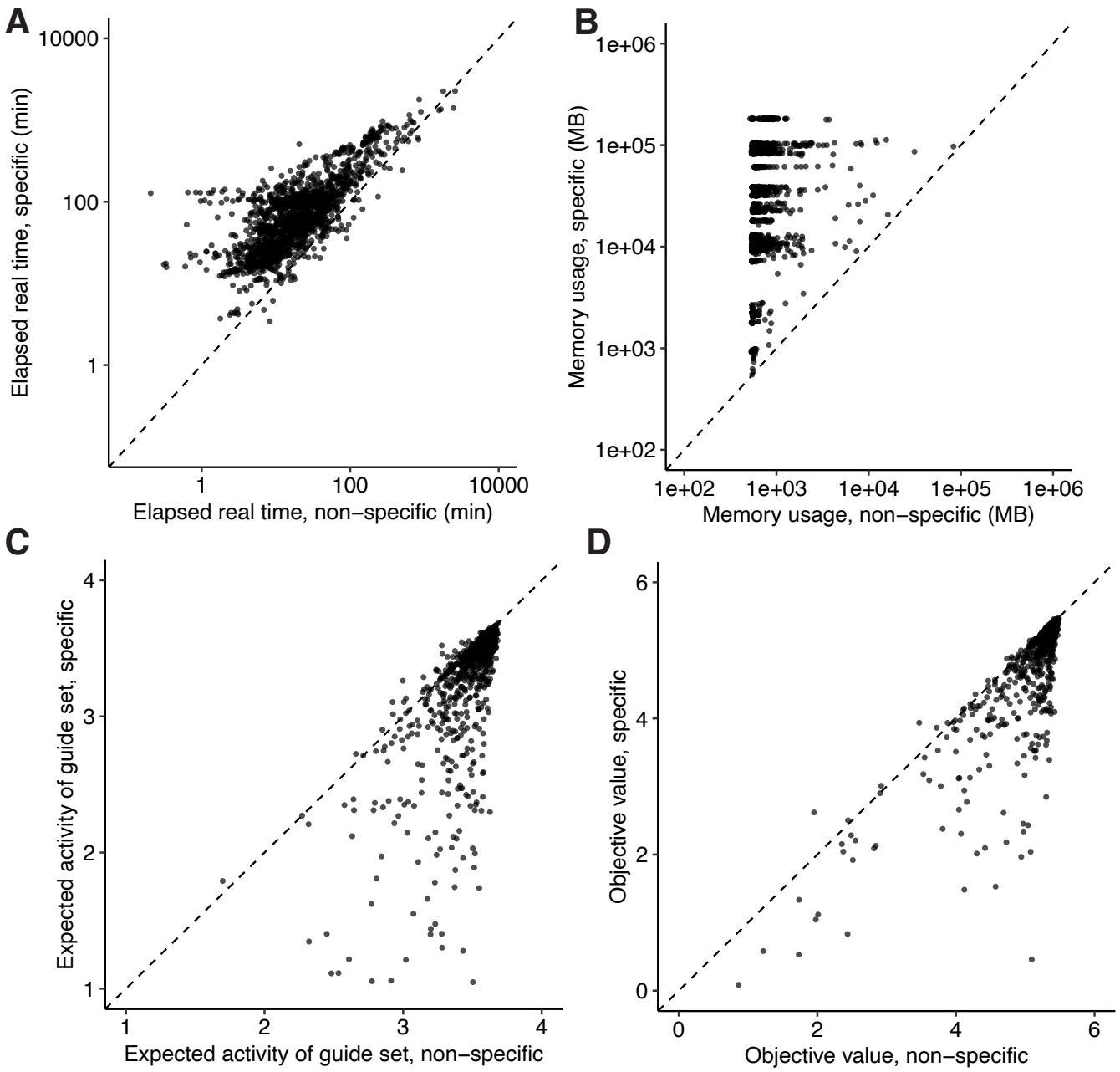
Supplementary Figure 23 — Dispersion in ADAPT's designs. For each species, we ran ADAPT 20 times. For each pair of runs, we calculated the Jaccard similarity comparing the top 5 design options from each. Violin plots show a smoothed density estimate of the pairwise Jaccard similarities; dot indicates the mean and bars show 1 standard deviation around the mean. **(a)** Using resampled input genomes for each run and considering two design options to be equal if they have exactly the same primers and probes. **(b)** Using the same input genomes for each run and considering two design options to be equal if they have exactly the same primers and probes. **(c)** Using resampled input genomes for each run and considering two design options to be equal if their endpoints are within 40 nt of each other. **(d)** Using the same input genomes for each run and considering two design options to be equal if their endpoints are within 40 nt of each other. When using resampled input genomes, the comparisons account for algorithmic randomness and input sampling. When using the same input genomes, the comparisons account only for algorithmic randomness. EBOV, Zaire ebolavirus; EVA, Enterovirus A; LASV L/S, Lassa virus segment L/S; NIPV, Nipah virus; ZIKV, Zika virus.



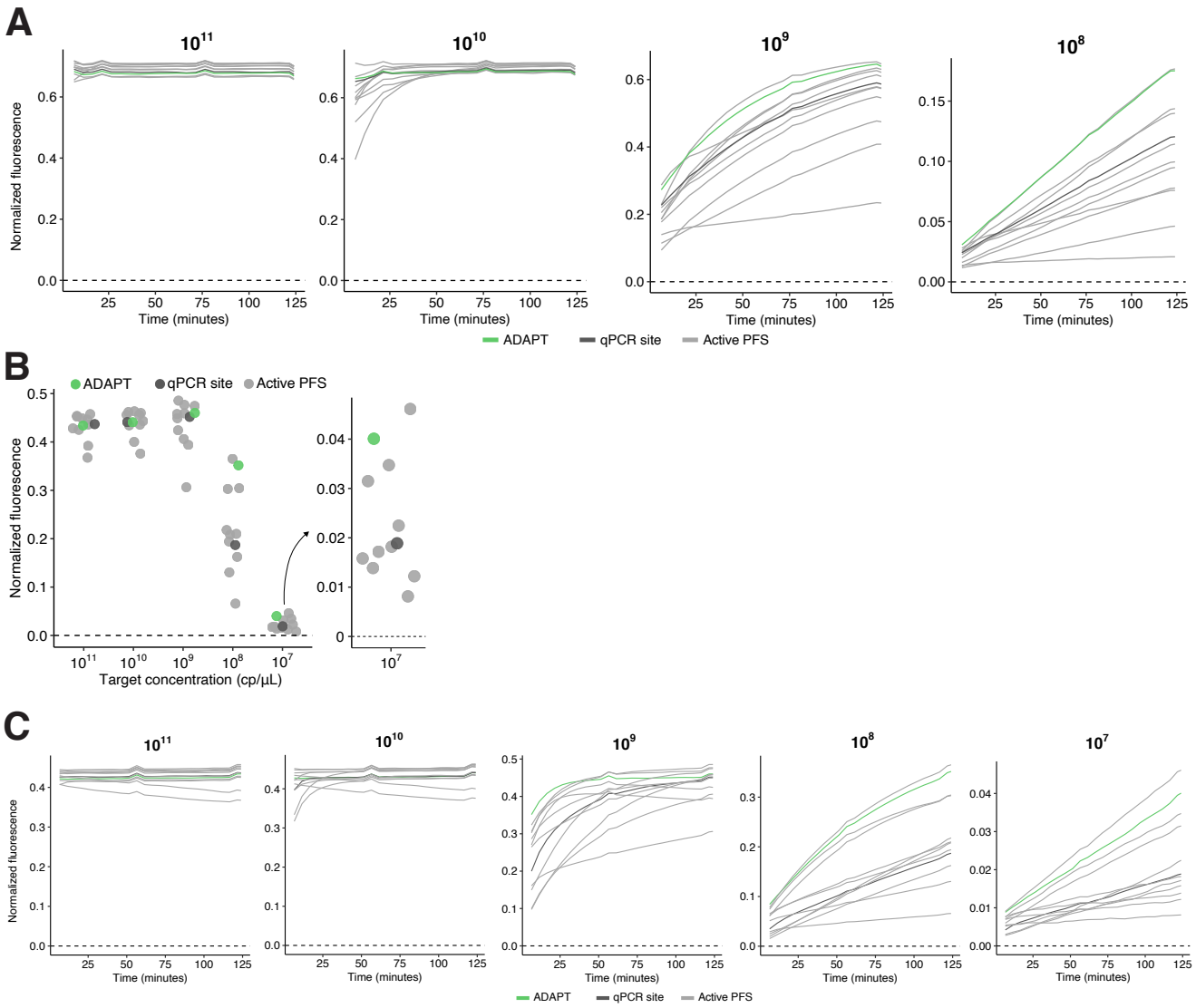
Supplementary Figure 24 — Results of ADAPT's designs for 1,926 vertebrate-infecting viruses. Running ADAPT on 1,933 vertebrate-infecting species produced designs on 1,926 (Methods). **(a)** Length of each target region (amplicon) in the highest-ranked design output by ADAPT for each species. As part of the design we restricted the length to $\leq$ 250-nt for all species except two (Methods). Horizontal axis is the number of input sequences for design. **(b)** Number of Cas13a guides in the highest-ranked design option for each species, produced using the objective function in which we minimize the number of guides subject to detecting $>$ 98% of sequences with high activity. Color indicates the length of the targeted region (amplicon) in the design. **(c)** Maximum resident set size (RSS), in MB, of the process running ADAPT on each species. **(d)** Distribution, across species, of the fraction of input sequences passing curation. **(e)** Fraction of input sequences passing curation for each species compared the number of input sequences for that species. **(f)** Number of clusters for each species compared to the number of input sequences for that species.



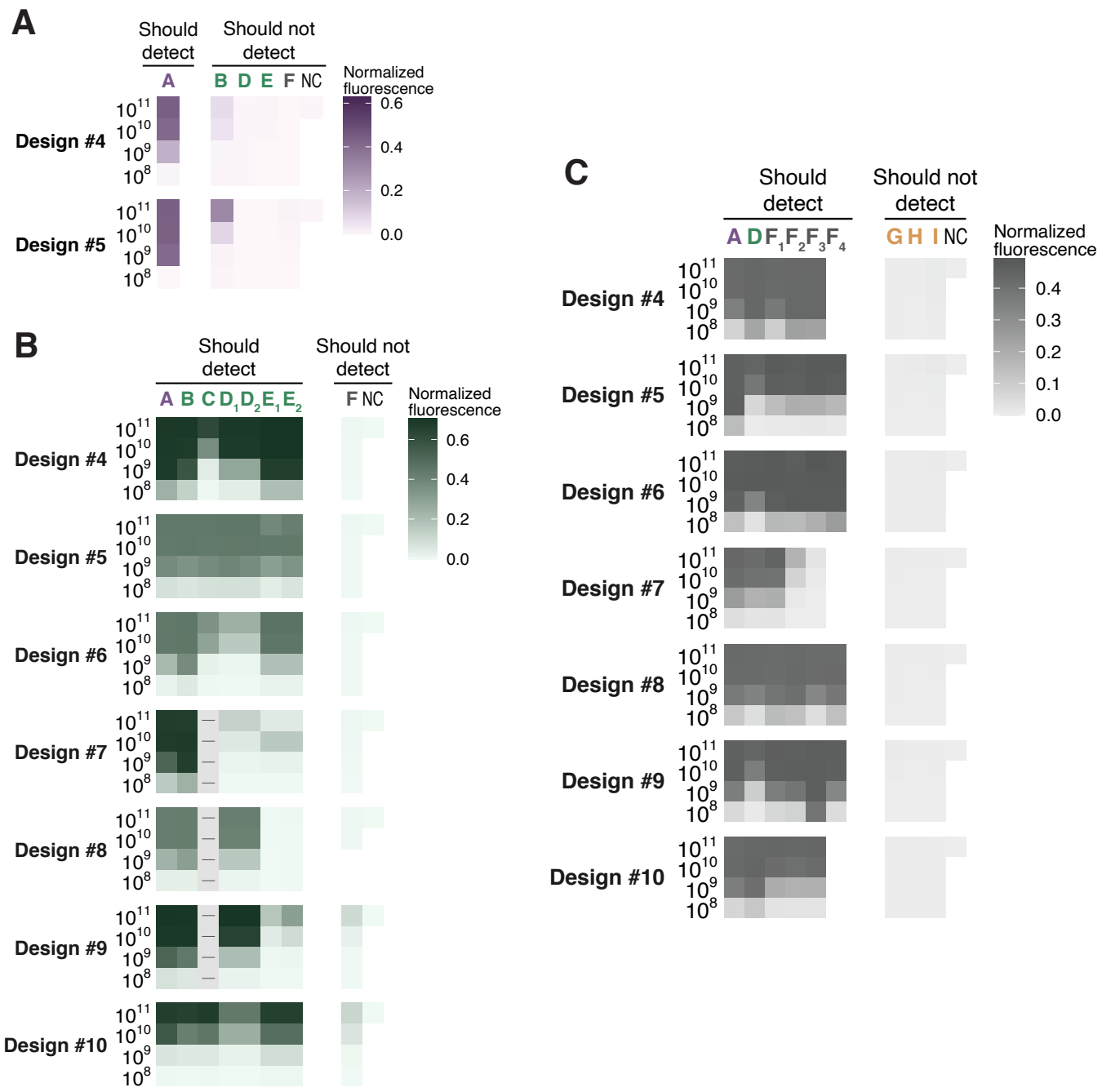
Supplementary Figure 25 — Evaluation of SARS-CoV-2 detection with assays designed, using genomes through 2018, to detect SARS-related coronavirus. The SARS-related CoV designs were generated using genomes available through the end of 2018, which simulates the design of broadly-effective assays a year before SARS-CoV-2's emergence. SARS-related CoV is a species that encompasses SARS-CoV-2, as well as SARS-CoV-1 and viruses sampled from animals. **(a)** Performance of Cas13a guides from each of the five highest-ranked design outputs from ADAPT (ordered by ranking; 1 is best). Points indicate the mean predicted activity of each design's guides in detecting targeted genomes. Purple, mean across the 311 genomes used for the design (all SARS-related CoV genomes through the end of 2018). Green, mean across the 184,197 SARS-CoV-2 genomes available through November 12, 2020. **(b)** Fraction of genomes predicted to be detected by each design's assay, accounting for both the primers and guides in the assay (details in Methods). Designs were produced as in (a) and colors are as in (a). **(c)** Same as (a), except the designs used input that downsampled SARS-CoV-1 to a single genome. Purple, mean across the 49 genomes used for the design. Green, mean across the 184,197 SARS-CoV-2 genomes available through November 12, 2020. **(d)** Fraction of genomes predicted to be detected by each design's assay, accounting for both the primers and guides in the assay. Designs were produced as in (c) and colors are as in (c). In (a) and (c), values at 0 indicate Cas13a guides that are classified to be inactive; values above 0 are classified to be active.



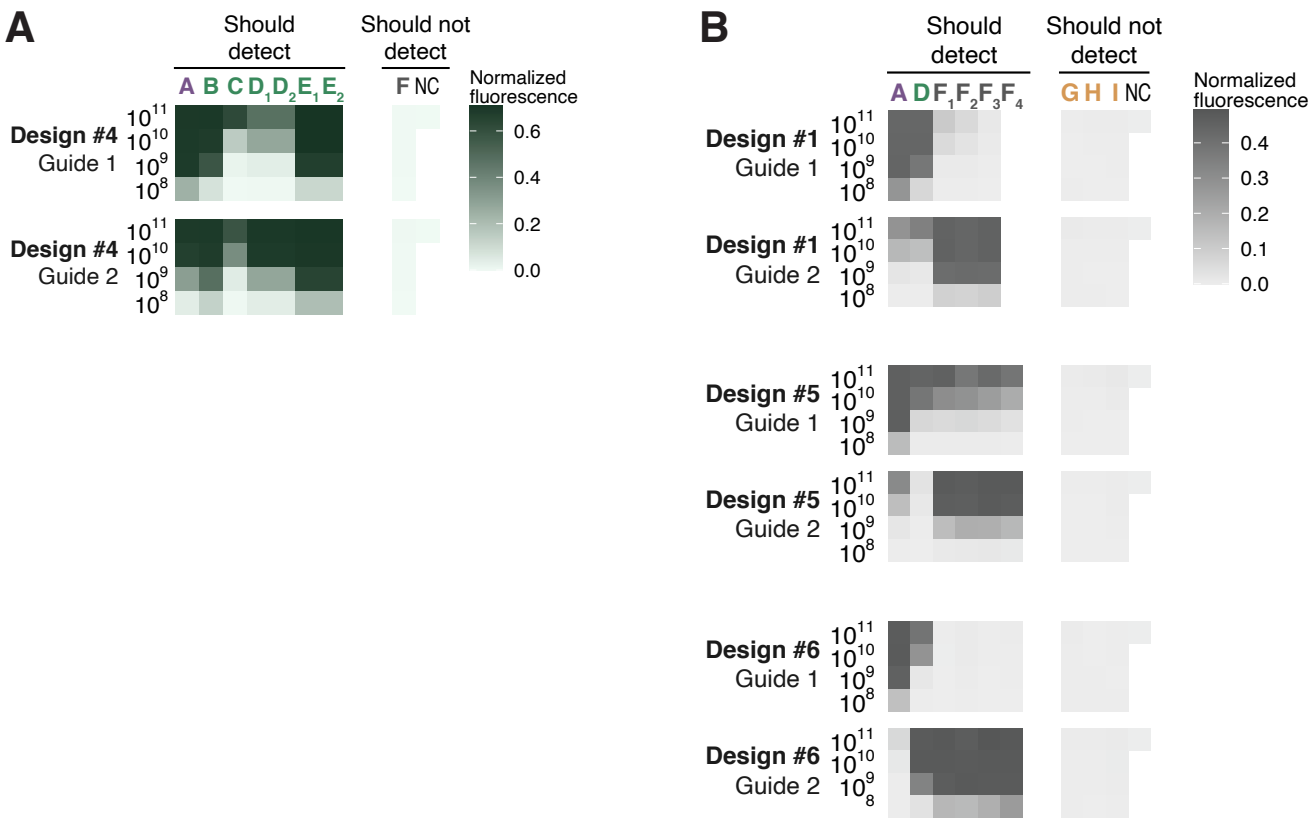
Supplementary Figure 26 — Effects of enforcing specificity on ADAPT's designs for 1,926 vertebrate-infecting viruses. In each panel, each point is a species and comparisons are with and without enforcing species-level specificity within each family. **(a)** End-to-end elapsed real time running ADAPT. **(b)** Maximum resident set size (RSS), in MB, of the process running ADAPT. **(c)** Mean activity of the guide set, from the highest-ranked design option, across input sequences. **(d)** Objective value of the highest-ranked design option, which incorporates expected activity of the guide set, the number of primers, and the target region length. Not shown, 9 species with objective value < 0. In all panels, 1,926 species are shown (7 of the 1,933 vertebrate-infecting species did not produce designs; Methods).



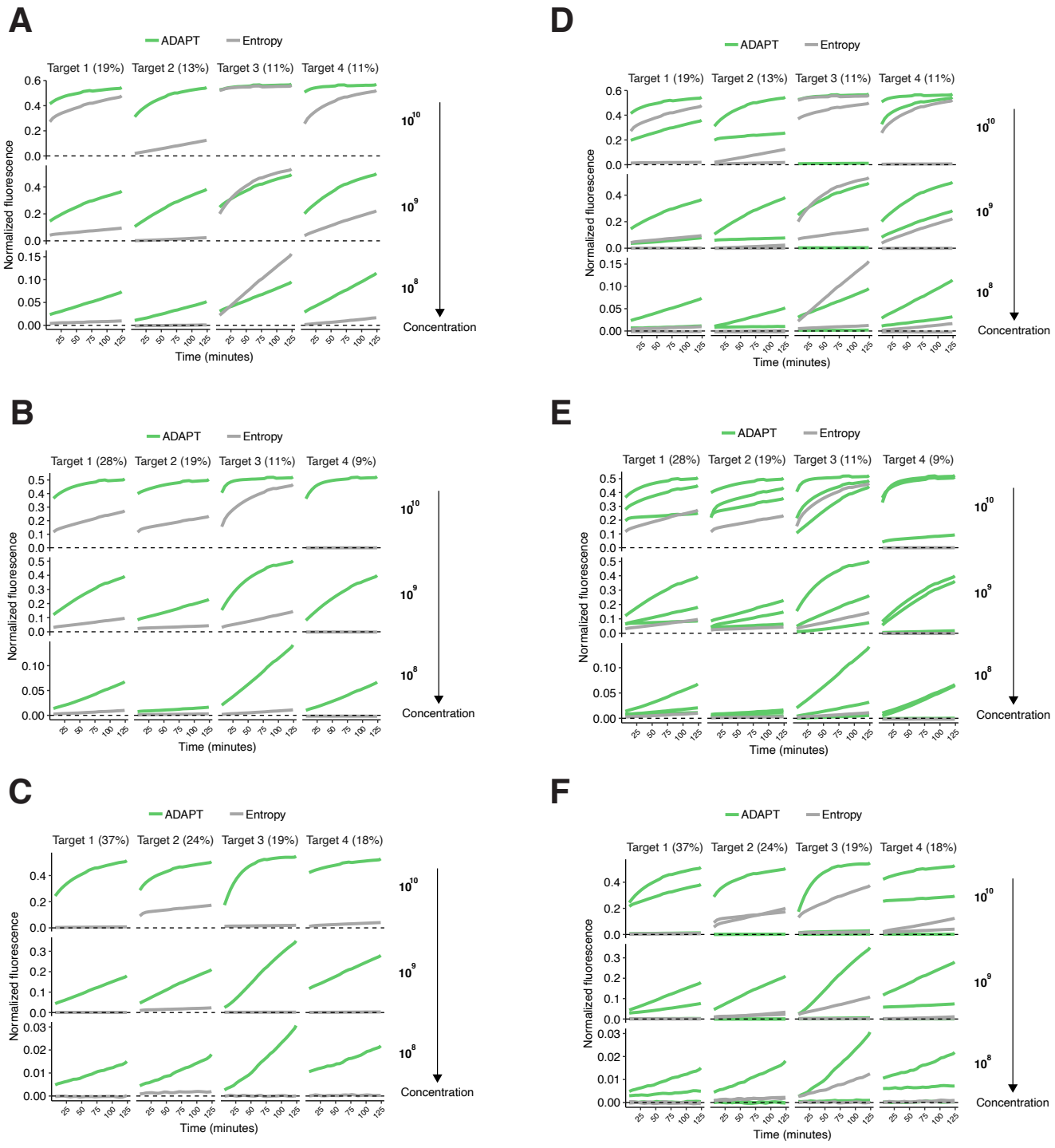
Supplementary Figure 27 — Benchmarking ADAPT's designs in US CDC SARS-CoV-2 amplicons. (a) Fluorescence over time for Cas13 guides at varying target concentrations (top of each plot, in cp/μL) within the US CDC's SARS-CoV-2 N1 RT-qPCR amplicon. Compared guides are ADAPT's design (green), a guide with an active (non-G) PFS at the site of the qPCR probe, and 10 randomly selected guides with an active PFS. Target concentration of 10^7 cp/μL is shown in Fig. 4b. **(b)** Fluorescence for Cas13 guides at varying target concentrations within the US CDC's SARS-CoV-2 N2 RT-qPCR amplicon. The final time point is shown (124 minutes). Guides are as in (a). **(c)** Fluorescence over time for Cas13 guides at varying target concentrations (top of each plot, in cp/μL) within the US CDC's SARS-CoV-2 N2 RT-qPCR amplicon.



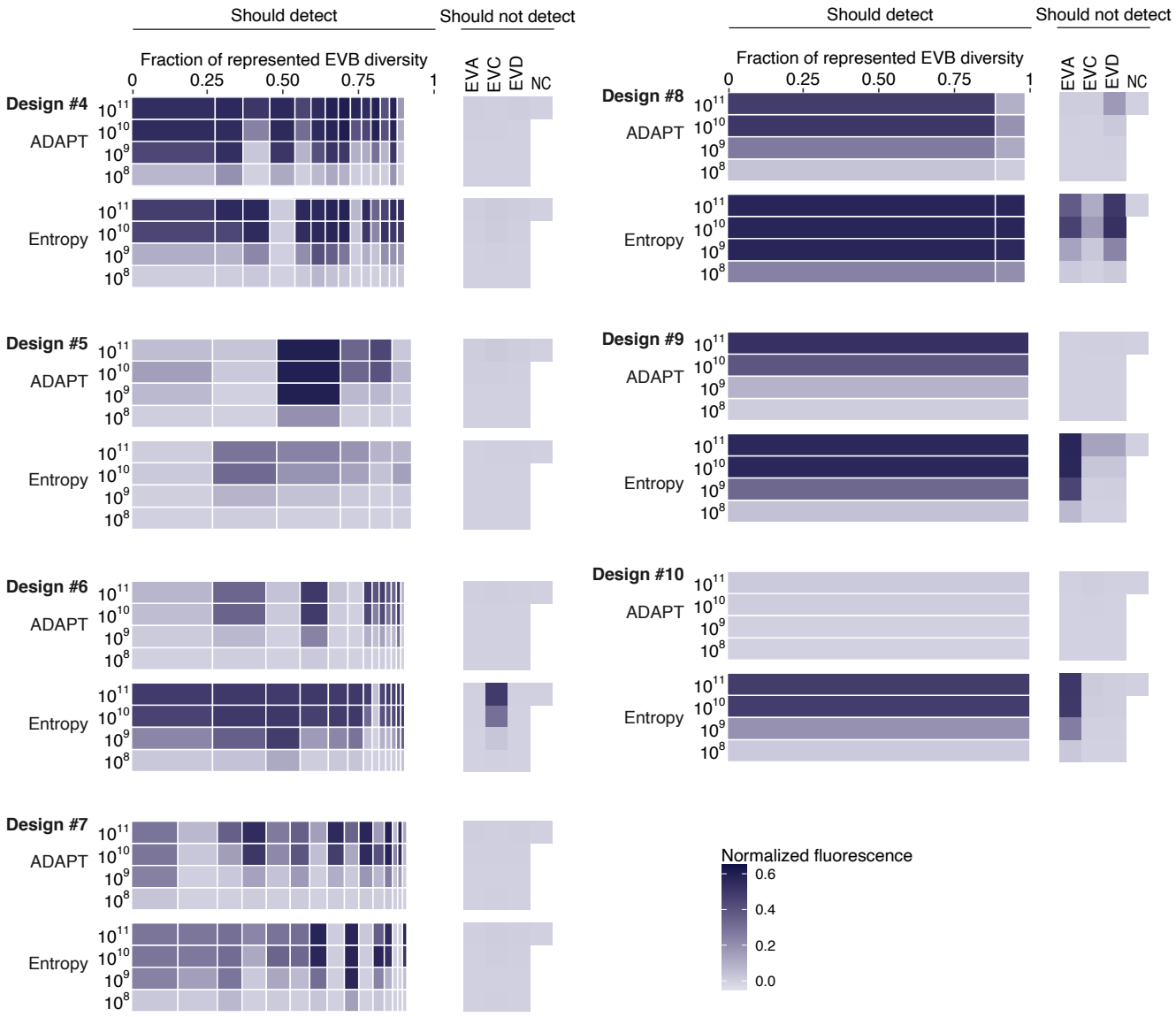
Supplementary Figure 28 — Sensitivity and specificity of additional designs for SARS-related taxa. Fluorescence for ADAPT's designs specific to **(a)** SARS-CoV-2, **(b)** SARS-CoV-2–related, and **(c)** SARS-related coronavirus species. Fig. 4c shows phylogenetic relationships of these taxa. Assays ranked from 4 through 10 are shown (only 5 tested for SARS-CoV-2); the top 3 are shown in Fig. 4. Target definitions are in Fig. 4c and the Fig. 4 legend provides additional details about each panel. In (c), clade F required a fourth representative target (F_4) in only some amplicons.



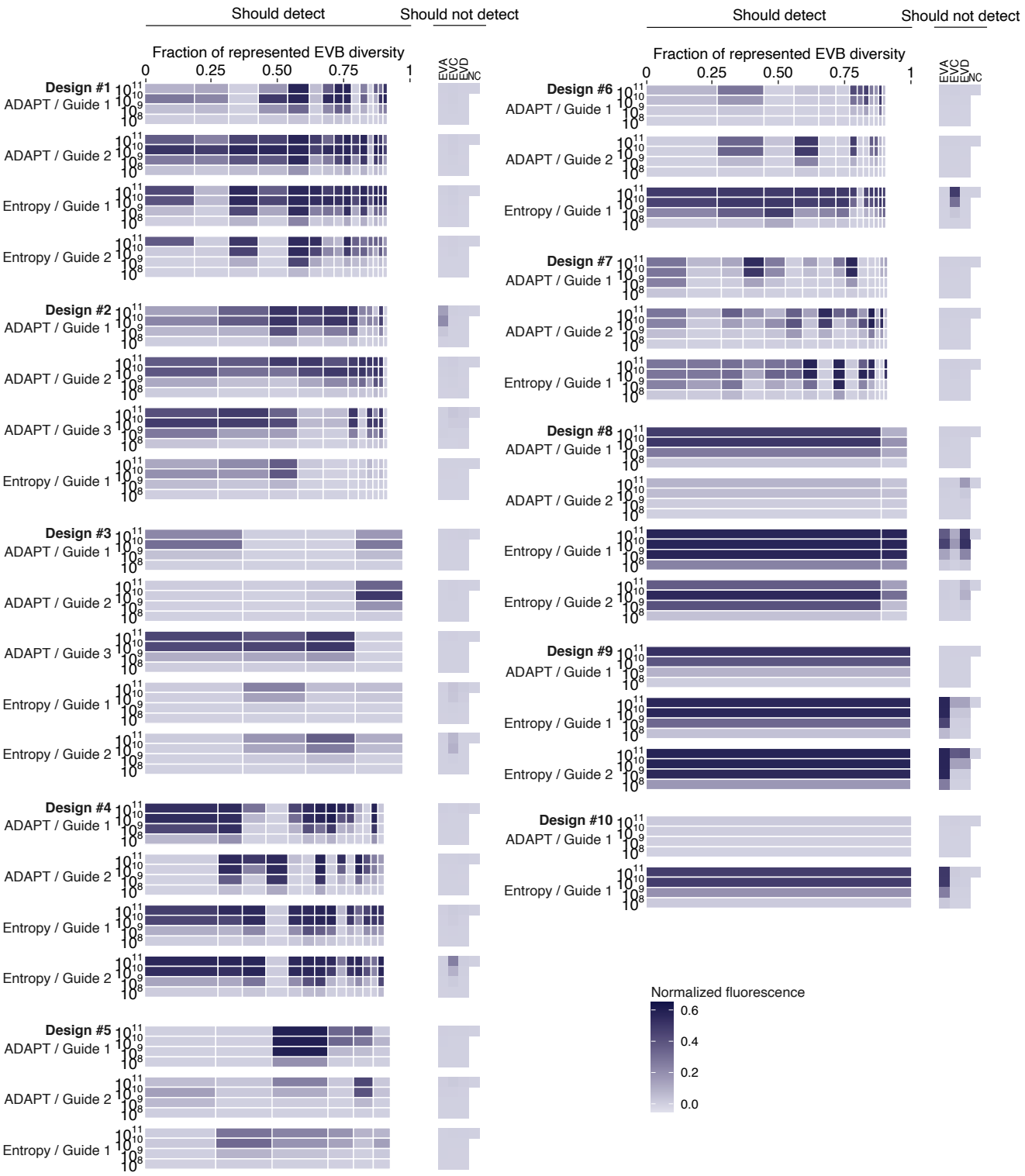
Supplementary Figure 29 — Separate guides for multi-guide designs in SARS-related taxa. Fluorescence for ADAPT's design options that use more than one Cas13 guide, separated by guide. **(a)** The two guides in Design #4 to detect SARS-CoV-2–related taxon (Supplementary Fig. 28b). **(b)** The two guides in Design #1 (Fig. 4e) and in Designs #5 and #6 (Supplementary Fig. 28c) to detect the SARS-related coronavirus species. In other figures, plotted value is the maximum across multiple guides.



Supplementary Figure 30 — Kinetic curves of designs for detecting Enterovirus B. (a–c) Fluorescence over time for ADAPT's designs in detecting EVB at varying target concentrations (right of each plot in cp/µL), for the 4 targets representing the largest fraction of EVB genomic diversity within the corresponding amplicon. (a) Design #1 (highest ranked output design); (b) Design #2; (c) Design #3. Plots at the target concentration of 10^8 cp/µL are also shown in Fig. 4h. The entropy-based guide targets the site with an active PFS and minimal entropy. **(d–f)** Same as (a–c) except with a separate line for each guide, when there are multiple guides in ADAPT's design or 2 guides tested for the entropy-based approach. (d) Design #1; (e) Design #2; (f) Design #3. When there are two entropy-based guides, they have an active PFS and the least and second-least entropy in the amplicon of ADAPT's design. In other figures, plotted value is the maximum across multiple guides.



Supplementary Figure 31 — Sensitivity and specificity of additional designs for Enterovirus B. Fluorescence for ADAPT's Enterovirus B (EVB) designs in detecting EVB and representative targets for Enterovirus A/C/D (EVA/C/D). Assays ranked from 4 through 10 are shown; the top 3 are shown in Fig. 4g. Each band is an EVB target having width proportional to the fraction of EVB genomic diversity represented by the target, within the amplicon of ADAPT's design. Immediately under each ADAPT design is a guide ("Entropy") from the site in the amplicon with an active PFS and minimal entropy.



Supplementary Figure 32 — Separate guides in designs for detecting Enterovirus B. Fluorescence for ADAPT's Enterovirus B (EVB) designs in detecting EVB and representative targets for Enterovirus A/C/D (EVA/C/D), separated by guide. Each band is an EVB target having width proportional to the fraction of EVB genomic diversity represented by the target, within the amplicon of ADAPT's design. Immediately under each ADAPT design is a guide ("Entropy") from the site in the amplicon with an active PFS and minimal entropy; when there are two Entropy guides, the second is from the site with an active PFS and the second-least entropy. In Fig. 4g and Supplementary Fig. 31, plotted value is the maximum across multiple guides.

References

[1] Gootenberg, J. S. *et al.* Nucleic acid detection with CRISPR-Cas13a/C2c2. *Science* **356**, 438–442 (2017).

[2] Gootenberg, J. S. *et al.* Multiplexed and portable nucleic acid detection platform with Cas13, Cas12a, and Csm6. *Science* **360**, 439–444 (2018).

[3] Myhrvold, C. *et al.* Field-deployable viral diagnostics using CRISPR-Cas13. *Science* **360**, 444–448 (2018).

[4] Chen, J. S. *et al.* CRISPR-Cas12a target binding unleashes indiscriminate single-stranded DNase activity. *Science* **360**, 436–439 (2018).

[5] Chiu, C. Cutting-Edge infectious disease diagnostics with CRISPR. *Cell Host & Microbe* **23**, 702–704 (2018).

[6] Pardee, K. *et al.* Paper-based synthetic gene networks. *Cell* **159**, 940–954 (2014).

[7] Pardee, K. *et al.* Rapid, low-cost detection of Zika virus using programmable biomolecular components. *Cell* **165**, 1255–1266 (2016).

[8] Stellrecht, K. A. The drift in molecular testing for influenza: Mutations affecting assay performance. *Journal of Clinical Microbiology* **56** (2018).

[9] Overmeire, Y. *et al.* Severe sensitivity loss in an influenza A molecular assay due to antigenic drift variants during the 2014/15 influenza season. *Diagnostic microbiology and infectious disease* **85**, 42–46 (2016).

[10] Klungthong, C. *et al.* The impact of primer and probe-template mismatches on the sensitivity of pandemic influenza A/H1N1/2009 virus detection by real-time RT-PCR. *Journal of Clinical Virology: the official publication of the Pan American Society for Clinical Virology* **48**, 91–95 (2010).

[11] Brault, A. C., Fang, Y., Dannen, M., Anishchenko, M. & Reisen, W. K. A naturally occurring mutation within the probe-binding region compromises a molecular-based West Nile virus surveillance assay for mosquito pools (diptera: Culicidae). *Journal of Medical Entomology* **49**, 939–941 (2012).

[12] Lee, H. K. *et al.* Missed diagnosis of influenza B virus due to nucleoprotein sequence mutations, Singapore, april 2011. *Euro surveillance: bulletin Europeen sur les maladies transmissibles = European communicable disease bulletin* **16** (2011).

[13] Cattoli, G. *et al.* False-negative results of a validated real-time PCR protocol for diagnosis of newcastle disease due to genetic variability of the matrix gene. *Journal of Clinical Microbiology* **47**, 3791–3792 (2009).

[14] Lengerova, M. *et al.* Real-time PCR diagnostics failure caused by nucleotide variability within exon 4 of the human cytomegalovirus major immediate-early gene. *Journal of Clinical Microbiology* **45**, 1042–1044 (2007).

[15] Stevenson, J., Hymas, W. & Hillyard, D. Effect of sequence polymorphisms on performance of two real-time PCR assays for detection of herpes simplex virus. *Journal of Clinical Microbiology* **43**, 2391–2398 (2005).

[16] Linhart, C. & Shamir, R. The degenerate primer design problem. *Bioinformatics* **18 Suppl 1**, S172–81 (2002).

[17] Fitch, J. P. *et al.* Rapid development of nucleic acid diagnostics. *Proceedings of the IEEE* **90**, 1708–1721 (2002).

[18] Jabado, O. J. *et al.* Greene SCPrimer: a rapid comprehensive tool for designing degenerate primers from multiple sequence alignments. *Nucleic Acids Research* **34**, 6605–6611 (2006).

[19] Vijaya Satya, R., Kumar, K., Zavaljevski, N. & Reifman, J. A high-throughput pipeline for the design of real-time PCR signatures. *BMC Bioinformatics* **11**, 340 (2010).

[20] Karim, S. *et al.* Development of the automated primer design workflow uniqprimer and diagnostic primers for the Broad-Host-Range plant pathogen Dickeya dianthicola. *Plant Disease* **103**, 2893–2902 (2019).

[21] Duitama, J. *et al.* PrimerHunter: a primer design tool for PCR-based virus subtype identification. *Nucleic Acids Research* **37**, 2483–2492 (2009).

[22] Zheng, J. *et al.* OligoSpawn: a software tool for the design of overgo probes from large unigene datasets. *BMC Bioinformatics* **7**, 7 (2006).

[23] Brodin, J. *et al.* A multiple-alignment based primer design algorithm for genetically highly variable DNA targets. *BMC Bioinformatics* **14**, 255 (2013).

[24] Wright, E. S. & Vetsigian, K. H. DesignSignatures: a tool for designing primers that yields amplicons with distinct signatures. *Bioinformatics* **32**, 1565–1567 (2016).

[25] Wright, E. S. *et al.* Exploiting extension bias in polymerase chain reaction to improve primer specificity in ensembles of nearly identical DNA templates. *Environmental Microbiology* **16**, 1354–1365 (2014).

[26] Kreer, C. *et al.* openPrimeR for multiplex amplification of highly diverse templates. *Journal of Immunological Methods* **480**, 112752 (2020).

[27] Roux, S. *et al.* Minimum information about an uncultivated virus genome (MIUViG). *Nature Biotechnology* **37**, 29–37 (2019).

[28] Brister, J. R., Ako-Adjei, D., Bao, Y. & Blinkova, O. NCBI viral genomes resource. *Nucleic Acids Research* **43**, D571–7 (2015).

[29] Vanaerschot, M. *et al.* Identification of a polymorphism in the N gene of SARS-CoV-2 that adversely impacts detection by a widely-used RT-PCR assay. *bioRxiv* 2020.08.25.265074 (2020).

[30] Artesi, M. *et al.* A recurrent mutation at position 26340 of SARS-CoV-2 is associated with failure of the E gene quantitative reverse transcription-PCR utilized in a commercial dual-target diagnostic assay. *Journal of Clinical Microbiology* **58** (2020).

[31] Indyk, P. & Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, 604–613 (ACM, New York, NY, USA, 1998).

[32] Buchbinder, N., Feldman, M., Naor, J. s. & Schwartz, R. Submodular maximization with cardinality constraints. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on*

Discrete algorithms, SODA '14, 1433–1452 (Society for Industrial and Applied Mathematics, USA, 2014).

[33] Nemhauser, G. L., Wolsey, L. A. & Fisher, M. L. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming. A Publication of the Mathematical Programming Society* **14**, 265–294 (1978).

[34] Abudayyeh, O. O. *et al.* C2c2 is a single-component programmable RNA-guided RNA-targeting CRISPR effector. *Science* **353**, aaf5573 (2016).

[35] Abudayyeh, O. O. *et al.* RNA targeting with CRISPR-Cas13. *Nature* **550**, 280–284 (2017).

[36] Tambe, A., East-Seletsky, A., Knott, G. J., Doudna, J. A. & O'Connell, M. R. RNA binding and HEPN-Nuclease activation are decoupled in CRISPR-Cas13a. *Cell Reports* **24**, 1025–1036 (2018).

[37] Ackerman, C. M. *et al.* Massively multiplexed nucleic acid detection with Cas13. *Nature* **582**, 277–282 (2020).

[38] Kim, H. K. *et al.* Deep learning improves prediction of CRISPR-Cpf1 guide RNA activity. *Nature Biotechnology* **36**, 239–241 (2018).

[39] Liu, L. *et al.* The molecular architecture for RNA-Guided RNA cleavage by Cas13a. *Cell* **170**, 714–726.e10 (2017).

[40] Wessels, H.-H. *et al.* Massively parallel Cas13 screens reveal principles for guide RNA design. *Nature Biotechnology* **38**, 722–727 (2020).

[41] Tavaré, S. Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on mathematics in the life sciences* (1986).

[42] Broughton, J. P. *et al.* CRISPR-Cas12-based detection of SARS-CoV-2. *Nature Biotechnology* (2020).

[43] Zhou, P. *et al.* A pneumonia outbreak associated with a new coronavirus of probable bat origin. *Nature* (2020).

[44] Lam, T. T.-Y. *et al.* Identifying SARS-CoV-2-related coronaviruses in Malayan pangolins. *Nature* (2020).

[45] Khetsuriani, N., Lamonte-Fowlkes, A., Oberst, S., Pallansch, M. A. & Centers for Disease Control and Prevention. Enterovirus surveillance–United States, 1970-2005. *Morbidity and mortality weekly report. Surveillance summaries* **55**, 1–20 (2006).

[46] Zell, R. *et al.* ICTV virus taxonomy profile: Picornaviridae. *The Journal of general virology* **98**, 2421–2422 (2017).

[47] WHO Regional Office for Europe and the United States Centers for Disease Control and Prevention. Enterovirus surveillance guidelines (2015).

[48] Tan, C. Y. Q. *et al.* A retrospective overview of enterovirus infection diagnosis and molecular epidemiology in the public hospitals of Marseille, France (1985-2005). *PLOS One* **6**, e18022 (2011).

[49] Metsky, H. C., Freije, C. A., Kosoko-Thoroddsen, T.-S. F., Sabeti, P. C. & Myhrvold, C. CRISPR-based surveillance for COVID-19 using genomically-comprehensive machine learning design. *bioRxiv* 2020.02.26.967026 (2020).

[50] Shu, Y. & McCauley, J. GISAID: Global initiative on sharing all influenza data - from vision to reality. *Euro surveillance: bulletin Europeen sur les maladies transmissibles = European communicable disease bulletin* **22** (2017).

[51] Barnes, K. G. *et al.* Deployable CRISPR-Cas13a diagnostic tools to detect and report Ebola and Lassa virus cases in real-time. *Nature Communications* **11**, 4131 (2020).

[52] MacKay, M. J. *et al.* The COVID-19 XPRIZE and the need for scalable, fast, and widespread testing. *Nature Biotechnology* **38**, 1021–1024 (2020).

[53] Vogels, C. B. F. *et al.* Analytical sensitivity and efficiency comparisons of SARS-CoV-2 RT-qPCR primer-probe sets. *Nature Microbiology* **5**, 1299–1305 (2020).

[54] Zhang, D. & Lu, J. In silico design of siRNAs targeting existing and future respiratory viruses with VirusSi. *bioRxiv* 2020.08.13.250076 (2020).

[55] Kugelman, J. R. *et al.* Evaluation of the potential impact of Ebola virus genomic drift on the efficacy of sequence-based candidate therapeutics. *mBio* **6** (2015).

[56] Freije, C. A. *et al.* Programmable inhibition and detection of RNA viruses using Cas13. *Molecular Cell* **76**, 826–837.e11 (2019).

[57] Plotkin, J. B., Dushoff, J. & Levin, S. A. Hemagglutinin sequence clusters and the antigenic evolution of influenza A virus. *Proceedings of the National Academy of Sciences of the United States of America* **99**, 6263–6268 (2002).

[58] Langat, P. *et al.* Genome-wide evolutionary dynamics of influenza B viruses on a global scale. *PLOS Pathogens* **13**, e1006749 (2017).

[59] Davies, M. R. *et al.* Atlas of group a streptococcal vaccine candidates compiled using large-scale comparative genomics. *Nature Genetics* **51**, 1035–1043 (2019).

[60] Federhen, S. The NCBI taxonomy database. *Nucleic Acids Research* **40**, D136–43 (2012).

[61] Bao, Y. *et al.* The influenza virus resource at the National Center for Biotechnology Information. *Journal of Virology* **82**, 596–601 (2008).

[62] Metsky, H. C. *et al.* Capturing sequence diversity in metagenomes with comprehensive and scalable probe design. *Nature Biotechnology* **37**, 160–168 (2019).

[63] Pedregosa, F. *et al.* Scikit-learn: Machine learning in python. *Journal of Machine Learning Research: JMLR* **12**, 2825–2830 (2011).

[64] Martín Abadi *et al.* TensorFlow: Large-Scale machine learning on heterogeneous systems (2015).

[65] Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization (2014). 1412.6980.

[66] Lefkowitz, E. J. *et al.* Virus taxonomy: the database of the International Committee on Taxonomy of Viruses (ICTV). *Nucleic Acids Research* **46**, D708–D717 (2018).

[67] Daher, R. K., Stewart, G., Boissinot, M. & Bergeron, M. G. Recombinase polymerase amplification for diagnostic applications. *Clinical Chemistry* **62**, 947–958 (2016).

[68] Ondov, B. D. *et al.* Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biology* **17**, 132 (2016).

[69] United States Centers for Disease Control and Prevention. Research use only 2019-novel coronavirus (2019-nCoV) real-time RT-PCR primers and probes. https://www.cdc.gov/coronavirus/2019-ncov/lab/rt-pcr-panel-primer-probes.html.

[70] Chvatal, V. A greedy heuristic for the Set-Covering problem. *Mathematics of Operations Research* **4**, 233–235 (1979).

[71] Johnson, D. S. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences* **9**, 256–278 (1974).

[72] Pearson, W. R., Robins, G., Wrege, D. E. & Zhang, T. On the primer selection problem in polymerase chain reaction experiments. *Discrete Applied Mathematics* **71**, 231–246 (1996).

[73] Huang, Y.-C. *et al.* Integrated minimum-set primers and unique probe design algorithms for differential detection on symptom-related pathogens. *Bioinformatics* **21**, 4330–4337 (2005).

[74] Feige, U. A threshold of ln n for approximating set cover. *Journal of the ACM* **45**, 634–652 (1998).

[75] Moshkovitz, D. The projection games conjecture and the NP-Hardness of ln n-Approximating Set-Cover. *Theory of Computing* **11**, 221–235 (2015).

[76] Har-Peled, S. & Jones, M. Few cuts meet many point sets (2018). 1808.03260.

[77] Varani, G. & McClain, W. H. The G x U wobble base pair. a fundamental building block of RNA structure crucial to RNA function in diverse biological systems. *EMBO Reports* **1**, 18–23 (2000).

[78] Saxena, S., Jónsson, Z. O. & Dutta, A. Small RNAs with imperfect match to endogenous mRNA repress translation. implications for off-target activity of small inhibitory RNA in mammalian cells. *The Journal of Biological Chemistry* **278**, 44312–44319 (2003).

[79] Du, Q., Thonberg, H., Wang, J., Wahlestedt, C. & Liang, Z. A systematic analysis of the silencing effects of an active siRNA at all single-nucleotide mismatched target sites. *Nucleic Acids Research* **33**, 1671–1677 (2005).

[80] Snøve, O., Jr & Holen, T. Many commonly used siRNAs risk off-target activity. *Biochemical and biophysical research communications* **319**, 256–263 (2004).

[81] Naito, Y., Yamada, T., Ui-Tei, K., Morishita, S. & Saigo, K. sidirect: highly effective, target-specific siRNA design software for mammalian RNA interference. *Nucleic Acids Research* **32**, W124–9 (2004).

[82] Qiu, S., Adema, C. M. & Lane, T. A computational study of off-target effects of RNA interference. *Nucleic Acids Research* **33**, 1834–1847 (2005).

[83] Yamada, T. & Morishita, S. Accelerated off-target search algorithm for siRNA. *Bioinformatics* **21**, 1316–1324 (2005).

[84] Zhao, W. & Lane, T. siRNA off-target search: A hybrid q-gram based filtering approach. In *Proceedings of the 5th International Workshop on Bioinformatics*, BIOKDD '05, 54–60 (ACM, New York, NY, USA, 2005).

[85] Alkan, F. *et al.* RIsearch2: suffix array-based large-scale prediction of RNA-RNA interactions and siRNA off-targets. *Nucleic Acids Research* **45**, e60 (2017).

[86] Doench, J. G. & Sharp, P. A. Specificity of microRNA target selection in translational repression. *Genes & Development* **18**, 504–511 (2004).

[87] Andoni, A. & Indyk, P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Proceedings of the Symposium on Foundations of Computer Science* (2006).

[88] Břinda, K., Sykulski, M. & Kucherov, G. Spaced seeds improve k-mer-based metagenomic classification. *Bioinformatics* **31**, 3584–3592 (2015).

[89] Katoh, K. & Standley, D. M. MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Molecular Biology and Evolution* **30**, 772–780 (2013).

[90] Yang, Z. *Computational molecular evolution* (Oxford University Press, Oxford; New York, 2006).