



# GGAssembler: Precise and economical design and synthesis of combinatorial mutation libraries

Shlomo Yakir Hoch  | Ravit Netzer | Jonathan Yaacov Weinstein |  
Lucas Krauss | Karen Hakeny | Sarel Jacob Fleishman 

Department of Biomolecular Sciences,  
Weizmann Institute of Science,  
Rehovot, Israel

## Correspondence

Shlomo Yakir Hoch and Sarel Jacob  
Fleishman, Department of Biomolecular  
Sciences, Weizmann Institute of Science,  
Rehovot 7600001, Israel.  
Email: hochshi@gmail.com and sarelj@  
weizmann.ac.il

## Present addresses

Ravit Netzer and Jonathan  
Yaacov Weinstein, Scala Biodesign LTD,  
50 Dizengoff, Tel Aviv, 6433222, Israel.

## Funding information

Dr. Barry Sherman Institute for Medicinal  
Chemistry; Donation in memory of Sam  
Switzer; Israel Science Foundation (1844);  
European Research Council through a  
Consolidator Award (815379);  
Volkswagen Foundation (94747)

**Review Editor:** Nir Ben-Tal

## Abstract

Golden Gate assembly (GGA) can seamlessly generate full-length genes from DNA fragments. In principle, GGA could be used to design combinatorial mutation libraries for protein engineering, but creating accurate, complex, and cost-effective libraries has been challenging. We present GGAssembler, a graph-theoretical method for economical design of DNA fragments that assemble a combinatorial library that encodes any desired diversity. We used GGAssembler for one-pot in vitro assembly of camelid antibody libraries comprising  $>10^5$  variants with DNA costs  $<0.007$  per variant and dropping significantly with increased library complexity.  $>93\%$  of the desired variants were present in the assembly product and  $>99\%$  were represented within the expected order of magnitude as verified by deep sequencing. The GGAssembler workflow is, therefore, an accurate approach for generating complex variant libraries that may drastically reduce costs and accelerate discovery and optimization of antibodies, enzymes and other proteins. The workflow is accessible through a Google Colab notebook at <https://github.com/Fleishman-Lab/GGAssembler>.

## KEYWORDS

biotechnology, combinatorial mutation libraries, DNA library synthesis, Golden Gate assembly

## 1 | INTRODUCTION

Over the past decade, high-throughput screening of protein variant libraries and deep sequencing have revolutionized protein characterization, engineering, and design (Fowler et al., 2010; Markel et al., 2020; Wójcik et al., 2015). Common library-synthesis approaches introduce random or systematic single-point mutations (Alejaldre et al., 2021). However, significant gains in protein activity and stability often require multiple simultaneous mutations (Goldenzweig et al., 2016; Khersonsky

et al., 2018; Listov et al., 2024; Whitehead et al., 2012), calling for gene-synthesis approaches that introduce combinatorial mutations. Such multipoint combinatorial libraries can be designed based on previous experimental screens (Fowler et al., 2010; Whitehead et al., 2012; Whitehead et al., 2013) and increasingly based on computational design calculations (Guntas et al., 2010; R. Lipsh-Sokolik et al., 2023; Listov et al., 2024; Treynor et al., 2007; Weinstein et al., 2023). However, synthesizing combinatorial libraries accurately and economically is challenging due to the large size of a typical binding or

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial](https://creativecommons.org/licenses/by-nc/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2024 The Author(s). *Protein Science* published by Wiley Periodicals LLC on behalf of The Protein Society.

enzyme active site and the distribution of active-site positions across multiple noncontiguous epitopes. An illustration of the potential complexity of effective variant libraries is provided by our recent computational active-site library design studies from which thousands of diverse and functional variants were isolated in a single experiment (R. Lipsh-Sokolik et al., 2023; Weinstein et al., 2023). In these studies, atomistic design calculations defined the desired diversity across dozens of positions, including, in one case, through combinatorial insertions and deletions at an enzyme active site (R. Lipsh-Sokolik et al., 2023). To synthesize such complex yet completely defined variant libraries, we considered different cloning strategies that would be general and economical, enable a high level of control over the location and identity of mutations and the ratio of mutations to the wild-type parental identities, and exhibit low representation bias among the variants.

Given these demands, we focused on Golden Gate Assembly (GGA) (Engler et al., 2008; Engler et al., 2009). GGA employs Type II restriction enzymes, such as *BsaI*, that cleave outside their DNA recognition sequence. After processing by the Type II restriction enzyme, the recognition site is eliminated, leaving terminal overhangs that can be programmed to encode immutable (nondiversified) amino acid positions that link subsequent gene fragments. Ligation of the processed fragments then seamlessly combines them into full-length genes (Engler et al., 2008; Engler et al., 2009). In principle, GGA enables scarless gene construction of any size from small fragments (Andreou & Nakayama, 2018; Püllmann et al., 2019; Sarrion-Perdigones et al., 2011, 2013; Vazquez-Vilar et al., 2017; Weber et al., 2011). High-fidelity GGA, however, is governed by several factors that limit the number and identity of overhangs: the nonspecific (star) activity of the restriction enzyme (Mayer, 1978), the sensitivity of the DNA ligase to mismatches (Lohman et al., 2016), and the relative propensity of DNA overhangs to anneal under different reaction conditions. These limitations forced previous GGA applications to use iterative DNA assembly methods with a restricted set of overhangs (Andreou & Nakayama, 2018; Püllmann et al., 2019; Sarrion-Perdigones et al., 2011, 2013; Vazquez-Vilar et al., 2017; Weber et al., 2011). As a step to address these limitations, recent studies have examined the results of GGA under different experimental conditions and expanded the set of allowed overhangs (Potapov, Ong, Kucera, et al., 2018; Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020). In principle, these developments allow the assembly of dozens of fragments for combinatorial gene synthesis (Potapov, Ong, Kucera, et al., 2018; Pryor et al., 2020). Despite much progress (Daffern et al., 2023;

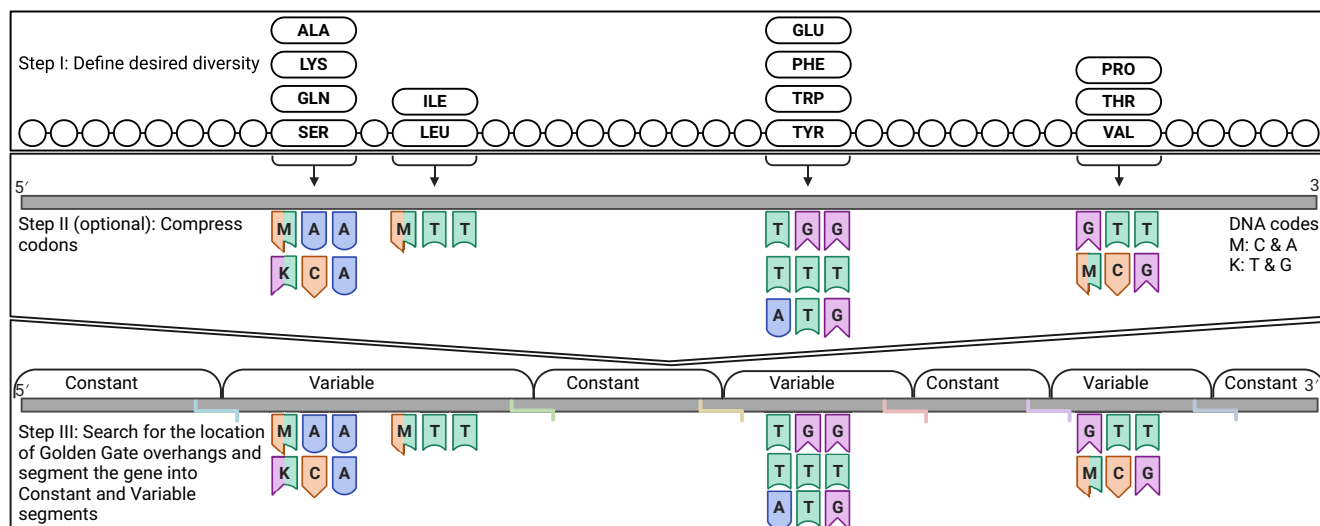
Kirby et al., 2021; Pryor et al., 2020; Püllmann et al., 2019), however, methods for economical and accurate construction of variant libraries from dozens of fragments have not yet been demonstrated.

We present a new method, called GGAssembler, for the economical design of DNA fragments that assemble into a gene library which encodes any specified diversity with minimal representation bias. We recently used GGAssembler to synthesize two complex libraries both encoding millions of combinatorial mutants at 14 positions in the green fluorescent protein (GFP) chromophore-binding pocket with a total DNA cost of \$455, translating to  $4.2 \times 10^{-3}$  ¢ per variant (Weinstein et al., 2023). We now demonstrate that this approach can accurately, efficiently, and economically encode complex multipoint mutation libraries of camelid single-domain (vHH) antibodies comprising hundreds of thousands of variants. GGAssembler can be applied, in principle, to genes of all sizes, but the relatively small size of the vHH gene allows us to rigorously demonstrate completeness, high accuracy, and low representation bias in the final assembly product. GGAssembler is freely available at <https://github.com/Fleishman-Lab/GGAssembler> and can be accessed through an online notebook at [https://github.com/Fleishman-Lab/GGAssembler/blob/master/example/colab\\_oligos\\_design.ipynb](https://github.com/Fleishman-Lab/GGAssembler/blob/master/example/colab_oligos_design.ipynb).

## 2 | RESULTS

### 2.1 | A general method to design combinatorial diversity

We developed GGAssembler to provide an economical approach to combinatorial library assembly while allowing nearly complete freedom to decide the mutation types, positions, and library size. Critically, GGAssembler guarantees a lower bound on the fidelity of the GGA reaction. To estimate the probability that a given set of overhangs assembles with high fidelity and accuracy, we leverage empirically derived overhang ligation fidelity (Equation 1) and efficiency (Equation 2) estimates (Potapov, Ong, Kucera, et al., 2018; Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020). The cost associated with each DNA fragment is the number of nucleotides required to produce all the diversity encoded in that fragment. For example, the cost associated with a 20 nucleotides DNA fragment that encodes a single mutated amino acid position that requires two codons would be 40 nucleotides (the length of the fragment times the encoded variability). Adding an extra codon at the mutated position would increase the cost to 60 nt,



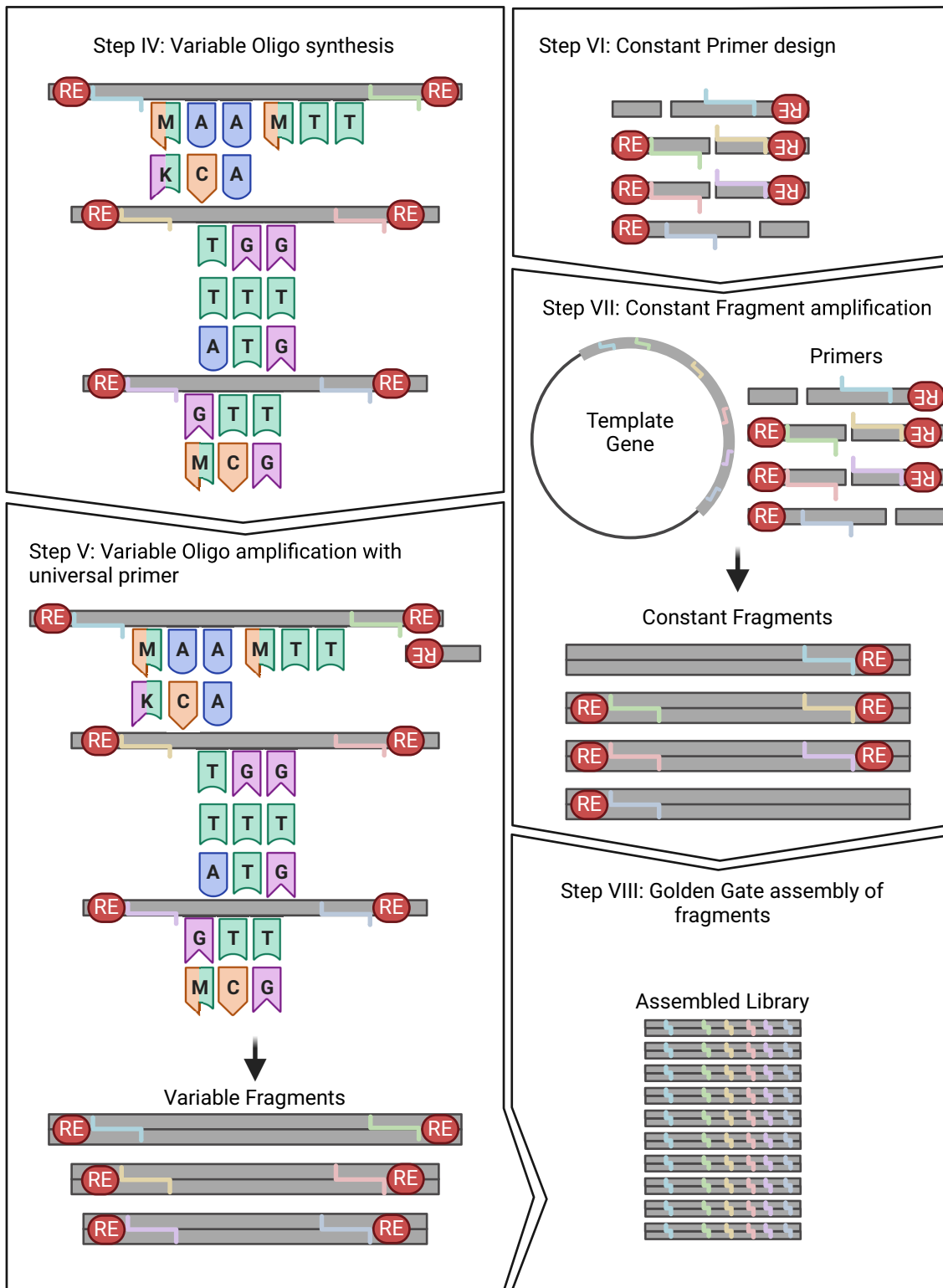
**FIGURE 1** Key steps in the GGAssembler computational workflow. Step I: Provide wild-type DNA sequence and the required diversity in each position. Optional Step II: Apply codon compression for the required diversity in each position. Step III: Apply a shortest-path and rainbow shortest-path search to find an economical segmentation of the DNA that maintains GGA reaction fidelity above a given threshold.

whereas adding another mutated position would multiply the cost by the number of codons in that position. GGAssembler uses degenerate codons to encode as much desired diversity as possible at a minimal DNA cost while completely excluding undesired amino acid identities and stop codons.

The GGAssembler workflow comprises three steps as schematically represented in Figures 1 and 2 and Figure S1. We start by defining the amino acids that encode the desired diversity with an optional codon-compression step that uses degenerate codons (Methods). This step excludes all unspecified amino acids and stop codons. In the next step, we employ a graph-theoretical cost-optimization algorithm to compute alternative solutions that vary in cost and number of overhangs. To accomplish this, we build a graph in which nodes represent restriction enzyme cleavage sites, and edges connect all pairs of sites that a single DNA oligo can traverse. To construct this graph, we start with a user-defined DNA sequence of the target gene before amino acid diversification, a list of desired amino acid mutations, the Type II restriction enzyme, the minimum required level of ligation efficiency, and criteria that define acceptable DNA fragments, such as minimal and maximal oligo lengths. We then search for potential cleavage sites given the size of the overhang generated by the restriction enzyme. To serve as a cleavage site, the DNA region encompassing the nucleotide and the overhang generated by the restriction enzyme must be immutable (it cannot encode diversity), and the overhang must exhibit at least the desired ligation efficiency according to empirical measurements (Potapov, Ong, Kucera, et al., 2018; Potapov,

Ong, Langhorst, et al., 2018; Pryor et al., 2020). We add sites that fulfill these requirements as nodes to the graph with their associated overhangs. An edge connects each pair of nodes if the two overhangs do not ligate to one another. We assign each edge a weight based on the number of nucleotides (proportional to the cost) required to generate the variability in the DNA fragment it represents.

A critical confounding factor in combinatorial assembly of multiple fragments that is not addressed by the scheme above is that nonadjacent overhangs might inadvertently exhibit high ligation efficiency resulting in undesired assembly products (see Figure 3 for examples). Some misassembled products may be shorter than the desired one and would therefore be preferentially amplified in downstream polymerase chain reactions (PCRs). Furthermore, some cases may result in loops of misassembled concatemers. To ensure strict assembly only of the desired product, we must ensure fidelity above the user-defined threshold. Providing the most economical solution while maintaining fidelity above the provided threshold is formally known as the constrained shortest-path problem, a recognized nonpolynomial (NP)-hard problem (Lozano & Medaglia, 2013). To overcome the hardness barrier we employ randomized color-coding (Alon et al., 1995) in which nodes that represent overhangs that are predicted to ligate at high efficiency are colored identically. We then apply a vertex rainbow shortest-path algorithm that ensures that valid paths include each color at most once, resulting in vertex rainbow paths (Figure 3). Selecting rainbow shortest-path solutions guarantees the assembly of an economical



**FIGURE 2** Key steps in the GGAssembler experimental workflow. Step IV: Order the variable segments as single-stranded DNA (ssDNA). Step V: Amplify variable segments ssDNA to generate double-stranded DNA. Step VI: Manually design and order primers for constant segments based on the output EMBL sequence file. Step VII: Amplify and purify constant segments from the input gene template. Step VIII: Execute GGA following one of several established protocols (Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020, 2022; Sikkema et al., 2023).

solution that excludes undesired products; however, this solution may not be the global optimum in terms of cost.

GGAssembler segments the target gene sequence into variable fragments that encode diversity and constant ones devoid of diversity. The algorithm generates an output table that specifies the variable fragments and the sequences of the constant fragments that can be custom-synthesized or amplified from a preexisting gene.

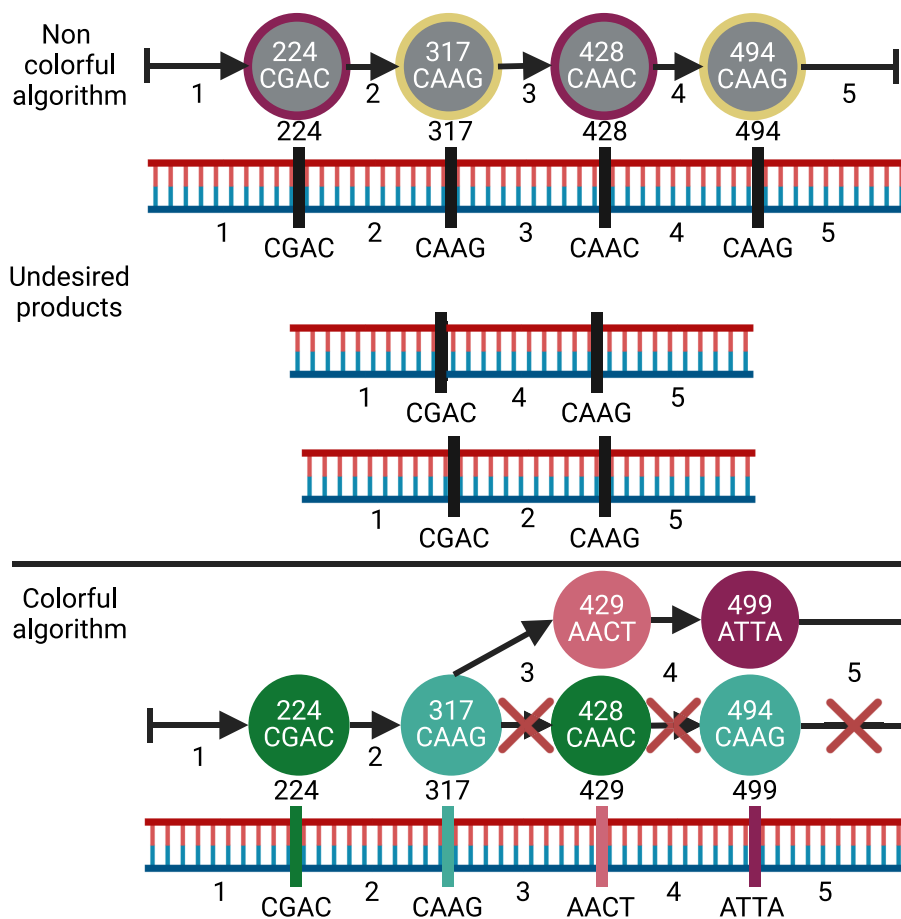
## 2.2 | Lab protocol for synthesizing a GGAssembler library

We next introduce a streamlined wet-lab protocol (Figure 2) for assembling a library generated by GGAssembler (Hoch et al., 2023). To maintain cost-

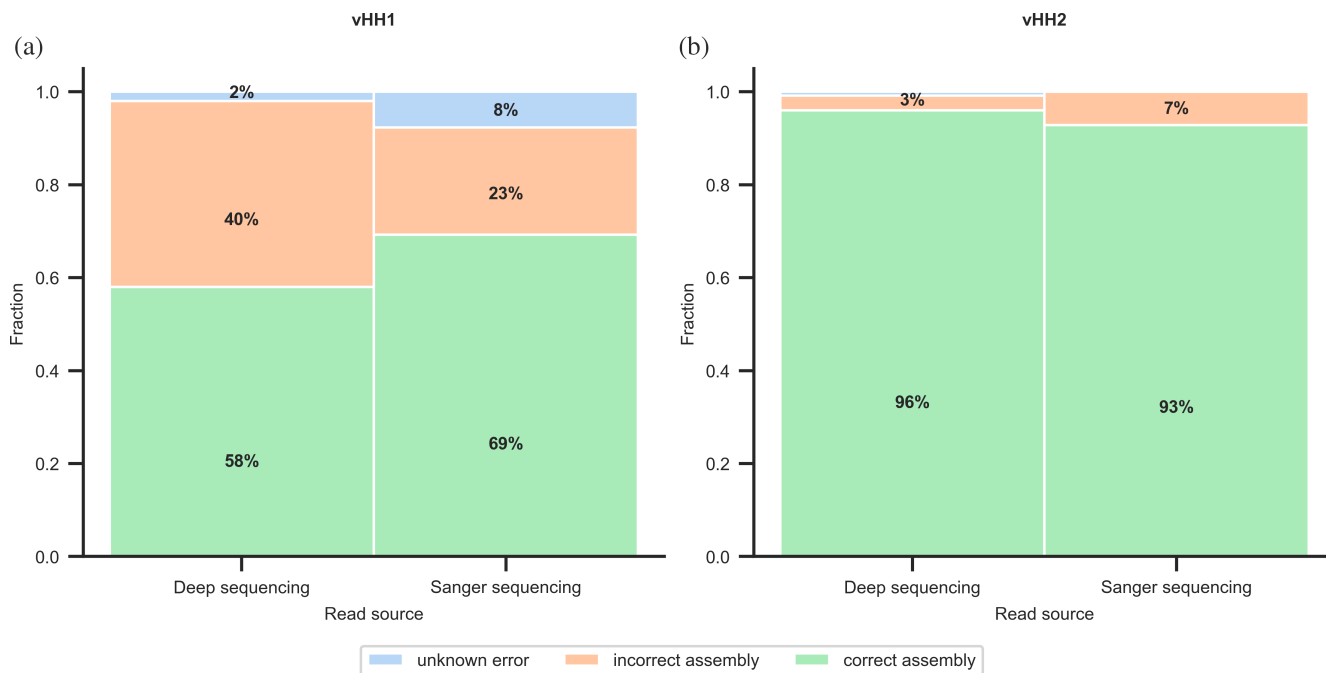
effectiveness, we amplify constant fragments with primers flanked by the appropriate restriction-enzyme recognition site and all variable DNA oligos with a single 3' primer, to allow filling the reverse strand by a polymerase. By adjusting the concentration of each DNA fragment in the assembly reaction the user can bias the resulting library toward or away from specific variants or mutations providing high operational freedom.

## 2.3 | Accurate assembly of vHH combinatorial libraries

We verify the accuracy of GGAssembler by constructing two complex camelid antibody (vHH) libraries, each comprising nine DNA segments. Unlike GFP, vHHs are



**FIGURE 3** Graph coloring in GGAssembler eliminates undesired assemblies. A subset of the path common to all shortest paths in a previously designed GFP library that encodes  $>10^7$  active-site variants (Weinstein et al., 2023). (Top) Nodes marked by cleavage-site location and overhang are colored identically if the overhangs ligate. Two nodes with the same overhang (CAAG, positions 317 and 494) are marked yellow, and nodes that are not Watson–Crick pairs but are predicted to exhibit high-efficiency ligation (Potapov, Ong, Kucera, et al., 2018; Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020) are in purple. All shortest-path solutions would result in misassembled products. In this case, segments 1 and 4 are complementary as are 2 and 5, leading to specific undesired products shown in the figure. In addition, the end of segment 3 can ligate to the beginning of segment 2 leading to an infinite number of different undesired products (not shown). (Bottom) A rainbow shortest-path search excludes undesired solutions, forcing the search to find alternative paths that do not misassemble.



**FIGURE 4** Sequencing read classification. (a, b) Reads were classified based on their alignment to each of the fragments. We classified reads containing both flanking constant segments based on the type of error we found “incorrect assembly,” “correct assembly,” and “unknown error.” We have omitted class fractions below 1%.

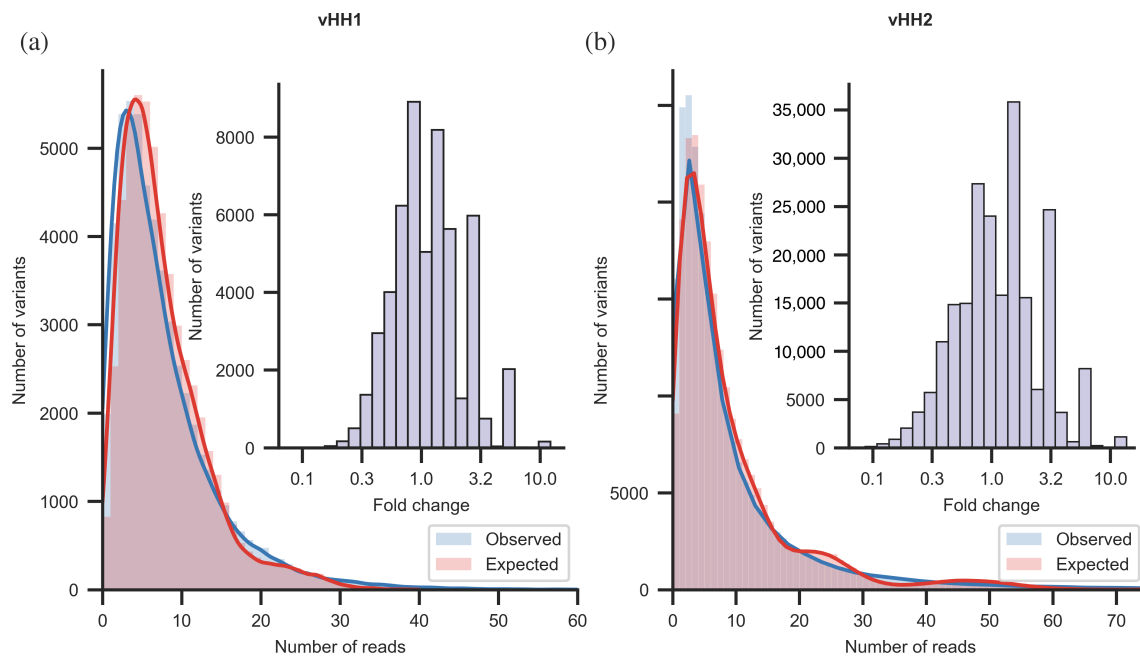
small (<130 amino acids), allowing us to analyze the accuracy and representation bias of the library through deep sequencing.

Using the htFuncLib protein design approach that generated the GFP libraries (Weinstein et al., 2023), we designed two camelid antibody (vHH) libraries, vHH1 and vHH2, that encoded diversity in the three complementarity-determining regions (CDRs) of each antibody. Briefly, htFuncLib uses the FuncLib design method (Khersonsky et al., 2018) and a machine-learning method called EpiNNet (R. Lipsh-Sokolik et al., 2023) to select mutations that are predicted to combine freely to generate low-energy multipoint mutation libraries. This approach circumvents experimental mutation scanning and nominates mutations for combinatorial variant libraries that are predicted to be tolerated both individually and in combination with one another, thus addressing the problem that many combinatorial mutations in protein active sites exhibit negative epistasis (Rosalie Lipsh-Sokolik & Fleishman, 2024). We demonstrated that such designed variant libraries are enriched in stable, well-folded, and potentially functional protein variants that exhibited vast changes in their functional profile (R. Lipsh-Sokolik et al., 2023; Weinstein et al., 2023). Applied to the vHHs, htFuncLib results in 55,296 and 233,280 designs, in 11 and 13 positions across the three CDRs for vHH1 and vHH2, respectively. Following

protein design calculations, we applied GGAssembler, including codon compression.

We transformed the libraries into bacterial cells and sequenced 12 and 14 colonies for vHH1 and vHH2, respectively. 75% and 93% of the resulting sequences were in frame and matched desired designs for vHH1 and vHH2, respectively, on par with the current state of the art (Choi et al., 2022; Daffern et al., 2023). In both errors observed in vHH1, a pair of overhangs misligated, resulting in a short product (Table S1). Retrospectively, the overhangs used to generate vHH1 were designed based on data from an early study on fidelity in GGA (Potapov, Ong, Kucera, et al., 2018), but a subsequent study (Potapov, Ong, Langhorst, et al., 2018) showed that the selected overhangs exhibit low ligation fidelity (60%) matching the fidelity we observed in Sanger sequencing (Figure 4a). Using the updated fidelity data (Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020) in GGAssembler, the overhangs in vHH1 would not have been selected.

Next, we subjected the assembled libraries (before transforming into bacteria so as not to introduce biological bias) to deep sequencing analysis. To assess biases in the library we started with stringent quality control, discarding any read that was not completely identical to a desired variant in any part of the variable segments (mismatches and deletions were allowed in the constant



**FIGURE 5** Low bias in GGAssembler libraries. Deep sequencing analysis of libraries (a) vHH1 and (b) vHH2. The observed distribution of variant reads compared with the expected mixed Poisson distribution. (Insets) Log fold differences between expected and observed read counts.

segments). 96% and 93% of designed variants were uniquely identified in vHH1 and vHH2, respectively, resulting in nearly complete coverage of the designed libraries. The distribution of variants is expected to be multimodal, and the observed distribution of variants reflects the expected mixed Poisson distribution very closely (Figure 5a,b). Only two variants exhibited more than a tenfold count difference in vHH1 (out of >55,000 variants) and approximately 1300 of vHH2 (out of >230,000 variants; 0.6%), and in both libraries, the discrepancy was at most 22 fold (Figure 5a,b, insets).

We also assessed the observed frequencies of variable segments relative to the expectation. The fold-change differences ranged between 0.58–1.76 and 0.47–2.67 in vHH1 and vHH2, respectively (Figure 6a,b). Mutation frequencies exhibited an even narrower range, with fold-change ranging 0.67–1.26 and 0.72–1.32 in vHH1 and vHH2, respectively (Figure 6c,d); this range is considered to be near-uniform (Daffern et al., 2023).

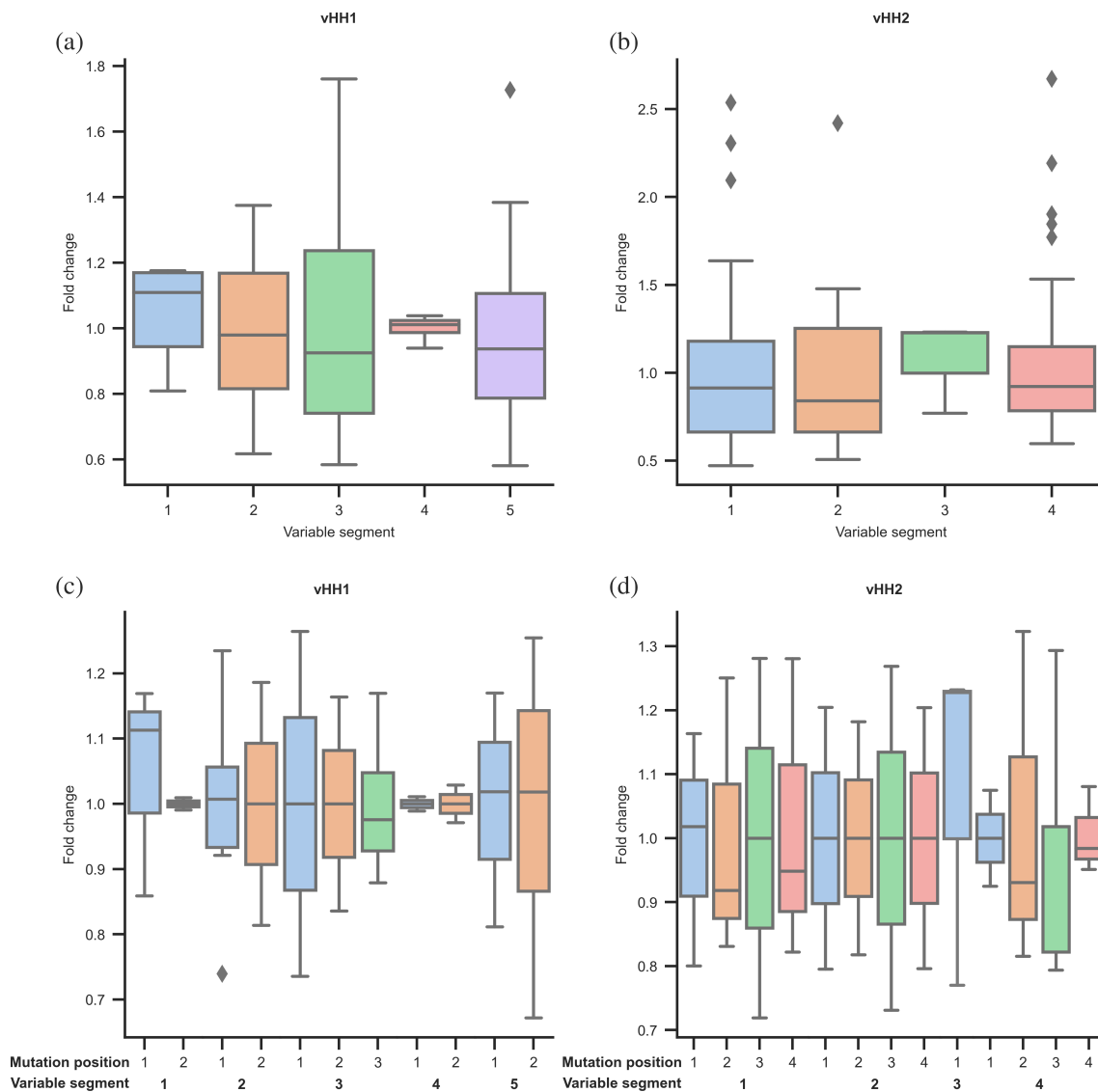
Finally, we analyzed misligations to understand the reasons for failures in assembly (see Methods) (Figure 4a,b; Table S2). Incorrect assemblies (40% and 3% for vHH1 and vHH2, respectively) were all shorter than the desired assembly. Additionally, <3% of the sequencing reads were assigned as misassembled for unknown reasons. The distribution of correctly and incorrectly assembled products is similar in Sanger and deep sequencing (Figure 4a,b; Table S2), highlighting again the high accuracy of the method and the importance of using the more recent and reliable GGA empirical

ligation data (Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020) as implemented in vHH2. We conclude that GGAssembler produces a nearly complete representation and low bias of the designed variants and correct assembly of the vast majority of the product. Thus, the GGAssembler method and the experimental assembly protocol generate highly complex variant libraries reliably, accurately, and economically.

The total reaction costs for vHH1 and vHH2, respectively, are \$392.16 and \$439.08 of which \$365.85 and \$425.70 (93% and 97%) are DNA-associated costs, translating to 0.7¢ and 0.2¢ per variant (Table S3). We further note that in many variant screening workflows, one may wish to generate alternative libraries that focus diversity on different regions of the protein. In such cases, parts of the assembly reaction can be reused to generate differentially biased products at no additional DNA cost.

### 3 | DISCUSSION

New approaches to probe the mutational tolerance of proteins experimentally and computationally fuel interest in methods for efficient synthesis of combinatorial mutation libraries (Alejaldre et al., 2021; Daffern et al., 2023; Jacobs et al., 2015; Kirby et al., 2021; Lund et al., 2024; Öling et al., 2022; Plesa et al., 2018; Pullmann et al., 2019; Shimko et al., 2020; Sidore et al., 2020; Tretyachenko et al., 2020; Wrenbeck et al., 2016; Yamamoto et al., 2020). GGAssembler exploits the fact that desired



**FIGURE 6** Low-bias in the representation of assembled segments and mutations. (a, b) Observed versus expected distribution of segments in the assembled product. The vHH1 and vHH2 libraries were encoded with 5 and 4 variable segments, respectively (for cost-effective representation of all diversity in complementarity-determining regions [CDR] H3). Comparison of the theoretical frequency of each variable segment compared to the observed frequency seen in deep sequencing. (c, d) Observed versus expected distribution of mutated positions. In all plots, box bounds signify the first, median and third quartiles. Whiskers represent 1.5 times the interquartile range and outliers are shown as diamonds.

diversity in active sites is typically clustered in several contiguous epitopes (e.g., antibody CDRs or enzyme active-site loops) that are separated by immutable regions. This clustering lends itself to breaking the assembly reaction into constant fragments that can be synthesized or amplified from a preexisting gene and variable regions that can be economically encoded by custom-synthesized short oligos or oligo pools. GGAssembler uses codon compression and chooses a set of empirically determined high-fidelity overhangs to reduce costs and ensure accurate assembly even in complex libraries with more than a dozen mutated positions across several noncontiguous epitopes. In the two

camelid antibody examples provided here and our previous application to the GFP chromophore-binding pocket (Weinstein et al., 2023), correct assemblies were made from 9 and 20 parts, respectively, demonstrating that even large, multiepitope active sites can be effectively generated with a simple experimental workflow that can be accomplished in a one-pot in vitro reaction. Critically, our variant libraries were designed strictly according to atomistic considerations without attempting to reduce the complexity of the assembly reaction like previous variant library studies (Jacobs et al., 2015; Shimko et al., 2020), yet the GGAssembler approach successfully produced economical, high-fidelity libraries. Thus,



GGAssembler opens the way to encoding user-defined diversity in large and complex functional epitopes.

While we strove for generality in developing GGAssembler, we note that it only applies to constructing combinatorial assembly libraries. With the current active research pushing the boundary of what is possible with GGA, we hope to extend GGAssembler capabilities further to allow the assembly of multiple homologous proteins in the future. Our results show the importance of inferring overhang fidelity from experiments under settings used in the assembly reaction (Potapov, Ong, Kucera, et al., 2018; Pryor et al., 2020), with notable improvement in predictive ability for library vHH2 relative to the previously assembled vHH1 library. Future research into the fidelity of multiple restriction enzymes and ligases in the same reaction pot could enable executing complex multistep reactions in a single step. We envision that GGAssembler will enable efficient and economical study of the mutational tolerance of antibodies, enzymes, pathogenic antigens and other functional proteins and an effective approach to screen for new or improved functions (R. Lipsh-Sokolik et al., 2023; Weinstein et al., 2023).

## 4 | METHODS

### 4.1 | GGA optimization

The fidelity  $F$  of a given overhang  $O$  is defined as the proportion of the number of times overhang  $O$  ligates to its Watson–Crick pair and vice versa ( $N_{\text{correct}}$ ) divided by the number of times overhang  $O$  ligates to any overhang ( $N_{\text{total}}$ ).

$$F(O) = \frac{N_{\text{correct}}}{N_{\text{total}}}. \quad (1)$$

The efficiency of a given overhang  $O$  is relative to overhang with the most number of ligation events  $O_{\text{max}}$  and defined as the proportion of times overhang  $O$  ligates to any overhang ( $N_{\text{total}}$ ) divided by the number of times overhang  $O_{\text{max}}$  ligates to any overhang ( $N_{\text{max}}$ ).

$$E(O) = \frac{N_{\text{total}}}{N_{\text{max}}}. \quad (2)$$

### 4.2 | Computational library design

#### 4.2.1 | Codon compression

For each diversified amino acid position, we applied codon compression by generating all ambiguous codons that encode a portion of the required diversity using

CodonGenie (Swainston et al., 2017) and formulating the compression as the set cover problem, where the desired amino acid diversity in each position is the set to be covered, and each ambiguous codon provided by CodonGenie is a subset of that required diversity. We have implemented Dancing Links (DLX) (Knuth, 2000) to solve the exact cover problem, which results in every amino acid identity appearing exactly once.

#### 4.2.2 | DNA segmentation

We model all possible segmentations as a directed graph with each node representing a cleavage site and overhang, and edges representing DNA segments between a pair of cleavage sites. Edges are weighted by the number of nucleotides required to produce that segment to represent relative DNA costs. As stated in the results, finding economical segmentations with fidelity above a given threshold is a known NP-hard problem (Lozano & Medaglia, 2013). To overcome the hardness barrier, we developed a randomized algorithm employing color-coding (Alon et al., 1995), drawing random colorings that color nodes identically if empirical ligation preferences (Potapov, Ong, Kucera, et al., 2018; Potapov, Ong, Langhorst, et al., 2018; Pryor et al., 2020) indicate the overhangs would ligate. Next, for each random coloring drawn, we run a rainbow version of the Dijkstra shortest-path algorithm (Dijkstra, 1959), which considers extending a rainbow path only if the added node color does not already appear in the path. The result is a DNA segmentation and set of overhangs that ensure high-fidelity assembly. Selecting rainbow shortest-path solutions guarantees the assembly of an economical solution that excludes undesired products.

#### 4.2.3 | Algorithm, time complexity, and correctness

Our proposed solution is a randomized algorithm that searches for the rainbow shortest path covering the entire wild-type DNA sequence. We employ randomness because given graph  $G$  of  $n$  vertices, vertices colored by  $k$  colors and vertices  $s, t$  (start, terminus), the  $s$ - $t$  rainbow vertex connection problem has been shown to be NP-Complete with runtime complexity of  $O(2^k n^{O(1)})$  (Chen et al., 2011). Because the number of colors used when coloring nodes by their empirical ligation preferences considerably exceeds the number of fragments in most cases, drawing a random coloring results in shorter runtimes. To illustrate that the number of colors typically far exceeds the number of fragments, consider the vHH designs shown in the Results. Using *BsaI* as the Golden

Gate restriction enzyme results in 4 bp overhangs translating to at least 128 overhang sequences and colors (for each overhang and its reverse complement) whereas the number of fragments in the vHH libraries is merely nine. Pseudo-code and proofs of correctness and run-time complexity are presented in Supporting Information File 2.

We found that while running the algorithm for  $(2e)^{\#fragments}$  iterations ensures finding the lowest-cost solution with high probability if such a solution exists; in many cases, the difference in cost between the nonrainbow lowest-cost solutions (using Dijkstra's algorithm; Dijkstra, 1959) and the rainbow solutions is negligible (e.g., on the order of tens of bps for the entire reaction). With such a slight difference in cost, we prefer solutions with fewer fragments or higher fidelity. We have also observed that even when running the randomized algorithm for 100–1000  $[(2e)^{2.72} - (2e)^{4.08}]$  iterations, the rainbow solutions obtained are similar in cost to the nonrainbow lowest-cost solutions found, resulting in run time of several minutes from start to finish.

### 4.3 | Experimental library assembly

The full protocol is described in Hoch et al., 2023 and in Supporting Information 1.

The primers used for the reactions are described in Tables S4 and S5.

### 4.4 | Library cloning

The assembled library after Golden Gate was cloned into pNACP plasmid (Uchański et al., 2019) (kindly provided by Jan Steyaert, Vrije Universiteit Brussels) using homologous recombination in yeast (Gietz & Schiestl, 2007). Then, 100  $\mu$ L yeast cells after recombination were plated on SDCAA plates. Colonies were collected with 1.5 mL sterile DDW and extracted using Zymoprep Yeast Plasmid Miniprep II (Zymo Research, CAT #D2004).

#### 4.4.1 | Colony PCR and Sanger sequencing

The cloned libraries after recombination to pNACP were transformed into E. Cloni 10G cells (Lucigen). Next, colony PCR and Sanger sequencing were performed using standard protocols.

### 4.5 | Deep sequencing

Amplicon libraries were prepared as previously described (Blecher-Gonen et al., 2013). The libraries after GGA

were sequenced using a paired-end V3 600 cycles Illumina kit (Illumina MS-102-3003) on an Illumina Miseq. We preprocessed Fastq sequences using the BBTools software suite Bushnell et al., 2017, and pairwise aligned the results to constant and variable DNA segments using Parasail (Daily, 2016). We further analyzed alignment results using Python (Cock et al., 2009; Granger & Pérez, 2021; Harris et al., 2020; McKinney, 2010; van der Walt et al., 2011), by assigning each read bp its corresponding segments. This allows us to quickly identify correctly assembled sequences and assign each incorrectly assembled sequence the reasons it was assembled as such. We then extracted variable segments and filtered out sequences that did not exactly match variable DNA segments for the bias analysis.

### AUTHOR CONTRIBUTIONS

**Shlomo Yakir Hoch:** Conceptualization; investigation; writing – original draft; methodology; validation; visualization; writing – review and editing; software; formal analysis. **Ravit Netzer:** Investigation; resources; data curation; methodology. **Jonathan Yaacov Weinstein:** Investigation; software; data curation; resources; methodology. **Lucas Krauss:** Data curation; resources; methodology. **Karen Hakeny:** Data curation; resources. **Sarel Jacob Fleishman:** Resources; supervision; project administration; funding acquisition.

### ACKNOWLEDGMENTS

We thank David Peleg for helpful discussions and comments on the algorithm and Olga Khersonsky and Ariel Tennenhouse for critical reading. Figures 1–3 were created with BioRender.com. The research was supported by the Volkswagen Foundation (94747), the Israel Science Foundation (1844), the European Research Council through a Consolidator Award (815379), the Dr. Barry Sherman Institute for Medicinal Chemistry, and a donation in memory of Sam Switzer.

### CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

### ORCID

Shlomo Yakir Hoch  <https://orcid.org/0000-0003-3991-1533>

Sarel Jacob Fleishman  <https://orcid.org/0000-0003-3177-7560>

### REFERENCES

- Alejaldre L, Pelletier JN, Quaglia D. Methods for enzyme library creation: which one will you choose? *BioEssays*. 2021;43(8): 2100052.
- Alon N, Yuster R, Zwick U. Color-coding. *J ACM*. 1995;42(4): 844–56.

- Andreou AI, Nakayama N. Mobius assembly: a versatile Golden Gate framework towards universal DNA assembly. *PLoS One*. 2018;13(1):e0189892.
- Blecher-Gonen R, Barnett-Itzhaki Z, Jaitin D, Amann-Zalcenstein D, Lara-Astiaso D, Amit I. High-throughput chromatin immunoprecipitation for genome-wide mapping of in vivo protein-DNA interactions and epigenomic states. *Nat Protoc*. 2013;8(3):539–54.
- Bushnell B, Rood J, Singer E. BBMerge—accurate paired shotgun read merging via overlap. *PLoS One*. 2017;12(10):e0185056.
- Chen L, Li X, Shi Y. The complexity of determining the rainbow vertex-connection of a graph. *Theor Comput Sci*. 2011;412(35):4531–5.
- Choi H, Choi Y, Choi J, Lee AC, Yeom H, Hyun J, et al. Purification of multiplex oligonucleotide libraries by synthesis and selection. *Nat Biotechnol*. 2022;40(1):47–53.
- Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*. 2009;25(11):1422–3.
- Daffern N, Francino-Urdaniz IM, Baumer ZT, Whitehead TA. Standardizing cassette-based deep mutagenesis by Golden Gate assembly. *Biotechnol Bioeng*. 2023;121:281–90. <https://doi.org/10.1002/bit.28564>
- Daily J. Parasail: SIMD C library for global, semi-global, and local pairwise sequence alignments. *BMC Bioinformatics*. 2016;17-(February):81.
- Dijkstra EW. A note on two problems in Connexion with graphs. *Numer Math*. 1959;1(1):269–71.
- Engler C, Gruetznern R, Kandzia R, Marillonnet S. Golden Gate shuffling: a one-pot DNA shuffling method based on type II restriction enzymes. *PLoS One*. 2009;4(5):e5553.
- Engler C, Kandzia R, Marillonnet S. A one pot, one step, precision cloning method with high throughput capability. *PLoS One*. 2008;3(11):e3647.
- Fowler DM, Araya CL, Fleishman SJ, Kellogg EH, Stephany JJ, Baker D, et al. High-resolution mapping of protein sequence-function relationships. *Nat Methods*. 2010;7(9):741–6.
- Gietz RD, Schiestl RH. Large-scale high-efficiency yeast transformation using the LiAc/SS carrier DNA/PEG method. *Nat Protoc*. 2007;2(1):38–41.
- Goldenzweig A, Goldsmith M, Hill SE, Gertman O, Laurino P, Ashani Y, et al. Automated structure- and sequence-based design of proteins for high bacterial expression and stability. *Mol Cell*. 2016;63(2):337–46.
- Granger BE, Pérez F. Jupyter: thinking and storytelling with code and data. *Comput Sci Eng*. 2021;23(2):7–14.
- Guntas G, Purbeck C, Kuhlman B. Engineering a protein–protein interface using a computationally designed library. *Proc Natl Acad Sci U S A*. 2010;107(45):19296–301.
- Harris CR, Jarrod Millman K, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. *Nature*. 2020;585(7825):357–62.
- Hoch SY, Netzer R, Hakeny K, Fleishman SJ. GGAssembler library construction. 2023 <https://doi.org/10.17504/protocols.io.81wgbxqkolkp/v3>
- Jacobs TM, Yumerefendi H, Kuhlman B, Leaver-Fay A. SwiftLib: rapid degenerate-codon-library optimization through dynamic programming. *Nucleic Acids Res*. 2015;43(5):e34.
- Khersonsky O, Lipsh R, Avizemer Z, Ashani Y, Goldsmith M, Leader H, et al. Automated design of efficient and functionally diverse enzyme repertoires. *Mol Cell*. 2018;72(1):178.e5–186.e5.
- Kirby MB, Medina-Cucurella AV, Baumer ZT, Whitehead TA. Optimization of multi-site nicking mutagenesis for generation of large, user-defined combinatorial libraries. *Protein Eng Des Sel*. 2021;34:gzab017.
- Knuth DE. Dancing links. *arXiv*. 2000 <https://doi.org/10.48550/arXiv.cs/0011047>
- Lipsh-Sokolik R, Khersonsky O, Schröder SP, de Boer C, Hoch S-Y, Davies GJ, et al. Combinatorial assembly and Design of Enzymes. *Science*. 2023;379(6628):195–201.
- Lipsh-Sokolik R, Fleishman SJ. Addressing epistasis in the design of protein function. *Proc Natl Acad Sci U S A*. 2024;121(34):e2314999121. <https://doi.org/10.1073/pnas.2314999121>
- Listov D, Goverde CA, Correia BE, Fleishman SJ. Opportunities and challenges in design and optimization of protein function. *Nat Rev Mol Cell Biol*. 2024;25(8):639–53.
- Lohman GJS, Bauer RJ, Nichols NM, Mazzola L, Bybee J, Rivizzigno D, et al. A high-throughput assay for the comprehensive profiling of DNA ligase Fidelity. *Nucleic Acids Res*. 2016;44(2):e14.
- Lozano L, Medaglia AL. On an exact method for the constrained shortest path problem. *Comput Oper Res*. 2013;40(1):378–84.
- Lund S, Potapov V, Johnson SR, Buss J, Tanner NA. Highly parallelized construction of DNA from low-cost oligonucleotide mixtures using data-optimized assembly design and Golden Gate. *ACS Synth Biol*. 2024;13(3):745–51.
- Markel U, Essani KD, Besirlioglu V, Schifffels J, Streit WR, Schwaneberg U. Advances in ultrahigh-throughput screening for directed enzyme evolution. *Chem Soc Rev*. 2020;49(1):233–62.
- Mayer H. Optimization of the EcoRI\*-activity of EcoRI endonuclease. *FEBS Lett*. 1978;90(2):341–4.
- McKinney W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, 56–61. <https://doi.org/10.25080/majora-92bf1922-00a>
- Öling D, Lan-Chow-Wing O, Martella A, Gilberto S, Chi J, Cooper E, et al. FRAGLER: a fragment recycler application enabling rapid and scalable modular DNA assembly. *ACS Synth Biol*. 2022;11(7):2229–37.
- Plesa C, Sidore AM, Lubock NB, Zhang D, Kosuri S. Multiplexed gene synthesis in emulsions for exploring protein functional landscapes. *Science*. 2018;359(6373):343–7.
- Potapov V, Ong JL, Kucera RB, Langhorst BW, Bilotti K, Pryor JM, et al. Comprehensive profiling of four base overhang ligation Fidelity by T4 DNA ligase and application to DNA assembly. *ACS Synth Biol*. 2018;7(11):2665–74.
- Potapov V, Ong JL, Langhorst BW, Bilotti K, Cahoon D, Canton B, et al. A single-molecule sequencing assay for the comprehensive profiling of T4 DNA ligase fidelity and bias during DNA end-joining. *Nucleic Acids Res*. 2018;46(13):e79.
- Pryor JM, Potapov V, Bilotti K, Pokhrel N, Lohman GJS. Rapid 40 kb genome construction from 52 parts through data-optimized assembly design. *ACS Synth Biol*. 2022;11(6):2036–42.
- Pryor JM, Potapov V, Kucera RB, Bilotti K, Cantor EJ, Lohman GJS. Enabling one-pot Golden Gate assemblies of unprecedented complexity using data-optimized assembly design. *PLoS One*. 2020;15(9):e0238592.

- Püllmann P, Ulpinnis C, Marillonnet S, Gruetzner R, Neumann S, Weissenborn MJ. Golden mutagenesis: an efficient multi-site-saturation mutagenesis approach by Golden Gate cloning with automated primer design. *Sci Rep.* 2019;9(1):10932.
- Sarrion-Perdigones A, Falconi EE, Zandalinas SI, Juárez P, Fernández-del-Carmen A, Granell A, et al. GoldenBraid: an iterative cloning system for standardized assembly of reusable genetic modules. *PLoS One.* 2011;6(7):e21622.
- Sarrion-Perdigones A, Vazquez-Vilar M, Palací J, Castelijn B, Forment J, Ziarsolo P, et al. GoldenBraid 2.0: a comprehensive DNA assembly framework for plant synthetic biology. *Plant Physiol.* 2013;162(3):1618–31.
- Shimko TC, Fordyce PM, Orenstein Y. DeCoDe: degenerate codon design for complete protein-coding DNA libraries. *Bioinformatics.* 2020;36(11):3357–64.
- Sidore AM, Plesa C, Samson JA, Lubock NB, Kosuri S. DropSynth 2.0: high-fidelity multiplexed gene synthesis in emulsions. *Nucleic Acids Res.* 2020;48(16):e95.
- Sikkema AP, Kasra Tabatabaei S, Lee Y-J, Lund S, Lohman GJS. High-complexity one-pot Golden Gate assembly. *Curr Protoc.* 2023;3(9):e882.
- Swainston N, Currin A, Green L, Breitling R, Day PJ, Kell DB. CodonGenie: optimised ambiguous codon design tools. *PeerJ Comput Sci.* 2017;3(July):e120.
- Tretyachenko V, Voráček V, Souček R, Fujishima K, Hlouchová K. CoLiDe: combinatorial library design tool for probing protein sequence space. *Bioinformatics.* 2020;37(4):482–9. <https://doi.org/10.1093/bioinformatics/btaa804>
- Treynor TP, Vizcarra CL, Nedelcu D, Mayo SL. Computationally designed libraries of fluorescent proteins evaluated by preservation and diversity of function. *Proc Natl Acad Sci U S A.* 2007;104(1):48–53.
- Uchański T, Zögg T, Yin J, Yuan D, Wohlkönig A, Fischer B, et al. An improved yeast surface display platform for the screening of Nanobody immune libraries. *Sci Rep.* 2019;9(1):382.
- van der Walt S, Colbert SC, Varoquaux G. The NumPy Array: a structure for efficient numerical computation. *Comput Sci Eng.* 2011;13(2):22–30.
- Vazquez-Vilar M, Quijano-Rubio A, Fernandez-Del-Carmen A, Sarrion-Perdigones A, Ochoa-Fernandez R, Ziarsolo P, et al. GB3.0: a platform for plant bio-design that connects functional DNA elements with associated biological data. *Nucleic Acids Res.* 2017;45(4):2196–209.
- Weber E, Engler C, Gruetzner R, Werner S, Marillonnet S. A modular cloning system for standardized assembly of multigene constructs. *PLoS One.* 2011;6(2):e16765.
- Weinstein JY, Martí-Gómez C, Lipsh-Sokolik R, Hoch SY, Liebermann D, Nevo R, et al. Designed active-site library reveals thousands of functional GFP variants. *Nat Commun.* 2023;14(1):2890.
- Whitehead TA, Baker D, Fleishman SJ. Computational design of novel protein binders and experimental affinity maturation. *Methods Enzymol.* 2013;523:1–19.
- Whitehead TA, Chevalier A, Song Y, Dreyfus C, Fleishman SJ, De Mattos C, et al. Optimization of affinity, specificity and function of designed influenza inhibitors using deep sequencing. *Nat Biotechnol.* 2012;30(6):543–8.
- Wójcik M, Telzerow A, Quax WJ, Boersma YL. High-throughput screening in protein engineering: recent advances and future perspectives. *Int J Mol Sci.* 2015;16(10):24918–45.
- Wrenbeck EE, Klesmith JR, Stapleton JA, Adeniran A, Tyo KEJ, Whitehead TA. Plasmid-based one-pot saturation mutagenesis. *Nat Methods.* 2016;13(11):928–30.
- Yamamoto Y, Terai T, Kumachi S, Nemoto N. In vitro construction of large-scale DNA libraries from fragments containing random regions using Deoxyinosine-containing oligonucleotides and endonuclease V. *ACS Comb Sci.* 2020;22(4):165–71.

## SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

**How to cite this article:** Hoch SY, Netzer R, Weinstein JY, Krauss L, Hakeny K, Fleishman SJ. GGAssembler: Precise and economical design and synthesis of combinatorial mutation libraries. *Protein Science.* 2024;33(10):e5169. <https://doi.org/10.1002/pro.5169>