ACS SyntheticBiology

Research Article

pubs.acs.org/synthbio

Codon Compression Algorithms for Saturation Mutagenesis

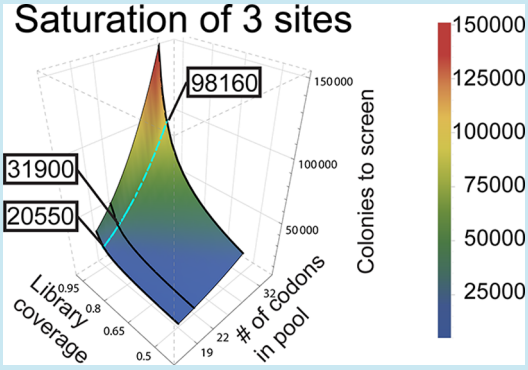
Gur Pines,[†] Assaf Pines, Andrew D. Garst,[†] Ramsey I. Zeitoun,[†] Sean A. Lynch,[†,‡] and Ryan T. Gill*[,†]

[†]Department of Chemical and Biological Engineering, University of Colorado Boulder, Boulder, Colorado 80309, United States

[‡]Biosciences Center, National Renewable Energy Laboratory, 15013 Denver West Parkway, Golden, Colorado 80401, United States

Supporting Information

ABSTRACT: Saturation mutagenesis is employed in protein engineering and genome-editing efforts to generate libraries that span amino acid design space. Traditionally, this is accomplished by using degenerate/compressed codons such as NNK (N = A/C/G/T, K = G/T), which covers all amino acids and one stop codon. These solutions suffer from two types of redundancy: (a) different codons for the same amino acid lead to bias, and (b) wild type amino acid is included within the library. These redundancies increase library size and downstream screening efforts. Here, we present a dynamic approach to compress codons for any desired list of amino acids, taking into account codon usage. This results in a unique codon collection for every amino acid to be mutated, with the desired redundancy level. Finally, we demonstrate that this approach can be used to design precise oligo libraries amendable to recombineering and CRISPR-based genome editing to obtain a diverse population with high efficiency.



KEYWORDS: *saturation mutagenesis, library size, codon redundancy, codon usage, genome editing, CRISPR selection*

Random mutagenesis techniques such as error-prone PCR provide a robust method for generating DNA libraries for use in protein, pathway, and genome engineering applications. However, such approaches generate a large screening load and often poorly sample the functional residues of interest. Semirational approaches such as saturation mutagenesis (SM)[1] can significantly reduce the screening load and improve the chances of sampling improved variants by focusing genetic diversity at codon position of interest. Indeed, SM has been successful in the study of protein−protein interactions,[2] protein engineering,[3] and for directed evolution[4] and are highly enriched for functionally interesting variants compared to randomly generated libraries.[5] Due to the redundancies in the genetic code, however, SM libraries constructed as NNK/S are inherently biased toward amino acids with greater codon degeneracy and include undesirable stop codons that generate noninformative protein variants. As a result, library size is increased, particularly when generating combinatorial libraries, thereby limiting the depth that the mutation space can be practically explored.

A number of strategies have been developed to further focus library diversity for protein engineering applications. For example, CAST (combinatorial active-site saturation test)[6] and SCOPE (structure-based combinatorial protein engineering)[7] leverage structural information for library size reduction. Another approach known as MAX randomization is designed to remove stop codons and eliminate redundancy by utilizing a different oligo for each codon.[8] Other strategies have aimed to reduce the codon space by reducing the number of amino acids that are coded. Reetz *et al.*[9] used the IUPAC notation[10,11] (Table 1) to define NDT as a single randomization (also termed "degenerate" or "ambiguity") codon that codes only for 12 amino acids (without redundancy), but with a good representation for all amino acid groups (Supporting Information Figure S2). The use of this codon enabled the identification of improved variants within a relatively small space.

Table 1. IUPAC Nucleic Acid Notation

	symbol	meaning
DNA bases	G	
	T	
	A	
	C	
degenerate (ambiguity) characters	R	G+A
	Y	T+C
	S	G+C
	W	T+A
	K	G+T
	M	A+C
	D	G+T+A
	H	T+A+C
	B	G+T+C
	V	G+A+C
	N	G+A+T+C

Received: July 24, 2014
Published: October 10, 2014

   dx.doi.org/10.1021/sb500282v | *ACS Synth. Biol.* 2015, 4, 604−614

Recent reports have demonstrated that a small set of degenerate codons can cover all 20 amino acids, remove all stop codons and reduce the codon redundancy significantly or even eliminate it completely. Tang *et al.*,[12] reported a codon design (NDT, VMA, ATG, and TGG, termed Small Intelligent) that covers all amino acids only once, without the stop codons and without including rarely used amino acids in *Escherichia coli*. Shortly after, another study reported a better compression ratio (i.e., oligos/codon site requirement) of only 3 codons (NDT, VHG, and TGG, termed "22c-trick"), essentially compressing the VMA and ATG codons into VHG.[13] This does, however, carry a cost in terms of redundancy that was increased from 20:20 (codon/amino acid ratio) to 22:20, with two amino acids being coded twice (Leu and Val). Both these methods were shown to be superior to the traditional NNN (64:20, including 3 stop codons) and NNK/S (32:20, including 1 stop codon) by means of library size and screening effort required for a given library coverage.

The aforementioned methods are static, produce solutions that include the wild type amino acid, and do not take into account organismal codon usage. As genetic tools continue to expand engineering capabilities to the chromosomes of phylogenetically distant organisms such solutions may therefore prove inadequate, as rare codon usage may adversely affect expression and folding of these engineered variants.[14−18] Here, we present two algorithms that accurately tailor a specific set of degenerate codons for a defined collection of amino acids. Both algorithms take into account the target organism's codon usage and designs the solution accordingly. DYNAMCC_0 (Dynamic management of codon compression, with no redundancy) reduces redundancy to zero, while keeping high usage-ranking codons within the collection, while the second algorithm, DYNAMCC_R (DYNAMCC Redundancy), includes redundancy within the boundaries of the defined amino acid list to allow the exploration of the redundant space. As several parameters are taken into account, namely, redundancy, usage, and compression efficiency, different weights can be given to every parameter, resulting in different codon collections.

To experimentally demonstrate the advantages of this approach, we used a combination of λ-red recombineering with CRISPR-based selection to target a specific amino acid in the *E. coli galK* gene for saturation mutagenesis. Our algorithm returned a collection of four degenerate codons that removed the wild type serine, the stop codons, and codon redundancy. This resulted in an efficient production of genomically varied population, harboring the 19 different designed codons that correspond to the 19 amino acids excluding wild type.

RESULTS AND DISCUSSION

Considerations for Creating a Compressed Codon Collection. There are several mathematical approaches to compute the oversampling necessary for obtaining an acceptable library coverage and completeness.[19,20] Directed evolution and SM studies commonly seek about 95% library coverage, which translates to an oversampling factor close to 3.[21] Figure 1 exemplifies the number of variants required to be screened with a given number of codons in the mutational pool, desired library coverage, and number of sites to be mutated. This illustrates that the more codons are used, more variants should be screened and that these numbers increase exponentially with every codon and site to be added. Ideally, 19 codons should be used, as this reduces the codon number as much as possible, without impairing the desired amino acid pool, allowing the wild type to be mutated to any other amino acid. An additional method computes the number of variants needed to be screened for getting 100% coverage in a certain degree of confidence. The numbers that derive from these calculations

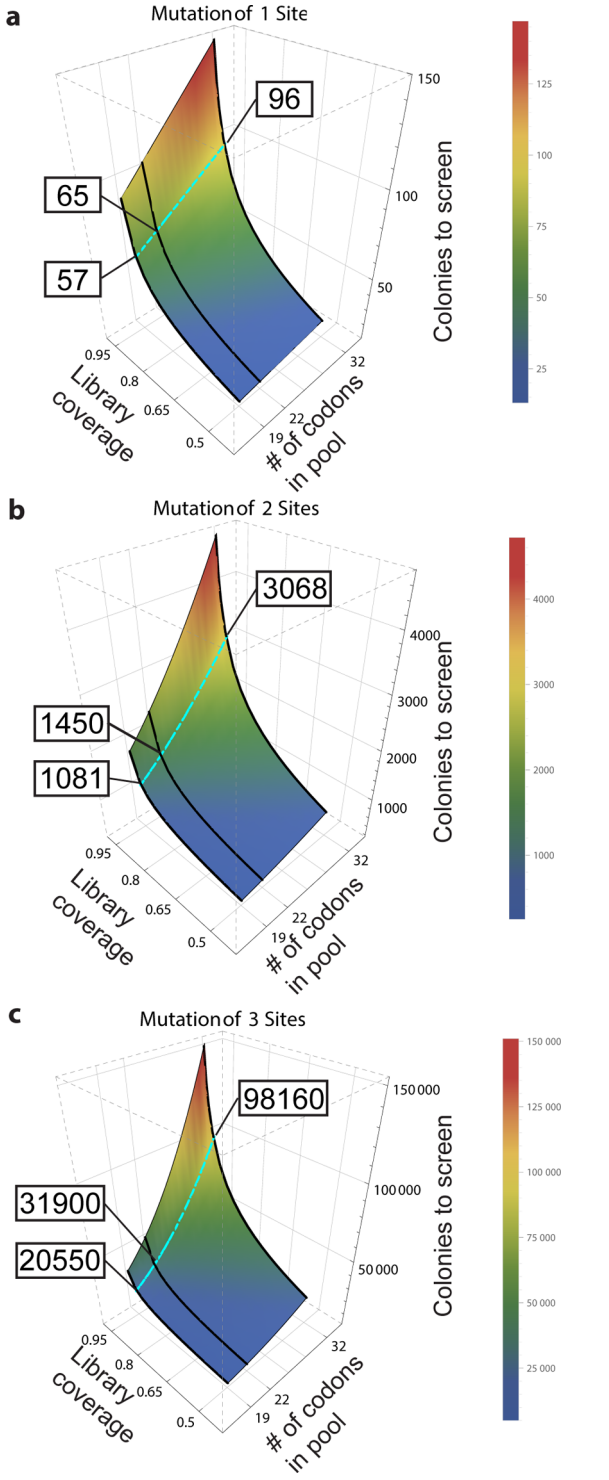


Figure 1. Screening effort calculations. The number of variants to be tested is determined by the number of mutated sites, the desired library coverage, and the number of codons that are in the mutational pool. Black lines represent, from top to bottom, 32, 22, and 19 codons, corresponding to the NNK, 22c-trick, and the method described here. Cyan line indicates 95% library coverage. Framed numbers indicate the intersects highlighting the number of variants needed to be screened in order to achieve 95% coverage in any given method.

dx.doi.org/10.1021/sb500282v | *ACS Synth. Biol.* 2015, 4, 604−614

are significantly higher and are presented in Supporting Information Figure S1. Recently, a mathematical analysis by Nov suggested that in some cases the need to identify the absolute best variant is not necessary, as the second or third best might be good enough in practical terms.[22] This methodology reduces screening effort and was demonstrated experimentally by Hoebenreich and colleagues.[23] This strategy complements methods for decreasing redundancy, which together can be used to further reduce unnecessary screening efforts.

As creating 19 oligonucleotides for every codon to be mutated is still expensive, we sought to develop a strategy for codon compression, similar to Small Intelligent and 22c-trick in its compression capabilities with the added advantage of working in a dynamic fashion. This dynamic approach enables the exclusion from the codon collection the wild type amino acid that varies between positions.

The use of multiple degenerate codons as implemented in the algorithms described in this study opens the door for many different designs, and enables the organism-specific codon usage tables as the ranking parameter. The algorithms return custom degenerate codon collection for a variety of user specifications and take into account three considerations: redundancy, compression, and codon usage. Naturally, these three requirements cannot be satisfied simultaneously, and different weights should be given to every parameter according to prior knowledge of the target protein's sequence and domain structure, as well as desired library size. Figure 2 illustrates the relationships between the three parameters: The triangle vertices represent a complete satisfaction of two parameters while neglecting the third. For example, the vertex "Best List", is the intersect of the "Redundancy" and the "Usage" edges, and is furthest away from the "Compression" edge, as no compression takes place; it therefore represents the collection of the desired amino acids with the best performing codons without any compression. Maximum possible compressions, such as the single codons NNN or NNK/S, include high usage ranking codons and some redundancy, are represented by the top vertex, and are the intersect of the maximum compression and the maximum usage lines. An additional example is represented by the solution marked with an "I" that puts the most weight on codon usage (shortest perpendicular line to the edge) and the least weight on reducing redundancy, (longest perpendicular line to its edge). The colored sections indicate areas in which each of the algorithms described here "travels" in order to find the ideal results, according to the user's inputs. These areas are variable in size and consistency and depend on parameters such as the identity and number of amino acids to be excluded from the library, and the codon usage of the specific organism. The two previously reported static solutions are also indicated in this figure: "Small Intelligent"[12] has no redundancy and resides on the "Redundancy" edge. "22c-trick",[13] on the other hand, is more compressed (3 codons instead of 4) and is closer to the "Compression" edge. However, "22c-trick" has some redundancy (22 codons for 20 amino acids) and is situated further away from the "Redundancy" edge than the "Small Intelligent" solution.

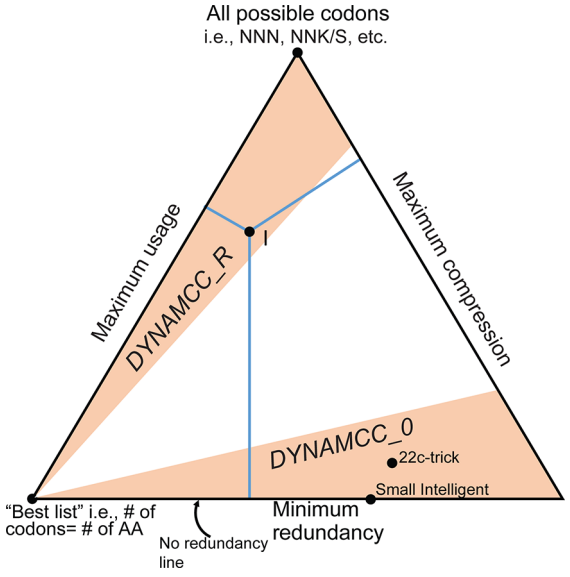


Figure 2. Schematic representation of the solution space searched by the algorithms described in this study. The triangle area represents the whole optional compressed codon space for a single point mutation that requires the removal of some codons (for example stop codons). Each edge represents one of the three considerations that are taken into account during codon compression, namely, redundancy, usage, and compression efficiency. For example, the solution marked with "I" puts the most weight on codon usage (it has the shortest distance to this edge) and the least weight on reducing redundancy. The colored sections indicate areas in which each algorithm "travels" in order to find the ideal results, according to the user's inputs. These areas are variable in size and depend on the identity and number of amino acids to be excluded from the library and the codon usage of the specific organism.

Using DYNAMCC_0 for Compressing Nonredundant Codons. To test our dynamic codon compression approach, we first removed only the stop codons, keeping all 20 amino acids with the requirement that every amino acid will be represented only once, using the *E. coli* usage table. These settings allowed the comparison of our solutions to previously published ones. This run resulted in the return of 4 codons, similar to the "Small Intelligent" solution, but with codons that are somewhat more compatible for expression in *E. coli* (Figure 3a, leftmost column for every indicated method). To further test the importance of the organism-specific codon usage, we repeated the same run, with different usage tables corresponding to various model organisms, restricting the usage rank to no less than 3 (i.e., if there are more than 3 codons for a given amino acid, use a codon that is in the top 3 most common, see Methods). Indeed, while the compatibility of previous static solutions is more suitable for some organisms than others, our dynamic approach described here resulted in high codon compatibility for every organism (Figure 3a). The expanded results, including the compressed codon solutions are summarized in Figure 3b, illustrating that for most organisms tested, a different codon mix is required for achieving best usage with full amino acid coverage. Not surprisingly, due to usage similarity, both runs for mouse and human resulted in the same codon collection. An expanded comparison between the methods is available in Supporting Information Figure S2.

A central feature of this algorithm is the ability to omit any number of amino acids and find the best corresponding degenerate codons in terms of usage, redundancy, and the lowest number of codons possible. The significance is that the repertoire of amino acids that replaces the wild type can be defined precisely, without any compromise of including nonrelevant amino acids or biasing toward a specific amino acid and using high ranking codons. For example, a common

dx.doi.org/10.1021/sb500282v | ACS Synth. Biol. 2015, 4, 604−614

a

	Small inteligent						22c trick						This study					
AA	E. coli	Yeast	C. elegans	D. melanogaster	Mouse	Human	E. coli	Yeast	C. elegans	D. melanogaster	Mouse	Human	E. coli	Yeast	C. elegans	D. melanogaster	Mouse	Human
A	3	2	2	4	3	3	1	4	4	3	4	4	1	1	1	3	3	3
C	2	1	1	2	2	2	2	1	1	2	2	2	2	1	1	1	1	1
D	1	1	1	1	2	2	1	1	1	1	2	2	1	1	1	2	1	1
E	1	1	1	2	2	2	2	2	2	1	1	1	2	1	1	1	2	2
F	1	1	1	2	2	2	1	1	1	2	2	2	1	1	1	1	1	1
G	2	1	2	3	4	4	2	1	2	3	4	4	2	1	1	1	3	3
H	1	1	1	2	2	2	1	1	1	2	2	2	1	1	1	1	1	1
I	1	1	1	2	2	2	1	1	1	2	2	2	1	1	1	1	1	1
K	1	1	1	2	2	2	2	2	2	1	1	1	2	1	2	1	2	2
L	4	4	1	4	4	4	4	4	1	4	4	4	1	1	1	1	3	3
M	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
N	2	1	1	2	2	2	2	1	1	2	2	2	2	1	1	1	1	1
P	2	1	1	3	3	3	1	4	2	2	4	4	1	2	3	2	3	3
Q	2	1	1	2	2	2	1	2	2	1	1	1	1	1	1	1	2	2
R	1	3	3	2	6	6	1	3	3	2	6	6	1	2	2	1	3	3
S	5	4	4	4	4	5	5	4	4	4	4	5	3	1	2	1	1	1
T	2	2	1	3	2	2	3	4	4	2	4	4	2	1	2	2	2	2
V	2	1	1	3	3	3	2	1	1	3	3	3	1	1	1	1	1	1
W	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Y	1	1	1	2	2	2	1	1	1	2	2	2	1	1	1	1	1	1
L							1	5	4	1	1	1						
V							1	4	2	1	1	1						
Total codons:	4						3						4	5	4	4	4	4

b

E. coli				Yeast				C. elegans				D. melanogaster				Mouse/Human			
Degenerate codon	Codons covered	Amino acids covered	usage rank	Degenerate codon	Codons covered	Amino acids covered	usage rank	Degenerate codon	Codons covered	Amino acids covered	usage rank	Degenerate codon	Codons covered	Amino acids covered	codon usage rank	Degenerate codon	Codons covered	Amino acids covered	usage rank
VHG	AAG	K	2	NMT	AAT	N	1	TGK	TGG	W	1	VHG	AAG	K	1	VMA	AAA	K	2
	ACG	T	2		ACT	T	1		TGT	C	1		ACG	T	2		ACA	T	2
	ATG	M	1		CAT	H	1	NHT	AAT	N	1		ATG	M	1		CAA	Q	2
	CAG	Q	1		CCT	P	2		ACT	T	2		CAG	Q	1		CCA	P	3
	CCG	P	1		GAT	D	1		ATT	I	1		CCG	P	2		GAA	E	2
	CTG	L	1		GCT	A	1		CAT	H	1		CTG	L	1		GCA	A	3
	GAG	E	2		TAT	Y	1		CCT	P	3		GAG	E	1	WKC	AGC	S	1
	GCG	A	1		TCT	S	1		CTT	L	1		GCG	A	3		ATC	I	1
	GTG	V	1	WKG	AGG	R	2		GAT	D	1		GTG	V	1		TGC	C	1
NRT	AAT	N	2		ATG	M	1		GCT	A	1	NRC	AAC	N	1		TTC	F	1
	AGT	S	3		TGG	W	1		GTT	V	1		AGC	S	1	DKG	AGG	R	3
	CAT	H	1		TTG	L	1		TAT	Y	1		CAC	H	1		ATG	M	1
	CGT	R	1	VAA	AAA	K	1		TCT	S	2		CGC	R	1		GGG	G	3
	GAT	D	1		CAA	Q	1		TTT	F	1		GAC	D	2		GTG	V	1
	GGT	G	2		GAA	E	1	SRA	CAA	Q	1		GGC	G	1		TGG	W	1
	TAT	Y	1	DTT	ATT	I	1		CGA	R	2		TAC	Y	1		TTG	L	3
	TGT	C	2		GTT	V	1		GAA	E	1		TGC	C	1	NAC	AAC	N	1
WTT	ATT	I	1		TTT	F	1		GGA	G	1	WTC	ATC	I	1		CAC	H	1
	TTT	F	1	KGT	GGT	G	1	AWG	AAG	K	2		TTC	F	1		GAC	D	1
TGG	TGG	W	1		TGT	C	1		ATG	M	1	TGG	TGG	W	1		TAC	Y	1

Figure 3. Comparison of codon compression methods. Usage ranking is color-coded with green having the highest and red the lowest ranking, respectively. (a) The method described here compared to Small Intelligent and 22c-trick by means of codon usage for several model organisms. (b) Detailed information on the compressed codons required for obtaining codons with the highest usage rank. Note that due to the dynamic nature of the method, every organism requires a different solution for optimal usage ranking.

solution for coding for hydrophilic amino acids is to use the degenerate VRK codon (covers N, S, Q, H, E, and D once, G twice, and R three times).[24] This solution, however, does not cover lysine, codes for glycine (an uncharged amino acid), and is biased toward arginine. Using the algorithms described here, a solution of two codons is found that covers precisely this group, without redundancy and with adequate usage (the example shown here is adjusted to *E. coli* usage): CGT (R) and VAW (K, N, Q, H, E, and D). Moreover, in the case where a hydrophilic amino acid is to be replaced with other hydrophilic ones, specific codons can be found to precisely code for the desired amino acid set, removing the wild type and allowing only the rest of the hydrophilic amino acids to be expressed: Figure 4a illustrates the solutions for removing different amino acids from this collection. The results span from 3 compressed codons (for the removal of D, E, H, K, or N) to a single codon only, in the case of the removal of arginine (R). This methodology of restricted randomization may be useful in cases of semi rational approaches, in which structural and other analyses suggests that certain side chain functionality may result in improved phenotypes. An additional advantage of this approach is the significant reduction of library size. This allows for either less sampling to find a desired mutation or for the ability to mutate more amino acids. For example, using the aforementioned hydrophilic group of seven codons as the library pool, one can mutate three amino acids with a screening effort that is on the same order of magnitude as mutating only two residues using the "22c-trick" method (Figure 4b).

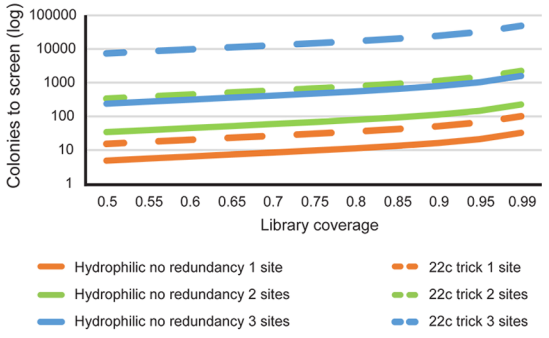
Using DYNAMCC_R for Exploring the Redundant Space. Since synonymous codons can sometimes have

dx.doi.org/10.1021/sb500282v | *ACS Synth. Biol.* 2015, 4, 604−614

a

Amino acid coded by	All	D	E	H	K	N	Q	R
	Amino acid to be removed							
D	VAW		VAT	RAW	SAW	SAT	RAW	VAW
E	VAW	VAA		RAW	SAW	VAA	RAW	VAW
H	VAW	CRT	VAT		SAW	SAT	CRC	VAW
K	VAW	VAA	MAA	RAW		VAA	RAW	VAW
N	VAW	AAC	VAT	RAW	AAC		RAW	VAW
Q	VAW	VAA	MAA	CAG	SAW	VAA		VAW
R	CGT	CRT	CGT	CGT	CGT	CGT	CRC	
# of deg. codons	2	3	3	3	3	3	2	1
# of codons covered	7	6	6	6	6	6	6	6

b



c

Amino acid coded by	All	D	E	H	K	N	Q	R
	Amino acid to be removed							
D	VAN		VAY	RAY	VAY	SAY	VAY	VAN
E	VAN	VAR		VAR	SAR	VAR	RAR	VAN
H	VAN	CAC	VAY		VAY	SAY	VAY	VAN
K	VAN	VAR	MRR	VAR		VAR	RAR	VAN
N	VAN	MAY	VAY	RAY	VAY		VAY	VAN
Q	VAN	VAR	MRR	VAR	SAR	VAR		VAN
R	MGR CGY	MGR CGY	MRR CGY	MGR CGY	MGR CGY	MGR CGY	MGR CGY	
# of deg. codons	3	4	3	4	4	4	4	1
# of codons covered	18	16	16	16	16	16	16	12

Figure 4. Example for defining a specific group of amino acids for mutagenesis and the resulting compressed codons. (a) Results using DYNAMCC_0 for no redundancy, using the *E. coli* usage table within a selected group of amino acids. Here, a defined a group of hydrophilic amino acids is shown. The left column represents the degenerate codons needed in order to cover the whole group, (namely, VAW and CGT). The rest of the columns show the degenerate codon solutions when removing a single amino acid from this collection (amino acid is indicated in the top of the column). Colors are used to group identical codons within every column. Note that the solutions range from 3 to a single codon, depending on the amino acid to be removed. (b) The effect of restricted randomization (the 7 codons from (a)) on the sampling effort needed for various degrees of library coverage, compared to the previously reported "22c-trick" solution is shown. Note that, in this case, using a restricted randomization allows for 3 amino acids mutations to be sampled with a less sampling effort as is required for mutating 2 sites only using the "22c-trick" method. (c) Similar analysis as in a, this time using DYNAMCC_R for maximum redundancy for the same group of amino acids.

biological effects (see below), it is desired to be able to perform controlled exploration of codon redundancy space. DYNAMCC_R finds a compressed codon collection for a defined set of amino acids with a variety of redundancies including the extreme complete saturation of all available codons per amino acid. The same case study as was described above for a defined set of hydrophilic amino acids (Figure 4c) produce a compressed codon collection representing the whole redundant codon space for this group that is universal to all organisms, regardless of codon usage. Here, restricted randomization might also be beneficial as the addition of the redundant codons occurs to a small set of amino acids; thus, it does not dramatically increase the screening effort. For example, the full redundant collection of the hydrophilic amino acids from Figure 4c sums to 18 codons, less than the 19 codons used in nonrestricted, nonredundant saturation mutagenesis.

Table 2 represent more general cases in which the genetic code is completely covered apart from the three stop codons and a selected single amino acid. As this represents an end point case, which covers all codon redundancies, this table is general for all organisms, regardless of their usage bias. This case is represented by the "all possible codons" vertex in Figure 2, satisfying both usage, as the highest ranking codons are present in the collection, and efficient compression. Removing only the three stop codons results in four degenerate codons that correspond to the whole amino acids repertoire. The removal of an additional single amino acid results in a collection ranging from 7 to 3 degenerate codons, depending on the identity of the amino acid to be removed.

Experimental Demonstration for Eliminating Redundancy. To demonstrate the algorithms described here, we targeted the endogenous *E. coli galK* gene for editing with various codon compressed oligomer libraries. We used λ-red recombineering in combination with CRISPR-based selection to saturate position 376.[25,26] In short, an editing cassette is PCR-amplified from a wild type genome using oligo pools designed with DYNAMCC_0. This cassette harbors two mutations: the first is within the open reading frame of *galK* and targets the desired amino acid, while the second is downstream from the open reading frame and mutates a protospacer-adjacent motif (PAM) sequence, a necessary element for Cas9 targeting (Figure 5a and b). The cassette is then recombineered to cells containing λ-red and an inducible Cas9 plasmids. Together with the editing cassette, cells are transformed with a specific guide RNA (gRNA) that targets the spacer sequence next to the PAM domain. This results in the recombineering of the editing cassette into the genome of cells by the λ-red system, and the death of nonrecombineered cells by the CRISPR machinery, thus enriching the population of engineered cells. Using DYNAMCC_0, we designed four editing cassettes that collectively code for highly ranking 19 codons and do not code for stop codons or the wild type serine (WWC, TGS, SRT, and VHG. Expansion of each codon to its corresponding nondegenerate codons is available in Supporting Information Figure S3. A comprehensive table including the codon collections for the removal of every amino acid, in addition to stop codons, optimized for *E. coli*, is available in Supporting Information Table S1). Additionally, we constructed an NNK cassette for comparison.

The *galK* gene was amplified from the recombineered cell population and sequenced. The Sanger sequencing confirmed successful mutagenesis of the open reading frame target site both in the case of using every editing cassette individually and when recombineered in multiplex with the mixture of all four editing cassettes. Quick Quality Control quantifications[23] showed acceptable agreement between the expected and experimental nucleotide distribution, although the NNK quantification shows some bias toward the wild type T and C in the first and second positions of the target codon, respectively (Supporting Information Figure S4). To follow the individual frequency of each codon, cell populations that were recombineered with either the mixture of all four editing

dx.doi.org/10.1021/sb500282v | ACS Synth. Biol. 2015, 4, 604−614

Table 2. Compressed Genetic Code after the Removal of Single Amino Acids, Using DYNAMCC_R

amino acid removed in addition to stop codons	minimal set of compressed codons							total number of codons
only stop	NYA	VAG	VRA	NBG				61
A	NDY	VAG	VRA	NTA	HCN	NKG		57
C	NYA	VAG	VRA	NBG	NHY	VGY		59
D	NYA	VAG	VRA	NBB	HAY			59
E	NYA	NBG	NNY	MAR	VGA			59
F	NYA	VAG	VRA	NBG	UTY	NUY		59
G	NYR	VAR	NHY	HGB	MGA			57
H	NYA	VAG	VRA	NBB	DAY			59
I	NCA	VAG	VRA	NBG	BTH	NUY		58
K	NYA	SAR	NBG	NNY	VGA			59
L	NCA	NSG	VAG	VRA	DTY	RTR	NUY	55
M	NSG	NYA	VAG	VRA	NNY	BTG		60
N	NYA	VAG	VRA	NBB	BAY			59
P	NDY	VAG	VRA	DCN	NTA	NKG		57
Q	NYA	NBG	NNY	RAR	VGA			59
R	NYR	VAR	DGY	NHY	KGG	GGA		55
S	VMG	UCY	NTA	BGY	NWY	VVA	NKG	55
T	BCN	NDY	VAG	VRA	NTA	NKG		57
V	NCA	NSG	HTN	VAG	VRA	NUY		57
W	NYR	VRR	NNY					60
Y	NYA	VAB	VRA	NBB				59

cassettes or the NNK cassette were further analyzed by high-throughput sequencing. The sequence reads of the 19 codon library summed to more than 220 000 and were divided into four groups, according to the presence of a mutation in either one of the targeted locations (i.e., S376 or the PAM domain), in both or in none (Supporting Information Figure S5). Remarkably, the counts of the double-mutated sequences summed to more than 185 000 of the counts. The fact that the wild type codon is not present in the pooled editing cassettes enabled the calculation of the saturation mutagenesis efficiency. While the PAM domain was mutated in more than 99.7% of the cases, S376 was mutated in 81% of cells, significantly higher than traditional λ-red methods that range between 0.1 and 10% efficiency.[27,28] Similar numbers were achieved with the NNK library (Supporting Information Figure S5). Next, we examined the codon frequency in the double mutant groups. The codon distribution for both methods is illustrated in Figures 5c and d. As expected, the mix of the four editing cassettes resulted in the prevalence of the designed codons, which summed to more than 99% of the total codons, with the rest accounting for random mutations or sequencing misreads. Similarly, the NNK library cells resulted in the coded 32 codons that represented more than 99% of the total codon counts (codon frequency is available in Supporting Information Table S2). While the 19 codons designed by our algorithm represent the exact desired codon collection, NNK codes both for serine and a stop codon, which account for 10.5% and 3.2% of the total codons, respectively (Figure 5e and f).

Conclusion. High throughput DNA writing and reading (i.e., oligonucleotide synthesis and sequencing) technologies are widespread and in routine use in many laboratories. As a result, the bottleneck has shifted to the screening methods for variants of interest. An ideal screening experiment requires good compatibility between library size and the screening method. Unless using a selection, a method that is not applicable for most screens, the screening method might significantly limit efficient searching through large libraries. Hence, methods to reduce library size without affecting its diversity are desired. A major parameter that affects library size is redundancy. We consider here two redundancy types: the first is codon redundancy, stemming from the inherent degeneracy of the genetic code, and the second is amino acid redundancy and is related to the presence of the wild type amino acid in the mutational pool. The wild type amino acid increases library size needlessly to an extent determined by its identity (with the amino acids coded by six codons contributing the most) and ideally should be removed from the oligonucleotide pool. Combined with the removal of codon redundancy, the oligonucleotide pool is further reduced here to include only 19 codons, compared to 64 when using NNN, 32 in NNK/S, or 22 and 20 in "22c-trick" and "Small Intelligent", respectively. The difference between 20 and 19 is significant, as a bias toward the wild type amino acid might exist within the constructed library,[13] thus increasing the screening effort. This bias toward wild type may be explained by the fact that an oligonucleotide harboring the wild type sequence is 100% homologous to the template and therefore integrates much more efficiently than other sequences, both in PCR and in recombination based approaches such as those demonstrated here. Removing the wild type amino acid from the mutational pool also assists with the evaluation of the mutational technique and mutation rates, as the wild type presence is directly derived from incomplete mutagenesis, rather than the additional possibility of its successful incorporation during the mutagenesis process. When saturating more than a single site, however, the wild type amino acid should sometimes remain in the pool to allow cooperative interactions. As DYNAMCC_0 asks for the amino acids to remove, removing only the stop codons results in an optimal codon collection including all 20 amino acids (as demonstrated in Figure 3).

With recent developments in genome engineering and editing tools such as zinc fingers, TALENS and CRISPR,[29,30] saturation mutagenesis is expanding outside of the bacterial plasmid context and is now available to be performed directly at the genome level of multiple organisms. As such, codon usage is an essential parameter that should be taken into account. As a

dx.doi.org/10.1021/sb500282v | ACS Synth. Biol. 2015, 4, 604−614

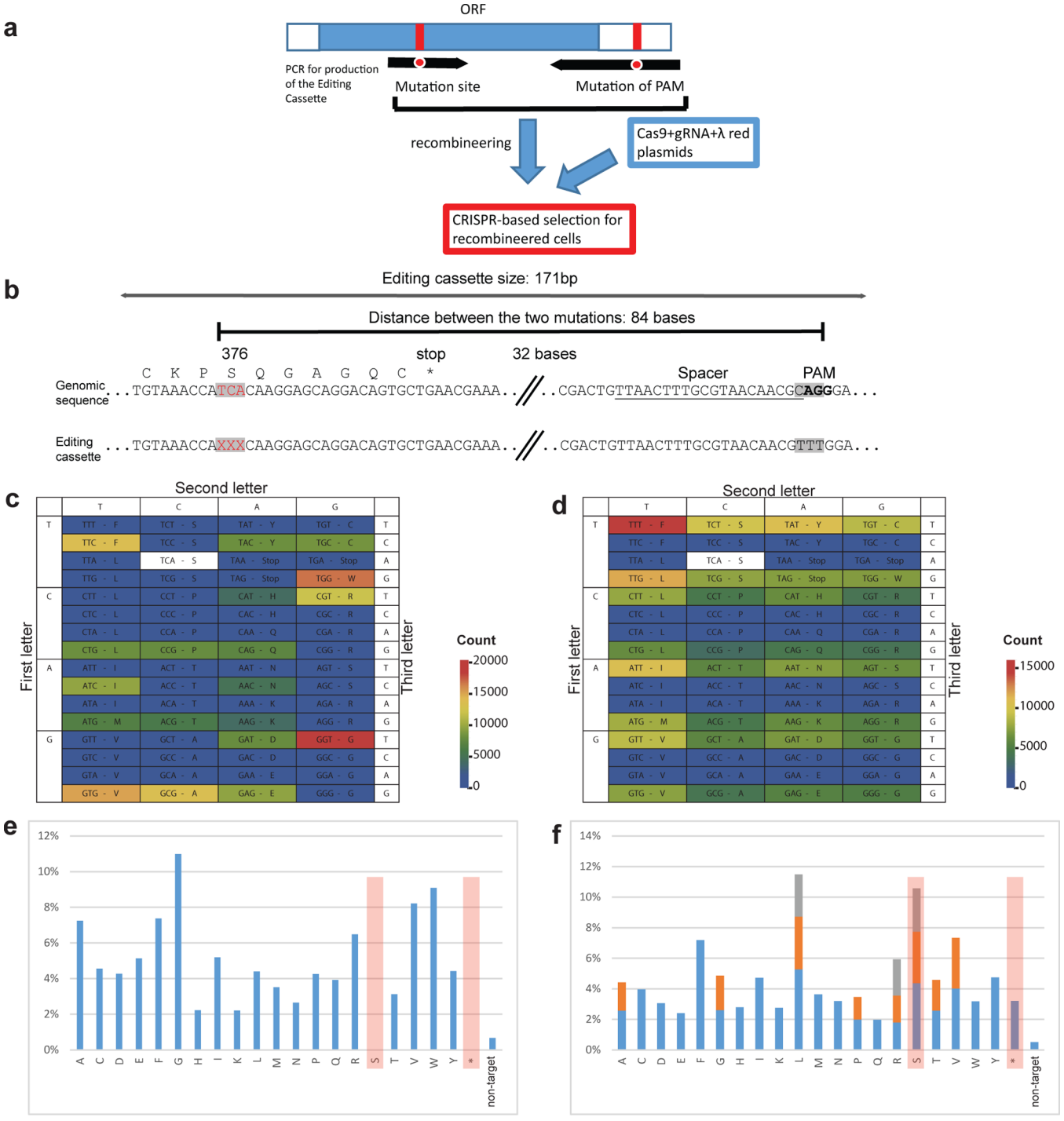


Figure 5. Saturation of the *E. coli* genomic *galK* S376 position. (a) General scheme of the genome editing method using a combination of recombineering and CRISPR selection. An editing cassette is amplified from a reference genome using primers incorporating two mutations: the first mutates the open reading frame, while the second mutates a downstream PAM domain (Methods). (b) Design of the editing cassette. Upper nucleotide sequence is the genomic 3′ end of the *galK* gene. Lower sequence is the editing cassette. Forward and reverse primers (Methods) harboring the mutations of S376 (red) and the PAM and spacer sequences (gray boxes) were used for cassette amplification from a wild type *E. coli* genome. XXX is either the collection of the WWC, TGS, SRT, and VHG compressed codons or the NNK codon. Editing cassette size and the distance between the two mutations are indicated. (c, d) The genetic code table highlighting the number of counts for each codon obtained by high throughput sequencing for the method described here and NNK, respectively. (e, f) Relative amino acid distribution in the method described here and NNK, respectively. Amino acids are indicated by a one letter code and sorted in an alphabetical order. Stop codon is indicated by an asterisk. Stacked bars indicate redundant codons for the same amino acid and the "non-target" bar indicates the proportion of nondesigned codons, which are less than 1% in both cases. Orange boxes highlight the wild type serine and stop codon that occurs only in the NNK experiment.

general rule, presence of high-usage codons correlates with tRNA abundance and predicts high protein translation rate and expression.[14−18] DYNAMCC_0 takes this into account and reduces codon redundancy while keeping high usage ranking codons, according to organism-specific usage tables. However, the redundant space may be of interest in some cases, as it was

dx.doi.org/10.1021/sb500282v | ACS Synth. Biol. 2015, 4, 604−614

shown that synonymous codons sometimes have effects on mRNA stability,[31] or protein expression,[32] folding,[33] and function.[34] The location of relatively rare codons within the open reading frame is thought to be nonrandom and affect the secondary and tertiary protein structure by dictating a varying translational speed.[35] The effect of synonymous codon mutations are difficult to predict, and approaches such as NNK saturation that employ static redundancy cannot fully explore such relationships. Similarly, the DYNAMCC_0 solutions reduce the codon coverage to the smallest possible nonredundant set, thus ignoring the potential effects of synonymous mutations. The DYNAMCC_R solution, however, enables the exploration of each codon in a case by case basis with a predefined level of redundancy (Figure 2). Thus, the combination of the DYNAMCC_0 and DYNAMCC_R offers flexibility depending on the experimental hypothesis of interest.

We provide here a toolbox that complements existing strategies for saturation mutagenesis and library construction. Every mutagenesis experiment should be designed according to various parameters such as the number of sites to be mutated, whether or not the sites should be mutated in a completely saturated manner, the importance of codon redundancy and the available screening tools that are directly linked to the desired library size. As such, there are multiple solutions for each case. While designing the algorithms described here, we aimed at reducing library size as much as possible (or desired) while keeping a relatively small amount of required oligonucleotides. As mentioned in previous multicodon solutions,[13] the mutation of more than a single amino acid per primer requires the synthesis of all possible combinations. For example, one can consider a mutation of two adjacent amino acids, one of which has a three codon solution while the second position has a four codon solution. In this scenario, in most PCR-based techniques, 12 different forward and reverse oligonucleotides should be synthesized, with the total of 24. This is more expensive than four NNK primers, but with current oligonucleotide synthesis prices, it is still affordable for most laboratories and will be even more so if taking into account the rate of synthesis price reduction with time.[36] Moreover, this investment is negligible, relative to the advantages of library size reduction and downstream screening effort reduction. Another option is directly synthesizing all different codons via resin splitting. This method results in a single randomized oligonucleotide that includes all selected codons and was successfully demonstrated for the "Small Intelligent" codon collection.[37] However, this requires access to an oligonucleotide synthesis apparatus, which is not widely available. Another approach to this problem was recently demonstrated by Tang and colleagues.[38] Limiting the amino acids to which the target sites can be mutated dramatically reduces the number of degenerate codons required for each site and as a result reduces the number of codon combinations required for synthesis. This methodology, however, is feasible only when the target protein has been characterized and analyzed in a way that provides data on the exact and reduced set of amino acids that should replace the wild type residue. This approach represents an extreme case of the restricted saturation shown in Figure 4a.

Our approach for dynamic codon compression is demonstrated by the genomic mutation of the *galK* gene in the S376 position. We employed a combination of λ-red recombineering with CRISPR-based selection to produce a library of engineered cells that does not include the wild type amino acid or stop codons, does not contain redundant codons, and includes highly used codons in *E. coli*. In this particular case, the alternative NNK approach coded for the unproductive wild type serine and a stop codon in more than 13.5% of the counts, increasing needlessly the screening effort.

The apparent bias in codon composition for both the DYNAMCC and NNK libraries of about 5-fold (Figure 5c and d) is similar to that observed in other recent literature ranging between 4- and 9-fold.[12,13,26] While there are many factors that influence library quality,[13] the most probable factor in our study is the use of equimolar amounts of each nucleotide during the synthesis of degenerate sites. While this approximation is suitable in many applications, the higher coupling efficiencies of T and G to the growing DNA strand during synthesis can lead to the biases observed in this work, which are composed of various combinations of G and T (GGT, TGG, and TTT, TTG in DYNAMCC and NNK, respectively). A correction bias can be applied to correct this (1.0T:1.15G:1.25C:1.5A) and should be taken into consideration in future applications of these techniques.[39,40]

The high-throughput sequencing results also revealed that, although very low in numbers, nontarget codons also appear in both libraries (Figure 5c and d). Since these nontarget codons are very rare (all nontarget codons sum to about 0.5% of the counts, Figure 5e and f), the chances of identifying them are very low when sampling several tens of variants, which is currently the standard practice of assessing such libraries.[12,13,26] The source for the presence of these nontarget codons is unclear but could result from errors in synthesis, errors introduced during the PCR reactions that were performed in preparation for high-throughput sequencing (see Methods), and/or from sequencing errors.

An additional advantage of removing the wild type codon is the ability to calculate the efficiency of our genome editing approach to be around 80% (Supporting Information Figure S5), a significantly higher number than traditional recombineering-only based protocols.

The two algorithms described here allow the intelligent and flexible sampling of the mutational space by precisely defining a subgroup of target amino acids, based on prior knowledge or model assumptions, with a minimal set of corresponding oligonucleotides (Figure 2). Importantly, with some adjustments the external usage tables we use here may be modified to contain any set of information varying from codon harmonization data[35] to the metabolic load of individual amino acids. Moreover, the same compression logic may be applied for engineering other genetic elements such as ribosomal binding sites, promoters, repressors etc.[41] This will allow to control library size and synthesize less oligonucleotides without an impact on library variance, similarly to what we describe here.

The source code is available for download from our Web site, https://sites.google.com/site/thegillgroupcu/software, allowing the community to fully understand the code logic, to further improve it, and to implement it in other tools. Moreover, the usage tables are external text files, allowing easy addition of custom tables for organisms that are not included in our release.

METHODS

We aimed at writing an algorithm that enables codon compression under user-defined constraints, such as the removal of specific amino acids and stop codons, redundancy levels, and organism-specific codon usage. The code was

written in Perl, so runs are made with no code compilations. This open-source approach allows future code improvements by others and its implementation in other pipelines. The algorithm's logic is depicted in Figure 6: The compression

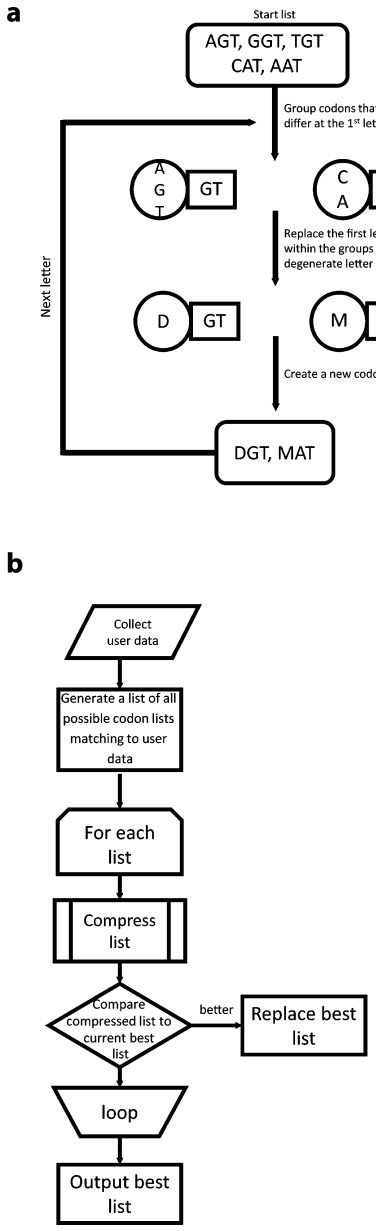


Figure 6. Compression and ranking algorithms. (a) The compression algorithm. Compressing a set of codons without adding new ones. Step 1: group codons that only differ in one letter. Step 2: replace this group with one codon using a degenerate letter. Repeat steps 1 and 2 until the set stops changing. (b) The ranking algorithm. After codon list compression, the list is evaluated against the current best list, using the compressed size and the codon list usage.

module (Figure 6a) is shared by both algorithms, while the ranking module (Figure 6b) is implemented only in DYNAMCC_0. The software uses external text files for the codon usage tables, making the addition of any custom organism-specific table manageable for users without a computational background. The codon usage tables that are in use in this study were taken from the GenScript Web site (http://www.genscript.com/cgi-bin/tools/codon_freq_table).

Algorithm Compression Logic. The compression module generates a list of all potential codons based on the user input data such as amino acids to remove and usage threshold. The second phase is the list compression, as depicted in Figure 6a: all codons that differ in the first position only are grouped. Then, the diversity of the first position is compressed to a degenerate codon according to the IUPAC notation table (Table 1). This process is repeated two more times and results in a minimal compressed set of codons that accurately represents the original codon collection.

DYNAMCC _R. This algorithm is designed for the exploration of the redundant space (Figure 1). Following a dialogue that defines the usage table and the amino acids to be removed, a codon table is generated. The algorithm initiates with the "best list", which comprises the highest ranking codons of the user-defined amino acid collection. Then, codons are added iteratively from the next best usage score until the run ends with the complete set of compressed codons that covers all codons coding for the defined amino acid list. This provides high-level flexibility in fine-tuning between the two opposing parameters, namely, redundancy and library size. The compression dynamics with the increasing redundancy are illustrated in Supporting Information Figure S6.

DYNAMCC _0. This algorithm is designed for reducing codon redundancy. Similar to DYNAMCC_R, the run begins with a dialogue for defining the usage table and amino acids to be removed. Additionally, the users also input the usage ranking they are willing to "pay" in order to get efficient compression. There are two alternative options for usage ranking: by ranking or by usage: Ranking spans from 1 to 6 (depending on the amino acid) with 1 representing the codon with the highest usage score. This allows an unbiased usage cutoff. The usage method's purpose is to define the usage ratio minimal threshold. The usage input maximum value is derived from the amino acid composition and is defined in the dialogue. During the run, codon lists are generated and constantly evaluated against previous lists (based on compression and usage, Figure 6b). Upon the run completion, the algorithm returns the best list that was found from all the potential lists, together with their rank and usage ratios, and displays the resultant compressed codons. This algorithm uses a "brute force" approach, as it checks all possible codon permutations. With an average CPU, the algorithm cycles through 1000 codon list permutations in about 300 ms. In order to utilize computers with more than a single core, the program can slice the codon lists into a number of smaller ones. Each sublist thread can be run on a different thread to find its best list, with the best of the best list being found at the end of the run by comparing the results from the different threads. This results in a standard run (removing stop codons and an additional amino acid, rank limit of 3) taking about 6−7 min using a common quad core i7 laptop.

To enable the user to have a high level of control, we added an additional feature not demonstrated here, allowing the definition of an acceptable level of redundancy (i.e., more than one codon per amino acid); however, this specific feature is computationally intensive and may result in very long runs. This "brute force" was chosen because, despite it taking a longer time to run, this algorithm is typically being run once for every amino acid mutation design, and it gives the absolute best result. In contrast, methods such as genetic algorithms[42] and annealing[43] methods, run significantly faster, but do not

dx.doi.org/10.1021/sb500282v | ACS Synth. Biol. 2015, 4, 604−614

guarantee the best result will be found, as they do not search through the whole solution space.

Library Coverage Calculations. Calculations are based on the following formula:[20,21]

$$T = -v \cdot \ln(1 - p)$$

where T is the library size, v is the number of variants, and p is the probability for each variant to occur. Graphics were made using Wolfram Mathematica.

Construction of Editing Cassettes. Editing cassettes were PCR amplified using boiled *E. coli* MG-1655 strain as template using the following primers.

Forward primers: AAAAACAGGTATTAAAGAGACTTTTTACGTTTGTAAACCAXXXCAAGGAGCAGGACAGTG, where XXX stands for WWC (codes for AAC, ATC, TAC, TTC), TGS (codes for TGC, TGG), SRT (codes for CAT,CGT,GAT,GGT), VHG (codes for AAG, ACG, ATG, CAG, CCG, CTG, GAG, GCG, GTG), or NNK.

Reverse primer: AAGTAAAGTCGCACCCCAGTCCATCAGCGTGACTACCATCCaaaCGTTGTTACGCAAAGTTAACAGTCGGTACGGCTGACCAT, where the lower case indicates the mutation of two nucleotides of the PAM domain, and one of the targeting spacer, essentially mutating CAG to TTT (Figure 5b).

Plasmids. pSim5 was previously described.[44] The gRNA plasmid was purchased from Addgene (#44251). Cas9 plasmid was generated by amplifying a 4107bp Cas9 open reading frame from genomic DNA of *Streptococcus pyogenes* Strain SF370 (ATCC #700294) and cloned in front of the ribosome binding site of the broad host range plasmid pBTBX-2 to create the X-2Cas9 plasmid.[45]

Creating a mutS Negative Strain. The mutS gene was deleted from the chromosome of the *E. coli* BW25115 strain using the method of Datsenko and Wanner.[46] Briefly, a linear PCR product containing kanamycin resistance marker was amplified from the pKD13 plasmid using primers containing 50 bp of homology to the mutS ORF. A culture of BW25113 carrying the pKD46 plasmid grown at 30 C to an OD600 of approximately 0.1 at which time arabinose was added to a final concentration of 1 mM. Following arabinose addition, cells were grown to an OD600 of ∼0.5, made electrocompetent, and transformed with the linear PCR product. After at least 2 h of recovery in SOB, cells were plated on solid LB media containing kanamycin (30 ug/mL). Knockouts were confirmed via PCR. To remove the antibiotic resistance marker, ΔmutS clones were then transformed with pCP20 and grown on solid media at 37 °C. Resistance marker deletions were confirmed via PCR and loss of kanamycin resistance.

λ-Red Recombineering and CRISPR-Based Selection. *E. coli* BW25113 ΔmutS cells were cotransformed with pSim5 and an arabinose-inducible Cas9 plasmid. Following a standard recombineering protocol[28] that includes also the transformation of a sequence-specific gRNA plasmid, cells were recovered for 2 h and incubated overnight under the appropriate antibiotics and arabinose (0.2%).

Library Cells Genomic Amplification and High Throughput Sequencing. Overnight cultures (30 μL) were washed with water, boiled, and served as a template for a PCR, adding MiSeq pre-adapters. The PCR product served as a template for another PCR, adding an index read barcode sequence, as the sample was mixed with other experiments. The product of the second reaction was loaded on an Ilumina MiSeq high-throughput sequencer according to the manufacture instructions and represented 5% of the total sequencing run.

Data Analysis. Reads from the raw fastq file were run through the usearch fastq quality filtering algorithm with a $q > 15$ threshold.[47] Reads passing this filter were then analyzed using a custom python script to anchor each read to an absolute position in the gene. The targeted codon and PAM positions from these anchored reads were then enumerated to generate codon count frequencies.

■ ASSOCIATED CONTENT

Ⓢ Supporting Information

Supporting figures and tables. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: rtg@colorado.edu.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This work was funded by the U.S. Department of Energy Grant DOE-FOA-0000640. We thank Emily Freed for helping in the manuscript preparation.

■ REFERENCES

(1) Myers, R. M., Lerman, L. S., and Maniatis, T. (1985) A general method for saturation mutagenesis of cloned DNA fragments. *Science 229*, 242−247.

(2) Sidhu, S. S., and Kossiakoff, A. A. (2007) Exploring and designing protein function with restricted diversity. *Curr. Opin. Chem. Biol. 11*, 347−354.

(3) Naqvi, T., Warden, A. C., French, N., Sugrue, E., Carr, P. D., Jackson, C. J., and Scott, C. (2014) A 5000-fold increase in the specificity of a bacterial phosphotriesterase for malathion through combinatorial active site mutagenesis. *PloS One 9*, e94177.

(4) Umeno, D., and Arnold, F. H. (2004) Evolution of a pathway to novel long-chain carotenoids. *J. Bacteriol. 186*, 1531−1536.

(5) Chen, M. M., Snow, C. D., Vizcarra, C. L., Mayo, S. L., and Arnold, F. H. (2012) Comparison of random mutagenesis and semi-rational designed libraries for improved cytochrome P450 BM3-catalyzed hydroxylation of small alkanes. *Protein Eng., Des. Sel. 25*, 171−178.

(6) Reetz, M. T., Bocola, M., Carballeira, J. D., Zha, D., and Vogel, A. (2005) Expanding the range of substrate acceptance of enzymes: Combinatorial active-site saturation test. *Angew. Chem., Int. Ed. Engl. 44*, 4192−4196.

(7) O'Maille, P. E., Bakhtina, M., and Tsai, M. D. (2002) Structure-based combinatorial protein engineering (SCOPE). *J. Mol. Biol. 321*, 677−691.

(8) Hughes, M. D., Nagel, D. A., Santos, A. F., Sutherland, A. J., and Hine, A. V. (2003) Removing the redundancy from randomised gene libraries. *J. Mol. Biol. 331*, 973−979.

(9) Reetz, M. T., Kahakeaw, D., and Lohmer, R. (2008) Addressing the numbers problem in directed evolution. *ChemBioChem 9*, 1797−1804.

(10) IUPAC-IUB Commission on Biochemical Nomenclature. (1971) IUPAC-IUB Commission on Biochemical Nomenclature. Abbreviations and symbols for the description of the conformation of polypeptide chains. Tentative rules (1969). *Biochem. J. 121*, 3471−3479.

(11) Cornish-Bowden, A. (1985) Nomenclature for incompletely specified bases in nucleic acid sequences: Recommendations 1984. *Nucleic Acids Res. 13*, 3021−3030.

(12) Tang, L., Gao, H., Zhu, X., Wang, X., Zhou, M., and Jiang, R. (2012) Construction of "small-intelligent" focused mutagenesis libraries using well-designed combinatorial degenerate primers. *BioTechniques 52*, 149−158.
(13) Kille, S., Acevedo-Rocha, C. G., Parra, L. P., Zhang, Z. G., Opperman, D. J., Reetz, M. T., and Acevedo, J. P. (2013) Reducing codon redundancy and screening effort of combinatorial protein libraries created by saturation mutagenesis. *ACS Synth. Biol. 2*, 83−92.
(14) Ikemura, T. (1985) Codon usage and tRNA content in unicellular and multicellular organisms. *Mol. Biol. Evol. 2*, 13−34.
(15) Sharp, P. M., and Li, W. H. (1987) The codon Adaptation Index—A measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res. 15*, 1281−1295.
(16) Makrides, S. C. (1996) Strategies for achieving high-level expression of genes in *Escherichia coli*. *Microbiol. Rev. 60*, 512−538.
(17) Lithwick, G., and Margalit, H. (2003) Hierarchy of sequence-dependent features associated with prokaryotic translation. *Genome Res. 13*, 2665−2673.
(18) Wells, K. D., Foster, J. A., Moore, K., Pursel, V. G., and Wall, R. J. (1999) Codon optimization, genetic insulation, and an rtTA reporter improve performance of the tetracycline switch. *Transgenic Res. 8*, 371−381.
(19) Bosley, A. D., and Ostermeier, M. (2005) Mathematical expressions useful in the construction, description and evaluation of protein libraries. *Biomol. Eng. 22*, 57−61.
(20) Patrick, W. M., and Firth, A. E. (2005) Strategies and computational tools for improving randomized protein libraries. *Biomol. Eng. 22*, 105−112.
(21) Reetz, M. T. (2011) Laboratory evolution of stereoselective enzymes: A prolific source of catalysts for asymmetric reactions. *Angew. Chem., Int. Ed. Engl. 50*, 138−174.
(22) Nov, Y. (2012) When second best is good enough: Another probabilistic look at saturation mutagenesis. *Appl. Environ. Microbiol. 78*, 258−262.
(23) Hoebenreich, S., Zilly, F. E., Acevedo-Rocha, C. G., Zilly, M., and Reetz, M. T. (2014) Speeding up directed evolution: Combining the advantages of solid-phase combinatorial gene synthesis with statistically guided reduction of screening effort. *ACS Synth. Biol.*, DOI: 10.1021/sb5002399.
(24) Balint, R. F., and Larrick, J. W. (1993) Antibody engineering by parsimonious mutagenesis. *Gene 137*, 109−118.
(25) Jiang, W., Bikard, D., Cox, D., Zhang, F., and Marraffini, L. A. (2013) RNA-guided editing of bacterial genomes using CRISPR-Cas systems. *Nat. Biotechnol. 31*, 233−239.
(26) Oh, J. H., and van Pijkeren, J. P. (2014) CRISPR-Cas9-assisted recombineering in *Lactobacillus reuteri*. *Nucleic Acids Res.*, DOI: 10.1093/nar/gku623.
(27) Costantino, N., and Court, D. L. (2003) Enhanced levels of λ-Red-mediated recombinants in mismatch repair mutants. *Proc. Natl. Acad. Sci. U.S.A. 100*, 15748−15753.
(28) Sharan, S. K., Thomason, L. C., Kuznetsov, S. G., and Court, D. L. (2009) Recombineering: A homologous recombination-based method of genetic engineering. *Nat. Protoc. 4*, 206−223.
(29) Gaj, T., Gersbach, C. A., and Barbas, C. F., 3rd (2013) ZFN, TALEN, and CRISPR/Cas-based methods for genome engineering. *Trends Biotechnol. 31*, 397−405.
(30) Wei, C., Liu, J., Yu, Z., Zhang, B., Gao, G., and Jiao, R. (2013) TALEN or Cas9—Rapid, efficient, and specific choices for genome modifications. *J. Genet. Genomics 40*, 281−289.
(31) Nackley, A. G., Shabalina, S. A., Tchivileva, I. E., Satterfield, K., Korchynskyi, O., Makarov, S. S., Maixner, W., and Diatchenko, L. (2006) Human catechol-O-methyltransferase haplotypes modulate protein expression by altering mRNA secondary structure. *Science 314*, 1930−1933.
(32) Kudla, G., Murray, A. W., Tollervey, D., and Plotkin, J. B. (2009) Coding-sequence determinants of gene expression in *Escherichia coli*. *Science 324*, 255−258.
(33) Komar, A. A., Lesnik, T., and Reiss, C. (1999) Synonymous codon substitutions affect ribosome traffic and protein folding during *in vitro* translation. *FEBS Lett. 462*, 387−391.
(34) Kimchi-Sarfaty, C., Oh, J. M., Kim, I. W., Sauna, Z. E., Calcagno, A. M., Ambudkar, S. V., and Gottesman, M. M. (2007) A "silent" polymorphism in the MDR1 gene changes substrate specificity. *Science 315*, 525−528.
(35) Angov, E., Hillier, C. J., Kincaid, R. L., and Lyon, J. A. (2008) Heterologous protein expression is enhanced by harmonizing the codon usage frequencies of the target gene with those of the expression host. *PloS One 3*, e2189.
(36) Carlson, R. (2009) The changing economics of DNA synthesis. *Nat. Biotechnol. 27*, 1091−1094.
(37) Gaytan, P., and Roldan-Salgado, A. (2013) Elimination of redundant and stop codons during the chemical synthesis of degenerate oligonucleotides. Combinatorial testing on the chromophore region of the red fluorescent protein mKate. *ACS Synth. Biol. 2*, 453−462.
(38) Tang, L., Wang, X., Ru, B., Sun, H., Huang, J., and Gao, H. (2014) MDC-Analyzer: A novel degenerate primer design tool for the construction of intelligent mutagenesis libraries with contiguous sites. *BioTechniques 56 301−302* (304), 306−308 passim..
(39) Ho, S. P., Britton, D. H., Stone, B. A., Behrens, D. L., Leffet, L. M., Hobbs, F. W., Miller, J. A., and Trainor, G. L. (1996) Potent antisense oligonucleotides to the human multidrug resistance-1 mRNA are rationally selected by mapping RNA-accessible sites with oligonucleotide libraries. *Nucleic Acids Res. 24*, 1901−1907.
(40) Palfrey, D., Picardo, M., and Hine, A. V. (2000) A new randomization assay reveals unexpected elements of sequence bias in model 'randomized' gene libraries: Implications for biopanning. *Gene 251*, 91−99.
(41) Wang, H. H., Isaacs, F. J., Carr, P. A., Sun, Z. Z., Xu, G., Forest, C. R., and Church, G. M. (2009) Programming cells by multiplex genome engineering and accelerated evolution. *Nature 460*, 894−898.
(42) Craig, R. A., Lu, J., Luo, J., Shi, L., and Liao, L. (2010) Optimizing nucleotide sequence ensembles for combinatorial protein libraries using a genetic algorithm. *Nucleic Acids Res. 38*, e10.
(43) Kirkpatrick, S. (1984) Optimization by simulated annealing—Quantitative studies. *J. Stat. Phys. 34*, 975−986.
(44) Datta, S., Costantino, N., and Court, D. L. (2006) A set of recombineering plasmids for gram-negative bacteria. *Gene 379*, 109−115.
(45) Prior, J. E., Lynch, M. D., and Gill, R. T. (2010) Broad-host-range vectors for protein expression across gram negative hosts. *Biotechnol. Bioeng. 106*, 326−332.
(46) Datsenko, K. A., and Wanner, B. L. (2000) One-step inactivation of chromosomal genes in *Escherichia coli* K-12 using PCR products. *Proc. Natl. Acad. Sci. U.S.A. 97*, 6640−6645.
(47) Edgar, R. C. (2010) Search and clustering orders of magnitude faster than BLAST. *Bioinformatics 26*, 2460−2461.