Protein Engineering, Design & Selection pp. 1–13, 2007
doi:10.1093/protein/gzm030

Optimal protein library design using recombination or point mutations based on sequence-based scoring functions

Robert J. Pantazes[1], Manish C. Saraf[1] and Costas D. Maranas[1,2]

[1]Department of Chemical Engineering, The Pennsylvannia State University, University Park, PA 16802, USA

[2]To whom correspondence should be addressed.
E-mail: costas@psu.edu

In this paper, we introduce and test two new sequence-based protein scoring systems (i.e. S1, S2) for assessing the likelihood that a given protein hybrid will be functional. By binning together amino acids with similar properties (i.e. volume, hydrophobicity and charge) the scoring systems S1 and S2 allow for the quantification of the severity of mismatched interactions in the hybrids. The S2 scoring system is found to be able to significantly functionally enrich a cytochrome P450 library over other scoring methods. Given this scoring base, we subsequently constructed two separate optimization formulations (i.e. OPTCOMB and OPTOLIGO) for optimally designing protein combinatorial libraries involving recombination or mutations, respectively. Notably, two separate versions of OPTCOMB are generated (i.e. model M1, M2) with the latter allowing for position-dependent parental fragment skipping. Computational benchmarking results demonstrate the efficacy of models OPTCOMB and OPTOLIGO to generate high scoring libraries of a prespecified size.

Keywords: hybrid scoring method/protein engineering/protein library optimization

Introduction

By creatively applying the ever-growing palette of molecular biology techniques, a variety of protocols are currently available for performing targeted mutations or constructing combinatorial libraries with customized statistics of mutations and/or parental fragments (see Moore and Maranas, 2004 for a review). Recent developments have made it clear that given sufficient resources, a directed evolution protocol can be set up to create the desired level of diversity. This shifts the challenge to *a priori* identifying the optimal level and type of diversity for a given protein engineering task.

Most protein design efforts can broadly be classified into two main categories. The first category involves the redesign of an existing protein structure. It can be accomplished through changing the existing binding pocket, to alter the current functionality of the protein (Miyazaki *et al.*, 2000; Santoro *et al.*, 2002; Baik *et al.*, 2003; Chen and Zhao, 2005; Chockalingam *et al.*, 2005; Korkegian *et al.*, 2005), or by introducing an entirely new binding pocket, when a different functionality is desired (Marvin and Hellinga, 2001; Dwyer *et al.*, 2003). This type of protein design is dependent on detailed structural information of the protein. The second group of protein design efforts considers cases when only protein sequence information is available with little or no structural knowledge of the protein system. Here, typically, the objective is to identify variants that preserve and/or enhance the desired activity, which is accomplished through the creation and screening of large protein combinatorial libraries. This reliance on larger combinatorial libraries is precipitated by the lack of structural information requiring that computations must focus on the design of libraries based on sequence information alone, rather than targeted structural redesign, to enhance or improve function. The source of diversity in the library can be either accumulated point mutations, recombination of a small set of parental sequences or both. A number of techniques have been developed to create protein libraries with customized diversity statistics (Stemmer, 1994a, 1994b, 2000; Zhao *et al.*, 1998; Ostermeier *et al.*, 1999; Coco *et al.*, 2001, 2002; Lutz *et al.*, 2001; Martin *et al.*, 2001; Sakamoto *et al.*, 2001; Sieber *et al.*, 2001; Richardson *et al.*, 2002; Hiraga and Arnold, 2003; Áň *et al.*, 2005). This has placed the emphasis on identifying how to optimally apportion this diversity for a given library so that the library is maximally enriched in functional sequences. A number of scoring methods such as SCHEMA (Voigt *et al.*, 2002), SIRCH (Moore and Maranas, 2003), FamClash (Saraf *et al.*, 2004), residue clash maps (Saraf and Maranas, 2003) and SVMs (Dubey *et al.*, 2005) have already demonstrated that substantial improvements over randomly created libraries can be obtained in the overall quality of the library by proactively targeting diversity.

In this paper we introduce two new protein scoring systems, S1 and S2, and the protein library optimization methods which make use of these scoring systems. These scoring systems make use of binned amino acid properties instead of using amino acid identities. This allows for the novel action of quantifying the degree of clashes in proteins, as opposed to simply identifying the number of clashes present. By identifying the degree of each clash in a protein, the alterations of the amino acid sequence can be more accurately assessed for their disruptions to the protein's functionality. Modifying the OPTCOMB (Optimal Pattern of Tiling for COMBinatorial library design) framework (Saraf *et al.*, 2005), developed recently in our group, we incorporated these new scoring functions to optimally design and subsequently assess the quality of recombination libraries. Furthermore, we also developed a corresponding framework, OPTOLIGO, to optimally design and assess mutation libraries. Collectively, the new scoring functions and optimal library design frameworks we introduce here provide an integrated framework to design functional proteins and distinguish them from non-functional ones.

In the rest of the paper we first describe the two new scoring systems, S1 and S2, followed by a discussion of the optimization formulations for the library designs. The proposed optimization formulations are applied for a Cytochrome P450 case-study and the results are contrasted against experimental data found in the literature (Otey *et al.*, 2006).

Scoring methods

Here, we introduce two new scoring systems that rely on the underlying properties of amino acids, exemplified by volume, hydrophobicity and charge, instead of amino acid identities. The key concept is that sequences with amino acid compositions yielding similar property triplets at each position are more likely to share similar functions (or lack thereof) than sequences whose amino acid compositions are more distant. The use of amino acid properties enables us to augment the original family sequence dataset by ‘filling in’ information about sequences with new amino acid compositions. We use this concept in the scoring systems (S1, S2) along with mathematical optimization to maximize the fraction of high scoring, and by extension functional, sequences in the designed library.

The volume, hydrophobicity and charge values for all 20 amino acids are given in Supplementary data and are available at *PEDS* online (Krigbaum and Komoriya, 1979; Klein *et al.*, 1984; Cid *et al.*, 1992). We use the concept of a property bin, as in the FamClash procedure developed by Saraf *et al.* (2004), to cluster amino acids with similar values for a property. The smaller the size of a bin, the fewer the amino acids that reside in the bin. In the limit, this yields bins which contain only a single amino acid, thus reverting back to solely an amino acid description. Alternatively, using too few property bins results in a coarse description that tends to group together dissimilar amino acids. We used a clustering analysis to find the best compromise between these two extremes. The bins were created in such a manner that the maximum property distance between two amino acids in a bin is less than the minimum distance between that bin and adjacent bins. Fig. 1 depicts the identified property bins for all three properties, as well as shows the bins to which the amino acids belong. Note that the amino acid volume and hydrophobicity values are not equidistant, but rather tend to predominantly cluster together. This binning procedure identifies groups of amino acids with similar property values and is property-specific. Amino acids binned together with respect to one property are not necessarily binned together for another property. For example, methionine, leucine and isoleucine are all in the same bins for volume and charge, but only methionine and leucine are in the same bin for hydrophobicity. With the amino acids separated into bins, it is possible to compare and score protein mutant and/or recombination hybrids based on the frequency of different bins, instead of focusing on the conservation of specific amino acids.

S1 scoring system

The S1 scoring system is the first of the two new scoring systems that we propose for the scoring of protein libraries using the concept of amino acid property bins. Essentially, S1 rewards amino acid choices that lead to property triplets that match the statistics of the protein family’s members. The first step in this scoring system is to structurally align, using the ClustalW software (http://align.genome.jp/), the members of the protein family whose mutation and/or recombination hybrids need to be scored. The alignment is customized for the parent proteins chosen from the protein family whose mutation and/or recombination will create the combinatorial protein library. It begins with the multiple sequence alignment of the parental proteins. Subsequently, each one of the sequences from the protein family is appended, one at a time, to the multiple sequence alignment and an alignment score is calculated. The sequences are then sorted in a descending alignment score order. Protein family sequences corresponding to distant homologues aligning poorly with the parent proteins and any repeat copies of sequences are then discarded, eliminating sequences that contribute non-useful information. The retained well-aligned sequences form the dataset for the calculation of the base statistics of the property triplets. The particular value of the cutoff score used to determine which sequences to discard will be specific to every case involved, depending on the number of sequences being aligned. It should be selected such that approximately 1 SD (63.2%) of the sequences are kept when duplicates of sequences are excluded. However, this does not need to be very precise, as the scoring system is not very sensitive to the precise number of sequences, as long as a sufficient number is provided to form a good basis.

Once the protein family dataset is aligned and screened, the frequency of each amino acid $a = 1, \ldots, 20$ at each sequence position $i = 1, \ldots, N$ is calculated. Positions with gaps are excluded from this calculation, and the amino acid frequency information is stored in parameter $A_{a,i}$ that quantifies the fraction of sequences in the family that have amino acid a at position i. This implies that $\sum_a \mathrm{A}_{a,i} = 1$ for all positions i. This parameter is in turn used to calculate the frequency distributions for all three properties at each position i. The frequency of bin b at position i for property $k = 1,2,3$ (volume, charge and hydrophobicity) is the sum of the frequencies of the amino acids at position i that belong to bin b for property k. This information is stored in parameter $F_{i,k,b}$.

The bin occupation frequencies of the protein family are next used to calculate the likelihood of a protein sharing functionality with the protein family. The amino acid composition of the sequences to be scored is encoded using the binary indicator variable $Y_{i,k,b}$, which equals 1 if the amino acid in position i belongs to bin b of property k. Figure SF1 (Supplementary data available at *PEDS* online) shows an example of the alignment of 275 DHFR sequences and the $F_{i,k,b}$ and $Y_{i,k,b}$ values for a specific residue position. Once determined, the protein family information, $F_{i,k,b}$, and protein sequence information, $Y_{i,k,b}$, can be used to score the protein. The S1 score is defined as follows:

$$\mathrm{S1} = \frac{\sum_i \sum_k \sum_b \quad \log_{10}(F_{i,k,b}) Y_{i,k,b}}{L}$$

where L is the number of amino acids in the protein being scored.

Essentially, the S1 scoring system additively, in logarithmic space, accumulates the individual scores for every position i and property k of the examined sequence. The more the positions in the examined sequence with amino acids in high frequency bins in $F_{i,k,b}$, the better the S1 score. It is important to note that S1 scores are always negative as they

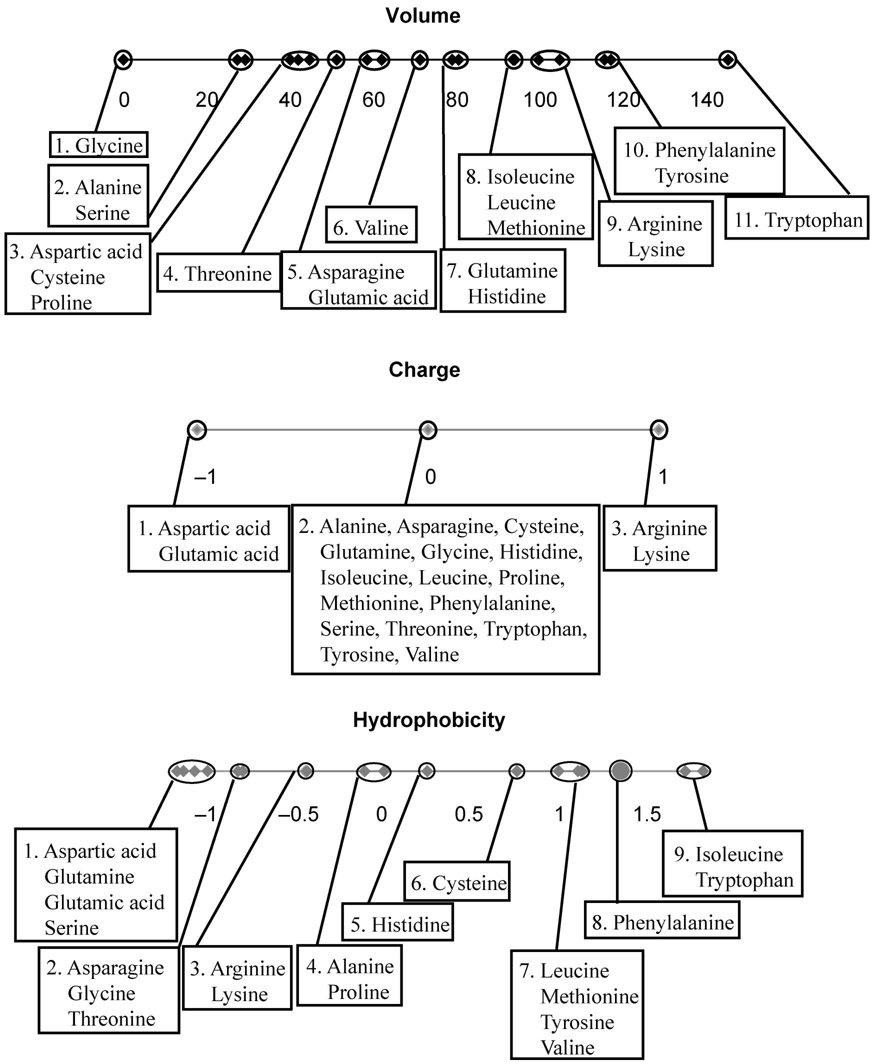
Volume
0
20
40
60
80
100
120
140
1. Glycine
2. Alanine Serine
3. Aspartic acid Cysteine Proline
4. Threonine
5. Asparagine Glutamic acid
6. Valine
7. Glutamine Histidine
8. Isoleucine Leucine Methionine
9. Arginine Lysine
10. Phenylalanine Tyrosine
11. Tryptophan
Charge
−1
0
1
1. Aspartic acid Glutamic acid
2. Alanine, Asparagine, Cysteine, Glutamine, Glycine, Histidine, Isoleucine, Leucine, Proline, Methionine, Phenylalanine, Serine, Threonine, Tryptophan, Tyrosine, Valine
3. Arginine Lysine
Hydrophobicity
−1
−0.5
0
0.5
1
1.5
1. Aspartic acid Glutamine Glutamic acid Serine
2. Asparagine Glycine Threonine
3. Arginine Lysine
4. Alanine Proline
5. Histidine
6. Cysteine
7. Leucine Methionine Tyrosine Valine
8. Phenylalanine
9. Isoleucine Tryptophan

Fig. 1. Clustering of all 20 amino acids into bins for volume, hydrophobicity and charge. The bins for each property have been circled and labeled.

are the sum of logarithms of numbers between 0 and 1. Conceptually, S1 assigns higher (less negative) scores to sequences that have amino acid compositions with property values that are similar to the ones found in the protein family. A pathological case arises when, at a given position i, the examined sequence has an amino acid whose property k value falls within a bin b that is completely unpopulated by all sequences in the protein family. We artificially assign all such instances a score of -200, which is of the same order of magnitude as the number of amino acids in most proteins. This penalizes any such occurrence in the examined sequences.

The S1 scoring system was tested on several different sets of data to assess its effectiveness. Representative results are provided for the DHFR family of proteins. Five hundred and fifty-four DHFR proteins were downloaded from the Pfam database (http://pfam.wustl.edu/index.html). We selected *Escherichia coli*, *Bacillus subtilis*, and *Lactobacillus casei* as the parental sequence set, and the members of the protein family were aligned using ClustalW (http://align.genome.jp/). Using an alignment cut-off score of 8225 and eliminating identical sequences, 275 DHFR proteins were retained to form the protein family dataset, as shown in Figure SF2 (Supplementary data available at *PEDS* online). The amino acid frequencies at each position were calculated, and using this information the $F_{i,k,b}$ parameter was populated. The S1 scores were then calculated for the 275 DHFR family sequences along with over 2500 protein sequences (not from the DHFR family) randomly selected from the protein data bank (PDB) (Berman *et al.*, 2000). The S1 score distributions for the two different protein populations, shown in Figure SF3 (Supplementary data available at *PEDS* online), are non-overlapping with the DHFR population forming a sharply peaked distribution with much higher scores than the randomly selected protein sequences. This result clearly demonstrates the ability of the S1 scoring system to isolate protein sequences sharing the same functionality from a background of randomly selected protein sequences.

Next, we examined the sensitivity of S1 to the presence of random mutations introduced in the DHFR sequences. Statistically speaking, an increasing number of random mutations in a sequence decrease its likelihood of functionality. Starting with the 275 DHFR family sequences, we created three additional protein sets. These data sets had 1, 5 and 10% mutation rates, which respectively correspond to 2, 8, and 16 mutations per sequence. The results, given in

Figure SF4 (Supplementary data available at *PEDS* online), indicate that as the random mutation level increases, the average S1 score for the dataset decreases. It is important to note that the average amino acid composition difference between any two sequences selected from the 275 top scoring DHFR sequences is 66.7% which is much higher than the imposed mutation rates. This implies that S1 is quite sensitive at detecting departures in the amino acid composition of the DHFR sequences away from the naturally occurring variance present in the protein family.

To ensure these results were not due to the DHFR family sequences acting as a 'training set,' we divided the dataset into five random, equally sized groups. Four of the groups were then used to form the basis for S1 and calculate the scores for the sequences in the fifth group. This was repeated to score each of the five groups of DHFR sequences. The scores of the DHFR sequences decreased by an average of 2.68%, and the worst change was 8.86%. In contrast, the average differences between the mutated sequences and the DHFR family scores when all 275 sequences were used as the basis were 2.98, 10.63 and 20.46% for the 2, 8 and 16 mutation datasets, respectively.

The final test of the S1 scoring system is designed to test the ability of the scoring system to discriminate between DHFR sequences and their recombination hybrids. Comparison of the S1 scores for 1, 5 and 10 crossover DHFR recombination hybrids were conducted. The results, as demonstrated in Figure SF5 (Supplementary data available at *PEDS* online), show no separation between recombination hybrids and DHFR sequences. This result is anticipated because the average amino acid composition for a given position *i* remains unchanged upon recombination between the parent proteins and the recombination hybrid population. This is because recombination does not introduce new amino acids but rather shuffles the juxtaposition of existing ones. For a scoring system to be able to distinguish between parental and recombination sequences, it is necessary to take into account not just every position in isolation but pairs of positions. This motivates the development and provides the starting point for the S2 scoring system, which is described next.

S2 scoring system

The S2 scoring system, like S1, relies on the distribution within bins of the amino acid properties for the protein family. However, unlike S1, which uses the statistics of sequence positions *one at a time*, S2 uses the statistics of *pairs of contacting residues*. By making use of pairs of positions, the S2 scoring system is sensitive to the presence of recombination events that bring in contact residues originating from different parental sequences, which have been shown to often cause disruptions (Voigt *et al.*, 2002). For every pair of contacting residue positions in the protein family, the frequency of finding any given pair of bins occupied for each property is calculated and subsequently used to determine the S2 score. Early experimentation with the S2 system revealed that using not just contacting but all residue pairs led to a weak ability to discriminate between sequences. This is because the number of non-contacting pairs is much higher than the total number of contacting pairs, whereas their information density content is lower. These extra pairs tended to partially obscure the 'signal' from only contacting pairs. This observation is consistent with the use of only contacting pairs in the popular SCHEMA (Voigt *et al.*, 2002) scoring method. In fact, the S2 scoring system can be viewed as an extension of the SCHEMA method where contacting residues originating from different parental sequences are penalized in proportion to how dissimilar their underlying pairs of property values are from the ones present in the parental sequences, thereby categorizing the degree of clashes.

Using only contacting residue pairs in the scoring system necessitates the creation of a residue contact map. Even though proteins with the same functionality can have very different amino acid sequences, the three-dimensional structure of the proteins is often highly conserved throughout the protein family, permitting one contact map to be used for all proteins without much loss of information. The creation of the residue contact map is performed by calculating residue–residue distances if the three-dimensional structure of at least one of the parent proteins is known. If no parental structure is known, then we rely on Swiss-model (http://swissmodel.expasy.org//SWISS-MODEL.html) to approximate the three-dimensional coordinates of every atom in the protein based on the homologues of known structure nearest to the parental sequences. We define the minimum distance between two amino acids as the distance between the two closest atoms in the amino acids, excluding hydrogens. Two amino acids are treated as being in contact with each other if the minimum distance between them is less than the widely employed cut-off (Voigt *et al.*, 2002) of 4.5 Å. If two amino acids $i1$,$i2$ are in contact with each other, then the corresponding binary parameter $C_{i1,i2}$ is set equal to 1.

The calculation of the S2 score proceeds in a similar fashion as in the case of S1. For all residue pairs that are in contact with each other, the frequency of finding occupied any pair of bins $b1$ and $b2$ for property k is calculated, and stored in the parameter $FF_{i1,i2,k,b1,b2}$. Figure SF6 (Supplementary data available at *PEDS* online), shows an example alignment of DHFR sequences and the charge bin pair distribution at contacting residues 34 and 115. Based on these definitions, the S2 score is calculated as follows:

$$\mathrm{S2} = \frac{\sum_{i1} \sum_{\substack{i2=i1+1 \\ C_{i1i2}=1}} \sum_{k} \sum_{b1} \sum_{b2} Y_{i1,k,b1} Y_{i2,k,b2} \log_{10}(FF_{i1,i2,k,b1,b2})}{T},$$

where T is the total number of contacting residue pairs in the protein. The S2 score is the logarithmic sum of the frequency of finding a pair of bins $b1$ and $b2$ occupied for a property k at positions $i1$ and $i2$. The more the high frequency pairs of bins from the protein family found occupied in the tested protein sequence, the higher (less negative) the S2 score is. Also, as in S1, the potential exists for a pair of bins to be present in a hybrid protein but to never occur simultaneously in the protein family. We examined a number of different values for this penalty parameter to determine the one that maximally separates the functional versus non-functional members of a set of cytochrome P450 recombination mutants (Otey *et al.*, 2006). The overlap between the S2 score distributions of functional and non-functional proteins rapidly reaches a maximum plateau when the magnitude of the penalty becomes approximately equal to -200. Therefore, we chose this value of -200 for the penalty parameter for the remainder of the analysis.

The S2 scoring system was tested by contrasting the S2 score distributions for the 275 DHFR sequences and the randomly created recombination libraries between *E. coli*, *B. subtilis* and *L. casei* involving 1, 5 and 10 crossovers, respectively. Results shown in Figure SF7 (Supplementary data available at *PEDS* online), indicate that the S2 scoring system is successful at separating wild-type DHFR proteins from recombination hybrids with random crossover locations of unknown functionality. For example, a score cut-off of -2.8, excludes only 3.64% of the wild-type DHFRs, and retains 68.4% of the single crossover hybrids, 25.8% of the five crossover proteins and only 9.45% of the 10 crossover proteins. Notably, as the total number of crossovers increases, the percent of retained sequences above the cut-off score decreases rapidly. This is consistent with the expectation that, statistically, more crossovers are likely to be detrimental for functionality. Therefore, it appears that S2 is successful at separating wild-type DHFR sequences from their recombination hybrids, presumably biasing the selection for sequences with the highest likelihood of functionality. Removing sequences from the DHFR dataset to examine their effects as a ‘training set’, as was done with S1, is not useful in this case. S2 is significantly more sensitive than S1 to changes in the sequence of a protein. As a result, the DHFR dataset gives the *appearance* of acting as a training set in this situation. However, the S2 scoring system is still detecting the presence of the crossovers in the hybrid DHFR sequences, and the more crossovers there are the worse the hybrid scores become. In the Computational Results, an examination of Cytochrome P450 proteins from the literature (Otey *et al.*, 2006) demonstrates S2’s ability to distinguish between functional and nonfunctional hybrids. In the next section, we describe how the S2 scoring system can be implemented in the context of the OPTCOMB (i.e., M1, M2) recombination library design procedures and the multiple mutation library design procedure OPTOLIGO.

Protein library optimization

OPTCOMB (M1 and M2 models)

Here, we switch focus from using S2 to score individual hybrids to optimizing an entire library of a predetermined size. Optimization variables include the location of junction points and the presence/absence of fragments from any given parental sequence at each junction point. Two separate models are described (M1 and M2), respectively, that either consider all parental fragments at every position or allow for specific fragment skipping. Both models have the same structure as those in the OPTCOMB procedure developed by Saraf *et al.* (2005) where, instead of a scoring function, the number of clashes was used to determine library fitness. We refer the reader to Saraf *et al.* (2005) for many of the details in the model derivation. Here, we highlight the parts of the model that change due to the use of the scoring function S2.

We first define two sets consisting of the aligned positions in the proteins, $i = 1, \ldots, N$, and the collection of parental proteins, $k = 1, 2, \ldots, K$. The contacting residue information described in the S2 scoring system is used to characterize which pairs of residues are in contact with one another. We then use the S2 scoring system to populate the parameter $S_{i1,i2}^{k1,k2}$, defined only over (i_1,i_2) with $C_{i1,i2} = 1$, which contains the scores of each pair of contacting residues with a different parental origin. The scores are calculated as before; however, the parameter is only populated for the pairs of contacting residues. As in the SCHEMA algorithm (Voigt *et al.*, 2002), we treat contacting residues from the same parent as being non-disruptive. Therefore, we do not include the scores of contacting pairs that originate from the same parental sequence since we are interested in pinpointing recombination locations.

Model M1: For model M1, the following additional parameters are needed—M, the total number of junctions; $L_{\min}$, the minimum length of a fragment; $L_{\max}$, the maximum length of a fragment.

The following variables are used to describe whether a particular sequence location i is a junction point:

$$Y_i = \begin{cases} 1 & \text{if a junction occurs at residue}\, i \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{i1,i2} = \begin{cases} 1 & \text{if there exist at least one junction} \\ & \text{between residues}\, i1 \text{ and}\, i2 \\ 0 & \text{otherwise.} \end{cases}$$

The objective function of the M1 model formulation is:

$$\text{maximize} \sum_{i1=1}^{N-1} \sum_{\substack{i2=i1+1 \\ C_{i1,i2}=1}}^{N} \sum_{k1=1}^{K} \sum_{k2 \neq k1}^{K} S_{i1,i2}^{k1,k2} Z_{i1,i2}. \qquad (1)$$

This function maximizes the overall score of the library by accounting for all possible fragment combinations. This is accomplished by scoring additively all fragment pairs originating from different parents with at least one junction between them. Constraint (2)

$$\sum_{i=1}^{N} Y_i = M, \qquad (2)$$

sets the sum of all junctions equal to the target M. Inequality (3)

$$\sum_{i=i'}^{i'+L_{\min}-1} Y_i \leq 1, \quad \forall i = 1, \ldots, N - L_{\min} + 1, \qquad (3)$$

ensures that no oligomer is shorter than the desired minimum fragment length of $L_{\min}$. For all positions i starting from the first to $L_{\min} - 1$ before the terminal position, the sum of the number of junctions between that position and the position which is $L_{\min}$ amino acids away can be no greater than one. Thus, if position i' is a junction, then the $L_{\min} - 1$ positions after i' cannot be a junction. Similar to inequality (3), constraint (4) ensures that the maximum desired fragment length, $L_{\max}$ is not exceeded.

$$\sum_{i=i'}^{i'+L_{\max}-1} Y_i \geq 1, \quad \forall i = 1, \ldots, N - L_{\max} + 1. \qquad (4)$$

For all positions i starting from the first to $L_{\max} - 1$ positions from the last, the sum of the number of junctions from any position i to $L_{\max} - 1$ positions ahead must be at least equal to one. This guarantees that no fragment is longer than $L_{\max}$. Equality (5) prevents the placement of a junction less than $L_{\min}$ amino acids from the terminal position by forcing the sum of the number of junctions within that range equal to zero.

$$\sum_{i=N-L\min+2}^{N} Y_i = 0, \tag{5}$$

Inequality (6) is used to determine which contacting pairs have junctions between them.

$$Z_{i1,i2} \geq Y_i, \quad i = i_1 + 1, \ldots, i_2 \tag{6}$$

If any position i between a contacting pair is a junction, then $Z_{i1,i2}$ must assume a value of at least one. Inequality (7) ensures that contacting pairs that do not have a junction between them are not included in the S2 score calculation.

$$Z_{i1,i2} \leq \sum_{i=i1+1}^{i2} Y_i \quad i_1, i_2 = 1, \ldots, N \\ \text{with } i_1 < i_2 \text{ and } C_{i1,i2} = 1 \tag{7}$$

This is accomplished by making $Z_{i1,i2}$ no greater than the sum of the junctions between the start and the end of a contacting pair.

$$Y_1 = 1, \tag{8}$$

Finally, equality (8) simply ensures that the first position in the protein is treated as a junction to flag the beginning of the first fragment. Also, the upper and lower bounds of $Z_{i1,i2}$ are set equal to one and zero to ensure that the above constraints can only assign integral values to Z. Yi is defined as a binary variable.

Collectively, Eqs. (1)–(8) define the formulation of the M1 model which belongs to the class of mixed-integer linear (MILP) optimization problems. The junctions are placed in such a manner that they maximize the overall S2 score of the library while still meeting all the constraints imposed by Eqs. (2)–(7).

Model M2: M2 departs from M1 by allowing for parental sequence-dependent fragment skipping at certain locations. Fig. 2 shows this difference between M1 and M2, with M1 being option *a* and M2 option *b*. By not including in the recombination mixture certain parental fragments contributing poor S2 scores, the M2 model explores whether the overall S2 library score can be substantially increased.

Model M2 retains the same set definitions described for M1 in addition to the parameters $L_{\min}$ and $L_{\max}$ and the binary variable Y_i. Since fragments can now be excluded, the library size is no longer simply the number of parents raised to the power of the number of fragments. A new parameter *LS* is needed that directly defines the desired library size. Variable $Z_{i1,i2}$ is generalized to $Z_{i1,i2}^{k1,k2}$ to reflect the fact that the presence or absence of a junction between any two sequence positions is now a function of the parental sequences, $k1$ and $k2$, examined. The following new variables were also added to M2 to model the additional complexity associated with fragment skipping:

$$w_{i,k} = \begin{cases} 1 & \text{if at position } i \text{ a fragment from parent } k \text{ exists} \\ & \text{in the protein library} \\ 0 & \text{otherwise} \end{cases}$$

$$y_{i,k} = \begin{cases} 1 & \text{if at position } i \text{ a fragment from parent } k \text{ begins} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,k} = \begin{cases} 1 & \text{if exactly } k \text{ parents are contributing at junction } Y_i \\ 0 & \text{otherwise.} \end{cases}$$

In the M2 model formulation, the objective function remains essentially unchanged from M1, with the only exception of using $Z_{i1,i2}^{k1,k2}$ instead of $Z_{i1,i2}$. This means that the objective function of M2 takes the form of:

$$\text{maximize} \sum_{i1=1}^{N-1} \sum_{\substack{i2=i1+1 \\ C_{i1,i2}=1}}^{N} \sum_{k1=1}^{K} \sum_{k2 \neq k1}^{K} S_{i1,i2}^{k1,k2} Z_{i1,i2}^{k1,k2}. \tag{9}$$

Equations (3), (4), (5) and (8) remain unchanged. The first additional constraint in M2 relates Y_i with $y_{i,k}$:

$$Y_i \geq y_{i,k}, \quad i = 1, \ldots, N \text{ and } k = 1, \ldots, K \tag{10}$$

This inequality is used to ensure that a fragment from any parent can only begin at a position that is a junction:

$$Y_i \leq \sum_{k1=1}^{3} y_{i,k}, \quad i = 1, \ldots, N \text{ and } k = 1, \ldots, K \tag{11}$$

Inequality (11) guarantees that there is no section of the protein that is left without at least one parent contributing a fragment at that location. Constraints (12–15) collectively ensure that variable $w_{i,k}$, assumes the proper value according to its definition given values for variables $y_{i,k}$ and Y_i.

$$\left.\begin{array}{l} w_{i+1,k} \leq w_{i,k} + y_{i+1,k} \\ w_{i+1,k} \leq 1 - Y_{i+1} + y_{i+1,k} \\ w_{i+1,k} \geq w_{i,k} - Y_{i+1} + y_{i+1,k} \\ w_{i,k} \geq y_{i,k} \end{array}\right\} \begin{array}{l} i = 1, \ldots, N \\ k = 1, \ldots, K \end{array} \qquad \begin{array}{r} (12) \\ (13) \\ (14) \\ (15) \end{array}$$

Specifically, constraint (12) makes sure that it is only possible for a fragment from parent k to exist at position $i + 1$ if a fragment from parent k existed at position i or if a fragment from parent k begins at position $i + 1$. Constraint (13) ensures that if position $i + 1$ is a junction point then parent k can only exist at that position if a fragment from parent k is starting at that position. Constraint (14) quantifies that if position i has a fragment from parent k then the next position $i + 1$ will also have a fragment from parent k unless there is a junction at $i + 1$. If there is a junction at position $i + 1$, than Equation (15) guarantees that if a fragment from parent k 'begins' at a position i then the same fragment must also 'exist' at the same position in the library. The next set of constraints relate the imposed library size, *LS*, to the

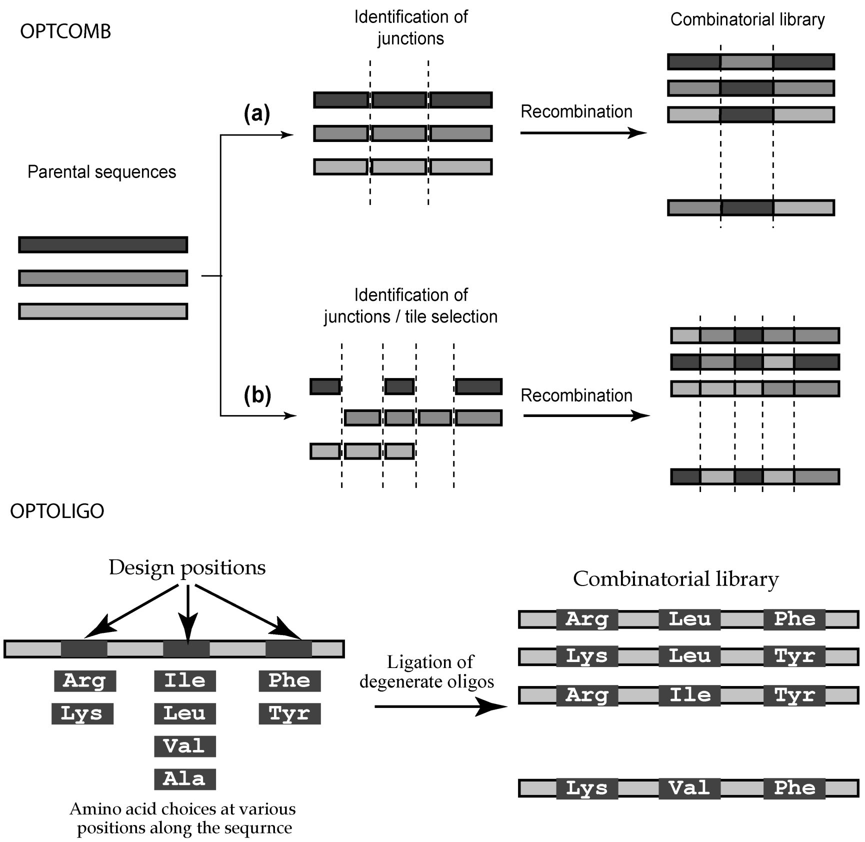

Fig. 2. The OPTCOMB and OPTOLIGO combinatorial library creation models. In options *a* (M1 model) and *b* (M2 model) junction points are selected and the parent sequences are recombined without and with fragment skipping, respectively. In OPTOLIGO, specific design locations are mutated to create the combinatorial library.

remaining variables.

$$\left.\begin{aligned} \sum_{k=1}^{3} y_{i,k} &= \sum_{k=1}^{3} kN_{i,k} \\ \sum_{k=1}^{3} N_{i,k} &= Y_i \\ \log_{10}(LS) &\le \sum_{i}^{N}\sum_{k}^{3} N_{i,k}\log_{10}(k) \end{aligned}\right\} \begin{array}{l} i=1,\ldots,N \\ k=1,\ldots,K \end{array} \qquad \begin{array}{r} (16) \\ (17) \\ (18) \end{array}$$

Inequality (16) forces the correct $N_{i,k}$ binary variable to assume a value of 1 at all positions of i. For example, if all three parents contribute a fragment at residue 49, a junction point, then $N_{49,1}=0$, $N_{49,2}=0$ and $N_{49,3}=1$. Inequality (17) makes sure that only one $N_{i,k}$ variable assumes a value of 1 at each junction. Finally, constraint (18) relates the library size parameter LS to the $N_{i,k}$ variables in logarithmic space to preserve the linearity of the model. The next group of constraints assigns the proper value to $Z_{i1,i2}^{k1,k2}$ which encodes the set of contacting pairs with a junction in between to be included in the S2 calculation.

$$\left.\begin{aligned} Z_{i1,i2}^{k1,k2} &\ge w_{i1,k1} + w_{i2,k2} + Y_i - 2 \\ Z_{i1,i2}^{k1,k2} &\le \sum_{i=i1+1}^{i2} Y_i \\ Z_{i1,i2}^{k1,k2} &\le w_{i1,k1} \\ Z_{i1,i2}^{k1,k2} &\le w_{i2,k2} \end{aligned}\right\} \begin{array}{l} i1, i2 = 1,\ldots,N \\ i1 < i2 \\ C_{i1,i2} = 1 \\ k = 1,\ldots,K \end{array} \qquad \begin{array}{r} (19) \\ (20) \\ (21) \\ (22) \end{array}$$

In essence, constraints (19–22) recast in an equivalent linear form the trilinear term $w_{i1,k1}\, w_{i2,k2}\, Y_i = Z_{i1,i2}^{k1,k2}$. It implies that $Z_{i1,i2}^{k1,k2}$ assumes a value of one only if both positions i_1 in parent k_1 and position i_2 in parent k_2 are contributing fragments and also there is a junction between positions i_1 and i_2. Finally, constraint set (23) identifies variables Y_i, $y_{i,k}$ and $N_{i,k}$ as binary and $Z_{i1,i2}^{k1,k2}$and $w_{i,k}$ as continuous. Both the continuous variables have their upper and lower bounds defined to be 1 and 0, respectively.

Collectively, equations (3)–(5), (8)–(23) define model M2, which, as in the case of M1, is an MILP. Because of the additional complexity associated with the flexibility of skipping parental fragments model M2 is typically more time consuming to run.

OPTOLIGO

Recombination of parental sequences is a conservative method of sampling sequence space as it simply rearranges the already present genetic diversity. New genetic diversity can be accessed through protocols that rely on the accumulation of point mutations to a single sequence. Given the popularity of such library generation protocols, we designed the OPTOLIGO optimization model that uses the S2 scoring system to determine the type of optimal amino acid substitutions for a given library size. The challenge here is to identify which residue positions should be allowed to mutate and also what mutations to permit for each location. Because of the exponential growth of the library size for an increasing repertoire of mutation choices the mutation sites and choices must be judiciously chosen. OPTOLIGO maximizes the average S2 library score by summing over contacting amino acid combinations present in the library. The key decision variable here is $Y_{i,a}$ which is equal to 1 if amino acid a is allowed at position i and 0 otherwise. Additional considerations include excluding the presence of two very similar amino acids for the same position or even constraints on the overall diversity of the library.

Conceptually, OPTOLIGO uses the S2 scoring system to evaluate and determine the optimal amino acid choices at select design positions. OPTOLIGO allows the user to define positions that can be mutated, as well as the size of the library desired. To accomplish this, OPTOLIGO makes use of three sets: the number of aligned positions in the protein, $i = 1, \ldots, N$; the amino acid alphabet, $a = 1, \ldots, 20$; and the maximum number of amino acid substitutions permitted at each design position, $k = 1, \ldots, K$. Several parameters are used in the formulation of the model.

$S^{a1,a2}_{i1,i2} = S2$ score for amino acid pair $a1$, $a2$ at positions $i1$, $i2$, respectively; $LS =$ library size.

$$y^{o}_{i,a} = \begin{cases} 1 & \text{if amino acid } a \text{ is present at non-design position } i \\ 0 & \text{otherwise.} \end{cases}$$

Parameter y^{o}_{ia} defines the original sequence used as a template before adding mutations to form the library. It can be either a specific wild-type sequence or simply the consensus sequence generated from a protein family. Other parameters used in the formulation include:

$$C_{i1,i2} = \begin{cases} 1 & \text{if positions } i1\text{, and } i2\text{, are in contact} \\ 0 & \text{otherwise} \end{cases}$$

$$d_i = \begin{cases} 1 & \text{if position } i \text{ is a design position} \\ 0 & \text{otherwise.} \end{cases}$$

Parameter $C_{i1,i2}$, as before, defines the contacting residues in the protein estimated from structural information of the template or any other protein homologue. Parameter d_i denotes the positions that are allowed to mutate. The number of positions that can be mutated must be kept small or the library size quickly makes the model extremely computationally intensive. Typically, the most variable positions in the protein family and/or positions close to the active site are chosen as design positions.

Binary variable $y_{i,a}$, denotes whether amino acid a can be present at position i. Binary variable $N_{i,k}$ identifies the number of different amino acids selected for design position i. It is equal to 1 if exactly k amino acids are selected for position i, and 0 otherwise. Variable $w^{a1,a2}_{i1,i2}$ is equal to the product of $y_{i1,a1}\ y_{i2,a2}$. The above defined parameters and variables are used to construct the OPTOLIGO formulation.

The objective function of OPTOLIGO is:

$$\text{Maximize} \sum_{i1=1}^{N-1} \sum_{\substack{i2=i1+1 \\ C_{i1,i2}=1 \\ d_{i1}+d_{i2}\geq 1}}^{N} \sum_{a1=1}^{20} \sum_{a2=1}^{20} S2^{a1,a2}_{i1,i2} w^{a1,a2}_{i1,i2} \tag{24}$$

This objective function maximizes the sum of the S2 scores of all positions that are in contact with at least one design position. This function is subject to the following constraints.

$$y_{i,a} = y^{o}_{i,a} \quad i = 1, \ldots, N \text{ and } a = 1, \ldots, 20 \\ \text{with } d_i = 0 \tag{25}$$

Constraint (25) ensures that non-design positions are assigned the correct amino acid choices.

$$\left. \sum_k kN_{i,k} = \sum_a y_{i,a} \right\} i = 1, \ldots, N \tag{26}$$

$$\sum_k N_{i,k} = 1 \tag{27}$$

Equalities (26) and (27), as described before for model M2, ensure that N, encodes the correct number of residues k for every position i.

$$\log(LS) \leq \sum_{i1=1}^{N-1} \sum_{\substack{k=1 \\ d_{i1}\neq 0}} N_{i,k} \log(k) \tag{28}$$

Inequality (28) uses the assigned values of z from equalities (26) and (27) to enforce the minimum desired library size. This constraint effectively ensures that the library size, which is the product of the number of amino acid selections at each position, must be at least the size of the desired library

$$\left. \begin{aligned} & w^{a1,a2}_{i1,i2} \leq y_{i1,a1} && (29) \\ & w^{a1,a2}_{i1,i2} \leq y_{i2,a2} && (30) \\ & w^{a1,a2}_{i1,i2} \geq y_{i1,a1} + y_{i2,a2} - 1 && (31) \end{aligned} \right\} \begin{aligned} & i1, i2 = 1, \ldots, N \\ & d_{i1} + d_{i2} \geq 1 \\ & C_{i1,i2} = 1 \\ & a1, a2 = 1, \ldots, 20 \end{aligned}$$

Inequalities (29) through (31) linearize exactly the product of $y_{i1,a1}$ and $y_{i2,a2}$.

Constraints (24)–(31) define the core requirements for the OPTOLIGO optimization model. New constraints can be appended to reflect additional requirements. For example, certain amino acids involve very similar property values (e.g. Met and Leu). This implies that OPTOLIGO will preferentially retain both amino acids to meet library size requirements, instead of expanding the amino acid usage repertoire. This can be prevented from happening by adding constraints of the following form: $y_{i,\text{methionine}} + y_{i,\text{leucine}} \leq 1$. Furthermore, one could impose requirements on the overall library diversity using similarity scores such as BLOSSUM and constraining the overall library diversity to be above a given cut-off.

Computational results

The effectiveness of both the OPTCOMB and OPTOLIGO library optimization procedures and the S2 scoring system are benchmarked by applying them to the design of combinatorial libraries of Cytochrome P450 proteins. The obtained computational predictions are contrasted against the results from the comprehensive study by Otey *et al.* (2006), where the SCHEMA algorithm was used to minimize clashes within a protein library generated from three parents with a total of eight fragment sections. We used the same parental sequences CYP102A1, CYP102A2 and CYP102A3 proteins and library size.

First, the members of the cytochrome P450 protein family were aligned one at a time to the three parent sequences. After the alignment process had been completed, the top 1001 aligning proteins, as well as the three parent sequences, were used to create the protein family used to calculate the S2 score. A contact map was subsequently generated using Swiss-Model (http://swissmodel.expasy.org//SWISS-MODEL.html) and the sequence of parent CYP102A2. For both the alignment process and the contact map, only residues 6–449 (on CYP102A2) were considered because of the poor alignment of the parent and family proteins for the terminal sections. With the protein family alignment and contact map prepared, sufficient information was available to score Cytochrome P450 proteins and compare the results against the SCHEMA scoring method.

We used as a basis of comparison the cytochrome P450 protein library of 6561 hybrid proteins developed by Otey *et al.* (2006). Out of 6561 hybrids, they sequenced 955 spanning both functional (620) and non-functional (335) sequences. Fig. 3 shows the distribution of SCHEMA scores for the two populations. The two populations are overlapping significantly. Fig. 4 depicts the S2 scores for the same proteins. We see that the S2 scoring system does a better job in this case of separating the population of functional versus

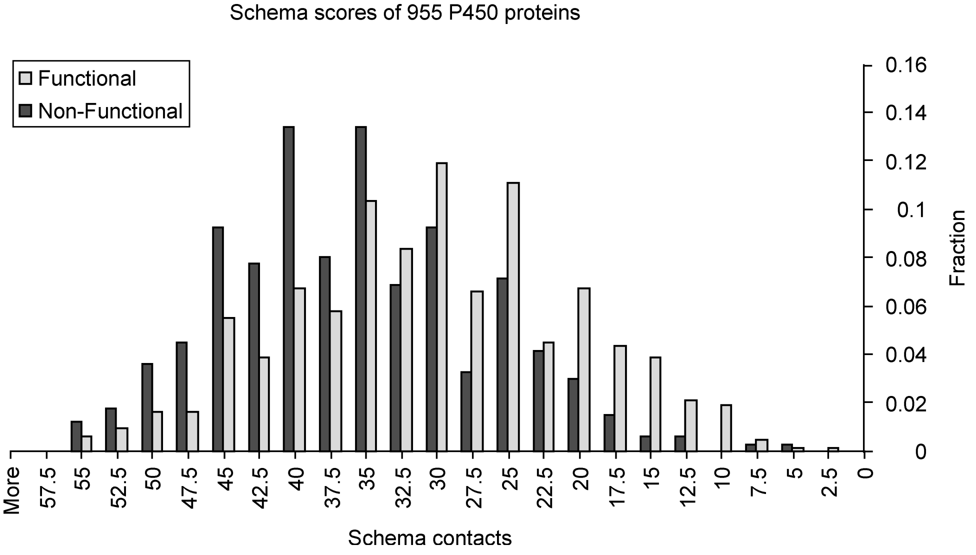

Fig. 3. These are the SCHEMA results of the 955 proteins examined by Otey *et al.* There is an overlap of 74.0% between the two groups, and using a cut-off of 30 leads to a library that is 77.0% functional proteins, and contains 54.5% of all the functional proteins.

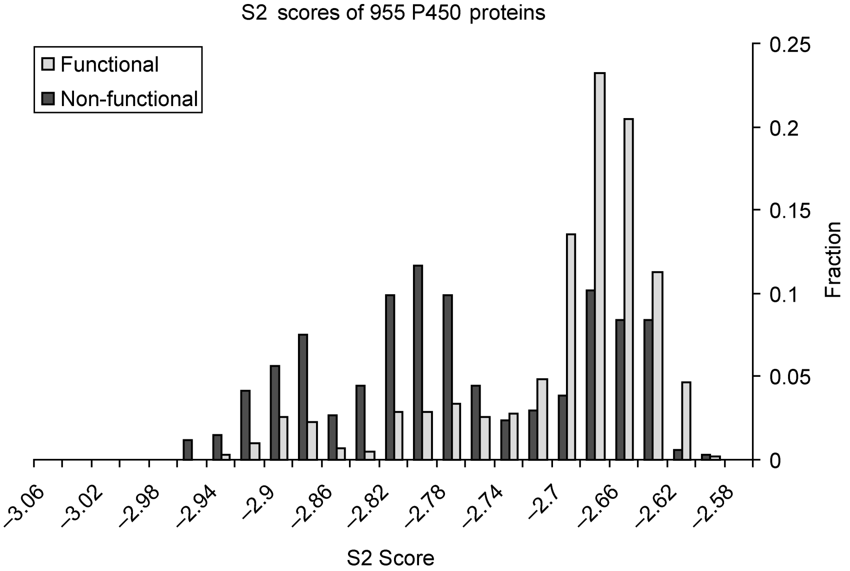

Fig. 4. The S2 score for the same 955 proteins scored by SCHEMA in *a* shows a much better separation between the two groups of proteins. There is only a 55.91% overlap, and using a cut-off of −2.74 leads to a library that is 80.2% functional proteins and contains 81.0% of all the functional proteins.

non-functional members. Specifically, a cut-off of −2.74 leads to a library that has 80.2% functional members while retaining 81.0% of the functional proteins.

The S2 scoring system is next used to score each one of the contacting pairs in the parental proteins for all six combinations resulting by enumerating all two at a time parental combinations. This provided the necessary information for the M1 and M2 models.

OPTCOMB results

We first tested the M1 model, as it yields recombination protein libraries with no fragment skipping like the protocol used by Otey *et al.* (2006). Minimum and maximum fragment lengths of 40 and 60 were used in accordance with experimental constraints. These upper and lower bounds on fragment lengths imply that the 444 amino acid long portion of the sequence affords a minimum of 8 and a maximum of 11 fragments in the generation of the protein library. In response to this, M1 was used to generate four protein libraries with 8, 9, 10 and 11 crossovers, respectively, thus covering all feasible fragment length combinations. Fig. 5 shows pictorially the four libraries that maximize the total S2 score using fragment sizes between 40 and 60 amino acids. It is evident that the location of the junction points are not equidistant with certain junction positions (e.g. 370, 410, etc.) reoccurring in most libraries independent of size. Fig. 6 plots the overall score as a function of number of crossovers and thus library size. Interestingly, the S2 scores worsen as the total number of fragments increases. That is because in this example additional junctions end up further fragmenting the parental sequences thus bringing in contact more residues originating in different parental sequences. As the number of junctions increases, it becomes increasingly difficult for M1 library designs to avoid having poor scoring S2 combinations interrupted by a junction.

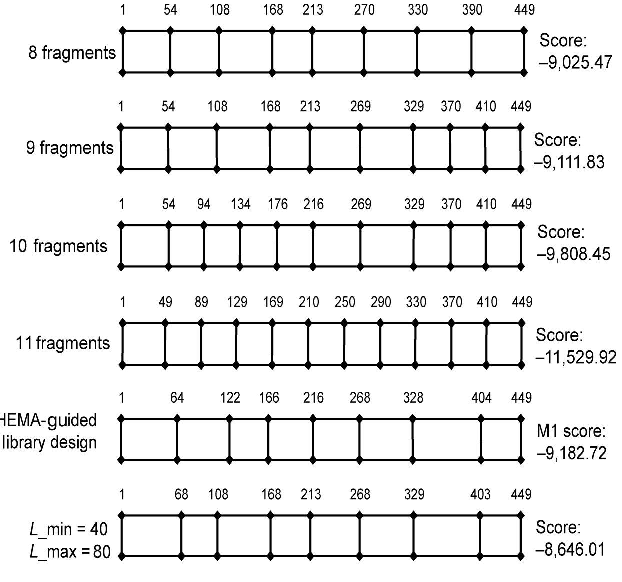

Fig. 5. These are junctions used in the six protein libraries. Each library contains the number of proteins equal to three raised to the power of the number of fragments. The first four libraries were generated by M1 using fragment lengths of 40–60 amino acids. The fifth library was generated by SCHEMA-guided recombination by Otey *et al.* The sixth library was generated by M1, using the same fragment sizes and number of fragments as used in the SCHEMA-guided recombination. All junction points use the actual numbering of the residues in the proteins, so the first amino acid is the actual first amino acid of the protein instead of the first aligned protein (which is the sixth).

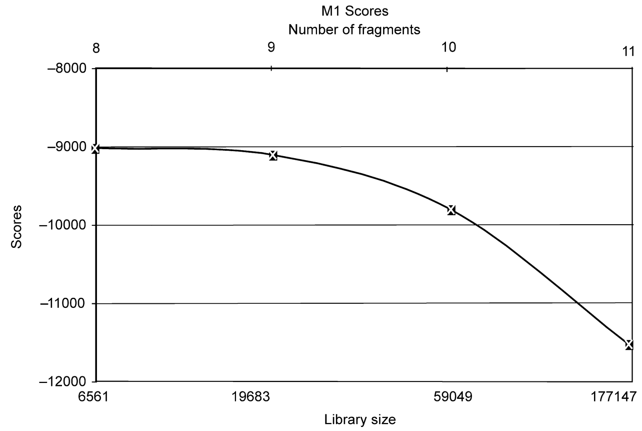

Fig. 6. Shown is a graphical representation of the S2 scores of the four M1-based protein libraries generated using fragment lengths of 40 and 60. The number of fragments used to generate the library is shown across the top of the graph, while the corresponding library size (equal to three raised to the number of fragments) is depicted across the bottom. As the number of fragments increases, the score of the library generated worsens.

Next we compare the junctions generated by SCHEMA in the Otey *et al.* work against the M1 model predictions. Because the maximum fragment length in the junctions generated by SCHEMA exceeded our imposed maximum of 60 amino acids we proceeded to adjust our upper limit to 80 amino acids for consistency in the comparison. Fig. 5 depicts the junctions generated by SCHEMA and model M1 along with the S2 scores for the two designs. The M1-based design yields a score improvement of 5.84% despite the small differences between the two designs. Interestingly, even when revisiting the more stringent constraint of a maximum fragment length of 60, an improvement of 1.71% in the S2 score is observed despite the presence of one extra crossover. In this example, we found that the differences between the M1 and SCHEMA designs are not dramatic, which alludes to the common element (i.e. use of contacting residues) between the two scoring methods. These differences are amplified when the M2 model is considered.

To facilitate the application of the M2 model we first used M1 in an iterative fashion to pre-identify all promising junction points for different parental combinations and numbers of fragments. Specifically, the M1 model was run for all three pair-wise combinations (i.e. 1–2, 1–3 and 2–3) and all three simultaneously of the three available parents for 8, 9,10 and 11 junctions, for a total of 16 runs. The identified junction points are shown in Figure SF8 (Supplementary data available at *PEDS* online). We subsequently used only these 'promising' junction locations to restrict the placement of junction points in model M2 so as to reduce computational requirements. Five separate M2 libraries were constructed, one for each junction set generated by M1 for fragment numbers of 8, 9, 10 and 11, respectively, as well as a fifth one using all the junctions generated by all four sets of fragments. In all M2 runs, a minimum library size requirement of 3^8, approximately 6500, was imposed to match the size obtained by Otey *et al.* (2006). The actual library sizes generated were in some cases somewhat larger, depending on how many junctions were present and how many fragments were excluded. The first library (eight fragments) exactly conserves the M1 results, since the minimum library size requirements prevent any fragment skipping. Library designs with 9, 10 and 11 junctions allow for skipping of certain fragments while still satisfying the same minimum library size requirement of 6500. No obvious/persistent trends are observed in the identified tiling patterns, implying that fragments' size limits rather than pair-wise S2 scores are the dominant factors in the obtained library designs. Notably, the overall S2 score progressively improves over the M1 design with the same number of junctions. This improvement is as much as 60.4% when all 11 junction positions are considered. This result alludes to the fact that fragment skipping can in principle be a powerful library design tool for improving functionality. These initial M2 results are shown in Figure SF9 (Supplementary data available at *PEDS* online).

Once the initial round of M2 runs had been completed, we explored whether improved solutions can be obtained by allowing the identified junction locations to shift one position to the left or right. This process was iteratively repeated until no additional junction location shifts were observed. We found that some junction points for the libraries with 9 and 10 junction points shifted after the first iteration but none did during the second. No junctions shifted for the other two libraries. The final results generated for each fragment size by M2 are given in Fig. 7. Figure 8 shows a comparison of the S2 scores before and after junction shifting for the four libraries. It is worthwhile to note that while for M1-based designs the overall S2 score decreases as the number of junctions increases, M2 designs follow the opposite trend. The reason for this seemingly inconsistent behavior is that while in the case of M1 increasing the number of junctions yields a corresponding increase in library size, in the case of M2 the increase of the number of junctions does not yield an increase in library size as it is offset by fragment skipping. The M2 model, by judiciously excluding poor fragment combinations, enables the systematic improvement of the overall S2 library score.

OPTOLIGO results

To examine the effectiveness of the OPTOLIGO model, we tested it on the Cytochrome P450 family of proteins. Using the same protein family alignment and information used to test the M1 and M2 models, we created a consensus sequence using the family sequence data. We then determine the 10 most variable positions to be Arg 25, Thr 66, Gly 69, Asn 161, Ser 177, Phe 181, Leu 184, Arg 185, Asp 191 and His 441 (the amino acids are the ones present in the consensus sequence). We also examined the properties of the amino acids to determine which pairs of amino acids should not be permitted to be substituted at the same design positions. A pair of amino acids was eliminated if they had all three properties in adjacent or identical bins. Table I gives the pairs of amino acids whose simultaneous presence is precluded.

OPTOLIGO was run for library sizes of 10^3, 10^4, 10^5, 10^6 and 10^7. The results revealed a number of interesting trends. Notably, the optimal library design seems to be hierarchical in nature. With only a handful of exceptions as the library size increases previous choices are retained and new ones are simply successively appended (Fig. 9). The only exception was that methionine and leucine were substituted for one another in consecutive size libraries. Interestingly, positions with the same wild-type amino acids do not necessarily end

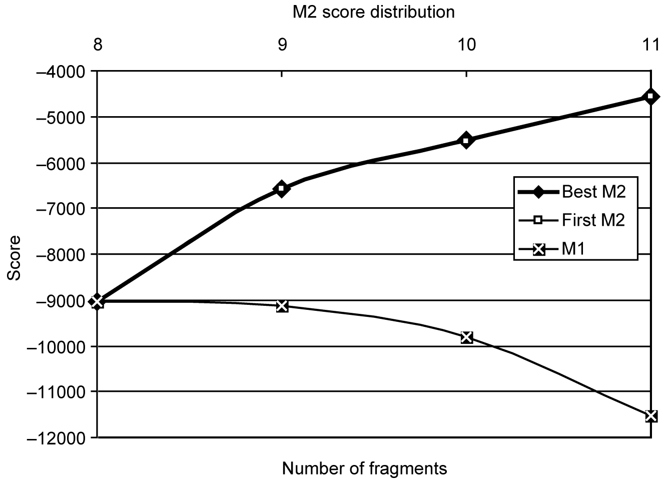

Fig. 8. These are the scores generated by the best M2 runs the initial M2 runs, and the M1 runs for each fragment size. While the M1 scores get worse as the number of fragments increases, the M2 scores improve. Also, minimal changes were generated by allowing the original solutions to perturb slightly; the two lines are essentially indistinguishable from one another.

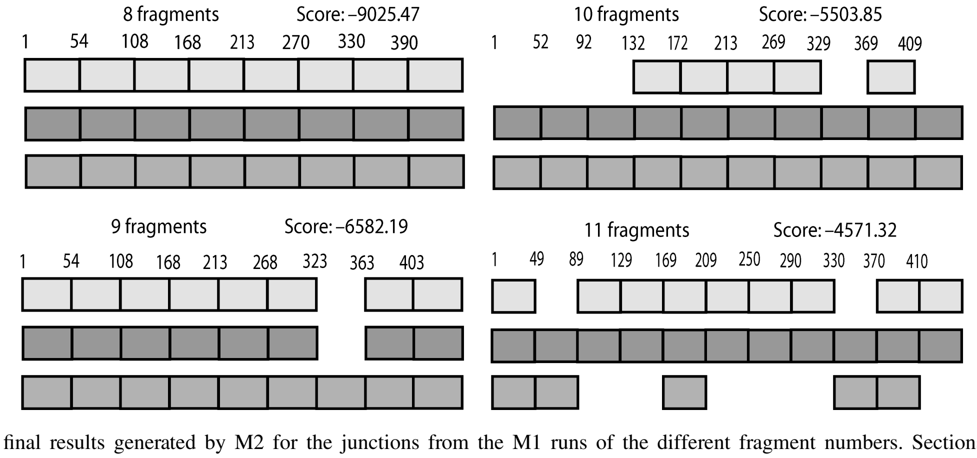

Fig. 7. These are the final results generated by M2 for the junctions from the M1 runs of the different fragment numbers. Sections where a parent is not contributing are excluded in the picture as well. The sizes of the libraries generated are 6561 for the 8 and 9 fragment libraries, 7776 for the 10 fragment library and 6912 for the 11 fragment library. The best overall solution, generated using all the junction points permitted in any of the libraries is identical to the 11 fragment solution. The labeled junction points are the actual positions in the protein as opposed to the aligned positions, which start at residue 6.

Table I. This is a list of the pairs of amino acids that were not permitted to simultaneously be substitutions in the OLIGOPT model

Eliminated pairs	
Alanine	Proline
Arginine	Lysine
Asparagine	Glutamic acid
Asparagine	Threonine
Aspartic acid	Serine
Aspartic acid	Threonine
Glutamic acid	Threonine
Glycine	Serine
Leucine	Methionine
Phenylalanine	Tryptophan
Phenylalanine	Tyrosine

They were eliminated because they have all 3 bins within +/−1 of one another.

up with the same redesigns. For example, in the consensus sequence, residue 25 is Arg and residue 191 is Asp. Neither of those amino acids is ever used in any OPTOLIGO protein library at those locations. Also, as the library size increases the overall library score decreases in accordance with the use of increasingly penalizing residues.

We also applied OPTOLIGO to the recently published green flourescent protein (GFP) library results (Treynor *et al.*, 2007) to contrast our findings. Unfortunately there are only 132 GFP sequences available in Pfam. After the alignment process, removing duplicate sequences, and retaining 63.2% of the sequences, only 83 sequences were retained to form the basis for S1 and S2. The size of this dataset is sufficient for S1 but not for S2. We did generate the OPTOLIGO results for a library of size 2^9, and the results were: P58D, T59I, T62A, T63A, F64L, T65S, V68I, Q69L and S72A. These results avoided the Q69R substitution that the paper pointed out as problematic while proposing the beneficial Q69L substitution. As expected, the S2 scores for the proteins from the DBIS^{ORBIT} and C^{ORBIT} libraries perform rather poorly because the V61L and T65A substitutions represent changes to amino acids that are never present in any of the 83 sequences used to create our basis.

Discussion and summary

In this paper, we have introduced two new protein scoring functions, S1 and S2, and corresponding library optimization techniques, OPTCOMB and OPTOLIGO, which make use of these scoring systems. S1 and S2 represent improvements over currently existing scoring functions because they can quantitatively identify the degree of clashes in hybrid and mutant proteins. All current scoring systems simply count the number or clashes present in a protein, and score it accordingly. S1 and S2 are the first scoring systems which can accurately predict the degree to which an individual clash will be detrimental to the functionality of a protein.

S1 and S2 accomplish this innovative feat by making use of binned amino acid properties. Each of the 20 amino acids was placed in a bin with other amino acids sharing similar values for each of the three properties of volume, charge and hydrophobicity. This allows S1 and S2 to identify clashes from changes in the amino acid property bins for a given position or pair of contacting positions, instead of being confined to using the amino acid identities. Furthermore, using the statistics of top aligning protein family members, S1 and S2 calculate the frequency with

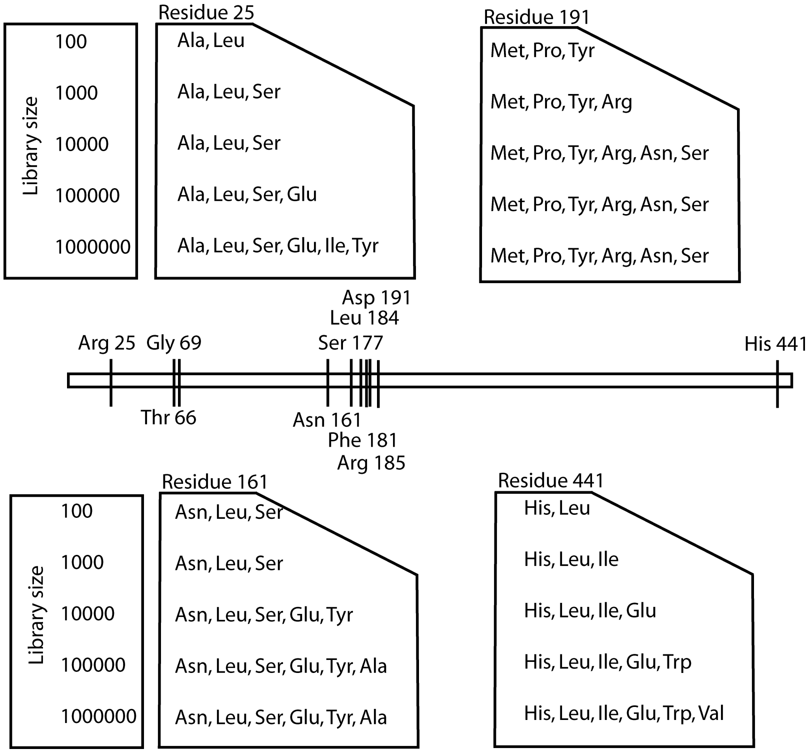

Fig. 9. These are the results for four of the 10 mutated residues used in OPTOLIGO. The substitution pattern is conservative from smaller to larger libraries.

which a particular property bin or pair of bins occurs. This information is in turn used to quantify the degree of the penalty for clashes in proteins. The two scoring systems generally need to be used in conjunction with one another. S1 is useful for identifying whether or not a particular protein belongs to a given protein family. S2, which is significantly more sensitive to changes in the amino acid sequence of a protein away from the statistics of the protein family, can then be used to tell whether a hybrid or mutant protein is functional or not. As shown in Figs. 3 and 4, the S2 scoring system is more effective in this case at distinguishing between functional and non-functional proteins than the currently employed SCHEMA algorithm.

The library optimization technique OPTCOMB uses scores derived from the S2 scoring system. OPTCOMB has two different formulation models, M1 and M2, each of which designs recombination protein libraries. M1 uses the S2 scores to determine the optimal locations to recombine parent proteins to maximize the overall score of the protein library. M2 expands on this application by not only determining where recombination junctions should be placed, but also by selecting which parents will contribute fragments at each location. This difference allows for the potential to skip parent fragments, potentially eliminating sections of one parent protein that cause many clashes with fragments from other proteins.

The OPTOLIGO optimization model is designed to create protein libraries through accumulated point mutations, as opposed to the recombination techniques of OPTCOMB. The user designates the number of design positions they desire, along with the minimum library size, number of substitutions permitted at each location and which amino acids cannot be substituted at the same locations. OPTOLIGO then uses the S2 scoring system to determine the optimal amino acid mutations to create a library of the desired size. As the size of the library is increased, the results from smaller library sizes are conserved with new additions added on to reach the desired library size. This is in comparison with M2, which shows no consistent tiling patterns, and M1, where the junction locations are forced to move around in order to accommodate the growing number of required junctions.

Acknowledgements

We gratefully acknowledge support from the National Science Foundation Grant BES-0331047 and also an REU in Biomolecular Engineering award to RJP (NSF Grant EEC-0353569).

References

An,Y., Ji,J., Wu,W., Lv,A., Huang,R. and Wei,Y. (2005) *Appl. Microbiol. Biotechnol.*, **68**, 774–778.

Baik,S.H., Ide,T., Yoshida,H., Kagami,O. and Harayama,S. (2003) *Appl. Microbiol. Biotechnol.*, **61**, 329–335.

Berman,H.M., Westbrook,J., Feng,Z., Gilliland,G., Bhat,T.N., Weissig,H., Shindyalov,I.N. and Bourne,P.E. (2000) *Nucleic Acids Res.*, **28**, 235–242.

Chen,Z. and Zhao,H. (2005) *J. Mol. Biol.*, **348**, 1273–1282.

Chockalingam,K., Chen,A., Katzenellenbogen,J.A. and Zhao,H. (2005) *Proc. Natl. Acad. Sci.*, **102**, 5691–5696.

Cid,H., Bunster,M., Canales,M. and Gazitua,F. (1992) *Protein Eng.*, **5**, 373–275.

Coco,W.M., *et al.* (2002) *Nat. Biotechnol.*, **20**, 1246–1250.

Coco,W.M., Levinson,W.E., Crist,M.J., Hektor,H.J., Darzins,A., Pienkos,P.T., Squires,C.H. and Monticello,D.J. (2001) *Nat. Biotechnol.*, **19**, 354–359.

Dubey,A., Realff,M.J., Lee,J.H. and Bommarius,A.S. (2005) *J. Theor. Biol.*, **234**, 351–361.

Dwyer,M.A., Looger,L.L. and Hellinga,H.W. (2003) *Proc. Natl Acad. Sci.*, **100**, 11255–11260.

Hiraga,K. and Arnold,F.H. (2003) *J. Mol. Biol.*, **330**, 287–296.

Klein,P., Kanehisa,M. and DeLisi,C. (1984) *Biochim. Biophys. Acta*, **787**, 221–226.

Korkegian,A., Black,M.E., Baker,D. and Stoddard,B.L. (2005) *Science*, **308**, 857–860.

Krigbaum,W.R. and Komoriya,A. (1979) *Biochim. Biophys. Acta.*, **576**, 204–248.

Lutz,S., Ostermeier,M., Moore,G.L., Maranas,C.D. and Benkovic,S.J. (2001) *Proc. Natl Acad. Sci.*, **98**, 11248–11253.

Martin,A., Sieber,V. and Schmid,F.X. (2001) *J. Mol. Biol.*, **309**, 717–726.

Marvin,J.S. and Hellinga,H.W. (2001) *Proc. Natl Acad. Sci.*, **98**, 4955–4960.

Miyazaki,K., Wintrode,P.L., Grayling,R.A., Rubingh,D.N. and Arnold,F.H. (2000) *J. Mol. Biol.*, **297**, 1015–1026.

Moore,G.L. and Maranas,C.D. (2004) *AIChE J.*, **50**, 262–272.

Moore,G.L. and Maranas,C.D. (2003) *Proc. Natl. Acad. Sci.*, **100**, 5091–5096.

Ostermeier,M., Nixon,A.E., Shim,J.H. and Benkovic,S.J. (1999) *Proc. Natl Acad. Sci.*, **96**, 3562–3567.

Otey,C.R., Landwehr,M., Endelman,J.B., Hiraga,K., Bloom,J.D. and Arnold,F.H. (2006) *PLOS Biol.*, **4**, e112.

Richardson,T.H., *et al.* (2002) *J. Biol. Chem.*, **277**, 26501–26507.

Treynor,T.P., Vizcarra,C.L., Nedelcu,D. and Mayo,S.L. (2007) *PNAS*, **104**, 48–53.

Sakamoto,T., Joern,J.M., Arisawa,A. and Arnold,F.H. (2001) *Appl. Environ. Microbiol.*, **67**, 3882–3887.

Santoro,S.W., Wang,L., Herberich,B., King,D.S. and Schultz,P.G. (2002) *Nat. Biotechnol.*, **20**, 1044–1048.

Saraf,M.C., Gupta,A. and Maranas,C.D. (2005) *Proteins*, **60**, 769–777.

Saraf,M.C., Horswill,A.R., Benkovic,S.J. and Maranas,C.D. (2004) *PNAS*, **101**, 4142–4147.

Saraf,M.C. and Maranas,C.D. (2003) *Protein Eng.*, **16**, 1025–1034.

Sieber,V., Martinez,C.A. and Arnold,F.H. (2001) *Nat. Biotechnol.*, **19**, 456–460.

Stemmer,W.P. (2000) *US Patent 6,132,970: Methods of shuffling polynucleotides.*

Stemmer,W.P.C. (1994a) *Nature*, **370**, 389–391.

Stemmer,W.P.C. (1994b) *Proc. Natl. Acad. Sci.*, **91**, 10747–10751.

Voigt,C.A., Martinez,C., Wang,Z.G., Mayo,S.L. and Arnold,F.H. (2002) *Nat. Struct. Biol.*, **9**, 553–558.

Zhao,H., Giver,L., Shao,Z., Affholter,J.A. and Arnold,F.H. (1998) *Nat. Biotechnol.*, **16**, 258–261.

**Received February 8, 2007; revised May 25, 2007;
accepted May 25, 2007**

Edited by Frances Arnold