 Nucleic Acids Research, 2002, Vol. 30, No. 11 2407–2416

*e*CodonOpt: a systematic computational framework for optimizing codon usage in directed evolution experiments

Gregory L. Moore and Costas D. Maranas*

Department of Chemical Engineering, The Pennsylvania State University, 112 Fenske Laboratory, University Park, PA 16802, USA

Received February 11, 2002; Revised and Accepted April 15, 2002

ABSTRACT

We present a systematic computational framework, *e*CodonOpt, for designing parental DNA sequences for directed evolution experiments through codon usage optimization. Given a set of homologous parental proteins to be recombined at the DNA level, the optimal DNA sequences encoding these proteins are sought for a given diversity objective. We find that the free energy of annealing between the recombining DNA sequences is a much better descriptor of the extent of crossover formation than sequence identity. Three different diversity targets are investigated for the DNA shuffling protocol to showcase the utility of the *e*CodonOpt framework: (i) maximizing the average number of crossovers per recombined sequence; (ii) minimizing bias in family DNA shuffling so that each of the parental sequence pair contributes a similar number of crossovers to the library; and (iii) maximizing the relative frequency of crossovers in specific structural regions. Each one of these design challenges is formulated as a constrained optimization problem that utilizes 0–1 binary variables as on/off switches to model the selection of different codon choices for each residue position. Computational results suggest that many-fold improvements in the crossover frequency, location and specificity are possible, providing valuable insights for the engineering of directed evolution protocols.

INTRODUCTION

The high-throughput screening of large combinatorial libraries is increasingly emerging as a dominant strategy for protein engineering challenges requiring specific functionalities [e.g., thermostability (1–3), enantioselectivity (4,5), gene therapy vectors (6,7), vaccines (8–10) and bioremediation (11–13)]. These combinatorial protein libraries are obtained by expressing, in appropriate prokaryotic hosts, the corresponding combinatorial DNA libraries. Thus, even though the screening step is performed at the protein level, the diversity generation step (i.e., combinatorialization) occurs at the DNA level. A number of methods have been proposed based on random mutagenesis [i.e., error-prone polymerase chain reaction (PCR) (14–16)] and various DNA recombination strategies (17) to generate combinatorial DNA libraries from a small set (i.e., from 2 to about 20) of homologous parental DNA sequences having to some, but not sufficient, extent the desired functionality. The key challenge herein is to ensure that DNA sequence space is sampled in an efficient and unbiased manner. While mutagenesis-based methods essentially probe DNA sequence diversity adjacent to the parental sequences, DNA recombination allows, in principle, the sampling of DNA sequences contained within the convex polytope defined by the vertices representing the parental sequences (Fig. 1). In practice, however, DNA recombination requires the annealing of complementary single-stranded fragments originating from different parental sequences (i.e., heteroduplex formation), which tends to occur primarily within stretches of near perfect sequence identity. This, in turn, gives rise to biased combinatorial DNA libraries or, even worse, libraries with no additional diversity over the parental one.

Here, we explore *in silico* the possibility of boosting, or even specifically directing, the formation of DNA recombination events by exploiting the inherent redundancy in the codon representation while recognizing that host preferences for specific patterns of codon usage may reduce the number of viable codon choices. For example, isoleucine has the following three synonymous codon representations: ATA, ATC and ATT. Therefore, it is possible to optimize the underlying parental DNA sequence codon representation for increasing and/or shaping diversity while at the same time preserving the parental amino acid encodings in the generated combinatorial protein libraries. This strategy is well suited in cases where parental sequences are synthetically generated (e.g., through oligomer ligation). The utility of this approach has been recognized and exploited in an empirical way in the context of industrially developed directed evolution protocols such as oligo shuffling (18) and GeneReassembly (19). In this work, a systematic computational framework is proposed for exploring the limits of performance that can be achieved through codon optimization. Specifically, mathematical optimization problems are formulated and solved for identifying the optimal codon representation of a parental protein set in light of different diversity objectives. DNA shuffling (20,21) is used as the benchmark recombination method to showcase the proposed framework. However, the formulations presented

*To whom correspondence should be addressed. Tel: +1 814 863 9958; Fax: +1 814 865 7846; Email: costas@psu.edu

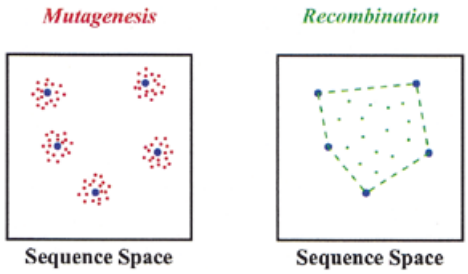

Figure 1. Depiction of the sequence space explored by mutagenesis and recombination. Large blue dots represent parental sequences, while smaller red (mutagenesis) and green (recombination) dots represent combinatorial DNA library members.

here can be extended in a straightforward manner to other annealing-based recombination protocols such as StEP (22), RACHITT (23) and SCRATCHY (24).

The DNA shuffling protocol has been described in detail previously (20,21). Briefly, it consists of two steps: (i) random fragmentation of a small set of parental nucleotide sequences and (ii) reassembly of the fragments through PCR without primers producing a library of full-length nucleotide sequences (Fig. 2). During the fragment annealing step, duplexes are formed through in-frame fragment annealing. Homoduplexes are formed when the annealed fragments originate from the same parental sequence, whereas heteroduplexes are formed when the two fragments are derived from different parental sequences (Fig. 3). Upon extension, heteroduplexes give rise to crossovers, the junction points between segments from different parental sequences (Fig. 2). Crossovers provide the quantitative means for assessing diversity through recombination in DNA shuffling. Because DNA shuffling utilizes annealing and extension steps during reassembly, crossover positions in turn are biased towards regions where pairs of parental sequences share a high degree of sequence identity. This has been observed experimentally (21) and has been quantitatively modeled (25).

In this paper, a systematic method, ***e***CodonOpt, is introduced for redirecting crossover positioning by engineering the sequence identity/free energy profile of a sequence set through codon usage optimization. Specifically, model formulations are described for (i) maximizing the average number of crossovers per recombined sequence, (ii) minimizing bias in family DNA shuffling (26) so that each of the parental sequence pair contributes a similar number of crossovers to the library and (iii) maximizing the relative frequency of crossovers in specific three dimensional (3D) structures such as loop or scaffold regions. In all cases the ***e***Shuffle software (25) is used to predict the number, position and type of crossovers.

*e*CodonOpt MODELING FRAMEWORK

The basic problem addressed in this work can be stated as follows: given a set of parental proteins, design the optimal nucleotide sequences encoding those proteins for a given diversity objective. A constraint-based modeling framework is introduced that only permits nucleotide sequences encoding the underlying parental proteins as solutions. It utilizes 0–1 binary variables as on/off switches to model the presence of a specific codon choice in a given residue position. Below, the index notation, variables, parameters and constraints utilized in the basic ***e***CodonOpt model are listed.

Indices

$i \in \{1,2,\ldots,B\}$ = set of nucleotide sequence positions

$k \in \{1,2,\ldots,K_{tot}\}$ = set of parental sequences

$n,n_1,n_2 \in \{A, T, C, G\}$ = set of nucleotides in positions i, $i + 1, i + 2$ in parental sequence k

Variable set

$$x_{ink} = \begin{cases} 1, & \text{if nucleotide } n \text{ is present at position } i \text{ in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

Parameters

$$a_{ink} = \begin{cases} 1, & \text{if nucleotide } n \text{ is permitted at position } i \text{ in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

$$b_{inn_1k} = \begin{cases} 1, & \text{if nucleotide pair } (n,n_1) \text{ is permitted at positions } (i,i+1) \text{ in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

$$c_{inn_2k} = \begin{cases} 1, & \text{if nucleotide pair } (n,n_2) \text{ is permitted at positions } (i,i+1) \text{ in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

Specifically, the proposed model utilizes the binary variable x_{ink} to represent the underlying nucleotide representation

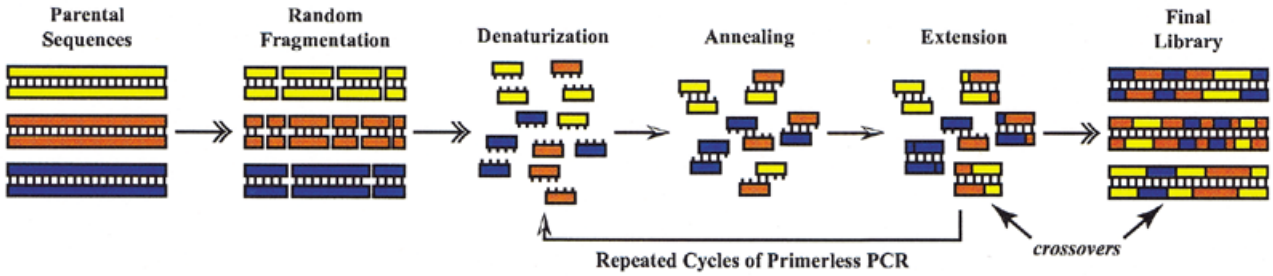

Figure 2. Diagram of DNA shuffling. First, the parental sequences are randomly fragmented by the enzyme DNase I. The fragments are then reassembled by repeated primerless PCR cycles. Each cycle consists of (i) denaturization, when double strands of DNA are separated into single strands, (ii) annealing, when DNA fragments reanneal forming duplexes and (iii) extension, when the addition of new nucleotides is catalyzed by a polymerase enzyme. Crossovers are generated during the extension step when duplexes composed of fragments from different parents have new nucleotides added. After many cycles, full-length sequences are reassembled.

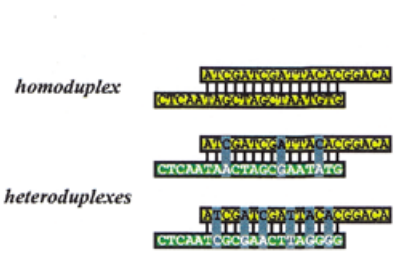

Figure 3. One homoduplex and two heteroduplex examples. Gray shading denotes mismatches in the heteroduplexes. Calculation of the annealing free energy change is for a DNA concentration of 10 ng/µl, 50 mM K^+ and 2.2 mM Mg^{2+} at 55°C.

$n = (A, T, C, G)$ at every sequence position i of the parental protein k. Parameter a_{ink} is equal to 1 only if there exists at least one codon representation that allows the use of nucleotide n at position i of parental sequence k. Parameter b_{inn_1k} is equal to 1 only if nucleotides (n,n_1) are both permitted at the first two codon positions whereas parameter c_{inn_2k} is equal to 1 if nucleotides (n,n_2) are allowed at the first and third codon positions. These parameter values are determined by scanning the parental proteins against the codon translation table. See Tables S1–S3 in the Supplementary Material for a complete list of parameter values for all 20 amino acids.

Codon constraints

Because only one nucleotide choice n can be present at each position i of sequence k, x_{ink} is allowed a non-zero value for only one of the (A, T, C, G) choices for n for every (i,k) pair (see constraint **1**). In addition, if a particular triplet (i,n,k) is not permitted ($a_{ink} = 0$) then variable x_{ink} is forced to zero (constraint **2**).

$$\sum_n x_{ink} = 1, \forall\ i,k \qquad \mathbf{1}$$

$$x_{ink} = 0, \forall\ i,n,k : a_{ink} = 0 \qquad \mathbf{2}$$

Constraints **1** and **2** suffice for residues with a single degenerate position (e.g., alanine). Additional constraints are needed for residues with multiple codon redundancies such as serine, arginine and leucine.

Constraint for serine encoding positions

Specifically for serine with degenerate first and second codon positions, if a consecutive pair (n,n_1) is disallowed ($b_{inn_1k} = 0$) then x_{ink} and $x_{i+1,n_1,k}$ cannot both be equal to 1.

$$x_{ink} + x_{i+1,n_1k} \le 1, \forall\ i,n,n_1,k : b_{inn_1k} = 0 \qquad \mathbf{3}$$

Constraint for arginine, leucine and serine encoding positions

Similarly, for degeneracies in the first and third position for arginine, leucine and serine residues, the following constraint is needed.

$$x_{ink} + x_{i+2,n_2,k} \le 1, \forall\ i,n,n_2,k : c_{inn_2k} = 0 \qquad \mathbf{4}$$

Host requirements

Substantial evidence exists that specific organisms prefer certain synonymous codons (i.e., for the same amino acid) over others. It has been shown that the frequency of codon usage is directly proportional to the corresponding tRNA population [e.g., *Escherichia coli* (27), *Drosophila melanogaster* (28) and *Caenorhabditis elegans* (29)]. Rare codons are generally undesirable because they decrease protein expression levels due to translational stalling (30). The proposed constraint framework is flexible enough to disallow the presence of rare codons by appropriately redefining parameters a_{ink}, b_{inn_1k} and c_{inn_2k}. For example, disallowing the rare isoleucine codon ATA simply requires setting $a_{i+2,Ak} = 0$ for all isoleucine positions in the DNA sequence, thus eliminating the use of A in the third position. However, is worthwhile noting that the removal of all rare codons can cause protein folding problems (31). Therefore, instead of completely eliminating rare codons it is possible to construct constraints that maintain codon usage ratios within some upper and lower bounds defined around the average organism-specific codon usage preferences.

A systematic approach for designing an organism-specific codon representation requires the use of a scoring metric to quantify the level of preferred codons present. Here we formulate constraints requiring that the host-specific score for each of the parental sequences is greater than a specified lower bound. The use of two such metrics is investigated: (i) the Codon Adaptation Index (CAI) (32) and (ii) Major Codon Usage (MCU) (27,33). In calculating the CAI, each codon (n,n_1,n_2) is assigned a weight $\omega_{nn_1n_2}$ that ranges from 0 (low frequency) to 1 (high frequency) based on how often it is utilized in the host organism. For instance, ATC is the most frequently used isoleucine codon, so $\omega_{ATC} = 1$, while the remaining isoleucine codons are assigned weights <1 ($\omega_{ATT} = 0.185$, $\omega_{ATA} = 0.008$). A complete table of weights can be found in Sharp and Li (32) for *E.coli*. The CAI_k for a particular parental sequence k is found by taking the geometric mean of all the individual codon weights.

$$CAI_k = \prod_{i = 1, 4, 7, \ldots} \left(\sum_{n, n_1, n_2} \omega_{nn_1n_2} (x_{ink} \cdot x_{i+1, n_1k} \cdot x_{i+2, n_2k}) \right)^{\frac{1}{B/3}}, \forall\ k \qquad \mathbf{5}$$

Two steps are necessary to express this relation in a linear form: (i) the logarithm is taken on both sides, transforming the geometric mean into an arithmetic one, and (ii) the three-term product is recast at the expense of introducing additional variables. Details of the exact linearization are found in Appendix A. Maintaining CAI_k above a desired lower bound CAI_{min} is attained with the following simple constraint:

$$CAI_k \ge CAI_{min}, \forall\ k \qquad \mathbf{6}$$

An alternative method for scoring a codon representation for a specific host is the calculation of the MCU metric, which quantifies the fraction of codons utilized in a given representation that are 'major' for that organism. Major codons are defined as those codons that appear with greater frequency in genes with high levels of codon bias (33). Whether a codon is a major codon or not is captured by the parameter $\mu_{nn_1n_2}$, which is equal to 1 if codon (n,n_1,n_2) is a major codon, and 0 otherwise (e.g., for isoleucine, $\mu_{ATC} = 1$ and $\mu_{ATT} = \mu_{ATA} = 0$). A tabulation of major codons for *E.coli* is found in Ikemura (27). The following expression is used to calculate the MCU_k metric for each parental sequence:

$$MCU_k = \frac{1}{B/3} \sum_{i = 1, 4, 7, \ldots} \left(\sum_{n, n_1, n_2} \mu_{nn_1n_2} (x_{ink} \cdot x_{i+1, n_1k} \cdot x_{i+2, n_2k}) \right), \forall\ k \qquad \mathbf{7}$$

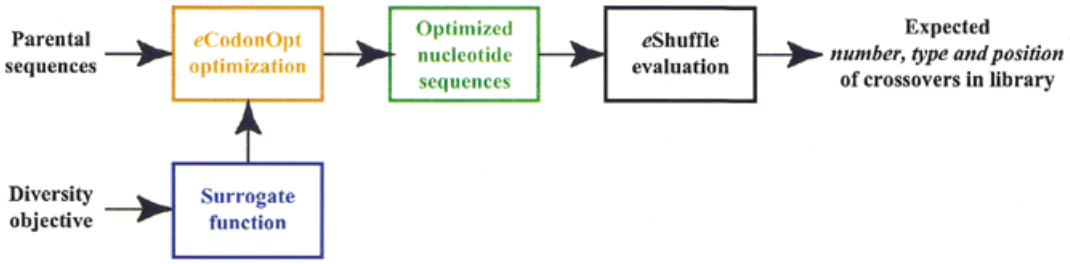

Figure 4. Flowchart showing the sequence of calculations followed in the *e*CodonOpt optimization procedure.

The three-term product is recast into an equivalent linear form in the same way as constraint **5**, and a lower limit on MCU is assigned as follows:

$$MCU_k \geq MCU_{min}, \forall\ k \qquad \mathbf{8}$$

By requiring CAI_k (with constraints **5** and **6**) or MCU_k (constraints **7** and **8**) to be greater than a desired lower bound, codon optimization can be performed while maintaining organism-specific usage ratios.

Limiting the number of codon manipulations

Alternatively, one may want to limit the number of codon representation changes (i.e., silent nucleotide mutations) made to the wild-type DNA sequences. Specifically, the total number of silent nucleotide point mutations in the designed sequences could be set to be less than an upper limit P. This requires the definition of the following additional parameters:

$$\delta_{nn'} = \begin{cases} 1, \text{ if } n = n' \text{ (nucleotide identity)} \\ 0, \text{ otherwise} \end{cases}$$

w_{ink} = codon representation corresponding to the wild-type (original) nucleotide sequences

P = maximum number of point mutations permitted from wild-type nucleotide sequences

Constraint **5** establishes an upper bound to the total number of allowable silent point mutations.

$$\sum_k \sum_i \sum_{n,n'} (1 - \delta_{nn'}) x_{ink}\ w_{in'k} \leq P \qquad \mathbf{9}$$

This constraint-based modeling framework allows searching the space of possible codon representations (codified in variable x_{ink} and subject to constraints **1**–**4**) for the one that optimizes a user-defined diversity objective. In the next section three such diversity objectives are discussed.

DIVERSITY OBJECTIVES

With the codon constraints in place, a number of different diversity objectives are explored: objective I, maximizing the number of crossovers; objective II, minimizing bias in family DNA shuffling; and objective III, maximizing the relative frequency of crossovers in specific structural regions. For objective I, the effect of *E.coli* preferred codon sets on the number of crossovers is studied by including constraints **5** and **6** or **7** and **8** in the optimization model. Optimized sequences for each of the objectives are provided in the Supplementary Material.

Objective I: maximizing the total number of crossovers

Crossover statistics for different parental sequence codon representations can be estimated by the ***e*Shuffle** program (25). However, because the clock time of an ***e*Shuffle** run can range from minutes to hours, utilizing ***e*Shuffle** in the context of optimization loops is impractical for all but the simplest cases. Instead, two simple surrogate objectives for crossover generation are postulated and subsequently tested: (i) maximization of the pairwise sequence identity between the parental DNA sequences and (ii) minimization of the total free energy change upon complete annealing of the two DNA sequences. Both of these surrogates for crossover generation capture the fact that crossover formation in DNA shuffling occurs predominantly within regions of near perfect sequence identity. A flowchart illustrating the sequence of calculations followed for this and other diversity objectives is shown in Figure 4.

Surrogate (i): maximizing pairwise sequence identity. This intuitive surrogate for crossover generation implies that the degree of sequence identity between a pair of DNA sequences correlates with the number of crossovers generated. The calculation of the sequence identity is performed by counting the total number of matching nucleotides, $M_{k\tilde{k}}$, between two aligned parental sequences k and $\tilde{k}$.

$$M_{k\tilde{k}} = \sum_i \sum_{n,\tilde{n}} \delta_{n\tilde{n}} x_{ink} x_{i\tilde{n}\tilde{k}}, \ \forall\ k, \tilde{k} > k \qquad \mathbf{10}$$

Note that the non-linearity introduced by the product of binary variables ($x_{ink} x_{i\tilde{n}\tilde{k}}$) is eliminated (see Appendix B for details). Therefore, the first surrogate for maximizing crossover generation upon codon optimization involves maximizing $M_{k\tilde{k}}$ subject to constraints **1**–**4** and **10**. Constraint sets **5** and **6** or **7** and **8** are added if a host-specific codon representation is desired, while constraint **9** is added if a limit on the total number of silent nucleotide mutations is needed. This problem belongs to the class of mixed-integer linear programming (MILP) problems and is solved using CPLEX 7.0 (34) accessed through the GAMS modeling environment (35). Note, without any additional restrictions such as **5**–**9**, this problem decomposes over codons and can be solved in linear complexity. This decoupling, however, does not hold for surrogate (ii).

Surrogate (ii): minimizing the free energy change of annealing. The second surrogate objective implies that crossover generation correlates with the total free energy change upon

$$\Delta G = \Delta G_{AT/TC} + \Delta G_{TC/CG} + \Delta G_{CG/GC} + \Delta G_{GA/CG} + \Delta G_{AT/GA}$$

Figure 5. Calculation of annealing free energy change using overlapping nearest-nucleotide pairs.

complete annealing of the recombining pair of DNA sequences. The free energy change is approximated using empirical nearest-neighbor parameters (36) which decompose the free energy calculation into the sum of the contributions of overlapping 2-nt units (Fig. 5). Matching pairs contribute negative free energy terms lowering the total free energy change of annealing, whereas mismatches contribute positive terms increasing the free energy change. Parameter set $\Delta G^{pair}_{nn_1\tilde{n}\tilde{n}_1}$ stores the free energy change associated with the annealing of nucleotide pair (n,n_1) with $(\tilde{n},\tilde{n}_1)$. The total free energy change $\Delta G_{k\tilde{k}}$, upon complete annealing of two parental sequences $(k,\tilde{k})$, is calculated by summing the contributions of all nucleotide pairs at positions $(i,i+1)$ along the entire sequence length.

$$\Delta G_{k\tilde{k}} = \sum_{i} \sum_{n,n_1,\tilde{n},\tilde{n}_1} \Delta G^{pair}_{nn_1\tilde{n}\tilde{n}_1}(x_{ink} \cdot x_{i+1,n_1k} \cdot x_{i\tilde{n}\tilde{k}} \cdot x_{i+1,\tilde{n}_1\tilde{k}}), \forall\, k,\tilde{k} > k \quad \mathbf{11}$$

Note that the four-term product in the expression is subsequently expressed in an equivalent linear form. The exact linearization is found in Appendix C. Therefore, the second surrogate for crossover generation in DNA shuffling involves minimizing $\Delta G_{k\tilde{k}}$ subject to constraints **1**–**4** and **11**, and optionally **5** and **6**, **7** and **8** or **9**.

These two surrogate choices are tested based on the DNA shuffling of two glycinamide ribonucleotide (GAR) transformylases. Specifically, the DNA shuffling of the *E.coli* and human versions of GAR transformylase is studied. The wild-type parental sequences share a very low nucleotide sequence identity of 47% even though the two enzymes share the same function and presumably the same structure. In the absence of any codon optimization, DNA shuffling crossovers are extremely rare for this system as shown previously in Moore *et al.* (25); therefore, there is clearly a need to increase the number of crossovers generated.

First, surrogate objective (i), maximizing the sequence identity of the two GAR transformylases, M_{12}, is examined. The maximum sequence identity upon codon optimization is identified for an increasing number of allowed silent nucleotide mutations. These codon-engineered parental sequences are next fed to ***e***Shuffle to predict the total number of crossovers expected to be formed upon DNA shuffling. Crossover numbers are plotted in Figure 6 from 0 (wild-type) to 320 permitted silent mutations. Interestingly, after 90–100 point mutations are accumulated, the total number of crossovers rapidly increases, reaching a maximum value of about 1.5 crossovers per sequence. Beyond this point, sequence identity ceases to correlate with crossover generation leading to the plateau effect beyond 140 silent mutations as shown in Figure 6. The second surrogate objective, involving the minimization of the free energy change of annealing, ΔG_{12}, provides much better correlation with the extent of crossover formation; almost twice as many crossovers are formed compared with the previous surrogate (Fig. 6). The key difference is that, unlike sequence identity, the free

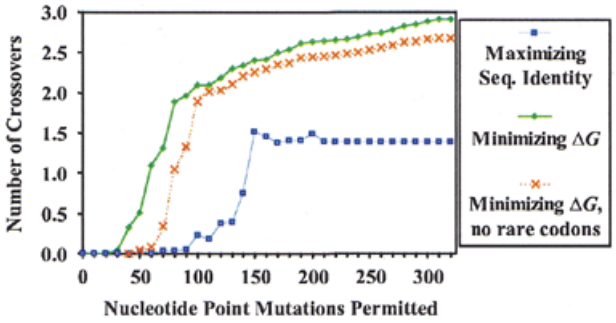

Figure 6. The total number of crossovers increases as more point mutations are permitted. Free energy change outperforms sequence identity as a surrogate. When rare *E.coli* codons are excluded, only a slight decrease is seen in the total number of crossovers.

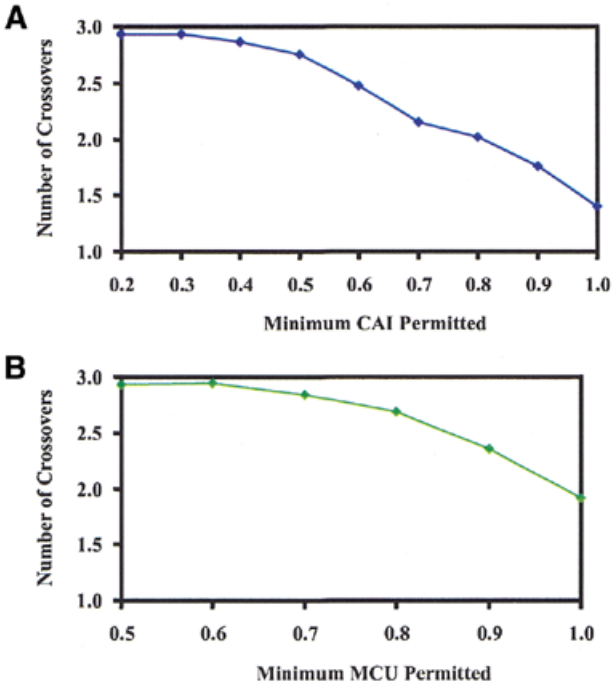

Figure 7. (**A**) As expected, the number of crossovers decreases as the lower bound on the Codon Adaptation Index (CAI_{min}) increases from 0.2 to 1. (**B**) The number of crossovers decreases as the minimum Major Codon Usage (MCU_{min}) increases from 0.5 to 1.

energy change continues to correlate strongly with crossover formation even for very high numbers of silent mutations preventing the onset of the plateau. Interestingly, the extent of crossover formation is only mildly affected by excluding all *E.coli* rare codons from consideration [i.e., ATA, AGA, AGG, TGT, CTA, CCC, CGA and CGG (37)]. Even when a lower bound is placed on the CAI metric (Fig. 7A) or MCU criterion (Fig. 7B) for the parental sequences, comparable numbers of crossovers are still generated. Even the most stringent requirement (CAI = MCU = 1) results in a <50% drop in the predicted number of crossovers from the theoretical maximum. These results demonstrate that codon optimization can be effectively performed for organism-specific codon sets leading to higher levels of protein expression in addition to a more diverse combinatorial library.

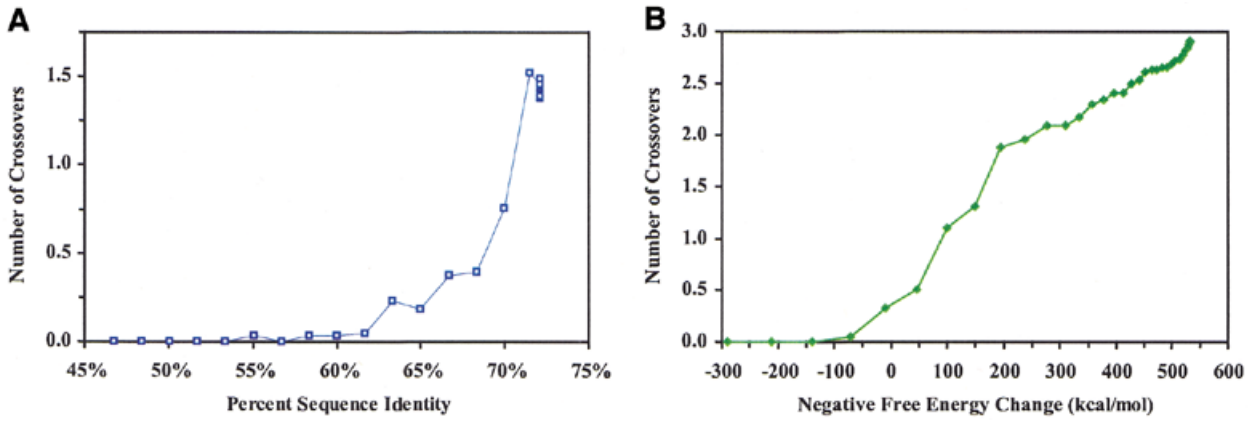

Figure 8. (**A**) Plot of the percent sequence identity of optimized (max M_{12}) sequences versus the total number of crossovers as a function of the number of silent mutations permitted. (**B**) Plot of the negative of the free energy change for optimized (min ΔG_{12}) sequences versus the number of crossovers as the number of silent mutations permitted is increased.

The strength of correlation of the two surrogate functions with the total number of crossovers generated is shown more clearly in Figure 8. It is noteworthy that increasing sequence identity beyond a certain level does not increase crossover generation (Fig. 8A). In fact, a reversal in the sign of correlation occurs near the end of the plot. On the other hand, free energy change correlates monotonically and almost linearly (Fig. 8B) with the extent of crossover formation. $\Delta G_{k\tilde{k}}$ out-performs sequence identity as a surrogate for crossover formation because it appropriately weighs the thermodynamic contribution of different matches and mismatches. In addition, by considering the contribution of overlapping nucleotide pairs, it places a higher emphasis on blocks of sequence identity over isolated nucleotide matches. Sequence identity is not as successful as a surrogate because the matching nucleotides do not necessarily group into crossover-generating blocks of sequence identity. The qualitative trends in the result hold for a wide range of example problems studied so far, implying that free energy of annealing is universally superior to sequence identity as a predictor of crossover formation. This result has a substantial implication on the way DNA shuffling studies are conducted and parental DNA sequences are engineered.

Objective II: minimizing bias in family DNA shuffling

Family DNA shuffling (26) extends DNA shuffling to more than two parental sequences allowing the simultaneous mixing of genetic information from many homologous DNA sequences. However, a strong possibility exists for a biased library in which only a small subset of the shuffled family generates crossovers while the remainder of the parental set does not participate in the recombination process. This results in a biased combinatorial library where the majority of crossovers originate from only a few pairs and the majority of parental sequences do not contribute to the genetic diversity of the combinatorial library.

Earlier, it was shown that the free energy change of annealing, $\Delta G_{k\tilde{k}}$, is a good predictor for the number of crossovers generated by a pair of parental sequences. By building on this constraint-based framework the goal here is to ensure that each parental sequence pair contributes an approximately equal number of crossovers to the library while the total number of generated crossovers stays as high as possible. This is ensured mathematically by minimizing the average free energy change over all parental sequence pairs while constraining all of the pairwise free energy changes within a window centered about the mean. The mean free energy change, ΔG_{mean}, is given by:

$$\Delta G_{mean} = \frac{\sum_{k,\tilde{k}>k} \Delta G_{k\tilde{k}}}{K_{tot}\,(K_{tot}-1)/2} \qquad \mathbf{12}$$

The parameter α is used to set the size of the window in which each of the pairwise free energy changes can fall. For example, setting $\alpha = 10\%$ ensures that all $\Delta G_{k\tilde{k}}$ are within 10% from ΔG_{mean}. Two linear constraints are utilized to set the upper and lower bounds on $\Delta G_{k\tilde{k}}$ separately.

$$\Delta G_{k\tilde{k}} \leq (1-\alpha)\Delta G_{mean},\ \forall\ k, \tilde{k} > k \qquad \mathbf{13}$$

$$\Delta G_{k\tilde{k}} \geq (1+\alpha)\Delta G_{mean},\ \forall\ k, \tilde{k} > k \qquad \mathbf{14}$$

Minimizing G_{mean} subject to constraints **1**–**4** and **11**–**14** increases overall crossover frequency while simultaneously reducing bias towards particular parental sequence pairs.

The family DNA shuffling of a family of four cephalosporinases (26) is used here to demonstrate the proposed framework. For the wild-type sequences, ***e***Shuffle predicts that 70% of the crossovers are generated by a single parental pair, *Citrobacter freundii* and *Enterobacter cloacae* (Fig. 9, 1 and 2, respectively). Solving the optimization problem posed above with $\alpha = 10\%$ for the four cephalosporinases greatly compacts the range of pairwise free energy changes by a factor of 4.5 (Fig. 10). This leads to a crossover distribution that is much more even (Fig. 9). Crossovers between *C.freundii* and *E.cloacae*, previously in excess of 70%, are reduced to a contribution of only 17%, while other types of crossovers are boosted. In addition to removing bias from the library, the optimization procedure greatly increases the total number of crossovers per sequence, from 0.87 up to 12.1. The ability to customize the number and type of crossovers for a sequence family can significantly affect the design of family DNA shuffling experiments. Codon optimization can substantially augment the set of feasible parental recombination candidates since homology can be custom-engineered.

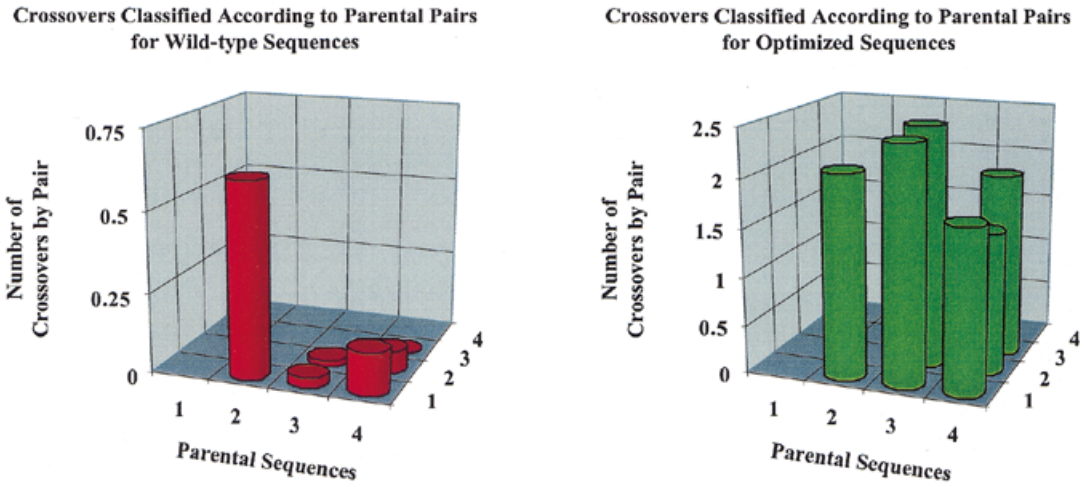

Figure 9. Crossover statistics before and after optimization for all possible pairs of parental sequences: (1) *C.freundii*, (2) *E.cloacae*, (3) *Y.enterocolitica* and (4) *K.pneumoniae*.

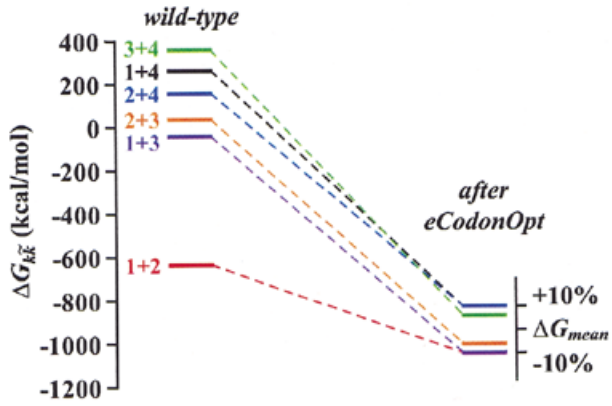

Figure 10. Free energy of annealing before and after optimization for all six pairs of parental sequences.

Objective III: directing crossovers to specific structural regions

Here we examine how crossovers can be directed to specific structural regions through codon optimization. These regions can be secondary structure units such as helices or sheets, specific domains of multi-domain proteins or sites that bind either substrates or co-factors. Currently there is substantial effort in the literature aimed at identifying regions where crossovers are more likely to be tolerated giving rise to functional hybrids. Some approaches are hypothesis driven such as multipool swapping (38) and minimum schema disruption (39), whereas others attempt to identify these regions by employing structural energy calculations (40). Given these desirable crossover regions a parameter L_i is defined that flags them along the sequence:

$$L_i = \begin{cases} 1, \text{ if position } i \text{ is a desirable crossover position} \\ 0, \text{ otherwise} \end{cases}$$

Preferentially directing crossovers to one region is achieved by minimizing $\Delta G_{annealing}$ in the preferred regions while maximizing $\Delta G_{annealing}$ in the remaining regions. Expressions for the change in free energy for desirable and undesirable crossover regions are given below.

$$\Delta G_{desirable} = \sum_{k,\tilde{k}<k} \sum_{\substack{i:L_i=1,\\ L_{i+1}=1}} \sum_{n,n_1,\tilde{n},\tilde{n}_1} \Delta G^{pair}_{nn_1\tilde{n}\tilde{n}_1} (x_{ink} \cdot x_{i+1,n_1k} \cdot x_{i\tilde{n}\tilde{k}} \cdot x_{i+1,\tilde{n}_1\tilde{k}})$$

$$\Delta G_{undesirable} = \sum_{k,\tilde{k}<k} \sum_{\substack{i:L_i=0,\\ L_{i+1}=0}} \sum_{n,n_1,\tilde{n},\tilde{n}_1} \Delta G^{pair}_{nn_1\tilde{n}\tilde{n}_1} (x_{ink} \cdot x_{i+1,n_1k} \cdot x_{i\tilde{n}\tilde{k}} \cdot x_{i+1,\tilde{n}_1\tilde{k}})$$

The proposed approach is demonstrated by preferentially allocating crossovers to the loop and scaffold regions of the phosphoribosylanthranilate isomerase (PRAI) domain of a bifunctional enzyme (41). PRAI is an α/β barrel protein with a scaffold region spanning the inner β-barrel and the eight outer α-helices (Fig. 11, purple and gold). Loops are defined as the connecting regions between the β-barrel and the α-helices and are shown in white in Figure 11. Parameter L_i indicates whether a sequence position is within a loop or not, and its values are superimposed on the PRAI 3D structure. Two design objectives are pursued: (i) directing crossovers to loop regions by minimizing ($\Delta G_{loop} - \Delta G_{scaffold}$) and (ii) directing crossovers to the scaffold by minimizing ($\Delta G_{scaffold} - \Delta G_{loop}$). Both of these two optimization problems are solved for the DNA shuffling of *E.coli* and *Salmonella enterica typhi* versions of the PRAI domain. For the wild-type sequences, ***e***Shuffle predicts that crossovers are predominantly located in the scaffold region (Fig. 12). Upon loop-optimization, crossovers in loop regions are increased almost 20-fold, outpacing those in the scaffold region by 40%. Alternatively, scaffold optimization increases the number of crossovers found in the scaffold region by 10-fold. The crossover locations after optimization are superimposed against the 3D structure in Figure 13. Codon optimization dramatically reshapes the crossover distribution (Fig. 13) by biasing it towards targeted regions. Interestingly, a by-product of the optimization is that, for both the loop and scaffold-optimized cases, overall crossover generation is greatly increased. The results obtained for this example

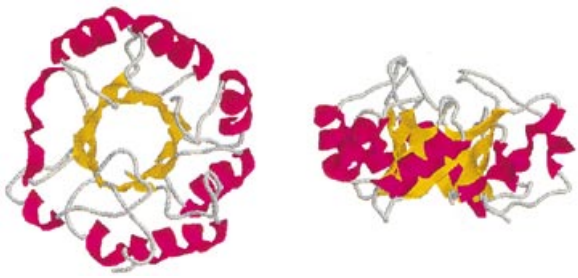

Figure 11. Top and side view of the *E.coli* PRAI protein domain. Scaffold regions are colored purple (α-helices) and gold (β-barrel), while connecting loop regions are colored gray.

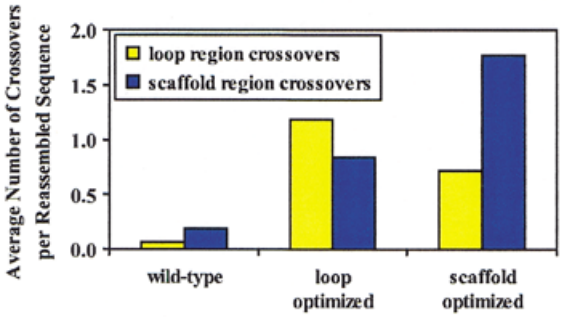

Figure 12. Codon optimization results for loop and scaffold regions in the PRAI domain.

demonstrate that codon optimization provides an effective strategy for directing crossovers to desirable protein regions.

IMPLEMENTATION

Optimization problems were solved using CPLEX 7.0 (34) accessed through the GAMS modeling environment (35) on an IBM RS6000-270 workstation. CPU times were in the order of seconds for objectives I(i), I(ii) and III, and hours for objective II. *e*Shuffle runs were performed assuming a standard DNA shuffling setup: annealing temperature 55°C, fragment length 25 nt, DNA concentration 10 ng/µl, 50 mM K^+ and 2.2 mM Mg^{2+}. Nucleotide and amino acid sequences utilized in the examples were downloaded from GenBank via the Entrez system (42). Accession numbers for wild-type proteins were: *E.coli* and human GAR transformylases, P08179 and P22102; *C.freundii*, *E.cloacae*, *Yersinia enterocolitica* and *Klebsiella pneumoniae* cephalosporinases, CAA35959, CAC08446, CAA44850 and AAK70221; and *E.coli* and *S.enterica typhi* PRAI domains, AAA57299 and CAD08407. The 3D structure of the PRAI domain (1PII, residues 256–452) was downloaded from the Protein Data Bank (43). Protein Explorer (http://proteinexplorer.org) was used to render 3D structures.

SUMMARY

In this paper, a systematic computational framework, *e*CodonOpt (http://fenske.che.psu.edu/faculty/cmaranas), for designing parental DNA sequences for directed evolution experiments through codon usage optimization was introduced. With the proposed MILP formulation, we designed parental sequence sets that met a variety of diversity objectives while observing host-specific codon preferences based on the CAI and MCU metrics. Initially, the number of crossovers generated by DNA shuffling was boosted substantially by optimizing the annealing free energy profile of two GAR transformylases. Then, crossover bias towards specific parental pairs was reduced for an engineered family of cephalosporinases while simultaneously increasing the total number of crossovers formed by family DNA shuffling. Finally, crossovers were preferentially allocated to specific structural regions in a PRAI domain allowing a customized crossover distribution. Much flexibility is present in the constraint-based framework, permitting the investigation of many other choices for diversity objectives.

We believe that codon engineering is capable of expanding and shaping the sequence space spanned by directed evolution experiments. As our knowledge of how recombination events preserve or disrupt protein structure improves, optimal design of the parental DNA sequence set will allow a more focused

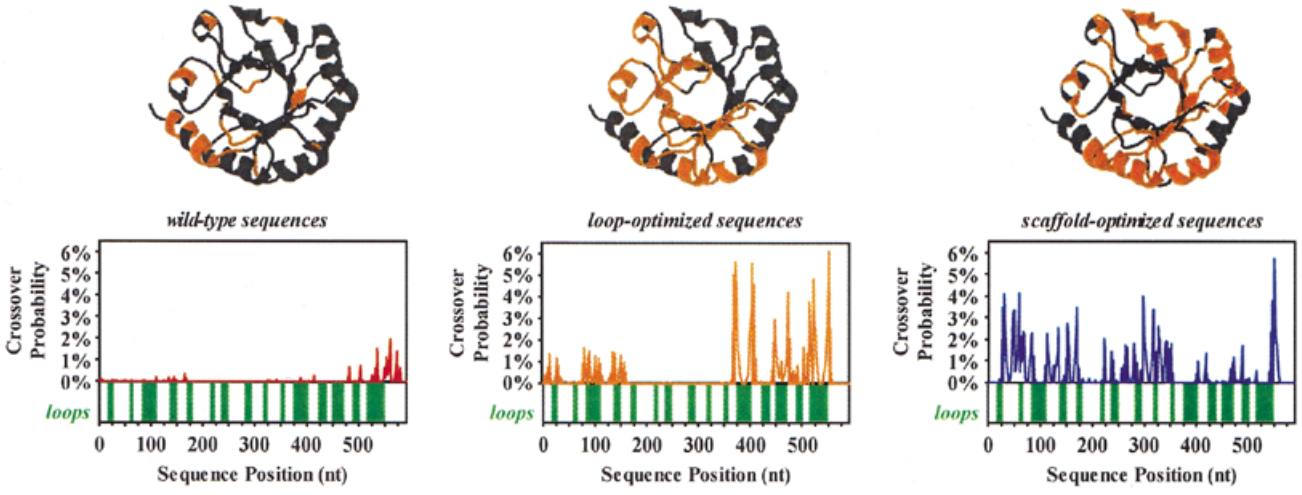

Figure 13. Crossover position statistics before and after codon optimization. Loop regions are represented by green bars in the strip chart. Orange residues in the 3D structures represent positions with crossover probability >0.1%.

probing of sequence space in only those regions that are likely to yield functional hybrids. This, in turn, will lead to a more efficient utilization of experimental resources, saving time and effort by reducing the number of evolutionary cycles that must be performed for a successful protein design effort.

SUPPLEMENTARY MATERIAL

Supplementary Material is available at NAR Online.

ACKNOWLEDGEMENTS

We thank Prof. Stephen Benkovic and Dr Stefan Lutz for helpful discussions. Financial support from National Science Foundation Award BES0120277 and hardware support by the IBM-SUR program are gratefully acknowledged.

REFERENCES

1. Giver,L., Gershenson,A., Freskgard,P. and Arnold,F. (1998) Directed evolution of a thermostable esterase. *Proc. Natl Acad. Sci. USA*, **95**, 12809–12813.
2. Matsumura,I. and Ellington,A. (1999) *In vitro* evolution of thermostable p53 variants. *Protein Sci.*, **8**, 731–740.
3. Miyazaki,K., Wintrode,P., Grayling,R., Rubingh,D. and Arnold,F. (2000) Directed evolution study of temperature adaptation in a psychrophilic enzyme. *J. Mol. Biol.*, **297**, 1015–1026.
4. Jaeger,K., Eggert,T., Eipper,A. and Reetz,M. (2001) Directed evolution and the creation of enantioselective biocatalysts. *Appl. Microbiol. Biotechnol.*, **55**, 519–530.
5. Reetz,M., Wilensek,S., Zha,D. and Jaeger,K. (2001) Directed evolution of an enantioselective enzyme through combinatorial multiple-cassette mutagenesis. *Angew. Chem. Int. Ed. Engl.*, **40**, 3589–3591.
6. Powell,S., Kaloss,M., Pinskstaff,A., McKee,R., Burimski,I., Pensiero,M., Otto,E., Stemmer,W. and Soong,N. (2000) Breeding of retroviruses by DNA shuffling for improved stability and processing yield. *Nat. Biotechnol.*, **18**, 1279–1282.
7. Soong,N., Nomura,L., Pekrun,K., Reed,M., Sheppard,L., Dawes,G. and Stemmer,W. (2000) Molecular breeding of viruses. *Nature Genet.*, **25**, 436–439.
8. Patten,P., Howard,R. and Stemmer,W. (1997) Applications of DNA shuffling to pharmaceuticals and vaccines. *Curr. Opin. Biotechnol.*, **8**, 724–733.
9. Marzio,G., Verhoef,K., Vink,M. and Berkhout,B. (2001) *In vitro* evolution of a highly replicating, doxycycline-dependent HIV for applications in vaccine studies. *Proc. Natl Acad. Sci. USA*, **98**, 6342–6247.
10. Whalen,R., Kaiwar,R., Soong,N. and Punnonen,J. (2001) DNA shuffling and vaccines. *Curr. Opin. Mol. Ther.*, **3**, 31–36.
11. Wackett,L. (1998) Directed evolution of new enzymes and pathways for environmental biocatalysis. *Ann. N. Y. Acad. Sci.*, **864**, 142–152.
12. Bruhlmann,F. and Chen,W. (1999) Tuning biphenyl dioxygenase for extended substrate specificity. *Biotechnol. Bioeng.*, **63**, 544–551.
13. Furukawa,K. (2000) Engineering dioxygenases for efficient degradation of environmental pollutants. *Curr. Opin. Biotechnol.*, **11**, 244–249.
14. Leung,D., Chen,E. and Goeddel,D. (1989) A method for random mutagenesis of a defined DNA segment using a modified polymerase chain reaction. *Technique*, **1**, 11–15.
15. Cadwell,R. and Joyce,G. (1992) Randomization of genes by PCR mutagenesis. *PCR Methods Appl.*, **2**, 28–33.
16. Lin-Goerke,J., Robbins,D. and Burczak,J. (1997) PCR-based random mutagenesis using manganese and reduced dNTP concentration. *Biotechniques*, **23**, 409–412.
17. Arnold,F. and Volkov,A. (1999) Directed evolution of biocatalysts. *Curr. Opin. Chem. Biol.*, **3**, 54–59.
18. Stemmer,W. (2000) US6,132,970: Methods of shuffling polynucleotides.
19. Short,J. (1999) US5,965,408: Method of DNA reassembly by interrupting synthesis.
20. Stemmer,W. (1994) Rapid evolution of a protein *in vitro* by DNA shuffling. *Nature*, **370**, 389–391.
21. Stemmer,W. (1994) DNA shuffling by random fragmentation and reassembly: *in vitro* recombination for molecular evolution. *Proc. Natl Acad. Sci. USA*, **91**, 10747–10751.
22. Zhao,H., Giver,L., Shao,Z., Affholter,J. and Arnold,F. (1998) Molecular evolution by staggered extension process (StEP) *in vitro* recombination. *Nat. Biotechnol.*, **16**, 258–261.
23. Coco,W., Levinson,W., Crist,M., Hektor,H., Darzins,A., Pienkos,P., Squires,C. and Monticello,D. (2001) DNA shuffling method for generating highly recombined genes and evolved enzymes. *Nat. Biotechnol.*, **19**, 354–359.
24. Lutz,S., Ostermeier,M., Moore,G., Maranas,C. and Benkovic,S. (2001) Creating multiple-crossover DNA libraries independent of sequence identity. *Proc. Natl Acad. Sci. USA*, **98**, 11248–11253.
25. Moore,G., Maranas,C., Lutz,S. and Benkovic,S. (2001) Predicting crossover generation in DNA shuffling. *Proc. Natl Acad. Sci. USA*, **98**, 3226–3231.
26. Crameri,A., Raillard,S., Bermudez,E. and Stemmer,W. (1998) DNA shuffling of a family of genes from diverse species accelerates directed evolution. *Nature*, **391**, 288–291.
27. Ikemura,T. (1985) Codon usage and tRNA content in unicellular and multicellular organisms. *Mol. Biol. Evol.*, **2**, 13–34.
28. Moriyama,E. and Powell,J. (1997) Codon usage bias and tRNA abundance in *Drosophila*. *J. Mol. Evol.*, **45**, 514–523.
29. Duret,L. (2000) tRNA gene number and codon usage in *C. elegans* genome are co-adapted for the optimal translation of highly expressed genes. *Trends Genet.*, **16**, 287–289.
30. Baca,A. and Hol,W. (2000) Overcoming codon bias: a method for high-level overexpression of *Plasmodium* and other AT-rich parasite genes in *Escherichia coli*. *Int. J. Parasitol.*, **30**, 113–118.
31. Komar,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.
32. Sharp,P. and Li,W. (1987) The codon adaptation index – a measure of directional synonymous codon usage bias and its potential applications. *Nucleic Acids Res.*, **15**, 1281–1295.
33. Akashi,H. (1995) Inferring weak selection from patterns of polymorphism and divergence at "silent" sites in *Drosophila* DNA. *Genetics*, **139**, 1067–1076.
34. Brooke,A., Kendrick,D., Meeraus,A. and Raman,R. (1998) *GAMS: The Solver Manuals*. GAMS Development Corporation, Washington, DC.
35. Brooke,A., Kendrick,D., Meeraus,A. and Raman,R. (1998) *GAMS: A User's Guide*. GAMS Development Corporation, Washington, DC.
36. SantaLucia.,J.,Jr (1998) A unified view of polymer, dumbbell and oligonucleotide DNA nearest neighbor thermodynamics. *Proc. Natl Acad. Sci. USA*, **95**, 1460–1465.
37. Nakamura,Y., Gojobori,T. and Ikemura,T. (2000) Codon usage tabulated from international DNA sequence databases: status for the year 2000. *Nucleic Acids Res.*, **28**, 292.
38. Bogarad,L. and Deem,M. (1999) A hierarchical approach to protein molecular evolution. *Proc. Natl Acad. Sci. USA*, **96**, 2591–2595.
39. Voigt,C., Wang,Z. and Arnold,F. (2000) A computational approach to directed evolution. Presented at American Institute of Chemical Engineers Annual Meeting, Los Angeles, CA.
40. Voigt,C., Mayo,S., Arnold,F. and Wang,Z. (2000) Computational method to reduce the search space for directed protein evolution. *Proc. Natl Acad. Sci. USA*, **98**, 3778–3783.
41. Altamirano,M., Blackburn,J., Aguayo,C. and Fersht,A. (2000) Directed evolution of new catalytic activity using the alpha/beta-barrel scaffold. *Nature*, **98**, 3288–3293.
42. Benson,D., Karsch-Mizrachi,I., Lipman,D., Ostell,J., Rapp,B. and Wheeler,D. (2002) GenBank. *Nucleic Acids Res.*, **30**, 17–20.
43. Westbrook,J., Feng,Z., Jain,S., Bhat,T., Thanki,N., Ravichandran,V., Gilliland,G., Bluhm,W., Weissig,H., Greer,D., Bourne,P. and Berman,H. (2002) The protein data bank: unifying the archive. *Nucleic Acids Res.*, **30**, 245–248.
44. Ahula,R., Magnanti,T. and Orlin,J. (1993) *Network Flows*. Prentice Hall, Inc., Upper Saddle River, NJ.

APPENDICES

Appendix A: equivalent linear representation of the CAI

In the expression for CAI_k, two sources of non-linearity are present: (i) the geometric mean and (ii) the product of three binary variables ($x_{ink} \cdot x_{i+1,n_1k} \cdot x_{i+2,n_2k}$). The geometric mean is transformed to an arithmetic one by taking the logarithm of both sides.

$$\log\ (CAI_k) =$$

$$\frac{1}{B/3} \sum_{i\ =\ 1,\ 4,\ 7,\ \ldots} \sum_{n,\ n_1,\ n_2} (x_{ink} \cdot x_{i\ +\ 1,\ n_1k} \cdot x_{i\ +\ 2,\ n_2k}) \log\ (\omega_{nn_1n_2}),\ \forall\ k$$

The product of binary variables is replaced by the continuous variable $z_{inn_1n_2k}$, which is defined as follows:

$$z_{inn_1n_2k} = \begin{cases} 1, & \text{if at positions } (i,i+1,i+2) \text{ codon } (n,n_1,n_2) \text{ is present in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

x_{ink} is linked to this new binary variable by the following three constraints:

$$\sum_{n_1,\ n_2} z_{inn_1n_2k} = x_{ink},\ \forall\ i,n,k$$

$$\sum_{n,\ n_2} z_{inn_1n_2k} = x_{i+1,n_1k},\ \forall\ i,n_1,k$$

$$\sum_{n,\ n_1} z_{inn_1n_2k} = x_{i+2,n_2k},\ \forall\ i,n_2,k$$

A new expression for $\log(\mathrm{CAI}_k)$ results.

$$\log\ (\mathrm{CAI}_k) = \frac{1}{B/3} \sum_{i\ =\ 1,\ 4,\ 7,\ \ldots} \sum_{n,\ n_1,\ n_2} z_{inn_1n_2k} \log\ (\omega_{nn_1n_2}),\ \forall\ k$$

Although $z_{inn_1n_2k}$ is defined as a continuous variable, it will only take 0–1 values as long as it is forced to be non-negative. This occurs because of the structure of the constraint set, which is that of an assignment problem (44).

Appendix B: equivalent linear representation of surrogate (i)

In surrogate objective (i), the product of two binary variables is present $(x_{ink}x_{i\tilde{n}\tilde{k}})$. This product is replaced by the continuous variable $z_{in\tilde{n}k\tilde{k}}$, which is defined as follows:

$$z_{in\tilde{n}k\tilde{k}} = \begin{cases} 1, & \text{if at position } i \text{ nucleotide } n \text{ is present in parental sequence } k \text{ and nucleotide } \tilde{n} \text{ is present in parental sequence } \tilde{k} \\ 0, & \text{otherwise} \end{cases}$$

x_{ink} is linked to this new binary variable by the following two constraints:

$$\sum_{\tilde{n}} z_{in\tilde{n}k\tilde{k}} = x_{ink},\ \forall\ i,\ n,\ k,\ \tilde{k} > k$$

$$\sum_{n} z_{in\tilde{n}k\tilde{k}} = x_{i\tilde{n}\tilde{k}},\ \forall\ i,\ \tilde{n},\ k,\ \tilde{k} > k$$

A new expression for $M_{k\tilde{k}}$ results.

$$M_{k\tilde{k}} = \sum_{i} \sum_{n,\ \tilde{n}} \delta_{n\tilde{n}} z_{in\tilde{n}k\tilde{k}},\ \forall\ k,\ \tilde{k} > k$$

Appendix C: equivalent linear representation of surrogate (ii)

The binary variable product $x_{ink} \cdot x_{i\ +\ 1,\ n_1k} \cdot x_{i\tilde{n}\tilde{k}} \cdot x_{i\ +\ 1,\ \tilde{n}_1\tilde{k}}$ is replaced in a similar manner by a new continuous variable defined below.

$$z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}} = \begin{cases} 1, & \text{if at positions } (i,i+1) \text{ nucleotide pair } (n,n_1) \text{ is present in parental sequence } k \text{ and nucleotide pair } (\tilde{n},\tilde{n}_1) \text{ is present in parental sequence } k \\ 0, & \text{otherwise} \end{cases}$$

Four new constraints and a new expression for $\Delta G^{tot}_{k\tilde{k}}$ result.

$$\sum_{n_1,\ \tilde{n},\ \tilde{n}_1} z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}} = x_{ink},\ \forall\ i,\ n,\ k,\ \tilde{k} > k$$

$$\sum_{n,\ \tilde{n},\ \tilde{n}_1} z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}} = x_{i\ +\ 1,\ n_1k},\ \forall\ i,\ n_1,\ k,\ \tilde{k} > k$$

$$\sum_{n,\ n_1,\ \tilde{n}_1} z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}} = x_{i\tilde{n}\tilde{k}},\ \forall\ i,\ \tilde{n},\ k,\ \tilde{k} > k$$

$$\sum_{n,\ n_1,\ \tilde{n}} z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}} = x_{i\ +\ 1,\ \tilde{n}_1\tilde{k}},\ \forall\ i,\ \tilde{n}_1,\ k,\ \tilde{k} > k$$

$$\Delta G^{tot}_{k\tilde{k}} = \sum_{i} \sum_{n,\ n_1,\ \tilde{n},\ \tilde{n}_1} \Delta G^{pair}_{nn_1\tilde{n}\tilde{n}_1} z_{inn_1\tilde{n}\tilde{n}_1k\tilde{k}},\ \forall\ k,\ \tilde{k} > k$$