Genome Biology

SOFTWARE Open Access

DiMSum: an error model and pipeline for analyzing deep mutational scanning data and diagnosing common experimental pathologies

Andre J. Faure[1†], Jörn M. Schmiedel[1*†], Pablo Baeza-Centurion[1] and Ben Lehner[1,2,3*]

* Correspondence: joern.schmiedel@gmail.com; ben.lehner@crg.eu
†Andre J. Faure and Jörn M. Schmiedel contributed equally to this work.
[1]Center for Genomic Regulation (CRG), The Barcelona Institute of Science and Technology, Doctor Aiguader 88, 08003 Barcelona, Spain
Full list of author information is available at the end of the article

Abstract

Deep mutational scanning (DMS) enables multiplexed measurement of the effects of thousands of variants of proteins, RNAs, and regulatory elements. Here, we present a customizable pipeline, DiMSum, that represents an end-to-end solution for obtaining variant fitness and error estimates from raw sequencing data. A key innovation of DiMSum is the use of an interpretable error model that captures the main sources of variability arising in DMS workflows, outperforming previous methods. DiMSum is available as an R/Bioconda package and provides summary reports to help researchers diagnose common DMS pathologies and take remedial steps in their analyses.

Keywords: Deep mutational scanning, Bioinformatic pipeline, Statistical model, Variant effect prediction, R package, Bioconda

Background

Deep mutational scanning (DMS), also known as massively parallel reporter assays (MPRAs) and multiplex assays of variant effect (MAVEs), enables parallel measurement of the effects of thousands of mutations in the same experiment [1, 2]. In a basic DMS experiment, a library of sequence variants is constructed and deep sequencing before and after selection for an in vitro or in vivo activity is used to quantify the relative activity (“molecular fitness”) of each genotype. Beyond assaying point mutations, the high-throughput nature of DMS facilitates the comprehensive study of combinations of mutations and their genetic interactions (epistasis) where fitness effects of individual mutations depend on the presence of other (background) mutations [3]. The resulting fitness landscapes are informative of protein [4–6], RNA [7–9], and regulatory element [10–18] function and have provided mechanistic insight into biological processes including the regulation of gene expression [10, 19], protein-protein interactions [20], alternative splicing [21, 22], and molecular evolution [7]. Deep mutational scans have the potential to improve human variant annotation [23, 24] and protein and RNA structure determination [25–27]. In recognition of the growing number and importance of DMS assays in biomedical research, a dedicated platform for sharing, accessing, and visualizing these datasets has recently been launched [28].

A key feature of a DMS experiment is that it preserves the link between quantitative phenotypic effects and their underlying causal genotypes measured for many variants simultaneously (Fig. 1a). The three main steps can be summarized as follows: (1) construction of a library of DNA variants corresponding to the assayed biomolecule (genotype), (2) selection (or separation) of variants according to a given molecular function (phenotype), and (3) quantification of the variant abundances before and after selection by DNA sequencing (measurement), which is either done by counting sequencing reads of variants directly or unique barcodes previously linked to them [29–33]. A fitness score for each variant is then calculated by comparing its relative abundance (with respect to a reference sequence, e.g., wild-type) before and after selection. Moreover, often multiple independent biological replicates of the experiment are performed to help estimate the error of variant fitness scores, that is, a measure of fitness score reliability.

Several software packages have been developed to simplify and standardize the calculation of fitness scores for each variant from deep sequencing data [34, 35], including

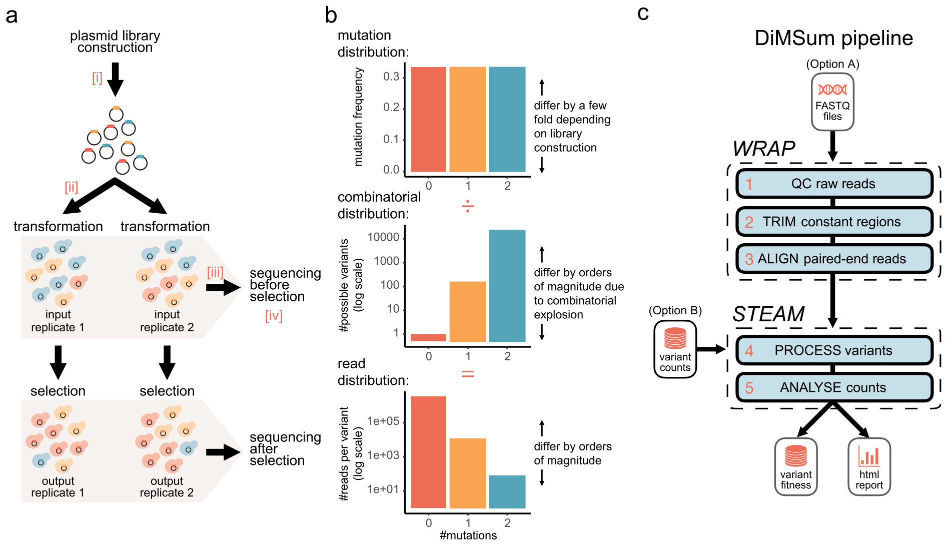

Fig. 1 Schematic overview of a minimal DMS experiment and the DiMSum pipeline. **a** Schematic of a basic, plasmid-based microbial growth DMS experiment: (1) construction of a plasmid library of mutant variants and independent transformation or integration of plasmid library into host cells, (2) exposure of cell population to selective conditions, and (3) high-throughput sequencing of samples to obtain variant counts before and after selection, which are used to derive fitness estimates for each variant. Indicated are steps at which bottlenecks could arise, potentially restricting variant pool size or complexity (red roman numerals): [i] inefficient library construction ("library bottleneck"), [ii] inefficient plasmid transformations ("replicate bottleneck"), and [iii] inefficient DNA extraction ("DNA extraction bottleneck"). Unforeseen bottlenecks can lead to over-sequencing [iv] of variant pools and thus underestimation of the errors associated with fitness scores or even appearance of sequencing counts for variants not contained in the original variant pool. **b** DMS experiments typically have a hierarchical abundance structure, where variants with more mutations are orders of magnitude less abundant than the wild-type sequence or single mutants. **c** DiMSum flow chart. The *WRAP* module performs low-level processing of raw DNA sequencing reads to produce sample-wise variant counts. The *STEAM* module transforms the resulting counts to estimates of variant fitness and associated error. See Additional file 1: Fig. S1-6 for example report plots

the estimation of errors for these fitness scores [36]. Unbiased estimation of fitness score reliability is crucial for the interpretation of DMS experiments, for example, when assessing the effects of a variant in a disease gene, and more generally for all kinds of hypothesis testing and when assessing genetic interactions.

The large-scale construction and high-throughput readout of thousands to hundreds of thousands of variants at once can, however, complicate basic quality control and identification of potential error sources and artifacts arising in DMS workflows. On the one hand, the many experimental steps of a DMS workflow can contribute errors to the final fitness measurements, especially when "bottlenecks" restrict the variant pool at certain steps in the workflow (Fig. 1a). On the other hand, libraries with a hierarchical variant abundance structure, arising from the combinatorial explosion of variants with multiple mutations (Fig. 1b), lead to distinct sources of error differentially affecting specific subsets of variants (see below). Moreover, the hierarchical variant abundance structure in combination with the typically low complexity of the genotype pool can lead to artifacts introduced by sequencing errors [37, 38].

To tackle these issues, we developed DiMSum, a pipeline that allows the end-to-end processing of DMS datasets using an interpretable model for the magnitude and sources of errors in fitness score. The workflow is freely available as an R/Bioconda package (DiMSum) that represents a complete solution for obtaining reliable variant fitness scores and error estimates from raw sequencing files.

Results and discussion

Overview of the DiMSum pipeline

The DiMSum pipeline is implemented as an R/Bioconda package and a command-line tool that can be easily configured to handle a variety of DMS experimental designs (see the "Methods" section). The pipeline is organized in two separate modules (Fig. 1c): *WRAP* processes raw read (FASTQ) files to produce sample-wise variant counts, and *STEAM* uses these sample-wise variant counts to estimate variant fitness scores and their measurement errors.

DiMSum *WRAP* performs the following sequence processing steps: (1) assessment of raw read quality using FastQC [39], (2) error-tolerant removal of constant regions (not subjected to mutagenesis but required for primer binding and isolation/amplification of variables regions) using cutadapt [40], and (3) alignment and filtering of paired-end reads in a base quality-aware manner using VSEARCH [41] if required. DiMSum *STEAM* accepts a table of counts, (4) isolates substitution variants of interest, and then (5) performs statistical analyses to obtain associated fitness scores and error estimates. Briefly, an error model is fit to a high confidence subset of variants to determine count-based, additive, and multiplicative errors of variant fitness scores for all replicates (see below).

To increase flexibility, *WRAP* or *STEAM* can each be run in stand-alone mode if desired, e.g., to obtain fitness scores from a user-generated table of variant counts (Fig. 1c, "option B" or to obtain sample-wise variant counts for a custom downstream analysis (Fig. 1c, "option A"). A detailed R markdown report—viewable with any web browser—including summary statistics, diagnostic plots, and analysis tips is also generated.

Estimates of variant fitness scores and associated errors

DiMSum calculates variant fitness scores as the natural logarithm of the ratio between sequencing counts in a replicate's output and input samples relative to the wild-type variant. It then uses replicate-specific error estimates to produce a weighted average of fitness scores across replicates for each variant.

DMS experiments are typically replicated to judge the reliability of fitness score estimates due to random variability in the workflow. However, the number of replicates performed is usually low (e.g., 3 to 6), and estimates of measurement errors on a variant-by-variant basis can thus lack statistical power. DiMSum instead estimates measurement errors of fitness scores by sharing information across all assayed variants to increase statistical power (Fig. 2, see the "Methods" section for full detail).

We assume that the error in fitness scores is, to a first approximation, primarily arising due to the finite sequencing counts, and thus, variants with similar counts in input and output samples should have similar measurement errors [42, 43]. If the error was purely arising due to sampling of variant frequencies by sequencing, the error could be well approximated by a Poisson distribution, with variance equal to the mean [44]. However, count data have been found to often be over-dispersed compared to this baseline Poisson expectation [45, 46]. To account for such over-dispersion, we introduce additive and multiplicative modifier terms of the baseline error, which has been shown to accurately describe variability in transcriptomic count data [47–49].

Multiplicative error terms modify the overall error proportional to the error resulting from a variant's sequencing counts and likely describe error sources in workflow steps linked to sequencing (see below for a discussion of potential sources and experimental remedies). Across different DMS datasets, we find such multiplicative error terms to range from one all the way to more than 100, suggesting that over-dispersion can be a grave issue in DMS experiments (Fig. 2, Table 2).

Additive error terms are independent of a variant's sequencing read counts, thus affecting all variants to the same extent, which we attribute to variability arising from differential handling of replicate selection experiments (see below). Additive error terms are typically small (s. d. < 10%) and therefore only become apparent in variants that have small errors from sequencing counts (those with many counts), constituting a lower error limit (Fig. 2, Table 2).

We assume that both multiplicative and additive error terms can differ between replicates but are the same for all variants in each replicate; our error model therefore has $3n$ parameters (where n is the number of replicates), which are estimated by minimizing the squared difference between the empirical and model-predicted variance of fitness scores across replicates for all variants simultaneously (see the "Methods" section).

Manipulating a DMS dataset to artificially increase either multiplicative error terms or additive error terms in one replicate suggests that the DiMSum error model is capable of accurately estimating the magnitude of the error model terms (Additional file 1: Fig. S7).

Error model benchmarking

To benchmark the error model, we performed leave-one-out cross-validation on published DMS datasets. Here, error model parameters were trained on all but one experimental replicate of a dataset. The resulting error estimates were used to judge whether the fitness scores of variants differ between the training replicates and the held-out test replicate. We find fitness score differences between training and test replicates are normally distributed with the magnitude predicted by the error model (Fig. 3a).

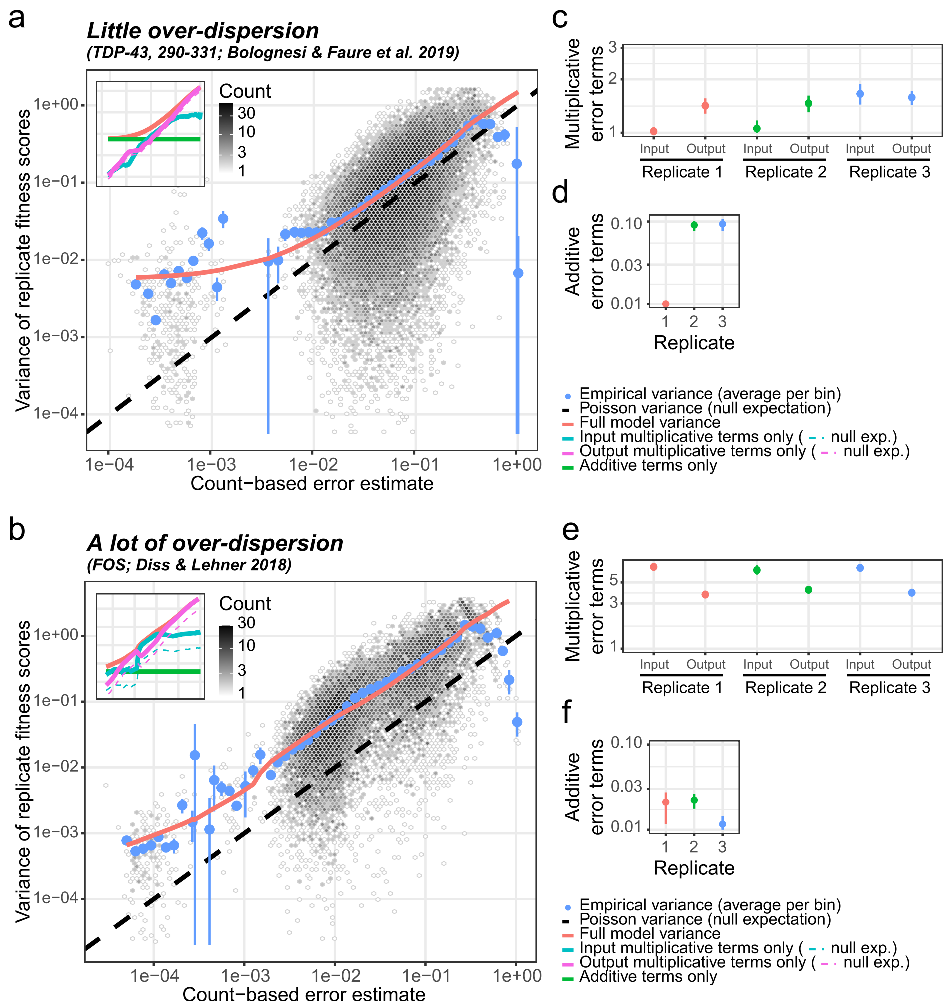

Fig. 2 DiMSum error model estimates multiplicative and additive error sources in fitness scores. **a** Empirical variance of replicate fitness scores as a function of error estimates based on sequencing counts under Poisson assumptions in a deep mutational scan of TDP-43 (positions 290-331) [6]. Empirical variance (blue dots show average variance in equally spaced bins, error bars indicate avg. variance × (1 ± 2/ # variants per bin)) is over-dispersed compared to baseline expectation of variance being described by a Poisson distribution (black dashed line). The bimodality of the count-based error distribution results from the relatively low number of single nucleotide mutants which have high counts (thus low count-based error) and the many double nucleotide mutants which have low counts (thus higher count-based error). The DiMSum error model (red line) accurately captures the deviations of the empirical variance from Poisson expectation. Inset: bold cyan and magenta lines indicate multiplicative error term contributions to variance corresponding to input and output samples, respectively (dashed thin lines give input or output sample contributions to variance if multiplicative error terms were 1). The horizontal green line indicates the additive error term contribution. The red line indicates the full DiMSum error model. **b** The same as **a** but for a deep mutational scan of FOS [20] that shows more over-dispersion. **c**–**f** Multiplicative (**c**, **e**) and additive (in s.d. units, **d**, **f**) error terms estimated by the error model on the two datasets. Dots give mean parameters, error bars 90% confidence intervals

Consequently, when testing for significant differences between the training and test replicates (using a z test), P values are uniformly distributed (Fig. 3b), as would be expected for replicates from the same experiment and indicating that the model correctly controls the type I error rate (rate of false positives).

We find that the DiMSum error model accurately estimates errors in fitness scores across twelve published DMS datasets that display various degrees of over-dispersion

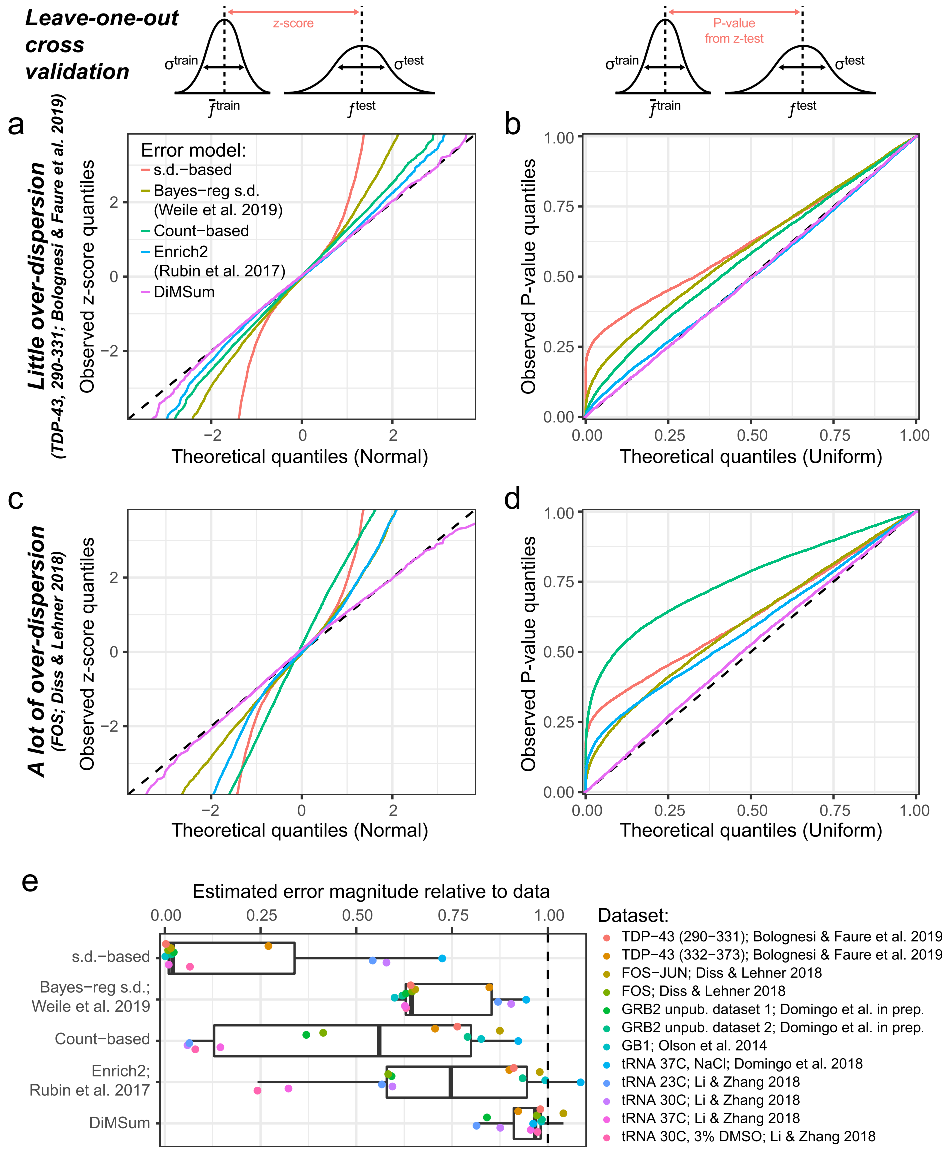

Fig. 3 DiMSum error model performance. Leave-one-out cross-validation to test error model performance. In turn, error models are trained on all but one replicate of a dataset, and *z*-scores of the differences in fitness scores between the training set $\bar{f}^{\mathrm{train}}$ and the remaining test replicate f^{test} are calculated (i.e., fitness score differences normalized by the estimated error in the training set σ^{train} and test replicate σ^{test}; importantly, σ^{test} is estimated from error model parameters fit only on the training set replicates). Because fitness scores from replicate experiments should only differ by random chance, if the error models estimate the error magnitude correctly, *z*-scores should be normally distributed, and corresponding *P* values from a *z* test should be uniformly distributed. The tested error models are described in the "Results and discussion" and "Methods" sections. **a**, **c** Quantile-quantile plots of *z*-scores in TDP-43 290-331 library (**a**) and FOS library (**c**) compared to the expected normal distribution. **b**, **d** Quantile-quantile plots of *P* values from two-sided *z* test in TDP-43 290-331 library (**a**) and FOS library (**c**) compared to the expected uniform distribution. **e** Estimated error magnitude relative to the differences observed between replicate fitness scores in twelve DMS datasets in leave-one-out cross-validation (see the "Methods" section). Relative error magnitude = 1 means the estimated magnitude of errors fits the data. Relative error magnitude < 1 means the estimated errors are too small. Boxplots indicate median and 1st and 3rd quartiles (box), and whiskers extend to 1.5× interquartile range

(Fig. 3, Table 2). Moreover, errors are accurately estimated no matter whether they are driven by low sequencing counts (variant with low counts, often higher-order mutants) or whether they appear to be independent of sequencing counts (variants with high counts, such as single mutants), suggesting that both multiplicative and additive error terms help to accurately model error sources in DMS experiments (Additional file 1: Fig. S8).

We compared the DiMSum error model performance to several popular alternative approaches that have previously been used to model error in DMS data (see Table 1). We note that this is not an exhaustive comparison against all statistical models previously used before to estimate measurement errors in DMS datasets. The chosen alternative approaches differ in whether they estimate errors for each variant from the observed variability of fitness scores or the sequencing counts, or a combination thereof, and how much information sharing across variants they allow.

On the one hand, several studies have used the empirical variance of fitness scores across replicates to calculate errors for each variant individually [8, 9, 20, 22, 31, 50]. Such error estimates are under-powered due to the typically low number of replicates in DMS studies, resulting in errors that are too large for some variants but too small for others. The latter results in an inflation of type I errors (Fig. 3b, d, "s.d.-based"). Error estimates improve with an increasing number of replicates, but type I error inflation persists even for a DMS dataset with 6 replicates (Fig. 3e, Domingo et al. [7]).

Building on this empirical variance approach, Weile et al. [51] used a Bayesian regularization of the empirical variance proposed by Baldi and Long [52], which uses a linear regression estimate of empirical variance across all variants as a prior. We find that this approach improves over only using the empirical variance to calculate errors, but still leads to inflation of type I errors (Fig. 3, "Bayes-reg s.d.").

On the other hand, several studies have assumed that errors can be modeled by a Poisson process based on a variant's sequencing counts [5–7, 53]. Not unexpectedly, the performance of the "Poisson-based" approach depends on the over-dispersion of the data. It works well for datasets with little systematic over-dispersion but fails dramatically in those cases where the DiMSum error model estimates high multiplicative or additive errors (Fig. 3 and Additional file 1: Fig. S8, "Count-based").

Table 1 Benchmarked statistical models to estimate measurement error in fitness scores

Error model type	Based on	Additional input?	Information sharing across variants?	Model parameters
Variance-based	Empirical variance of replicate fitness	**No**	**No**	0
Bayesian regularization of variance [51]	Empirical variance of replicate fitness	**Yes**: Bayesian prior to regression on seq. counts	**Yes**: prior estimated across all variants	3
Count-based	Sequencing counts follow Poisson distribution	**No**	**No**	0
Enrich2 [36]	Sequencing counts follow Poisson distribution	**Yes**: mixed-effects from empirical variance	**No**	# variants
DiMSum	Sequencing counts follow Poisson distribution	**Yes**: modifier terms from empirical variance	**Yes**: replicate-specific modifier terms estimated across all variants	3 × # replicates

Enrich2 [36] uses a random-effects model to account for over-dispersion over and above the count-based Poisson expectation on a variant-by-variant basis. In short, variant-specific random-effects terms increase the modeled error towards the empirical variance if it is larger than the count-based Poisson expectation. While this leads to accurate error estimates in datasets with little systematic over-dispersion (small multiplicative error terms, Fig. 3a, b, "Enrich2"), the under-powered estimation of the variant-specific random-effects terms leads to an inflation of type I errors in those datasets with systematic over-dispersion, similar to the other approaches based on variant-specific empirical variances (Fig. 3c, d, e, "Enrich2").

In summary, the DiMSum error model captures the major error sources arising in DMS workflows and improves in accuracy over previous approaches, while needing fewer replicate experiments and having fewer, but interpretable model parameters.

DiMSum provides diagnostic plots similar to Fig. 3a, c to help judge whether errors have been accurately modeled. Failure of the model to accurately estimate the errors suggests shortcomings, potentially due to systematic error sources in the DMS workflow which cannot be accounted for by the error model, urging further action by the user (see below).

Potential sources of increased error in fitness score estimates

In what follows, we provide suggestions for error sources that might be captured by DiMSum's additive and multiplicative error model terms as well as error sources that cannot be captured by the error model and how their impact on DMS experiments can be minimized.

Additive error terms are independent of variant read counts and therefore likely result from differential handling of replicate selection experiments. Because these error terms are typically small compared to errors resulting from low sequencing counts, they most often only affect fitness score estimates of very abundant variants, such as single mutants of the wild-type sequence in question. However, if such highly abundant variants are of interest, increasing sequencing coverage will not lead to reductions in the measurement errors of their fitness scores. DiMSum performs a simple scale and shift procedure to minimize inter-replicate differences in fitness score distributions prior to estimating error model parameters, therefore minimizing additive error terms that arise from linear differences between replicate selection experiments (see the "Methods" section and Additional file 1: Fig. S4b,c). Additional mitigation strategies to reduce additive error contributions should focus on streamlining the handling of replicate samples through the workflow (e.g., using master mixes, increasing pipetting volumes, reducing time lags in time-sensitive steps) as well as increasing the number of replicate experiments [53], even at similar overall sequencing coverage, as this will lead to a reduction of errors for variants that are dominated by sequencing-independent errors due to the weighted averaging of fitness scores across replicates.

Multiplicative error terms increase variants' errors by a multiple of their sequencing count-based error estimate. Potential error sources are thus likely linked to the sequencing steps in the DMS workflow, in particular, related to the start of the selection step, DNA extraction from input, and output samples as well as the subsequent PCR amplification for sequencing library construction.

First, consider a bottleneck at the DNA extraction step, which arises if the number of unique DNA molecules extracted from the input/output samples does not exceed the number of molecules that are subsequently sequenced, i.e., the extracted variant pool is "over-sequenced." This restriction in the numbers of variant molecules along the workflow will introduce additional random variability in variant frequencies that significantly contribute to—or even dominate—the overall count-based error, and errors calculated solely from the number of downstream sequenced molecules will thus be an underestimate of the true error.

In addition, Kowalsky et al. [54] found that PCR amplification protocols for sequencing library construction can introduce additional random variability to variant frequencies. Using our DiMSum error model, we find that multiplicative errors differ fivefold between the three PCR protocols tested (see the "Methods" section), thus showing that multiplicative errors can arise during the PCR amplification steps of the DMS workflow.

Lastly, another source of multiplicative errors that can potentially arise in input samples is a bottleneck at the start of the selection experiment. Here, if the number of variant molecules used to start the selection is similar to or smaller than the number of variant molecules extracted and sequenced from the input sample, this will randomly alter true variant frequencies at the start of the selection with error magnitudes on the order of or even larger than the error due to sequencing a finite subset of variant molecules.

For example, we recently performed a deep mutational scan of part of the protein GRB2 (Domingo et al., manuscript in preparation), for which the error model indicated a sixfold multiplicative error in the input replicates. A similar error was not observed in a second, related deep mutational scan for the same protein, suggesting a technical bottleneck specific to the input library preparation in the first experiment.

To minimize multiplicative error sources, thus reducing measurement errors and ultimately save sequencing costs, DMS workflows should ensure an excess of variant molecules (~ 5–10×) is used in all experimental steps upstream of the sequencing step [55] and PCR amplification protocols are optimized [54]. Additionally, sources of multiplicative errors due to bottlenecks at the DNA extraction step and other downstream steps, but not during the selection experiment, should be detectable (and correctable) if using unique molecular identifiers (UMIs) ligated to variant molecules during PCR-based sequencing library preparation [31, 56, 57].

Systematic error sources. Apart from sources of increased measurement error due to random error in DMS workflows, there are potentially also sources of systematic error that the DiMSum error model cannot account for and which might therefore inflate error or bias fitness scores in undetectable ways.

One potential source of systematic errors is (non-linear) differences in the replicate selection experiments. For example, we recently used DMS to quantify the toxicity of variants of TDP-43 when expressed in yeast in which we mutagenized two sections of the C-terminal prion-like domain [6]. Variants displayed a range of fitness values relative to the wild-type sequence, both detrimental and beneficial. Importantly, one replicate experiment showed a marginal fitness distribution whose shape differed from those of three other replicate experiments. In particular, non-toxic mutant variants were limited in how much faster they could grow compared to wild-type TDP-43, which perhaps resulted from nutrient limitation during the selection experiment (Additional file 1: Fig. S4c). Such non-linear effects that only affect a subset of variants (e.g., beneficial variants) cannot be corrected with simple linear normalization schemes (e.g., DiMSum's shift and scale normalization procedure) and will introduce systematic errors that the error model cannot adequately describe, thus potentially leading to biased fitness estimates as well as incorrect estimates of errors (Additional file 1: Fig. S6e,f). Thus, systematic differences in replicate selection experiments identify the need for better normalization strategies or exclusion of affected replicates, as we decided for the TDP-43 replicate [6].

In summary, the DiMSum error model and diagnostic plots can also serve to judge and improve the experimental workflow and downstream analyses of DMS experiments.

Diagnosing sources of systematic errors in DMS workflows

The particular combination of low genotype complexity and hierarchical abundance structure in DMS experiments (Fig. 1b) can lead to issues arising from sequencing errors.

On the one hand, sequencing errors in reads of highly abundant variants can contribute counts to closely related, but low abundant, variants [37, 38]. That is, sequencing errors in wild-type reads will contribute counts to single mutant variants, and sequencing errors in single mutant variants will contribute counts to double mutant variants and so on. DiMSum displays estimates of this sequencing error-induced “variant flow” in diagnostic plots of marginal count distributions to give the user an estimate of what fraction of reads of a set of mutants might be caused by sequencing errors (Fig. 4a, left column). Mitigation strategies to lower the fraction of reads per variant from sequencing errors include using higher minimum base quality (Phred score) thresholds, using paired-end sequencing to decrease the number of base call errors, or circumventing these issues altogether by using highly complex barcode libraries that are linked to variants [38].

On the other hand, a potential pitfall linked to the combination of low genotype complexity, hierarchical abundance structure, and sequencing errors in DMS experiments is to mistake sequencing reads purely arising from sequencing errors for the presence of a variant in the assayed genotype pool. That is, at deep enough sequencing coverage, reads for any low-order nucleotide mutant variant will appear in the sequencing record, even if the variant was not actually present in the experiment.

Consider two examples from published DMS experiments. The first example of a DMS experiment in which NNS (N=A, T, C, or G; S=C or G) saturation mutagenesis was used to introduce individually mutated codons into the wild-type sequences of FOS and JUN [20]. Variants that have one mutated codon show a bimodal count distribution in the input samples (Fig. 4a, middle column). Variants in the higher peak have similar read counts no matter whether one, two, or three nucleotides were mutated, consistent with NNS mutagenesis operating on the codon level and the number of mismatched base-pairs having little impact on mutation efficacy. In contrast, read counts for variants in the lower peaks show a dependency on the number of nucleotides mutated and coincide with DiMSum’s estimate for sequencing error-induced variant flow. The second example is from a DMS experiment in which doped oligonucleotide synthesis was used to introduce nucleotide mutations into a tRNA [50]. Variants with one or two mutated nucleotides show a bimodal count distribution in the input samples (Fig. 4a, right column). The read counts for variants in the upper peaks depend on the number of nucleotides mutated, consistent with read counts per variant being strongly affected by the combinatorics of mutational space (Fig. 1b). The read counts for variants in the lower peaks also depend on the number of nucleotides mutated and coincide with DiMSum’s estimate for sequencing error-induced variant flow.

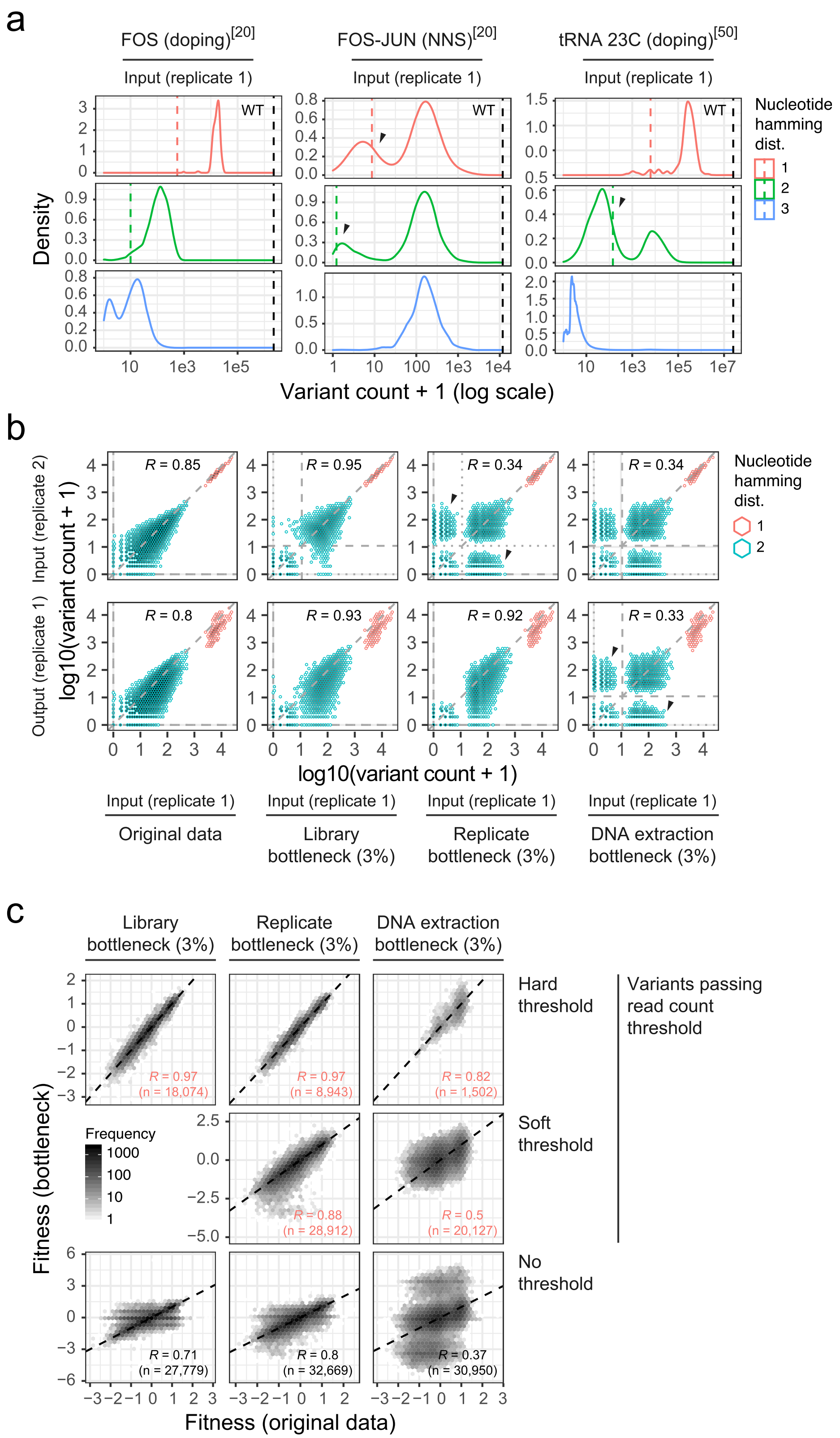

Fig. 4 (See legend on next page.)

(See figure on previous page.)

Fig. 4 Effects of bottlenecks on variant count distributions and fitness scores. **a** Input sample count distributions of previously published DMS experiments [20, 50]. For FOS and FOS-JUN datasets, counts of single AA variants with one, two, or three nucleotide substitutions in the same codon are shown. For the tRNA dataset, all variants with one, two, or three nucleotide substitutions are shown. Wild-type counts are indicated by the black dashed line. Expected count frequencies purely due to sequencing errors are indicated by red and green dashed lines for single and double nucleotide substitution variants, respectively. Black arrows indicate sets of variants that have likely not been assayed but whose sequencing reads are arising due to sequencing errors. **b** Simulation of bottlenecks at various steps of the DMS workflow based on a previously published DMS dataset [6]. Scatterplots show input and output sample counts for variants with one or two nucleotide substitutions in the original data or after simulating 3% library, replicate, or DNA extraction bottlenecks (from left to right). Hexagon color indicates the number of nucleotide substitutions and fill number of variants per 2d bin (see legend). Black arrows indicate sets of double nucleotide variants whose sequencing reads solely originate from sequencing errors. Dotted (or dashed) horizontal/vertical lines indicate soft (or hard) variant count thresholds used in downstream DiMSum analyses (see **c**). **c** Comparison of fitness scores from simulated datasets with (*y*-axis) or without (*x*-axis) the indicated bottlenecks. Variants are categorized by their robustness to filtering with hard (variants have to appear above the threshold in all replicates) or soft thresholds (variants have to appear above the threshold in at least one replicate) of 10 read counts. For the DNA extraction bottleneck, read count thresholds were also applied to output samples. Pearson correlation coefficients are indicated. The dashed line indicates the relationship $y = x$. Note that correlation coefficients are lower for soft than hard thresholds, because a subset of variants has fewer replicate measurements

Are variants in the lower read count peaks of these experiments really not present in the variant libraries before sequencing? And at which steps in the DMS workflow were the variants lost?

Potential bottlenecks (or ineffecacies) might arise during library construction, transfer of the library into the assay cell population or at subsequent DNA extraction and sequencing library preparation steps (Fig. 1a).

We find that comparisons of count distributions between sequencing samples can provide additional support to determine whether subsets of variants purely arose from sequencing errors and help to diagnose at which workflow step variants might have been lost, in order to improve future DMS experiments and serve to inform the strategy to avoid systematic errors in fitness calculations for a present dataset. We exemplify this in Fig. 4b on simulated bottlenecks in a deep mutational scan of TDP-43.

If variants have not been constructed or have been lost at initial library preparation steps and therefore are not present in any replicate experiment, count distributions between replicate input samples should be highly correlated and the same variants should fall into the same peaks of bimodal read count distributions (Fig. 4b, “library bottleneck”), as is also apparent in the FOS-JUN dataset (Additional file 1: Fig. S9). Variants in the lower peak of the distribution should be discarded from all replicates, e.g., using DiMSum’s “hard” read count thresholds for variant filtering (Fig. 4c), and downstream analysis should proceed as normal, as in the published analysis of the FOS-JUN dataset [20].

In contrast, if the variant loss was replicate-specific, e.g., if transformations into replicate cell populations were incomplete, read count distributions should display “flaps”—subsets of variants that appear at high counts in one replicate (variant was assayed) but at low counts in another (variant counts arise solely from sequencing errors) (Fig. 4b, “replicate bottleneck”). A conservative approach to avoid systematic errors in fitness score calculations is to use “hard” read count thresholds to discard all variants appearing in lower read count peaks in any replicate. Additionally, DiMSum allows the user to choose a “soft” threshold to discard variants only in the replicates where they appear in the low count peaks, therefore allowing their fitness to still be estimated from replicates in which they are actually present, resulting in an increased number of variants that can be used for downstream analyses (Fig. 4c).

Finally, if variants were lost at the DNA extraction steps, this should not only show up as flaps in count distributions between replicate input samples, but also between input and output samples of the same replicate (Fig. 4b, "DNA extraction bottleneck"), as it is observed for the tRNA dataset (Additional file 1: Fig. S9). Here, in order to avoid biased fitness estimates, all variants that do not appear in the high read count peak of both input and output samples from the same replicate experiment need to be discarded to avoid systematic errors in downstream analyses. Often, fitness differences between variants also result in bimodal output count distribution, meaning that practically, it can be hard or impossible to assign whether variants with low counts in output samples are due to low fitness or because they were not assayed. As for the replicate bottlenecks, "soft" thresholds can be used to obtain fitness estimates for all variants that appear in the input and output samples of at least one replicate, therefore increasing the number of variants that can be used for downstream analyses (Fig. 4c).

To further illustrate how experimental bottlenecks can adversely affect the conclusions of a study, we evaluated their impact on the central conclusion of a previous publication. We previously showed that the fitness effects of amino acid substitutions in the prion-like domain of TDP-43 are correlated with the increase in a principal component of amino acid properties (PC1) strongly related to the hydrophobicity of the protein [6]. Repeating this same analysis after simulating library, replicate, or DNA extraction bottlenecks in the original data results in lower correlations in all cases (Additional file 1: Fig. S10, left column). Imposing both hard and soft minimum read count filtering as described above, the result is an increase in the correlation between measured fitness of amino acid substitutions and their corresponding predicted effects on PC1/hydrophobicity (Additional file 1: Fig. S10, middle column).

Together, this demonstrates that it is crucial to discard variants purely arising from sequencing errors to avoid systematic errors and shows how DiMSum can be used successfully to prioritize variants, minimize biases in downstream analyses, and improve biological conclusions.

Conclusions

We have developed a customizable pipeline—DiMSum—that provides a complete solution for the analysis of DMS data. DiMSum is easy to run, can handle a wide variety of different library designs, provides detailed reporting, and produces fitness and error estimates from raw DNA sequencing data in a matter of hours. Importantly, DiMSum's interpretable error model is able to identify and account for measurement errors in fitness scores resulting from random variability in DMS workflows and additionally provides the user with diagnostics to identify and deal with common causes of systematic errors. We have also shown that the DiMSum error model provides accurate error estimates across many published DMS datasets, outperforming previously used methods, and that diagnostic plots enable simple remedial steps to be taken that have the potential to dramatically improve the reliability of results from downstream analyses.

Methods

DiMSum software implementation

DiMSum is implemented as an R/Bioconda package and a command-line tool compatible with Unix-like operating systems (see installation instructions: https://github.com/

lehner-lab/DiMSum). The pipeline consists of five stages grouped into two modules that can be run independently: *WRAP* (DiMSum stages 1–3) processes raw FASTQ files generating a table of variant counts and *STEAM* (DiMSum stages 4–5) analyses variant counts generating variant fitness and error estimates. *WRAP* requires common software tools for biological sequence analysis (FastQC [39], cutadapt [40], VSEARCH [41], and starcode [58]) whereas *STEAM* has no external binary dependencies other than Pandoc. A detailed R markdown report including summary statistics, diagnostic plots, and analysis tips is automatically generated. DiMSum takes advantage of multi-core computing if available. Further details and installation instructions are available on GitHub (https://github.com/lehner-lab/DiMSum).

DiMSum data preprocessing

FastQ files from paired-end sequencing of the TDP-43 290-331 library [6] were processed with DiMSum v1.1.3 using default parameters with minor adjustments. First, 5′ constant regions were trimmed in an error-tolerant manner ("cutadaptErrorRate" = 0.2). Read pairs were aligned, and those that contained base calls with posterior Phred scores (posterior score takes both Phred scores of aligned bases into account) below 30 were discarded ("vsearchMinQual" = 30, "vsearchMaxee" = 0.5). Finally, variants with greater than two amino acid mutations were removed ("maxSubstitutions" = 2). One out of four input replicates (and all associated output samples) was discarded ("retainedReplicates" = 1,3,4) from all results shown in main text figures because the shape of its fitness distribution significantly differed from those of three other replicate experiments (see Additional file 1: Fig. S4b,c). Note that Additional file 1: Fig. S1-6 show DiMSum summary report plots when using all four replicates.

Simulated bottlenecked datasets were similarly processed with DiMSum v1.1.3 using hard, soft, and no filtering. For datasets with library and replicate bottlenecks, filtering was performed on the input samples only ("fitnessMinInputCountAll" = 10 for hard threshold or "fitnessMinInputCountAny" = 10 for soft threshold), whereas for datasets with DNA extraction bottlenecks, output samples were additionally filtered ("fitnessMinOutputCountAll" = 10 for hard or "fitnessMinOutputCountAny" = 10 for soft threshold).

DMS datasets for leave-one-out cross-validation were processed with DiMSum v1.1.3 except the data for Protein G B1 domain (GB1 [5]) whose variant counts were obtained from Otwinoski [59]. tRNA datasets [50] obtained from SRA (SRP134087) were analyzed using DiMSum with default parameters except fitnessMinInputCountAll = 2000 and fitnessMinOutputCountAll = 200 to remove flaps likely due to DNA extraction bottlenecks, resulting in an average number of 2400 variants that could be analyzed per selection experiment. The use of soft thresholds would result in an average increase in variant counts of 200% across the four selection experiments. For datasets with only one input sample (GB1 and tRNA), we replicated the input sample to create as many matched input-output samples as necessary for the error model analysis. All experimental design files and bash scripts with command-line options required for running DiMSum on the above datasets are available on GitHub (https://github.com/lehner-lab/dimsumms).

DiMSum fitness estimation and error modeling

DiMSum calculates fitness scores of each variant i in each replicate r as the natural logarithm of the ratio between output read counts N_i^{output} and input read counts N_i^{input} relative to the wild-type variant wt:

$$f_{r,i} = \log\left(\frac{N_{r,i}^{\text{output}}}{N_{r,i}^{\text{input}}} \times \frac{N_{r,wt}^{\text{input}}}{N_{r,wt}^{\text{output}}}\right)$$

Optionally, DiMSum applies a scale and shift procedure to minimize linear differences in fitness scores between replicates. This is done by fitting a slope and an offset parameter to each replicate's fitness scores in order to minimize the sum of squared deviations between variants' replicate fitness scores and their respective averages. Moreover, it is ensured that wild-type variants have an average fitness score of 0 across replicates.

The measurement error of fitness scores is modeled based on Poissonian statistics from sequencing counts of the variant ($\sigma^2(\log(N)) = \frac{\sigma^2(N)}{E[N]^2} = \frac{1}{N}$), with multiplicative (m_r^{input} for input sample and m_r^{output} for output sample) and additive (a_r) modifier terms that are common to all variants, but specific to each replicate experiment performed, as:

$$\sigma^2\left(f_{r,i}\right) = \frac{m_r^{\text{input}}}{N_{r,i}^{\text{input}}} + \frac{m_r^{\text{output}}}{N_{r,i}^{\text{output}}} + a_r$$

Note that we omit the inclusion of error terms arising from the wild-type normalization, as these error terms are typically small due to high wild-type counts.

The error model is fit to a high confidence set of variants (variants that have enough sequencing reads in the input samples to display the full range of fitness scores and for which at least one sequencing read has been observed in all output samples, see Additional file 1: Fig. S4a).

The error model parameters are estimated by sharing information across all variants, that is by minimizing the sum over all variants' squared deviation between the average error model prediction across replicates and the observed variance of fitness scores across replicates:

$$\arg\ \min_{m_r\in[1,\infty),a_r\in[0,\infty)} \sum_R \sum_i \omega_{R,i} \left(\operatorname{var}\left(f_{R,i}\right) - \frac{1}{n_R} \times \left(\sum_{r\in R} \sigma^2\left(f_{r,i}\right)\right)\right)^2$$

In order to reliably estimate the replicate-specific additive error terms a_r, the error model fit is performed not only with variances/error model predictions across all replicates of the DMS experiment, but across all possible subsets of replicates R of size at least two simultaneously (e.g., for three replicates, $R \in (\{1,2,3\},\{1,2\},\{1,3\},\{2,3\})$ and $n_R = \{3,2,2,2\}$). This is done because estimation of additive error terms depends mostly on high count variants (which have little to no error contribution from sequencing counts) and the error model cannot distinguish how much additive variability was contributed by any one replicate unless further constrained (by subsets of lower-order combinations). However, this means that if only two replicates of the DMS experiment have been performed, the error model tends to split additive error contributions equally between replicates for lack of more information, i.e., additive error terms cannot be used as diagnostic.

Moreover, squared deviations between variance and average error model predictions per variant and replicate subset are weighted ($\omega_{R,\ i}$) according to three factors: first, the number of replicates in the replicate subset R, to account for the differential uncertainty in empirical variance estimates; second, the inverse of the average count-based error according to Poissonian statistics, to minimize relative, not absolute, deviations between the variance of fitness scores and respective error model estimates; and third, a term re-weighting all variants with the same number of mutations according to $\sqrt{\max(n_m, \sqrt{n_i})}$, where n_m is the number of variants with that number of mutations and n_i is the overall number of variants in the high-confidence variant pool. This is done to place more weight on the typically fewer lower-order mutant variants (e.g., single mutants) and therefore to improve estimates of additive error terms.

The error model is fit 100 times on bootstrapped data. For each bootstrap, at maximum 10,000 variants are drawn with replacement from the high-confidence variant pool. The average parameters across bootstraps are used to calculate measurement error estimates.

The error estimates are then used to merge fitness scores across replicates by weighted averaging:

$$\bar{f}_i = \frac{\sum_{r=1}^{n} f_{r,i}/\sigma_{r,i}^2}{\sum_{r=1}^{n} \sigma_{r,i}^{-2}}$$

The corresponding error of these merged fitness scores is calculated as:

$$\bar{\sigma}_i^2 = \frac{1}{\sum_{r=1}^{n} \sigma_{r,i}^{-2}}$$

DiMSum reports merged fitness scores and associated errors for all variants that have been observed in at least one experimental replicate (actual merging is performed for variants observed in two or more replicates; for variants only observed in one experimental replicate, merged fitness scores and error are simply those computed for this one replicate).

DiMSum diagnoses consistency of the error model with the data by estimating how well it describes fitness score differences between replicates. If all error sources have been accounted for and model parameters accurately attribute error contributions to the different replicates, the predicted error magnitude should match the randomly arising differences in fitness scores between replicates of the same experiment, which we find to be normally distributed across all DMS datasets investigated, i.e.:

$$\bar{f}_{r\neq j,i} - f_{j,i} \sim N\left(0, \sqrt{\bar{\sigma}_{r\neq j,i}^2 + \sigma_{j,i}^2}\right) \text{ with } j \,\forall\, 1, \ldots, n \text{ and } i \,\forall\, 1, \ldots N.$$

We thus calculate a z-score of the fitness score differences replicates as:

$$z_{j,i} = \frac{\bar{f}_{r\neq j,i} - f_{j,i}}{\sqrt{\bar{\sigma}_{r\neq j,i}^2 + \sigma_{j,i}^2}}$$

that should follow a normal distribution centered on zero and with unit standard deviation. DiMSum outputs quantile-quantile plots of $z_{j,\ i}$ as well as its corresponding mean and standard deviations and the P value distribution from a two-sided z test (Additional file 1: Fig. S6e,f). Mean values of $z_{j,\ i}$ different from zero suggest that fitness score estimates are biased, which suggests the presence of systematic errors not accounted for by the “scale and shift” normalization procedure and the error model. A standard deviation different from one suggests that the error has been over-estimated (s.d. < 1) or underestimated (s.d. > 1).

Error model validation and benchmarking

Artificial increases in multiplicative and additive error terms in Additional file 1: Fig. S7

In order to show that the error model can accurately capture multiplicative and additive error sources, we performed a manipulation of the TDP-43 290-331 library (using only replicates 1, 3, and 4) in which we artificially increased multiplicative input terms (by multiplying input count reads by factor 3 or 10) or additive error terms (by adding values of 0.3 or 1 to normalized fitness scores immediately before error model fitting) for replicate 1.

Leave-one-out cross-validation

To benchmark the error model and compare it against alternative approaches to quantify measurement error, we performed leave-one-out cross-validation on published DMS datasets. In contrast to the error model benchmarking performed as a diagnostic output from the DiMSum pipeline (see above), we trained the error models on all but one replicate of a dataset in turn. These error models were then used to calculate a *z*-score of the fitness score differences between the unseen replicate and the average over the training replicates ($r \neq j$) as:

$$z_{j,i}^{\text{leave - one - out}} = \frac{\bar{f}_{r\neq j,i} - f_{j,i}}{\sqrt{\bar{\sigma}_{r\neq j,i}^{2} + \left\langle \sigma_{r\neq j,i}^{2} \right\rangle}}$$

where $\langle \sigma_{r\neq j,i}^{2} \rangle$ is the prediction of error in the test replicate *j* using the error model parameters of the training replicates.

For the DiMSum error model, this is $\langle \sigma_{r\neq j,i}^{2} \rangle = \frac{\bar{m}_{r\neq j}^{\text{input}}}{N_{j,i}^{\text{input}}} + \frac{\bar{m}_{r\neq j}^{\text{output}}}{N_{j,i}^{\text{output}}} + \bar{a}_{r\neq j}$, with $\bar{m}_{r\neq j}^{\text{input}}$, $\bar{m}_{r\neq j}^{\text{output}}$ and $\bar{a}_{r\neq j}$ as the averages of replicate-specific error model terms over the training replicates.

In Fig. 3a,c $z_{j,i}^{\text{leave - one - out}}$ is compared against a normal distribution, with the expectation that they should match if the error magnitude is correctly predicted. Figure 3b, d displays the *P* values from a two-sided *z* test using $z_{j,i}^{\text{leave - one - out}}$, with the expectation that *P* values should be uniformly distributed, because there should not be significant differences in fitness scores between replicate experiments. Finally, to analyze $z_{j,i}^{\text{leave - one - out}}$ systematically across many datasets, we calculated the inverse of its standard deviation for each dataset (Table 2, Fig. 3e, Additional file 1: Fig. S8), i.e., $\sigma\left(z_{j,i}^{\text{leave - one - out}}\right)^{-1} = 1$ if the error model has correctly predicted the magnitude of measurement errors, $\sigma\left(z_{j,i}^{\text{leave - one - out}}\right)^{-1} > 1$ if errors have been over-estimated, or $\sigma\left(z_{j,i}^{\text{leave - one - out}}\right)^{-1} < 1$ if errors have been underestimated.

Alternative error models

We compared the DiMSum error model to four alternative error models.

Table 2 DiMSum error model parameters and error model performance in leave-one-out cross-validation across twelve DMS datasets

DMS dataset	No. of replicates	Error model parameters (avg ± s.d. across replicates)			Estimated error magnitude relative to data‡				
		m^{input}	m^{output}	$\sqrt{a}$^	Variance-based	Bayesian reg. of variance [51]	Count-based	Enrich2 [36]	DiMSum
FOS-JUN [20]	3	1.1 ± 0.0	1.6 ± 0.7	0.02 ± 0.01	0.02	0.65	0.88	0.98	1.04
FOS [20]	3	6.3 ± 0.4	5.0 ± 0.3	0.02 ± 0.01	0.01	0.65	0.41	0.58	0.97
GB1 [5]	3	1.1 ± 0.1	1 ± 0	0.04 ± 0.02	0.001	0.6	0.83	0.99	0.98
GRB2 unpublished dataset 1	3	6.2 ± 1.2	1.1 ± 0.1	0.05 ± 0.02	0.024	0.63	0.37	0.59	0.84
GRB2 unpublished dataset 2	3	1 ± 0	1 ± 0	0.03 ± 0.02	0.017	0.62	0.79	0.93	0.98
TDP-43 (290-331) [6]	3	1.3 ± 0.4	1.5 ± 0.1	0.07 ± 0.05	0.003	0.64	0.76	0.91	0.98
TDP-43 (332-373) [6]	4	1.5 ± 0.6	1.2 ± 0.4	0.1 ± 0.06	0.27	0.85	0.71	0.9	0.92
tRNA NaCl + 37C [7]	6	1 ± 0	1.1 ± 0.1	0.03 ± 0.02	0.72	0.94	0.92	1.09	0.96
tRNA 23C [50]	5	85 ± 55	96 ± 78	0.15 ± 0.04	0.54	0.87	0.064	0.57	0.81
tRNA 30C [50]	5	201 ± 187	121 ± 99	0.14 ± 0.07	0.58	0.90	0.059	0.59	0.88
tRNA 37C [50]	3	38 ± 19	39 ± 11	0.04 ± 0.01	0.01	0.63	0.15	0.32	0.96
tRNA DMSO [50]	3	101 ± 48	192 ± 124	0.04 ± 0.01	0.066	0.63	0.08	0.24	0.97

‡The inverse standard deviation of the *z*-score distribution from leave-one-out cross-validation (see the "Methods" section)

^Square root of additive error term *a* gives a standard deviation-based estimate of lower variability bound

First, a "variance-based" error model, where the error of fitness scores for each variant is calculated from the empirical variance of fitness scores between replicates, i.e., with the measurement error of fitness scores merged across replicates as $\bar{\sigma}_i^2 = \mathrm{var}(f_i)/n$.

Second, an error model using a Bayesian regularization of the empirical variance, as introduced by Weile et al. [51]. Here, the empirical variance of each variant's fitness scores between replicates is regularized with a prior, which is a regression of the empirical variance on input sequencing counts and fitness scores. Here, the measurement error of fitness scores merged across replicates is $\bar{\sigma}_i^2 = \frac{d\times\sigma_{i,\mathrm{prior}}^2+(n-1)\times\mathrm{var}(f_i)}{(d+n-2)\times n}$, with d as the degrees of freedom of the regression, $\sigma_{i,\mathrm{prior}}^2$ as the prior estimate of the variance for variant i, and n as the number of replicate experiments. For the variance-based error models, the measurement error for the unseen test replicate was estimated as the average of individual training replicates. The z-scores for the variance-based error model in the leave-one-out cross-validation were thus calculated as: $z_{j,i}^{\mathrm{variance-based}} = \frac{\bar{f}_{r\neq j,i} - f_{j,i}}{\sqrt{\bar{\sigma}_{r\neq j,i}^2 \times \frac{n}{n-1}}}$.

Third, a minimal "count-based" error model, where the error of fitness scores is estimated from sequencing read counts in input and output samples under the assumption that sequencing counts follow a Poisson distribution, i.e., as for the DiMSum error model but without multiplicative or additive terms.

Fourth, the Enrich2 error model by Rubin et al. [36], which is based on sequencing counts but modified with variant-specific correction terms ("random-effects model"). Here, error estimates are calculated from input and output sequencing read counts under Poisson assumptions, but with an additional variant-specific random-effects term. This term corrects error estimates if the observed variability of fitness scores across replicates is larger than the estimated count-based error alone. That is, if $\sigma_i^2 < \operatorname{var}(f_i)$, the random-effects term s_i^2 is estimated greater than 0 such that $\sigma_i^2 = \frac{1}{\sum_{r=1}^{n} {(\sigma_{r,i}^2 + s_i^2)}_{r,i}^{-1}} \cong \operatorname{var}(f_i)$, i.e., the error estimate becomes equivalent to that of the variance-based error model described above. To calculate *z*-scores in the leave-one-out cross-validation for the Enrich2 error model, we estimated random-effects terms across the training replicates and then also used them to modify the count-based error estimate of the test replicate, i.e., $z_{j,i}^{\text{Enrich2}} = \frac{\bar{f}_{r \neq j,i} - f_{j,i}}{\sqrt{\bar{\sigma}_{r \neq j,i}^2 + \left(\frac{1}{N_{j,i}^{\text{output}}} + \frac{1}{N_{j,i}^{\text{input}}} + s_{r \neq j,i}^2 \right)}}$.

Multiplicative errors from PCR amplification

Kowalsky et al. [54] previously reported increased variability in sequencing read counts due to PCR amplification protocols (see Table S3 of Kowalsky et al. [54]). The raw sequencing data for the three PCR amplification protocols tested was obtained from the authors. Paired-end reads were merged with USEARCH [60] using the *usearch -fastq_mergepairs* command with a minimum per base posterior Qscore of 20, and reads for unique variants were counted using the *usearch -fastx_uniques* command. To allow estimation of multiplicative and additive error terms, we treated the sequencing data from each PCR amplification protocol as replicate experiments. Variant fitness scores were calculated as the natural logarithm of read count frequency (read counts divided by the total number of reads in each replicate). Error of fitness scores was calculated as the inverse of variant read counts. DiMSum error model was adjusted to only fit one multiplicative error term and the additive error term per replicate. Additive errors were small compared to the variability observed. Multiplicative error terms were 1.9 ± 0.4 for method A (using one amplification cycle with all primers at once), 1.4 ± 0.1 for method B (two amplification cycles interspersed with a ExoI degradation step) and 6.4 ± 0.6 for method C (two amplification cycles).

Simulated bottlenecks in a previously published DMS dataset

We used a DiMSum processed DMS dataset from Bolognesi and Faure et al. [6] (290-331 library) to simulate the effects of various experimental bottlenecks.

Simulating a library bottleneck

A library bottleneck of size $\alpha = 0.03$ (meaning that only 3% of molecules pass through the bottleneck) was simulated based on the observed average frequencies of variants in the input samples. A bottleneck factor $b_i = Pois(N_i^{\text{input}} \times \alpha)/N_i^{\text{input}}$ was calculated to capture the subsequent changes in read count frequencies that would occur during such a bottleneck. For variants with high counts in input samples, the bottleneck factor will be close to α. However, for low count variants, the bottleneck factor will vary considerably. Some variants, especially variants with $N_{1,i}^{\text{input}} < \alpha$, will not pass through the bottleneck, i.e., $b_i = 0$, while others may pass through the bottleneck even though there was only one molecule of that variant present in the pool, i.e., $b_i = 1$.

To simulate how read counts in sequencing samples (both input and output sequencing samples) change due to this bottleneck, we sampled N times from a multinomial distribution $Mult(1, \pi_r)$ where N is the total number of sequencing reads in the "original" sample s, and π_s is a vector of probabilities given by:

$$\pi_s = \left(\pi_{s,1}, \pi_{s,2}, \ldots, \pi_{s,k}\right)$$

where k is the total number of different variant sequences, and $\pi_{r,\ i}$ is the frequency of variant i in sample s after the library bottleneck (e.g., for replicate *1* output sample):

$$\pi_{s,i} = \frac{N_{1,i}^{\text{output}} \times b_i}{\sum_1^k N_{1,i}^{\text{output}} \times b_i}$$

To simulate sequencing errors in the new modified data, we assumed that the probability that a given sequencing read is misidentified to be 0.02, based on a length of the mutated sequence of 126 nt and a per base misread frequency of 0.0001 [6], and that all errors involve WT molecules being misclassified as single mutants, or single mutant molecules being misidentified as double mutants, or double mutant molecules being misidentified as triple mutants, and triple mutant molecules being misidentified as quadruple mutants. The total number of triple mutant molecules that will be misidentified as quadruple mutants is $0.02N$, where N is the total number of triple mutant reads. Those counts were randomly subtracted from the triple mutant counts and added to the counts of all the quadruple mutants. The total number of double mutant molecules misidentified as triple mutants is $0.02N'$, where N' is the total number of double mutant reads. Those counts were subtracted from the double mutant counts and randomly distributed among all the triple mutants. This process was repeated to simulate single mutants being misidentified as double mutants and WT molecules being misidentified as single mutants.

Simulating a replicate bottleneck

The procedure was similar to the library bottleneck procedure described above, but a bottleneck factor was calculated on each replicate input sample independently, allowing for different variants to be present in each replicate.

Simulating a DNA extraction bottleneck

The procedure was similar to the library/replicate bottleneck simulations, but a bottleneck factor was calculated for each sequencing sample independently.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s13059-020-02091-3.

Additional file 1. Supplementary Figures S1 to S10.

Additional file 2. Review history.

Acknowledgements

The authors thank members of the Lehner Lab for feedback on the DiMSum pipeline and methods.

Review history

The review history is available as Additional file 2.

Peer review information

Yixin Yao was the primary editor of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Authors' contributions

All authors jointly conceived the project. A.J.F. and J.M.S. developed the pipeline software and performed the analyses of DMS datasets. J.M.S. developed the error model. P.B. conducted the bottleneck simulations. A.J.F. and J.M.S. wrote the manuscript with input from B.L. and P.B. The authors read and approved the final manuscript.

Funding

This work was supported by a European Research Council (ERC) Consolidator grant (616434), the Spanish Ministry of Economy and Competitiveness (BFU2017-89488-P and SEV-2012-0208), the Bettencourt Schueller Foundation, Agencia de Gestio d'Ajuts Universitaris i de Recerca (AGAUR, 2017 SGR 1322.), and the CERCA Program/Generalitat de Catalunya. We also acknowledge the support of the Spanish Ministry of Economy, Industry and Competitiveness (MEIC) to the EMBL partnership and the Centro de Excelencia Severo Ochoa. This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement 752809 (J.M.S.).

Availability of data and materials

Raw datasets analyzed during the current study (see Table 2) are available in the GEO repository:
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE102901
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE128165
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE99418
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE111508
All experimental design files and bash scripts with command-line options required for running DiMSum on the above datasets can be found on GitHub (https://github.com/lehner-lab/dimsumms). We also provide an R package with custom scripts to perform all downstream analyses, simulations, and performance comparisons and reproduce all figures described here.
DiMSum source code, installation instructions, command-line options, common use cases, input file formats (example templates), and the DiMSum demo mode are available on GitHub (https://github.com/lehner-lab/DiMSum) [61], under the MIT open source license. An archived version of DiMSum is available on Zenodo (https://doi.org/10.5281/zenodo.3925155) [62].

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Center for Genomic Regulation (CRG), The Barcelona Institute of Science and Technology, Doctor Aiguader 88, 08003 Barcelona, Spain. [2]Universitat Pompeu Fabra (UPF), Barcelona, Spain. [3]Institució Catalana de Recerca i Estudis Avançats (ICREA), Passeig Lluís Companys 23, 08010 Barcelona, Spain.

Received: 4 December 2019 Accepted: 5 July 2020
Published online: 17 August 2020

References

1. Kinney JB, McCandlish DM. Massively parallel assays and quantitative sequence–function relationships. Annual Review of Genomics and Human Genetics. 2019. p. 99–127.
2. Fowler DM, Fields S. Deep mutational scanning: a new style of protein science. Nat Methods. 2014;11:801–7.
3. Domingo J, Baeza-Centurion P, Lehner B. The causes and consequences of genetic interactions (epistasis). Annu Rev Genomics Hum Genet. 2019;20:433–60.
4. Fowler DM, Araya CL, Fleishman SJ, Kellogg EH, Stephany JJ, Baker D, et al. High-resolution mapping of protein sequence-function relationships. Nature Methods. 2010. p. 741–6.
5. Olson CA, Wu NC, Sun R. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. Curr Biol. 2014;24:2643–51.
6. Bolognesi B, Faure AJ, Seuma M, Schmiedel JM, Tartaglia GG, Lehner B. The mutational landscape of a prion-like domain. Nat Commun. 2019;10:4162.
7. Domingo J, Diss G, Lehner B. Pairwise and higher-order genetic interactions during the evolution of a tRNA. Nature. 2018;558:117–21.
8. Li C, Qian W, Maclean CJ, Zhang J. The fitness landscape of a tRNA gene. Science. 2016;352:837–40.

9. Puchta O, Cseke B, Czaja H, Tollervey D, Sanguinetti G, Kudla G. Network of epistatic interactions within a yeast snoRNA. Science. 2016;352:840–4.
10. Kinney JB, Murugan A, Callan CG, Cox EC. Using deep sequencing to characterize the biophysical mechanism of a transcriptional regulatory sequence. Proceedings of the National Academy of Sciences. 2010. p. 9158–63.
11. Kosuri S, Goodman DB, Cambray G, Mutalik VK, Gao Y, Arkin AP, et al. Composability of regulatory sequences controlling transcription and translation in Escherichia coli. Proc Natl Acad Sci U S A. 2013;110:14024–9.
12. Birnbaum RY, Patwardhan RP, Kim MJ, Findlay GM, Martin B, Zhao J, et al. Systematic dissection of coding exons at single nucleotide resolution supports an additional role in cell-specific transcriptional regulation. PLoS Genetics. 2014. p. e1004592.
13. Kheradpour P, Ernst J, Melnikov A, Rogov P, Wang L, Zhang X, et al. Systematic dissection of regulatory motifs in 2000 predicted human enhancers using a massively parallel reporter assay. Genome Res. 2013;23:800–11.
14. Melnikov A, Murugan A, Zhang X, Tesileanu T, Wang L, Rogov P, et al. Systematic dissection and optimization of inducible enhancers in human cells using a massively parallel reporter assay. Nat Biotechnol. 2012;30:271–7.
15. Patwardhan RP, Hiatt JB, Witten DM, Kim MJ, Smith RP, May D, et al. Massively parallel functional dissection of mammalian enhancers in vivo. Nat Biotechnol. 2012;30:265–70.
16. Kwasnieski JC, Fiore C, Chaudhari HG, Cohen BA. High-throughput functional testing of ENCODE segmentation predictions. Genome Res. 2014;24:1595–602.
17. Kwasnieski JC, Mogno I, Myers CA, Corbo JC, Cohen BA. Complex effects of nucleotide variants in a mammalian cis-regulatory element. Proceedings of the National Academy of Sciences. 2012. p. 19498–503.
18. White MA, Myers CA, Corbo JC, Cohen BA. Massively parallel in vivo enhancer assay reveals that highly local features determine the cis-regulatory function of ChIP-seq peaks. Proc Natl Acad Sci U S A. 2013;110:11952–7.
19. Patwardhan RP, Lee C, Litvin O, Young DL, Pe'er D, Shendure J. High-resolution analysis of DNA regulatory elements by synthetic saturation mutagenesis. Nat Biotechnol. 2009;27:1173–5.
20. Diss G, Lehner B. The genetic landscape of a physical interaction. Elife. 2018. p. 7.
21. Baeza-Centurion P, Miñana B, Schmiedel JM, Valcárcel J, Lehner B. Combinatorial genetics reveals a scaling law for the effects of mutations on splicing. Cell. 2019. p. 549–63.e23.
22. Julien P, Miñana B, Baeza-Centurion P, Valcárcel J, Lehner B. The complete local genotype–phenotype landscape for the alternative splicing of a human exon. Nature Communications. 2016;7:11558.
23. Starita LM, Young DL, Islam M, Kitzman JO, Gullingsrud J, Hause RJ, et al. Massively parallel functional analysis of BRCA1 RING domain variants. Genetics. 2015;200:413–22.
24. Findlay GM, Daza RM, Martin B, Zhang MD, Leith AP, Gasperini M, et al. Accurate classification of BRCA1 variants with saturation genome editing. Nature. 2018;562:217–22.
25. Schmiedel JM, Lehner B. Determining protein structures using deep mutagenesis. Nat Genet. 2019;51:1177–86.
26. Rollins NJ, Brock KP, Poelwijk FJ, Stiffler MA, Gauthier NP, Sander C, et al. Inferring protein 3D structure from deep mutation scans. Nat Genet. 2019;51:1170–6.
27. Zhang Z, Xiong P, Zhang T, Wang J, Zhan J, Zhou Y. Accurate inference of the full base-pairing structure of RNA by deep mutational scanning and covariation-induced deviation of activity. Nucleic Acids Res. 2019;48:1451–65.
28. Esposito D, Weile J, Shendure J, Starita LM, Papenfuss AT, Roth FP, et al. MaveDB: an open-source platform to distribute and interpret data from multiplexed assays of variant effect. Genome Biol. 2019;20:223.
29. Hiatt JB, Patwardhan RP, Turner EH, Lee C, Shendure J. Parallel, tag-directed assembly of locally derived short sequence reads. Nat Methods. 2010;7:119–22.
30. Kitzman JO, Starita LM, Lo RS, Fields S, Shendure J. Massively parallel single-amino-acid mutagenesis. Nature Methods. 2015. p. 203–6.
31. Matreyek KA, Starita LM, Stephany JJ, Martin B, Chiasson MA, Gray VE, et al. Multiplex assessment of protein variant abundance by massively parallel sequencing. Nature Genetics. 2018. p. 874–82.
32. Kircher M, Xiong C, Martin B, Schubach M, Inoue F, Bell RJA, et al. Saturation mutagenesis of twenty disease-associated regulatory elements at single base-pair resolution. Nat Commun. 2019;10:3583.
33. Poelwijk FJ, Socolich M, Ranganathan R. Learning the pattern of epistasis linking genotype and phenotype in a protein. Nature Communications. 2019;10:4213.
34. Fowler DM, Araya CL, Gerard W, Fields S. Enrich: software for analysis of protein function by enrichment and depletion of variants. Bioinformatics. 2011. p. 3430–1.
35. Hietpas RT, Jensen JD, Bolon DNA. Experimental illumination of a fitness landscape. Proc Natl Acad Sci U S A. 2011;108:7896–901.
36. Rubin AF, Gelman H, Lucas N, Bajjalieh SM, Papenfuss AT, Speed TP, et al. A statistical framework for analyzing deep mutational scanning data. Genome Biol. 2017;18:150.
37. Bloom JD. Software for the analysis and visualization of deep mutational scanning data. BMC Bioinformatics. 2015;16:168.
38. Zhang T-H, Wu NC, Sun R. A benchmark study on error-correction by read-pairing and tag-clustering in amplicon-based deep sequencing. BMC Genomics. 2016;17: 108.
39. Andrews S. FastQC A Quality control tool for high throughput sequence data. Available from: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/. Accessed 20 July 2020.
40. Martin M. Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet.journal. 2011;17(1):10–12.
41. Rognes T, Flouri T, Nichols B, Quince C, Mahé F. VSEARCH: a versatile open source tool for metagenomics. PeerJ. 2016;4:e2584.
42. Robinson MD, McCarthy DJ, Smyth GK. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics. 2010;26:139–40.
43. Anders S, Huber W. Differential expression analysis for sequence count data. Nature Precedings. 2010;11(10):R106.
44. Marioni JC, Mason CE, Mane SM, Stephens M, Gilad Y. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. Genome Res. 2008;18:1509–17.
45. Nagalakshmi U, Wang Z, Waern K, Shou C, Raha D, Gerstein M, et al. The transcriptional landscape of the yeast genome defined by RNA sequencing. Science. 2008;320:1344–9.
46. Robinson MD, Smyth GK. Moderated statistical tests for assessing differences in tag abundance. Bioinformatics. 2007;23:2881–7.

47. Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 2014;15:550.
48. Anders S, Reyes A, Huber W. Detecting differential usage of exons from RNA-seq data. Genome Res. 2012;22:2008–17.
49. Reyes A, Anders S, Weatheritt RJ, Gibson TJ, Steinmetz LM, Huber W. Drift and conservation of differential exon usage across tissues in primate species. Proc Natl Acad Sci U S A. 2013;110:15377–82.
50. Li C, Zhang J. Multi-environment fitness landscapes of a tRNA gene. Nat Ecol Evol. 2018;2:1025–32.
51. Weile J, Sun S, Cote AG, Knapp J, Verby M, Mellor JC, et al. A framework for exhaustively mapping functional missense variants. Mol Syst Biol. 2017;13:957.
52. Baldi P, Long AD. A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes. Bioinformatics. 2001;17:509–19.
53. Matuszewski S, Hildebrandt ME, Ghenu A-H, Jensen JD, Bank C. A statistical guide to the design of deep mutational scanning experiments. Genetics. 2016;204:77–87.
54. Kowalsky CA, Klesmith JR, Stapleton JA, Kelly V, Reichkitzer N, Whitehead TA. High-resolution sequence-function mapping of full-length proteins. PLoS One. 2015;10:e0118193.
55. Fowler DM, Stephany JJ, Fields S. Measuring the activity of protein variants on a large scale using deep mutational scanning. Nat Protoc. 2014;9:2267–84.
56. Kivioja T, Vähärautio A, Karlsson K, Bonke M, Enge M, Linnarsson S, et al. Counting absolute numbers of molecules using unique molecular identifiers. Nat Methods. 2011;9:72–4.
57. Grün D, Kester L, van Oudenaarden A. Validation of noise models for single-cell transcriptomics. Nat Methods. 2014;11:637–40.
58. Zorita E, Cuscó P, Filion GJ. Starcode: sequence clustering based on all-pairs search. Bioinformatics. 2015;31:1913–9.
59. Otwinowski J. Biophysical inference of epistasis and the effects of mutations on protein stability and function. Mol Biol Evol. 2018;35:2345–54.
60. Edgar RC. Search and clustering orders of magnitude faster than BLAST. Bioinformatics. 2010;26:2460–1.
61. Faure AJ, Schmiedel JM, Baeza-Centurion P, Lehner B. DiMSum. GitHub. 2020. Available from: https://github.com/lehner-lab/DiMSum. Accessed 20 July 2020.
62. Faure AJ, Schmiedel JM, Baeza-Centurion P, Lehner B. DiMSum. Zenodo. 2020. Available from: https://doi.org/10.5281/zenodo.3925155. Accessed 20 July 2020.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.