Non-contiguous SCHEMA protein recombination

Matthew A. Smith and Frances H. Arnold*

Division of Chemistry and Chemical Engineering, California Institute of Technology, Pasadena, CA, 91125, USA.

* To whom correspondence should be addressed: frances@cheme.caltech.edu

Summary

SCHEMA is a method of designing protein recombination libraries that contain a large fraction of functional proteins with a high degree of mutational diversity. In the previous chapter we illustrated the method for designing libraries by swapping contiguous sequence elements. Here, we introduce the NCR ("noncontiguous recombination") algorithm to identify optimal designs for swapping elements that are contiguous in the 3-D structure but not necessarily in the primary sequence. To exemplify the method, NCR is used to recombine 3 fungal cellobiohydrolases (CBH1s) to produce a library containing more than 500,000 novel chimeric sequences.

Key words: protein engineering, homologous recombination, SCHEMA, non-contiguous recombination, NCR, chimeragenesis,

Running head: SCHEMA and NCR

1. Introduction

As discussed in the previous chapter, SCHEMA ***(1)*** seeks to maximize the probability that a library of chimeric proteins will be functional by using structural information to identify sequence elements ("blocks") that can be swapped. It is advantageous to minimize the average SCHEMA energy (<E>) of all the chimeras in a library, as this increases the fraction of functional chimeras ***(2)***. When recombining sequence elements that are contiguous along the polypeptide chain, RASPP ***(3)*** is used to identify optimal crossovers that minimize <E>.

In this chapter, we describe the designing of SCHEMA libraries with even lower <E>s by removing the constraint that blocks be contiguous along the polypeptide chain. These non-contiguous blocks of sequence are still contiguous blocks of structure in the folded protein. We use non-contiguous recombination (NCR) ***(4)*** to computationally search for optimal non-contiguous SCHEMA library designs. This approach to chimera design has become feasible now that the genes can be made by total gene synthesis.

Here, we design a SCHEMA library that recombines 3 fungal cellobiohydrolases (CBH1s) splitting each homolog into 12 blocks. Shuffling these blocks generates a chimera library of 3^{12} = 531,441 possible sequences. Analysis of a library designed in the same manner as that described here ***(5)*** allowed the identification of several stabilizing sequence elements. NCR-designed libraries can have significantly lower disruption than RASPP (contiguous) designs from the same parent sequences. Alternatively, NCR enables recombination of parents with lower sequence identity. We recommend analysis of NCR-designed libraries by making an informative sample set of genes and using those to build predictive models, as we have done for RASPP-designed libraries ***(6)***.

2. Materials

1. A Unix-based computer that can run python scripts (*see* **Note 1**). Python can be downloaded from: http://www.python.org/download/
2. The NCR toolbox downloaded and unpacked. This is available from: http://cheme.che.caltech.edu/groups/fha/media/ncr.zip
3. MUSCLE (*see* **Note 2**). This is available for download from: http://www.drive5.com/muscle/downloads.htm

Unpack the compressed file and place the executable in the directory 'ncr/tools/muscle' (*see* **Note 3**).

4. hmetis (*see* **Note 4)**. This is available for download from:

   http://glaros.dtc.umn.edu/gkhome/metis/hmetis/download

   Unpack the compressed file and place the hmetis folder in the directory 'ncr/tools' (*see* **Note 3**).

5. A multiple sequence alignment of the parental sequences to be recombined (*see* **Note 5**). This alignment should be in FASTA format (*see* **Note 6**) and the file should be named 'alignment.fasta'. As recombination parents for the example used here, we selected the CBH1 sequences from *Chaetomium thermophilum*, *Hypocrea jecorina*, and *Talaromyces emersonii*, which have about 60% pairwise amino acid sequence identity. These CBH1s have a catalytic domain, a linker and a cellulose-binding domain. The available crystal structures are for the catalytic domain, thus we only considered this domain for recombination (*see* **Note 7**). To eliminate the possibility of generating unpaired disulfide bonds, we mutated two residues in the *T. emersonii* CBH1 sequence to cysteine (*see* **Note 8**). We used PROMALS3D ***(7)*** to align the parental sequences.

6. A PDB structure file of one of the parental sequences (*see* **Note 9**). We use the *T. emersonii* structure, '1Q9H.pdb'. Alternatively, if no structure is provided, the NCR tools can search for suitable structures from the PDB database (*see* **Note 10**).

3. Methods

1. Place the parent sequence alignment file (alignment.fasta) in the 'ncr' folder. Place the PDB structure file (1Q9H.pdb) in the directory 'ncr/structures'.

2. Set the ‘Number of blocks’ to 12 and ‘Find all PDB structures’ to 0 in the ‘init.txt’ file (*see* **Note 10**).
3. Run the following command (*see* **Note 11**) in the ‘ncr’ directory:

   ```
   python ncr.py
   ```

   This NCR script identifies a set of candidate libraries with low <E> and sends these results to the terminal window (*see* **Note 12**) (**Fig. 1**). These libraries are saved in the directory ‘ncr/output’ and listed in the text file ‘library12_result_list.csv’ (*see* **Note 13**).
4. Pick an NCR library (*see* **Note 14**). In this case, we pick the library ‘library12_2.output’, with <E> = 16.8 and <m> = 83.9 (**Fig. 2**).
5. Certain non-conserved residues still need to be assigned to blocks (*see* **Note 15**). Open ‘ncr/output/library12_2.output’ and assign residues 41, 175, 197, 199, 202, and 442 to blocks G, C, A, A, A, and J, respectively (*see* **Note 16**).
6. Run the following command (*see* **Note 17**) in the ‘ncr’ directory:

   ```
   python picklibrary.py library12_2
   ```

   This generates a list of all the chimeras in the chosen library along with their SCHEMA energies, number of mutations, and sequences (*see* **Note 18**). This list is saved as a text file ‘chimeras.output’ in the directory ‘ncr/picked_libraries/library12_2’.
7. We synthesize the genes encoding a subset of the chimera library (*see* **Note 19**). Before expressing the CBH1 chimeras, we add a linker and cellulose-binding domain to the recombined catalytic domains.

4. Notes

1. The NCR toolbox ‘ncr’ is written for python 2.6 on a Unix-based system. We recommend using this python release for the NCR toolbox.

2. Ensure you download the correct distribution of MUSCLE for your system. For example, on Apple OS X it might be 'muscle3.8.31_i86darwin64.tar.gz'. The NCR tools were written for muscle 3.8.
3. The NCR toolbox unpacks as a folder called 'ncr'. Directories are given relative to this folder. For example there is a folder in 'ncr' called 'tools' and the directory would be 'ncr/tools'.
4. Ensure you download the correct distribution of hmetis for your system. For example, on Apple OS X it might be 'hmetis-1.5-osx-i686.tar.gz'. The NCR tools were written for hmetis 1.5.
5. We assume the parental proteins share the same structural fold. If structures are available for more than one parental protein, we confirm the parents have the same fold by aligning the parental structures. It is important that the sequence alignment is accurate, especially when the parental sequence identities are low.
6. In FASTA format, the name of each sequence begins with '>', for example '>Temersonii'. After each name there should be a return, followed by the corresponding aligned sequence.
7. SCHEMA library designs require a protein structure. If no structural information is available for a parent sequence, but there are structures of homologs, use MODELLER to build a structure model ***(8)***. An inaccurate homology model hinders SCHEMA library design; an actual structure is preferred.
8. We assumed but did not verify that broken disulfide bonds are destabilizing. In this case, *C. thermophilum* and *H. jecorina* CBH1s have 10 disulfide bonds while *T. emersonii* has 9 disulfide bonds. If the cysteines from the missing disulfide bond are in separate sequence blocks, chimeras with unpaired cysteines can result. We avoided this by modifying the parental sequence of *T. emersonii* to include the remaining cysteine pair.
9. One or more structures is needed to identify the residue-residue contacts. When possible, we select high-resolution structures (< 2.0 Å). If a PDB file contains more than one chain, each

chain is automatically split into its own structure file labeled XXXX.A.pdb, XXXX.B.pdb, etc. The NCR tools can handle multiple structures. Residue-residue contacts from multiple structures of the same parent form a parental contact map if these contacts are present in at least 50% of the structures. If structures from multiple parents are used, each contact is weighted by the fraction of parental contact maps it appears in.

10. The 'init.txt' file is in the 'ncr' folder. It specifies two parameters for the NCR toolbox:
    - 'Number of blocks': The number of blocks in the designed libraries. It can either be a number (e.g. 8) or a range of numbers (e.g. 2-6) for designing a range of libraries with different block sizes.
    - 'Find all PDB structures': If 1, the NCR script will search, download and use all suitable structures from the PDB database. If 0, the user will provide one or more structures.

    Increasing the number of blocks in a library increases library size and reduces the average number of mutations in a block. The user may want smaller blocks if searching for single mutations that cause specific functional changes. However, it is harder to find desirable chimeras in larger libraries and increasing the number of blocks increases the <E> of a library. We chose to split our 3 parental proteins into 12 blocks.
11. The python script 'ncr.py' generates one or more parental contact maps, calculates the SCHEMA contacts and searches for low <E> libraries. This script may take several hours to complete, depending on protein size and computer specifications. Progress is displayed in the terminal window. The script uses heuristic algorithms to find near optimal solutions, thus results will vary slightly each time 'ncr.py' is run. However, these variations are very small and we find we do not identify significantly lower <E> library designs by running the algorithm multiple times.

12. In the terminal window, NCR lists <E> and <m> for each library as well as the distribution of mutations among the 12 blocks. This distribution is given as a list of 12 numbers, each referring to the number of mutations in a block with blocks counting A, B, C, etc. There is a trade-off between the average SCHEMA energy of a library (<E>) and how evenly distributed mutations are among the blocks. If all the blocks are evenly sized, the solution space of possible libraries is small and so <E> is large. As block sizes become uneven, the solution space of possible libraries increases. This enables NCR to find libraries with lower <E>, but libraries with very unevenly sized blocks may not be useful, as it will be more difficult to identify sequence-function relationships in very large blocks, and very small blocks contain few mutations. NCR is designed to find low <E> libraries for a range of block sizes.
13. In non-contiguous recombination, a library is defined by assigning every non-conserved residue to a block. In the library text file 'library12_2.output', a designated block (named 'A', 'B', 'C', etc.) appears beside every non-conserved residue. A dash ('-') is placed next to every conserved residue. Residues are numbered based on the parental sequence alignment. The results file 'library12_result_list.csv' lists <E> and <m> for each library.
14. NCR returns a set of candidate libraries with a range of <m> values. A lower <E> implies more functional chimeras in the library. For moderately-sized proteins (250-500 amino acids) we try to pick SCHEMA libraries with <E> less than 30. For non-contiguous recombination of homologs with > 55% sequence identity, often all the candidate libraries have <E> below 30. In our case, we pick a library with evenly-sized blocks. This will make it easier to identify stabilizing point mutations within a stabilizing block. Protein-specific biochemical and structural knowledge may also help users select from the candidate libraries. For example, one may wish to conserve a specific region of protein structure, such as an allosteric site or active site, by choosing a library design that has the structural feature in a single block. Note that the

<E> value is lower and the <m> value higher in this NCR design than the RASPP design from the previous chapter.

Blocks are not always one contiguous piece of structure. Sometimes, a group of residues will only have SCHEMA contacts with one another and not with the rest of the protein. These 'disconnected blocks' can belong to any block without altering <E>. NCR will assign these disconnected blocks to blocks such that <m> is maximized. This can result in a block comprising two separate pieces of structure. These disconnected blocks are apparent when blocks are visualized on the PDB structure. In this case, blocks 'A', 'G', and 'J' each contain a disconnected block.

15. Some non-conserved residues do not have any SCHEMA contacts. These residues often appear on the surface of the protein, in a region that is highly conserved or in a region where structural information is missing. NCR does not assign these residues to a block and instead the decision is left to the user. Unassigned residues are printed to the terminal. In this case residues 41, 175, 197, 199, 202, and 442 have not been assigned a block.
16. Looking at the structure '1Q9H.pdb', we designate each unassigned residue to the same block as one of its neighboring residues. This will slightly alter <m> for the library, but leave <E> unaffected. We can alter the block assignments by editing the text file 'ncr/output/library12_2.output'. In this file unassigned residues, like conserved residues, have a dash ('-') in place of a block ('A', 'B', 'C', etc.).
17. The python script 'picklibrary.py' generates all the chimeras in a given library. The name of the library 'library12_2' needs to be provided as a parameter, and appears after 'python picklibrary.py'. Any non-conserved residues that have not been assigned to a block will be automatically assigned to block A. For a large library such as this one (more than 500,000 chimeras), this script may take several hours to complete.

18. Chimeras are numbered according to the parental sequence of each block with the numbers ordered from the first to the last block. Parents are numbered based on the order they appear in the parental sequence alignment. For example, chimera '132213131322' has parent 1 as the sequence of its first block ('A'), parent 3 as its second block ('B'), etc. The amino acid sequence provided alongside each chimera in 'chimeras.output' is built from the parent sequence alignment. It contains dashes ('-') where there are gaps in the alignment. These dashes should be removed when ordering the synthetic genes.
19. These chimeras are very difficult to construct with traditional cloning techniques; each non-contiguous block will usually have mutations distributed throughout the protein sequence. We pick a subset of the library to synthesize and analyze. Typically this subset contains 20-40 chimera sequences and is limited by synthesis cost. We ensure every block from every parent is represented independently of one another in this subset. This enables us to model the effects of the different blocks on biochemical properties such as stability ***(9)***. We pick a set of chimeras to be most informative using the Submodular Function Optimization Matlab toolbox ***(10, 11)***. Alternatively, we could have selected a set of chimeras that substitute one block at a time into the background of a parent that expresses well, such as *T. emersonii* CBH1 ***(12)***.

Acknowledgements

The authors acknowledge funding from the Institute for Collaborative Biotechnologies through grant W911NF-09-D-0001 from the U.S. Army Research Office and The National Central University, Taiwan, through a Cooperative Agreement for Energy Research Collaboration. MAS is supported by a Resnick Sustainability Institute fellowship.

References

1. Voigt, CA, Martinez C, Wang Z-G, Mayo SL, Arnold FH (2002) Protein building blocks preserved by recombination. *Nat Struct Biol* **9**: 553–558
2. Meyer M, Hochrein L, Arnold FH (2006) Structure-guided SCHEMA recombination of distantly related β-lactamases. *Protein Eng Des Sel* **19**: 563–570
3. Endelman J, Silberg J, Wang Z, Arnold FH (2004) Site-directed protein recombination as a shortest-path problem. *Protein Eng Des Sel* **17**: 589–594
4. Smith MA, Romero PA, Wu T, Brustad EM, Arnold FH (2013) Chimeragenesis of distantly-related proteins by noncontiguous recombination. *Protein Sci* **22**: 231-238
5. Smith MA, Bedbrook CN, Wu T, Arnold FH (2013) Hypocrea jecorina cellobiohydrolase I stabilizing mutations identified using noncontiguous recombination. *ACS Syn Biol* doi:10.1021/sb400010m
6. Heinzelman P, Romero PA, Arnold FH (2013) Efficient sampling of SCHEMA Chimera Families for Identification of Useful Sequence Elements. In: Keasling, A (ed) Methods in Enzymology: Methods in Protein Design, Elsevier Ltd, Oxford, U.K.
7. Pei J, Kim B-H, Grishin NV (2008) PROMALS3D: a tool for multiple protein sequence and structure alignments. *Nucleic Acids Res* **36**: 2295–2300
8. Eswar N, Webb B, Marti-Renom MA, Madhusudhan M, Eramian D, Shen MY, Pieper U, Sali A (2007) Comparative protein structure modeling using Modeller. *Curr Protoc Protein Sci* **2**: 15–32
9. Li Y, Drummond DA, Sawayama AM, Snow CD, Bloom JD, Arnold FH (2007) A diverse

family of thermostable cytochrome P450s created by recombination of stabilizing fragments. *Nat Biotechnol* **25**: 1051–1056

10. Krause A (2010) SFO: A Toolbox for Submodular Function Optimization. *J Mach Learn Res* **11**: 1141–1144

11. Romero P, Stone E, Lamb C, Chantranupong L, Krause A, Miklos A, Hughes R, Fechtel B, Ellington AD, Arnold FH, Georgiou G (2012) SCHEMA-designed variants of human arginase I and II reveal sequence elements important to stability and catalysis. *ACS Synth Biol* **1**: 221–228

12. Heinzelman P, Komor R, Kanaan A, Romero PA, Yu X, Mohler S, Snow C, Arnold FH (2010) Efficient screening of fungal cellobiohydrolase class I enzymes for thermostabilizing sequence blocks by SCHEMA structure-guided recombination. *Protein Eng Des Sel* **23**: 871–880

Figures

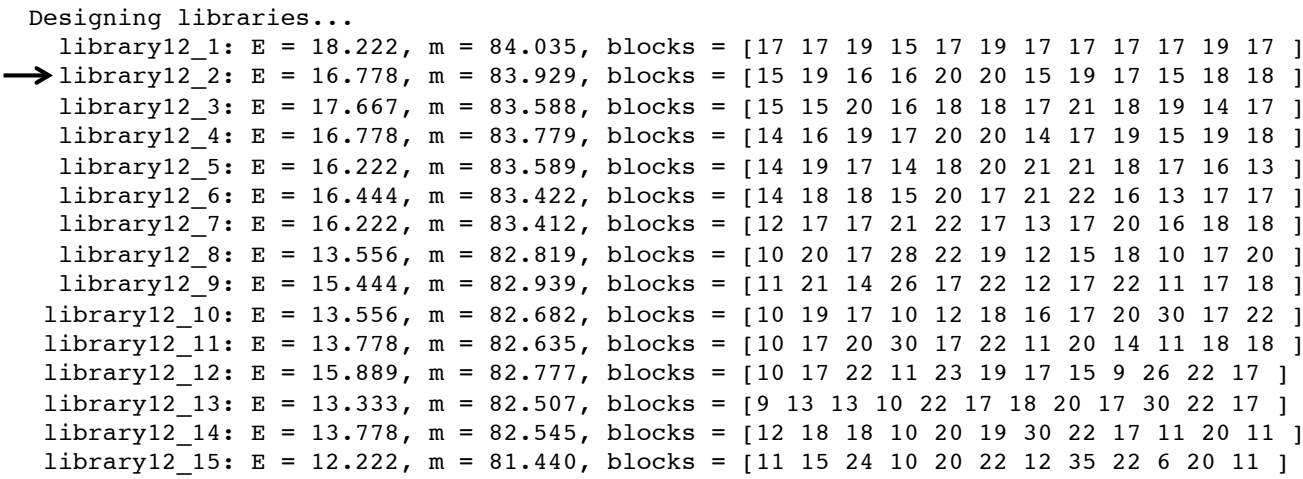

Fig. 1. Libraries returned by NCR. The average SCHEMA energy (<E>) and average number of mutations (<m>) for each library is printed to the terminal window. In addition, the output displays the distribution of the mutations among the 12 blocks. Libraries with higher <E> have more evenly-sized blocks. The chosen library is highlighted with an arrow. We picked this library because we wanted evenly sized blocks to help us efficiently search for stabilizing mutations.

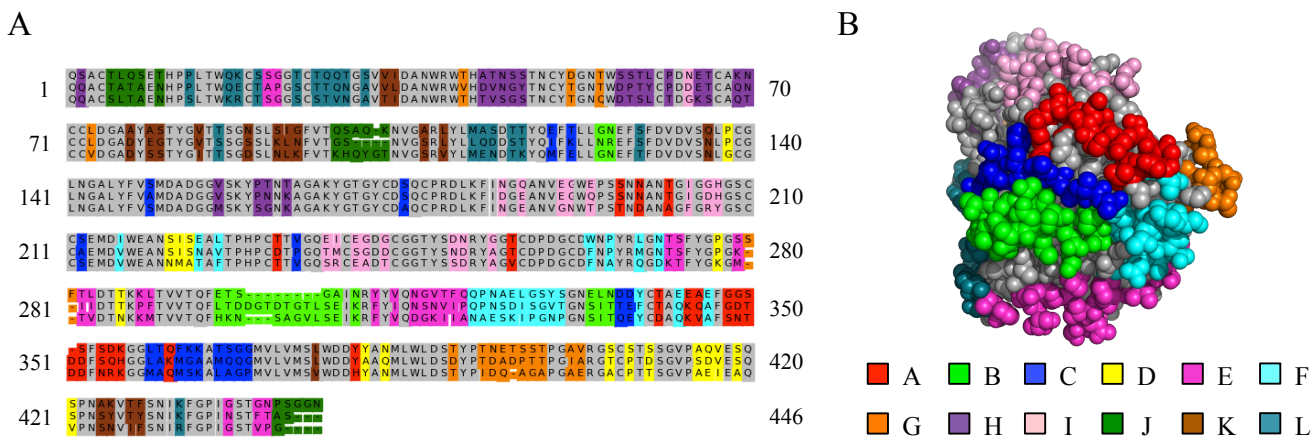

Fig. 2. Visualizing the chosen NCR design. (**a**) The multiple sequence alignment of the parent CBH1s with each of the 12 blocks highlighted in a different color. Conserved residues are colored grey. It is clear that the blocks are non-contiguous along the polypeptide chain. (**b**) The blocks highlighted on the CBH1 structure '1Q9H.pdb'. Most of the blocks are contiguous structural elements in 3-D.