GGAssembler: precise and economical design and synthesis of combinatorial mutation libraries

Authors: Shlomo Yakir Hoch[1], Ravit Netzer[1,2], Jonathan Yaacov Weinstein[1,2], Lucas Krauss[1] , Karen Hakeny[1], Sarel Jacob Fleishman[1]

Affiliations:

1. Department of Biomolecular Sciences, Weizmann Institute of Science, Rehovot 7600001, Israel

2. Present Address: Scala Biodesign LTD, 50 Dizengoff St., Tel Aviv 6433222, Israel

Correspondence: hochshi@gmail.com; sarel@weizmann.ac.il

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 to design DNA fragments for the most economical way to 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 <0.005$ 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.

Introduction

Over the past decade, high-throughput screening of protein variant libraries and deep sequencing have revolutionized protein characterization, engineering, and design[1,2]. Common library-synthesis approaches introduce random or systematic single-point mutations[3]. However, significant gains in protein activity and stability often require multiple simultaneous mutations[4,5], calling for gene-synthesis approaches that introduce

combinatorial mutations. Such multipoint combinatorial libraries can be designed based on previous experimental screens[5–7] and increasingly based on computational design calculations[8–10]. However, synthesizing combinatorial libraries accurately and economically is challenging due to the large size of a typical binding or 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[9,11]. 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[11]. 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)[12,13]. GGA employs Type IIs restriction enzymes, such as *BsaI*, that cleave outside their DNA recognition sequence. Following processing by the Type IIs restriction enzyme, the recognition site is eliminated, leaving terminal overhangs that can be programmed to encode immutable (non diversified) amino acid positions. Subsequent ligation reactions of the processed fragments then seamlessly combine them into full-length genes[12,13]. In principle, GGA enables scarless gene construction of any size from small fragments[14–19]. 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[20], the sensitivity of the DNA ligase to mismatches[21], 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[14–19]. 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[22–24]. In principle, these developments allow the assembly of dozens of fragments for combinatorial gene synthesis[22,23]. Despite much progress[15,23,25,26], 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 that 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 GFP chromophore-binding pocket with a total DNA cost of $455, translating to $4.2 * 10^{-3}$ ¢ per

variant[9]. 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.

Results

A general method to design combinatorial diversity

We developed GGAssembler to provide the most economical approach to combinatorial library assembly while allowing nearly complete freedom to decide on mutation types, positions, and the size of the library. 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 (**Eq. 1**) and efficiency (**Eq. 2**) estimates[22–24]. 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 20nt DNA fragment that encodes a single mutated amino acid position that requires two codons would be 40nt (the length of the fragment times the encoded variability). Adding an extra codon at the mutated position would increase the cost to 60nt, 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**. We start by defining the amino acids that encode the desired diversity with an optional codon-compression step that uses degenerate codons[27] (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 the cost and number of overhangs. We then 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 prior to amino acid diversification, a list of desired amino acid mutations, the Type IIs restriction enzyme, the minimum required level of ligation efficiency, and a set of criteria that define acceptable DNA fragments, such as their minimal and maximal 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[22–24]. 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 non-adjacent overhangs might inadvertently exhibit high ligation efficiency resulting in undesired assembly products (see **Fig. 3** for examples). Some misassembled products may be shorter than the desired one and would therefore be preferentially amplified in downstream PCR reactions. Furthermore, some cases may result in loops of misassembled concatemers. To ensure strict assembly only of the desired product, we employ color-coding[28] in which nodes that represent overhangs that are predicted to ligate at high efficiency are colored identically. We then apply a colorful shortest-path algorithm that ensures that valid paths include each color at most once (**Fig. 3**). Selecting colorful shortest-path solutions guarantees the assembly of the most economical solution that excludes undesired products.

GGAssembler segments the target gene sequence into variable fragments that encode diversity and constant ones devoid of diversity. We provide 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.

We next introduce a streamlined wet-lab protocol (**Fig. 2**) for assembling a library generated by GGAssembler[29]. 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 towards or away from specific variants or mutations providing high operational freedom.

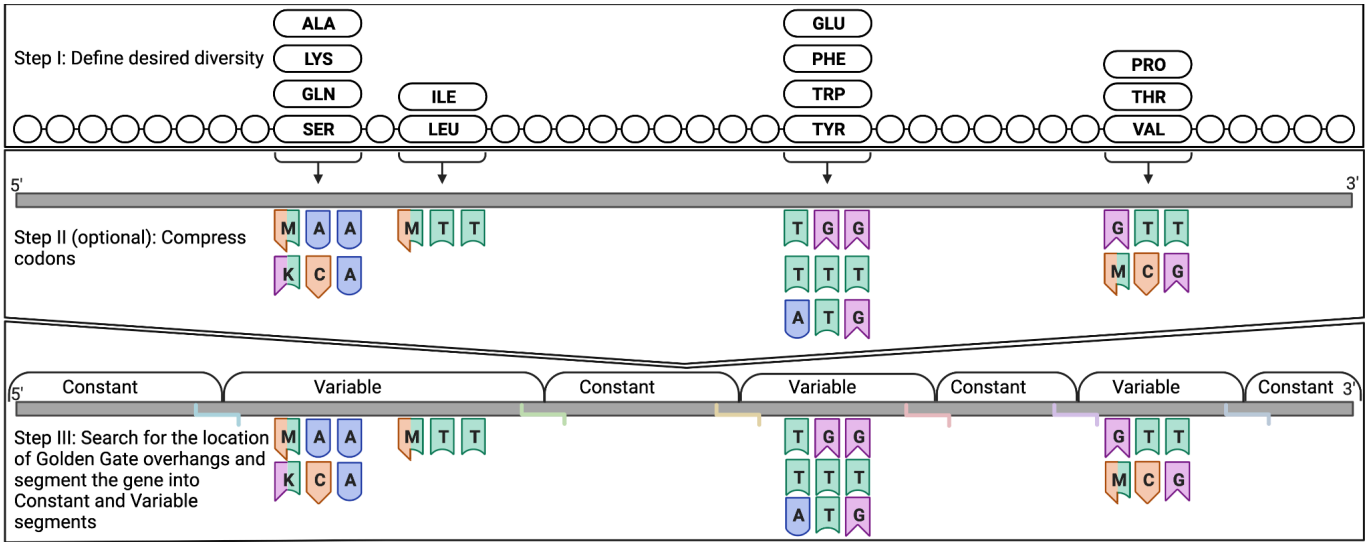

Figure 1. Key steps in the GGAssembler computational workflow.

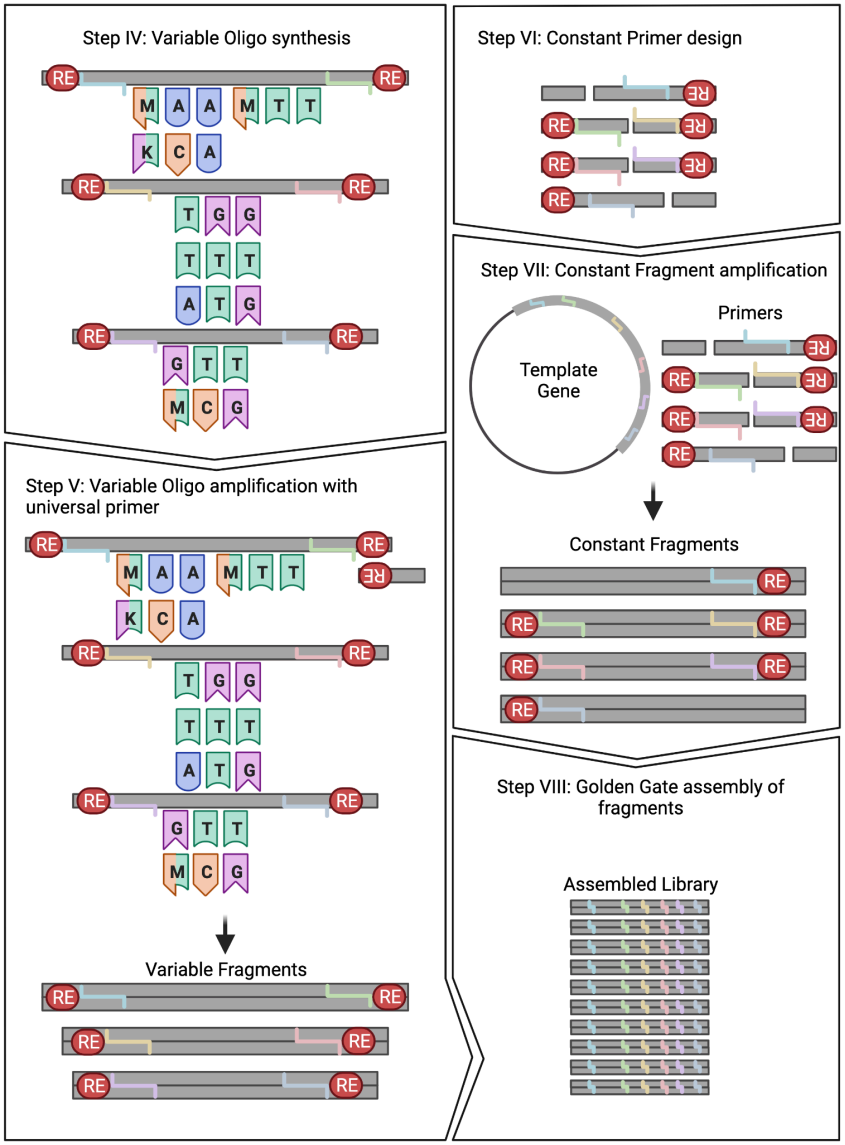

Figure 2. Key steps in the GGAssembler experimental workflow.

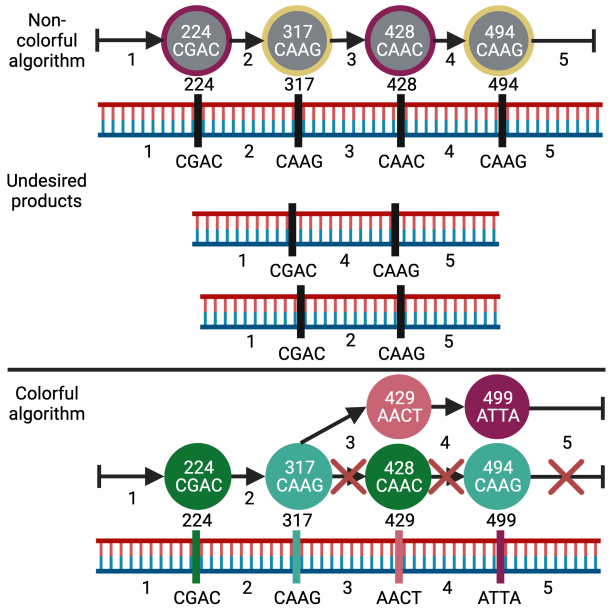

Figure 3. **Graph coloring in GGAssembler eliminates undesired assemblies.** A subset of the path common to all shortest paths in a previously designed GFP library that encodes >10^7 active-site variants[9]. (top) Nodes marked by cleavage-site location and overhang are colored identically if the overhangs 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[22–24] 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 colorful shortest-path search excludes undesired solutions, forcing the search to find alternative paths that do not mis-assemble.

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 small (<130 amino acids), allowing us to analyze the accuracy and representation bias of the library through deep sequencing. The entire reaction cost is <$423 per library of which $396 (94%) are DNA-associated costs, translating to 0.8¢ and 0.2¢ per variant in the two libraries. Costs scale roughly logarithmically with the number of fragment combinations, and larger libraries substantially reduce cost per variant. 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.

Using the htFuncLib protein design approach that generated the GFP libraries[9], we designed two camelid antibody (vHH) libraries, vHH1 and vHH2, that encoded diversity in the three complementarity-determining regions (CDRs). Briefly, htFuncLib uses Rosetta atomistic design calculations and a machine-learning method called EpiNNet[11] 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. We demonstrated that such designed variant libraries are enriched in stable, well-folded, and potentially functional protein variants that exhibited vast changes in functional profile[9,11]. 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[26,30]. In both errors observed in vHH1, a pair of overhangs misligated resulting in a short product **(Supplemental Table 1)**. Retrospectively, the overhangs used to generate vHH1 were designed based on data from an early study on fidelity in GGA[22]; but a subsequent study[22] showed that the selected gates exhibit low ligation fidelity (60%) matching the fidelity observed in Sanger sequencing (**Figure 6A**). Using the updated fidelity data[23] in GGAssembler, the overhangs in vHH1 would not have been selected.

Next, we subjected the assembled libraries (before transforming 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 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 (**Fig. 4A and B**). Only two variants exhibited more than a tenfold count difference in vHH1 (out of >55,000 variants) and approximately 1,300 of vHH2 (out of >230,000 variants; 0.6%), and in both libraries, the discrepancy was at most 22 fold (**Fig. 4A and 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 5A, 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 5C, D**).

Finally, we analyzed misligations to understand the reasons for failures in assembly (see Methods) (**Figure 6A, B**). Correct and incorrect assembly products were distributed similarly to the results of the Sanger sequencing. 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 reason. The distribution of correctly and incorrectly assembled product is similar in Sanger and deep sequencing (**Figure 6A, B**), highlighting again the high accuracy of the method when using the more recent and reliable GGA empirical ligation data 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.

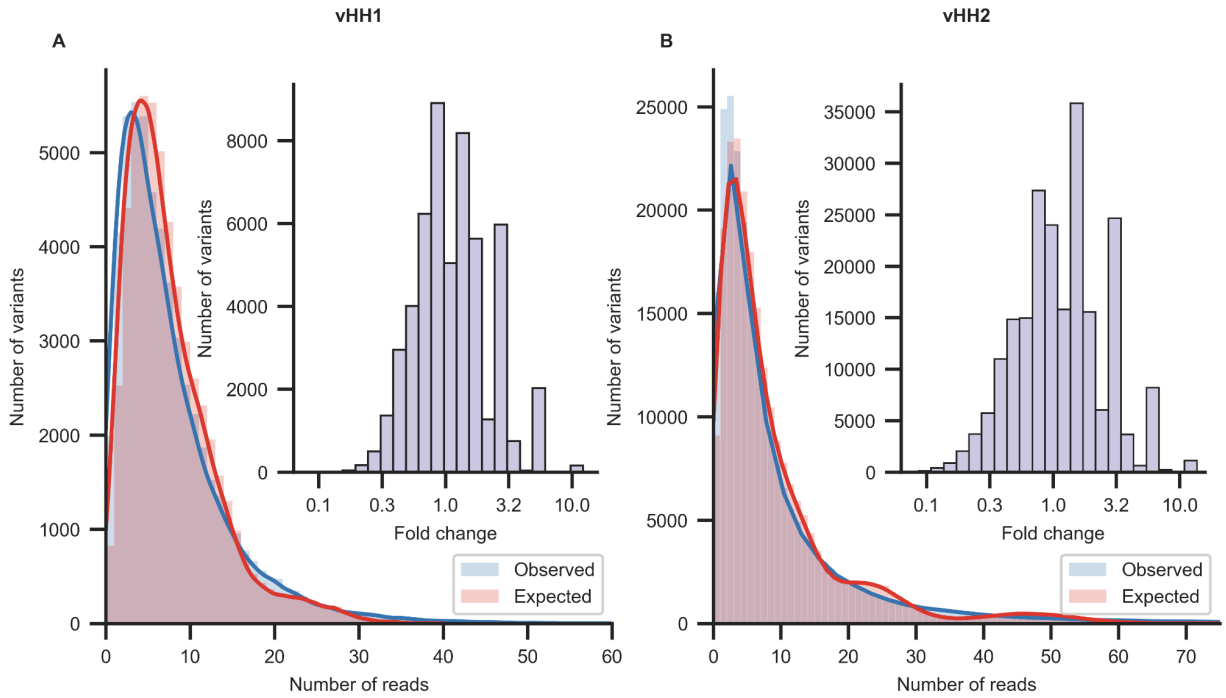

Figure 4. **Low bias in GGAssembler libraries.** Deep sequencing analysis of libraries (A) vHH1 and (B) vHH2. The observed distribution of variant reads compared to the expected mixed Poisson distribution. (insets) Log fold differences between expected and observed read counts.

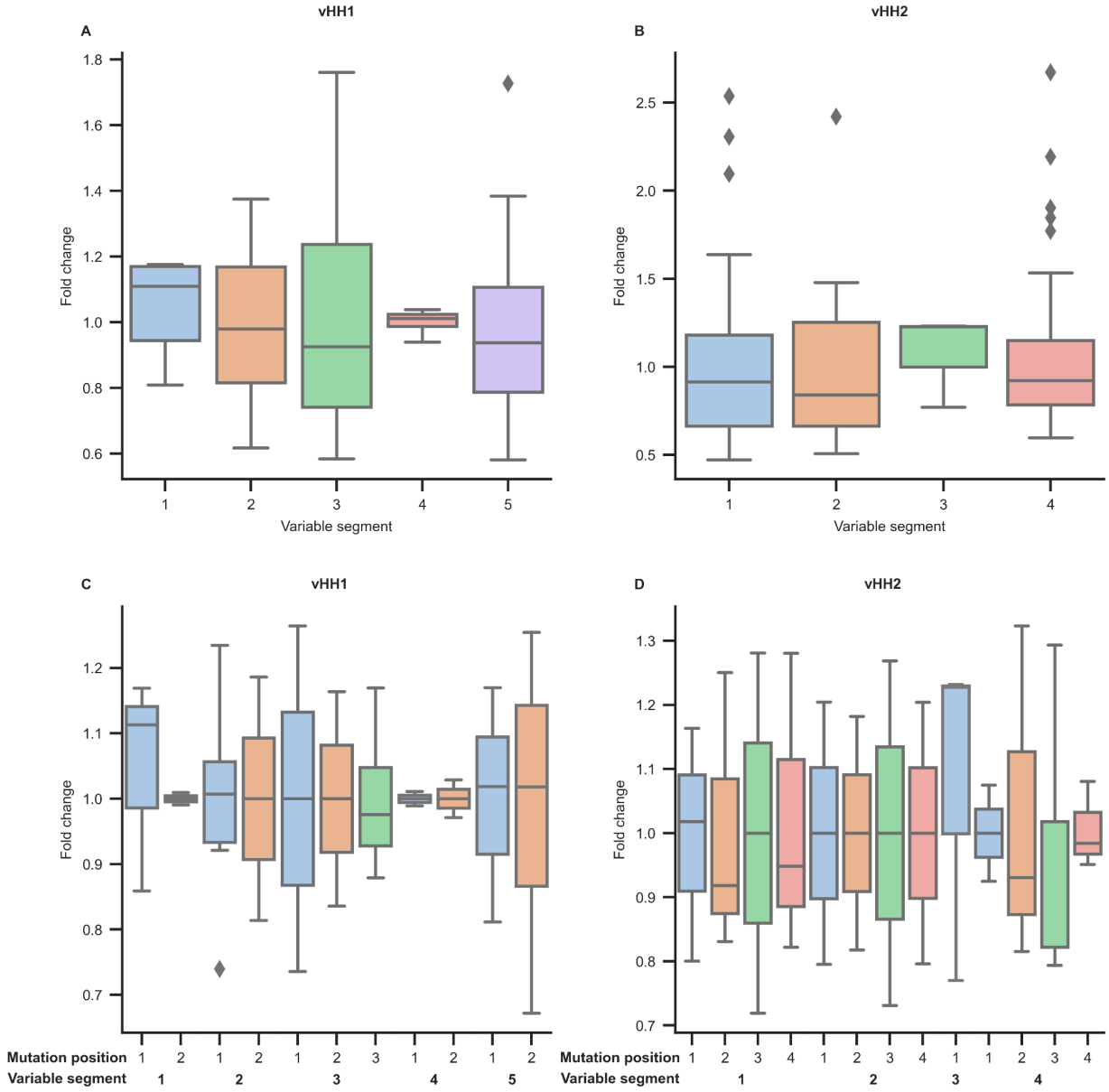

Figure 5. Low-bias in the representation of assembled segments and mutations. (**A, B**) Observed versus expected distribution of segments in the assembled product. The vHH1 and vHH2 libraries were encoded with 5 and 4 variable segments, respectively (for cost effective representation of all diversity in CDRH3). 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.

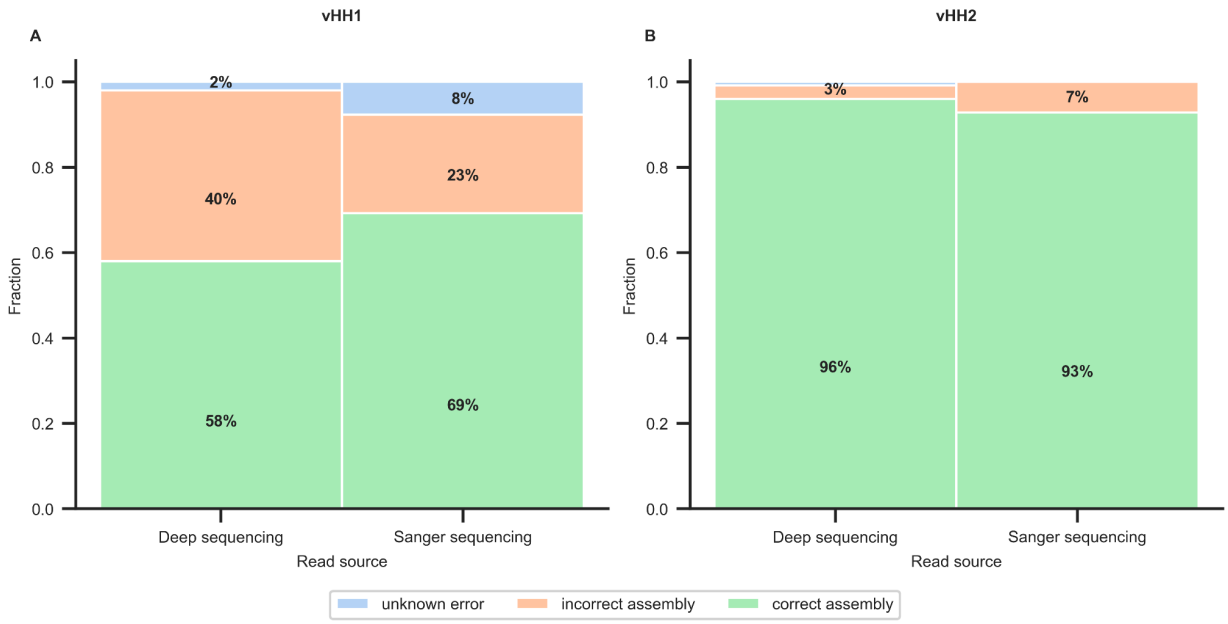

Figure 6. Sequencing read classification. (**A, B**) Reads were classified based on their alignment to each of the fragments. We classified reads containing both flanking constant segments based on the type of error we found "incorrect assembly", "correct assembly" and "unknown error". We have omitted class fractions below 1%.

Discussion

New approaches to probe the mutational tolerance of proteins experimentally and computationally fuel interest in methods for efficient synthesis of combinatorial mutation libraries[3,15,25,26,31–37]. GGAssembler exploits the fact that desired 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 minimize 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[9], correct assemblies were made from 9 and 20 parts, respectively, demonstrating that even large, multi-epitope active sites can be effectively generated. The experimental workflow is simple and 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[33,34]; 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 proteins. 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[9,11].

Methods

Golden gate assembly 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 ($N_{correct}$) divided by the number of times overhang *O* ligates to any overhang (N_{total}).

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

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

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

Computational Library design:

Codon compression

For each diversified amino acid position, we codon compressed the required diversity by generating all ambiguous codons that encode a portion of the required diversity using CodonGenie[38] and formulating the compression as the set cover problem. We have implemented Dancing Links (DLX)[27] to solve the exact cover problem, which results in every amino acid identity appearing exactly once.

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. To find segmentations that will assemble correctly, we employ color coding[28], coloring nodes with the same color if empirical ligation preferences[22–24] indicate the overhangs would ligate. Next, we run a colorful version of Dijkstra's shortest-path algorithm[39], which considers extending a 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 ensures high-fidelity assembly.

Experimental Library Assembly:

The full protocol is described in ref. [29].

The primers used for the reactions are described in Supplementary Tables 2 and 3.

Library Cloning

The assembled library after Golden Gate was cloned into pNACP plasmid[40] (kindly provided by Jan Steyaert, Vrije Universiteit Brussels) using homologous recombination in yeast[41]. 100µl yeast cells after recombination were plated on SDCAA plates. Colonies were collected with 1.5mL sterile DDW and extracted using Zymoprep Yeast Plasmid Miniprep II (Zymo Research, CAT #D2004).

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.

Deep Sequencing

Amplicon libraries were prepared as previously described[42]. 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[43], and pairwise aligned the results to constant and variable DNA segments using Parasail[44]. We further analyzed alignment results using Python[45–50] , 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.

Acknowledgements

We thank David Peleg for helpful discussions and comments on the algorithm and Olga Khersonsky and Ariel Tennenhouse for critical reading. Figures 1, 2 and 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.

References

1. Wójcik, M., Telzerow, A., Quax, W. J. & Boersma, Y. L. High-Throughput Screening in Protein Engineering: Recent Advances and Future Perspectives. *Int. J. Mol. Sci.* **16**, 24918–24945 (2015).
2. Markel, U. *et al.* Advances in ultrahigh-throughput screening for directed enzyme evolution. *Chem. Soc. Rev.* **49**, 233–262 (2020).
3. Alejaldre, L., Pelletier, J. N. & Quaglia, D. Methods for enzyme library creation: Which one will you choose? *Bioessays* **43**, 2100052 (2021).
4. Khersonsky, O. *et al.* Automated Design of Efficient and Functionally Diverse Enzyme Repertoires. *Mol. Cell* **72**, 178–186.e5 (2018).
5. Whitehead, T. A. *et al.* Optimization of affinity, specificity and function of designed influenza inhibitors using deep sequencing. *Nat. Biotechnol.* **30**, 543–548 (2012).
6. Whitehead, T. A., Baker, D. & Fleishman, S. J. Computational design of novel protein binders and experimental affinity maturation. *Methods Enzymol.* **523**, 1–19 (2013).
7. Fowler, D. M. *et al.* High-resolution mapping of protein sequence-function relationships. *Nat. Methods* **7**, 741–746 (2010).
8. Guntas, G., Purbeck, C. & Kuhlman, B. Engineering a protein--protein interface using a computationally designed library. *Proceedings of the National Academy of Sciences* **107**, 19296–19301 (2010).
9. Weinstein, J. Y. *et al.* Designed active-site library reveals thousands of functional GFP variants. *Nat. Commun.* **14**, 2890 (2023).
10. Treynor, T. P., Vizcarra, C. L., Nedelcu, D. & Mayo, S. L. Computationally designed

libraries of fluorescent proteins evaluated by preservation and diversity of function. *Proc. Natl. Acad. Sci. U. S. A.* **104**, 48–53 (2007).

11. Lipsh-Sokolik, R. *et al.* Combinatorial assembly and design of enzymes. *Science* **379**, 195–201 (2023).
12. Engler, C., Kandzia, R. & Marillonnet, S. A One Pot, One Step, Precision Cloning Method with High Throughput Capability. *PLoS One* **3**, e3647 (2008).
13. Engler, C., Gruetzner, R., Kandzia, R. & Marillonnet, S. Golden Gate Shuffling: A One-Pot DNA Shuffling Method Based on Type IIs Restriction Enzymes. *PLoS One* **4**, e5553 (2009).
14. Andreou, A. I. & Nakayama, N. Mobius Assembly: A versatile Golden-Gate framework towards universal DNA assembly. *PLoS One* **13**, e0189892 (2018).
15. Püllmann, P. *et al.* Golden Mutagenesis: An efficient multi-site-saturation mutagenesis approach by Golden Gate cloning with automated primer design. *Sci. Rep.* **9**, 10932 (2019).
16. Sarrion-Perdigones, A. *et al.* GoldenBraid: An Iterative Cloning System for Standardized Assembly of Reusable Genetic Modules. *PLoS One* **6**, e21622 (2011).
17. Sarrion-Perdigones, A. *et al.* GoldenBraid 2.0: A Comprehensive DNA Assembly Framework for Plant Synthetic Biology. *Plant Physiol.* **162**, 1618–1631 (2013).
18. Vazquez-Vilar, M. *et al.* GB3.0: a platform for plant bio-design that connects functional DNA elements with associated biological data. *Nucleic Acids Res.* **45**, 2196–2209 (2017).
19. Weber, E., Engler, C., Gruetzner, R., Werner, S. & Marillonnet, S. A Modular Cloning System for Standardized Assembly of Multigene Constructs. *PLoS One* **6**, e16765 (2011).

20. Mayer, H. Optimization of the EcoRI*-activity of EcoRI endonuclease. *FEBS Lett.* **90**, 341–344 (1978).

21. Lohman, G. J. S. *et al.* A high-throughput assay for the comprehensive profiling of DNA ligase fidelity. *Nucleic Acids Res.* **44**, e14 (2016).

22. Potapov, V. *et al.* Comprehensive Profiling of Four Base Overhang Ligation Fidelity by T4 DNA Ligase and Application to DNA Assembly. *ACS Synth. Biol.* **7**, 2665–2674 (2018).

23. Pryor, J. M. *et al.* Enabling one-pot Golden Gate assemblies of unprecedented complexity using data-optimized assembly design. *PLoS One* **15**, e0238592 (2020).

24. Potapov, V. *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.* **46**, e79 (2018).

25. Kirby, M. B., Medina-Cucurella, A. V., Baumer, Z. T. & Whitehead, T. A. Optimization of multi-site nicking mutagenesis for generation of large, user-defined combinatorial libraries. *Protein Eng. Des. Sel.* **34**, gzab017 (2021).

26. Daffern, N., Francino-Urdaniz, I. M., Baumer, Z. T. & Whitehead, T. A. Standardizing cassette-based deep mutagenesis by Golden Gate assembly. *Biotechnol. Bioeng.* (2023) doi:10.1002/bit.28564.

27. Knuth, D. E. Dancing links. *arXiv:cs/0011047* (2000).

28. Alon, N., Yuster, R. & Zwick, U. Color-coding. *J. ACM* **42**, 844–856 (1995).

29. Hoch, S. Y., Netzer, R., Hakeny, K. & Fleishman, S. J. GGAssembler Library Construction. doi:10.17504/protocols.io.81wgbxqkolpk/v3.

30. Choi, H. *et al.* Purification of multiplex oligonucleotide libraries by synthesis and

selection. *Nat. Biotechnol.* **40**, 47–53 (2022).

31. Tretyachenko, V., Voráček, V., Souček, R., Fujishima, K. & Hlouchová, K. CoLiDe: Combinatorial Library Design tool for probing protein sequence space. *Bioinformatics* (2020) doi:10.1093/bioinformatics/btaa804.
32. Wrenbeck, E. E. *et al.* Plasmid-based one-pot saturation mutagenesis. *Nat. Methods* **13**, 928–930 (2016).
33. Jacobs, T. M., Yumerefendi, H., Kuhlman, B. & Leaver-Fay, A. SwiftLib: rapid degenerate-codon-library optimization through dynamic programming. *Nucleic Acids Res.* **43**, e34 (2015).
34. Shimko, T. C., Fordyce, P. M. & Orenstein, Y. DeCoDe: degenerate codon design for complete protein-coding DNA libraries. *Bioinformatics* **36**, 3357–3364 (2020).
35. 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.* **22**, 165–171 (2020).
36. Sidore, A. M., Plesa, C., Samson, J. A., Lubock, N. B. & Kosuri, S. DropSynth 2.0: high-fidelity multiplexed gene synthesis in emulsions. *Nucleic Acids Res.* **48**, e95–e95 (2020).
37. Plesa, C., Sidore, A. M., Lubock, N. B., Zhang, D. & Kosuri, S. Multiplexed gene synthesis in emulsions for exploring protein functional landscapes. *Science* **359**, 343–347 (2018).
38. Swainston, N. *et al.* CodonGenie: optimised ambiguous codon design tools. *PeerJ Computer Science* **3**, e120 (2017).

39. Dijkstra, E. W. A note on two problems in connexion with graphs. *Numer. Math.* **1**, 269–271 (1959).

40. Uchański, T. *et al.* An improved yeast surface display platform for the screening of nanobody immune libraries. *Sci. Rep.* **9**, 382 (2019).

41. Gietz, R. D. & Schiestl, R. H. Large-scale high-efficiency yeast transformation using the LiAc/SS carrier DNA/PEG method. *Nat. Protoc.* **2**, 38–41 (2007).

42. Blecher-Gonen, R. *et al.* High-throughput chromatin immunoprecipitation for genome-wide mapping of in vivo protein-DNA interactions and epigenomic states. *Nat. Protoc.* **8**, 539–554 (2013).

43. Bushnell, B., Rood, J. & Singer, E. BBMerge - Accurate paired shotgun read merging via overlap. *PLoS One* **12**, e0185056 (2017).

44. Daily, J. Parasail: SIMD C library for global, semi-global, and local pairwise sequence alignments. *BMC Bioinformatics* **17**, 81 (2016).

45. McKinney, W. Data Structures for Statistical Computing in Python. in *Proceedings of the 9th Python in Science Conference* (SciPy, 2010). doi:10.25080/majora-92bf1922-00a.

46. The pandas development team. *pandas-dev/pandas: Pandas*. (2023). doi:10.5281/zenodo.7857418.

47. Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585**, 357–362 (2020).

48. Van Der Walt, S., Colbert, S. C. & Varoquaux, G. The NumPy array: A structure for efficient numerical computation. *Computing in Science and Engineering* **13**, 22–30 (2011).

49. Cock, P. J. A. *et al.* Biopython: Freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009).

50. Granger, B. E. & Pérez, F. Jupyter: Thinking and Storytelling With Code and Data. *Comput. Sci. Eng.* **23**, 7–14 (2021).