IMMUNOME RESEARCH

RESEARCH Open Access

Bridging the gap between single-template and fragment based protein structure modeling using Spanner

Mieszko Lis[1]*, Taeho Kim[2]*, Jamica J. Sarmiento[2], Daisuke Kuroda[3,4], Huy Viet Dinh[2], Akira R. Kinjo[3], Karlou Amada[2], Srinivas Devadas[1], Haruki Nakamura[3§], and Daron M. Standley[2§]

Abstract

Background: As the coverage of experimentally determined protein structures increases, fragment-based structural modeling approaches are expected to play an ever more important role in structural modeling. Here we introduce a structural modeling method by which an initial structural template can be extended by the addition of structural fragments to more closely match an aligned query sequence. A database of protein fragments indexed by their internal coordinates was created and a novel methodology for their retrieval was implemented. After fragment selection and assembly, sidechains are replaced and the all-atom model is refined by restrained energy minimization. We implemented the proposed method in the program *Spanner* and benchmarked it using a previously published set of 367 immunoglobulin (Ig) loops, 206 historical query-template pairs and alignments from the Critical Assessment of protein Structure Prediction (CASP) experiment, and 217 structural alignments between remotely homologous query-template pairs. The constraint-based modeling software MODELLER and previously reported results for RosettaAntibody, were used as references.

Results: The error in the modeled structures was assessed by root-mean square deviation (RMSD) from the native structure, as a function of the query-template sequence identity. For the Ig benchmark set, for which a single fragment was used to model each loop, the average RMSD for *Spanner* (3 +/- 1.5 Å) was found to lie midway between that of MODELLER (4 +/- 2 Å) and RosettaAntibody (2 +/- 1 Å). For the CASP and structural alignment benchmarks, for which gaps represent a small fraction of the modeled residues, the difference between *Spanner* and MODELLER were much smaller then the standard deviations of either program. The *Spanner* web server and source code are available at http://sysimm.ifrec.osaka-u.ac.jp/Spanner/.

Conclusions: For typical homology modeling, *Spanner* is at least as good, on average as the template-free constraint-driven approach used by MODELLER. The Ig model results suggest that when gap regions represent a significant fraction of the alignment, *Spanner*'s efficient use of fragment libraries, along with local sequence and secondary structural information, significantly improve model accuracy without a dramatic increase in computational cost.

Background

Homology-based protein structural modeling plays an important role in biomedical research by linking genomics and structural biology. As the number of known protein sequences and structures grows, so do the number of sequences that can be modeled. Knowledge of even an approximate three-dimensional protein structure can provide valuable information about its structural neighbors. This knowledge can, in turn, shed light on the protein's evolutionary history, biochemical and biological functions. For example, structural modeling was recently used to predict the Mg-dependent RNase activity of Zc3h12a, a protein essential for regulating inflammatory cytokines in toll-like receptor 4 signaling[1]. Here, we introduce a novel structural modeling method using a wider range of protein targets, including a representative set of all known antibody structures.

Currently, the most accurate methods for modeling protein structure are extensions of the fragment assembly method originally implemented in the program Rosetta [2], and now found in the successful TASSER program [3]. In this class of methods, short fragments of known structure are mapped on to the query sequence and then assembled by combinatorial optimization using structure-dependent scoring

[1]MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139, USA
[2]Systems Immunology Lab, WPI Immunology Frontier Research Center (IFReC), Osaka University,3-1 Yamadaoka, Suita, Osaka 565-0871, Japan.
[3]Institute for Protein Research, Osaka University, 3-2 Yamadaoka, Suita, Osaka 565-0871, Japan
[4]Department of Pharmaceutical Chemistry, University of California San Francisco, CA 94158, USA.
*The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.
[§]Corresponding author
Email addresses:
HN: harukin@protein.osaka-u.ac.jp
DMS: standley@ifrec.osaka-u.ac.jp

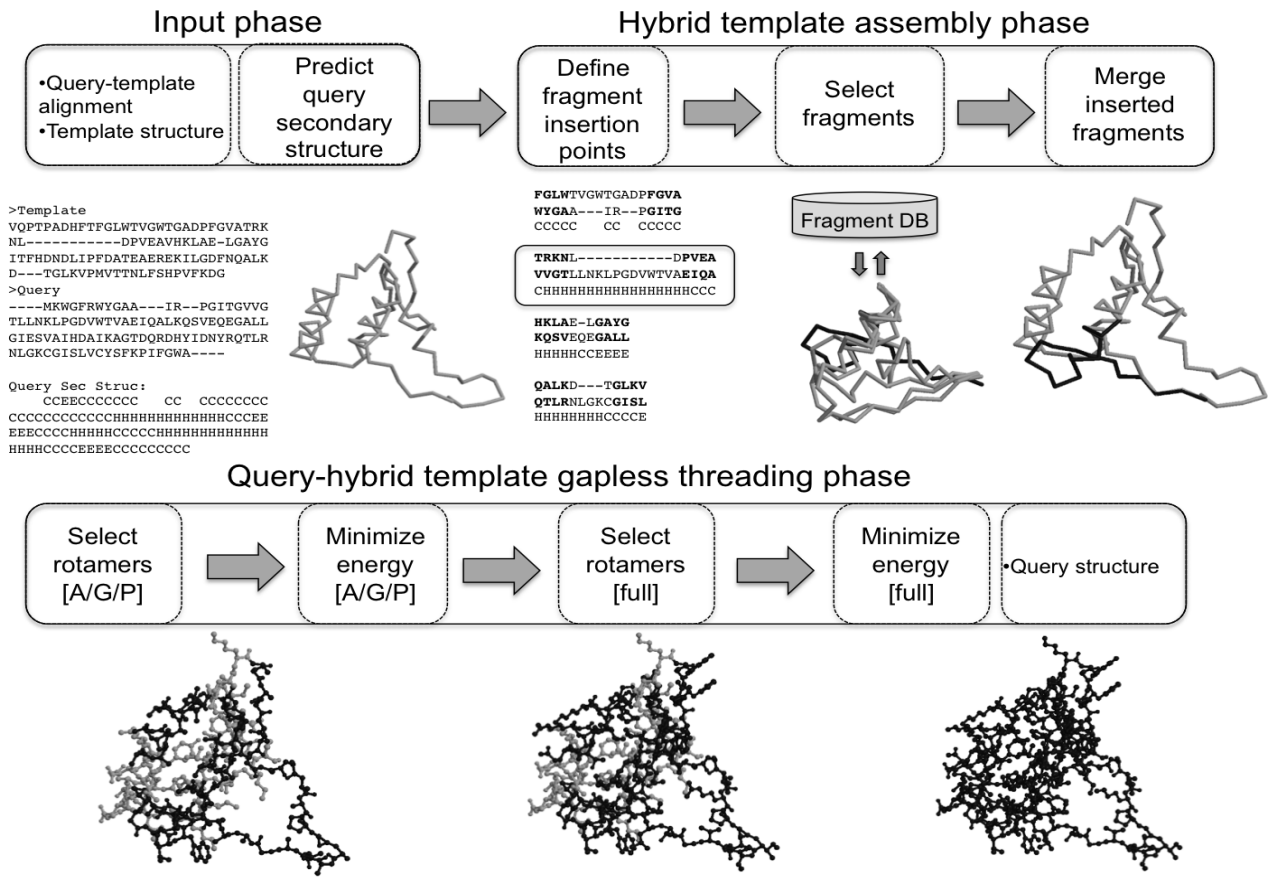

Figure 1 - Spanner Pipeline

The three main phases of the *Spanner* pipeline are shown. In the hybrid template assembly phase, the selection of one fragment is illustrated. In the threading phase, dark regions indicate parts of the structure that undergo energy minimization.

functions to create a hybrid template model. The strength and drawback of this approach, at least in current implementations, is that the size of the effective conformational space is very large. In practice, this means that the optimization procedure takes a long time. Waiting times on the most popular servers can be weeks to months, and users are usually limited to one query at a time.

For this reason, single-template threading, using profile-based scoring functions, is more widely used for routine homology model building. Results can be computed in minutes to hours, which fits well with a typical researcher's timeframe. Unfortunately, the single-template methods typically result in a significant number of insertions and deletions for query-template pairs with low sequence homology. Large insertions present challenges for constraint-based modeling software, such as MODELLER [4], since the inserted sequence is effectively unconstrained within the gaps and can appear as a random coil in the final model, even when the insertion is predicted to be structured.

Here we introduce a novel modeling method, implemented in the program *Spanner*, which uses fragment assembly to extend an initial single template such that there are no insertions or deletions with respect to the query. Because *Spanner* starts with an initial 'anchor' template, the search for fragments is constrained by the geometry of the gap endpoints, resulting in an efficient optimization protocol. Crucially, the use of internal coordinates as a database index allows fragments matching the

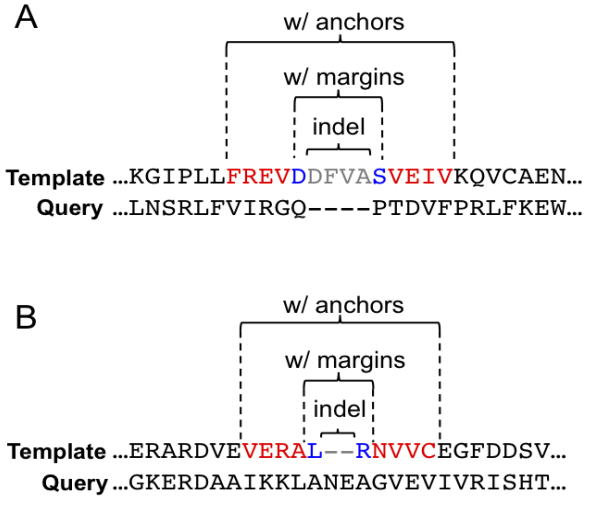

Figure 2 - Insert selection

Inserts are selected from an alignment of the query sequence (bottom line) with the template (top line). A small margin (in this case one residue) is added each gap boundary, and a four-residue anchor on each side is used to search the fragment database for suitable replacement candidates.

anchor regions within a given tolerance to be retrieved efficiently using a single PGSQL query. Furthermore, since the fragments are selected based on sequence and secondary structure similarity to the query, the insertions are likely to be structured if the corresponding query segment is predicted to be so. *Spanner* makes use of native and 3rd-party software, including utilities for populating and updating fragment relational databases, fragment scoring and assembly, sidechain replacement, and energy refinement. A web interface that supports 3D graphical visualization and export of the resulting model to the *SeSAW* functional annotation server [5] is available, along with source code and data for local installation.

Results

In this section we describe results for the Ig and for the CASP and ASH data sets using the fragment retrieval module and the full *Spanner* pipeline, respectively.

Figure 3A contains results for the entire Ig set binned by loop length. From this figure we can see that, overall, the RMSD grows roughly linearly with the loop length, as has been reported before [7]. In general, the backbone (N, C α, C, O) RMSDs of the *Spanner* loops lie below those of the loops built

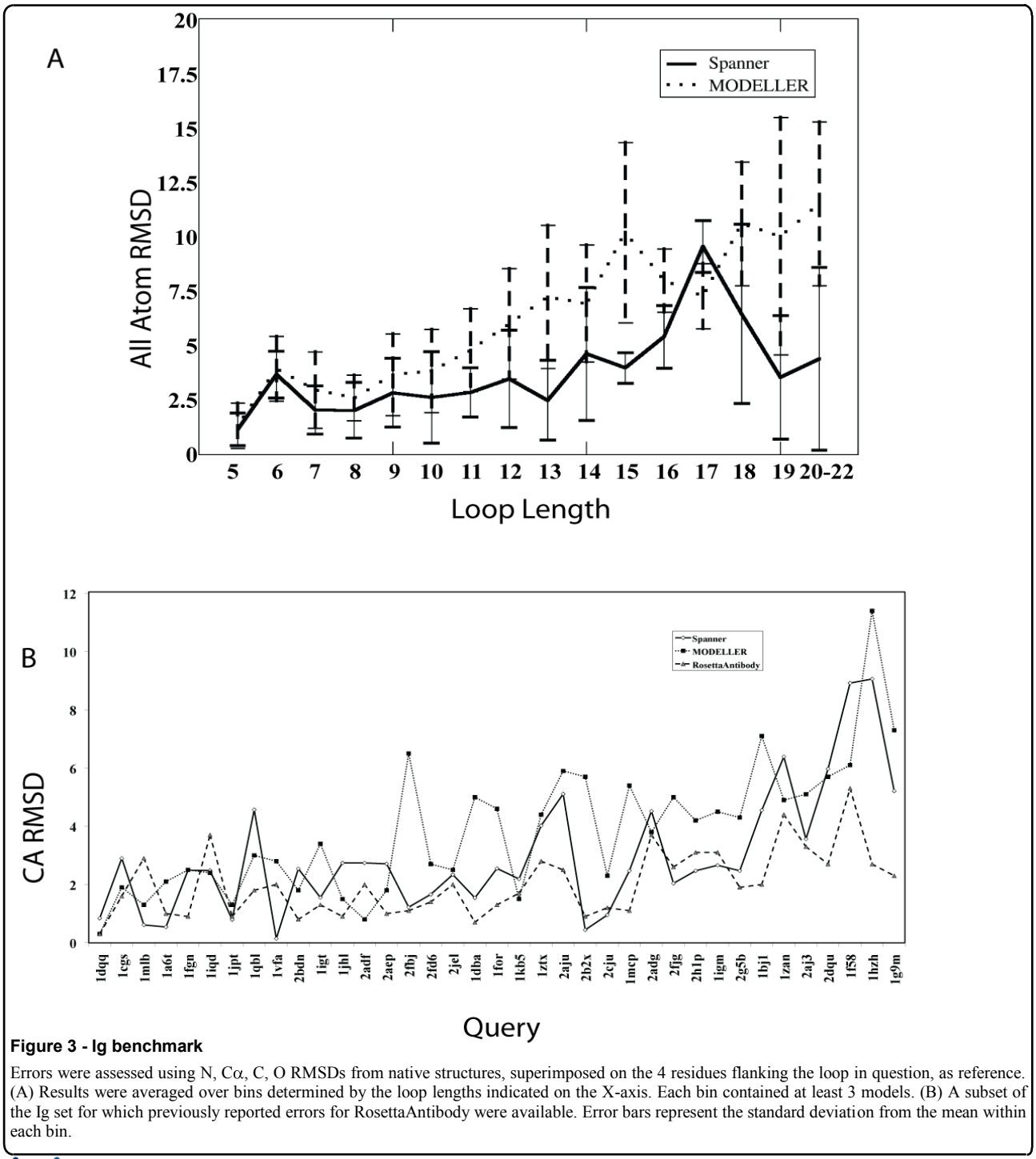

Figure 3 - Ig benchmark

Errors were assessed using N, Cα, C, O RMSDs from native structures, superimposed on the 4 residues flanking the loop in question, as reference. (A) Results were averaged over bins determined by the loop lengths indicated on the X-axis. Each bin contained at least 3 models. (B) A subset of the Ig set for which previously reported errors for RosettaAntibody were available. Error bars represent the standard deviation from the mean within each bin.

with MODELLER. The loops in the Ig set that have been assessed previously using the RosettaAntibody program are shown in Figure 3B. For this sub-set of loops, the mean and standard deviation of the *Spanner* backbone RMSD (3 +/- 1.5 Å) lies between that of MODELLER (4 +/- 2 Å) and RosettaAntibody (2 +/- 1 Å). We note that the *Spanner* results in Figures 3A and 3B are overall consistent with each other, so we can expect similar performance to that shown in Figure 3B on larger data sets.

For the Ig results, MODELLER jobs required an average of 3 +/- 1 CPU hours, while the *Spanner* fragment retrieval module required 2.4 +/- 0.6 minutes.

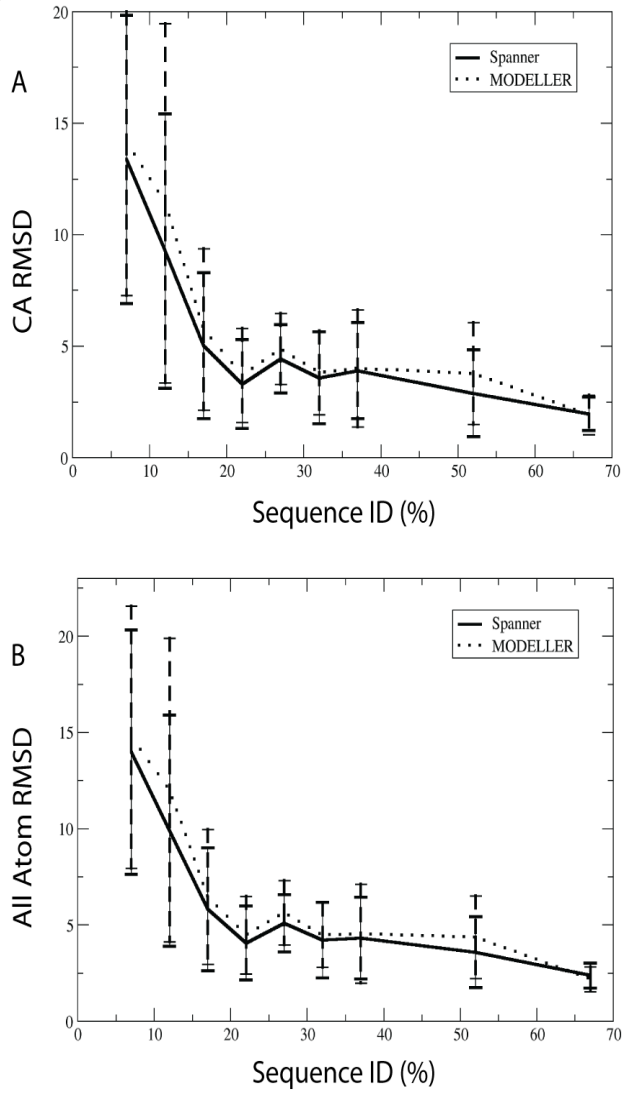

Figure 4 - Accuracy of Spanner using CASP set

The Cα RMSD (A) and all-atom RMSD (B) are shown for *Spanner* and MODELLER using the CASP test set. Results were binned by sequence identity such that each bin contained at least 10 data values. The plots represent averages within each bin. Error bars represent the standard deviation from the mean within each bin.

CASP and ASH sets: *Spanner* accuracy

Here, we examine the accuracy of the full *Spanner* pipeline using the CASP set, which allows us to evaluate the performance of *Spanner* with actual alignments generated for a range of query-template pairs using typical alignment tools. For benchmarking purposes we ran MODELLER using the automodel class with the same alignments. Note that in this exercise, we excluded any fragment from the DB if its sequence identity to the query was 30% or more, in order to make the comparison with MODELLER, which does not make use of existing structural fragments, as fair as possible. Figure 4 shows the average RMSD (C α and all-atom) within a range of sequence identity bins. From this figure we can see that, as expected, the average accuracy increased while the standard deviations decreased with sequence identity. There is a slight improvement in terms of RMSD for *Spanner* over MODELLER in some cases, but the differences are much smaller than the spread in the data. These results confirm that, on average, *Spanner* produces models that are at least as accurate as those of MODELLER, a state-of-the-art structural modeling tool.

The ASH set represents 'perfect' input for a set of low-homology query-template pairs. The results, shown in Figure 5, are consistent with the CASP alignment results. Here too, we see that the differences between *Spanner* and MODELLER are very small compared with the deviations for each program within a given sequence identity bin.

We also assessed the CPU times for *Spanner* and MODELLER for the CASP and ASH sets. In this case, MODELLER average CPU times (17 +/- 14 s) were over 20 times shorter than *Spanner* (377 +/ 4 s). There are two reasons for the reverse trend here as compared to the Ig set. First, the MODELLER automodel class is much faster than the dope_loopmodel class. Second, the fragment retrieval module (used in the previous section) is much faster than the full *Spanner* pipeline, which, in addition to fragment selection, performs sidechain replacement and energy refinement.

Discussion and Conclusions

In this article we present an approach for utilizing the strengths of both single and multiple-template protein modeling. The results clearly demonstrate that for gap regions, a fragment-based approach is at least as good, on average as the template-free constraint-driven approach, at a much lower computational cost; however, the performance is not yet equal to that reported for RosettaAntibody. Whether this is due to the superior sampling in the Rosetta program or the use of a specialized fragment database and sequence rules to identify kinked conformations is not known. Nevertheless, the contrast between the Ig model results and those for the CASP and ASH sets suggests that when gap regions repre-

sent a significant fraction of the alignment, local sequence and secondary structural information can be exploited to improve model accuracy. When gaps represented a small fraction of the alignment (CASP and ASH sets), we found that the differences between *Spanner* and MODELLER were less than the deviations within either program. In such cases, *Spanner* performed marginally better, in terms of accuracy, but at a higher computational cost. The average computational cost for *Spanner* (approx. 6 minutes) were, nevertheless, consistent with that of most profile-based threading methods, which typically finish within minutes to hours. Taken together, these results suggest that *Spanner* represents a rational and scalable approach to fragment-based structural modeling.

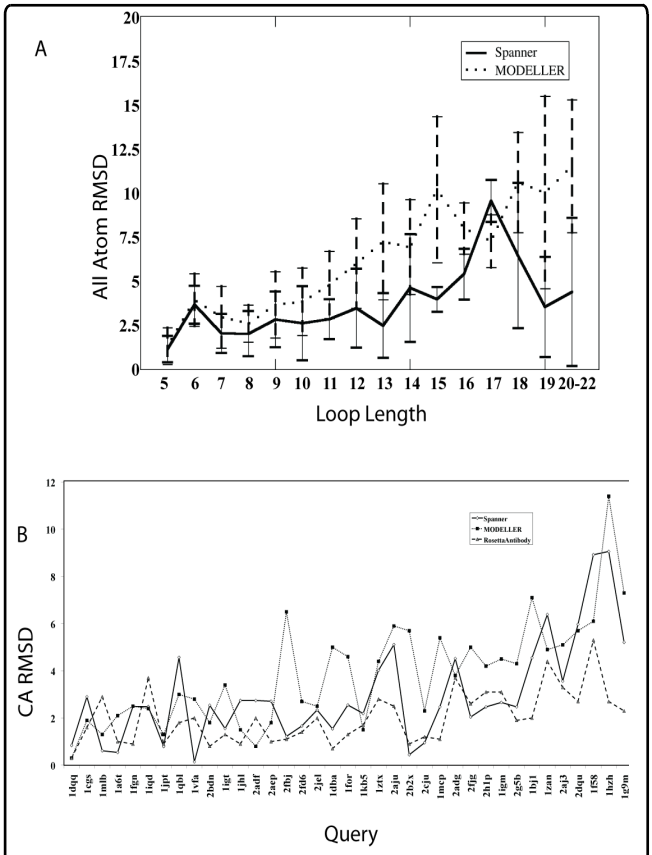

Figure 5 - Accuracy of Spanner using ASH set

The Ca RMSD (A) and all-atom RMSD (B) are shown for *Spanner* and MODELLER using the ASH test set. Results were binned by sequence identity such that each bin contained at least 10 data values. The plots represent averages within each bin. Error bars represent the standard deviation from the mean within each bin.

Methods

The modules in *Spanner* are arranged as a pipeline, as illustrated in Figure 1. The inputs to this pipeline are the query-template alignment, and template structure. The output is a model structure of the query. There are two main phases in the calculation: hybrid template assembly and gapless threading to the hybrid template. In the first phase, fragments are chosen based on their geometric similarity to the template at the anchor points and on their primary and secondary structural similarity to the query. The second phase involves sidechain replacement for the selected fragments and overall structural refinement. These individual steps are described in detail below.

Inputs

Spanner requires a template structure (in PDB format) and a template-query alignment (in FASTA format). In addition, the fragment selection process (described below) uses secondary structure information for the query; the secondary structure can be specified as optional input, or computed automatically using PSIPRED [8].

Definition of fragments

Spanner replaces continuous sequence segments (indels) in the template for every gap in the query-template alignment. The procedure is illustrated in Figure 2. First, the insertion point around each indel is established: for deletions (Figure 2A) in the alignment, the deleted residues are excised from the template; for insertions (Figure 2B), the template sequence gap itself identifies the insert location. To allow for small geometrical differences between the template and the inserted fragment, a margin of 1 or more residues on each side of the insert location is also excised from the template. Adjacent insertion points separated by fewer than 4 residues—i.e., too short to support an anchor—are merged into one insertion point with a larger insertion sequence. For greater user control, the indels and margins can be specified as optional input.

The four residues on each side of the insert point (the anchors) are used to efficiently search a fragment database for suitable insertion candidates, as described below.

Hybrid template preparation

Fragment storage

A representative set of protein chains is maintained using the cd-hit program [9] at 100% sequence identity. All continuous fragments of length 8 or more are regularly extracted from this set of chains and stored in a PostgreSQL relational database (RDB), indexed by the internal coordinates of the fragment endpoints. The internal coordinates consist of C α-C α distances between the following 4 residue pairs: first, last; first+1, last-1; first+2, last-2; first+3, last-3. In addition to the internal coordinates themselves, the PDB identifier, chain ID, sequence, secondary structure, as defined by STRIDE [10], and the beginning and ending atom indices of the fragment in the corresponding PDB entry, are stored in

the RDB. As an example, the first 10 fragments of length 12 for PDB entry 1nag, chain A are stored as:

```
PDBID  Atoms   Sequence    Sec Struc    D1-12  D2-11   D3-10  D4-9
1nagA   1   96 RPDFCLEPPYTG LLHHHHLLLLLL 23.532 20.037 14.990  9.769
1nagA  12  103 PDFCLEPPYTGP LHHHHLLLLLLL 24.821 19.648 15.086 11.978
1nagA  19  109 DFCLEPPYTGPC HHHHLLLLLLLL 24.970 19.725 17.073 15.364
1nagA  27  118 FCLEPPYTGPCK HHHLLLLLLLLL 23.583 22.074 20.992 13.916
1nagA  38  123 CLEPPYTGPCKA HHLLLLLLLLLL 22.959 24.464 19.605 15.500
1nagA  44  134 LEPPYTGPCKAR HLLLLLLLLLLL 22.291 21.479 19.027 14.439
1nagA  52  142 EPPYTGPCKARI LLLLLLLLLLLE 18.893 17.453 15.394 12.856
1nagA  61  150 PPYTGPCKARII LLLLLLLLLLEE 15.128 12.799 12.369  9.599
1nagA  68  161 PYTGPCKARIIR LLLLLLLLLEEE  9.252 11.808  9.340 10.781
1nagA  75  173 YTGPCKARIIRY LLLLLLLLEEEE 10.610  9.553 12.963 12.603
```

A separate RDB is prepared for each fragment length. Currently, fragments of length 8-60 are stored. In addition to the fragment RDB described above, two additional types of RDBs are created to store fragments used to fill N and C-terminal gaps. For N-terminal (C-terminal) gaps, the internal coordinates consist of all unique Cα-Cα distances pairs in the last (first) 4 residues of the fragment. Other fragment information is the same in the terminal RDBs.

Fragment retrieval

For a given fragment, a fragment index is generated from the template anchor residues. A tolerance in the fit to the anchor residues is used to specify a range of index values. The index range is used to generate a PostgreSQL query to the appropriate fragment database and all fragments satisfying the range of indices are returned. PDB entries that should be excluded from the RDB search can be specified by the user, a feature that was utilized in the present work in order to screen out close homologs when benchmarking the program. Since the number of returned fragments is sensitive to the tolerance in the fit to the anchor residues, the retrieval step starts with a small value (0.5 Å by default), and incrementally increases the tolerance until the required number of fragments (1000 by default) or a maximum tolerance (2.5 Å by default) is reached. Each of the above parameters can be modified on the command line.

The fragments returned from the RDB are then sorted by a simple score that is a function only of the primary and secondary structure similarities between the query and the candidate fragment

$$(1) \qquad S_{2D} = S_{seq} + S_{\sec}$$

where S_{seq} is proportional to a log-odds sequence substitution matrix score derived from a large number of structure alignments [11] and S_{sec} is proportional to a secondary structure substitution matrix score [12]. A specified number of candidate fragments (100 by default) is then retained. These retained candidates are then re-scored using a more sensitive function that takes structure into account and is given by

$$(2) \qquad S_{frag} = \frac{S_{2D} - S_{clash}}{RMSD_{fit} + 1}$$

where S_{clash} is a weighted sum of clashes between the fragment and the rest of the template structure, excluding residues that are to be replaced by the fragment. Since side chains are not expected to fit perfectly at this point, the weight of sidechain-sidechain and sidechain-backbone clashes is set to 1/6 that of backbone-backbone clashes. Also, only severe clashes (interatomic distance < 2 Å) are counted at this point. $RMSD_{fit}$ is given by the root-mean square deviation of Cα atoms in the fitted anchor residues. The user-specified number of top-scoring fragments (1 by default) is then output. The weights and number of resulting models can be adjusted on the command-line.

Threading to the hybrid template

After the model's backbone has been established by splicing in all of the indels, as described above, the query sequence is 'threaded' onto the template and the resulting structure is optimized via energy minimization. We use the term threading loosely, as the alignment is trivial (there are no gaps); the procedure involves only the sidechain replacement and relaxation steps of threading.

First, the sidechains from the query sequence are placed on the template structure's backbone and their rotamers optimized by using either the dead-end elimination (DEE) algorithm [13, 14] or the SCWRL4 rotamer selection algorithm [15]. To allow the inserted fragments maximum flexibility, their sidechains are first replaced by amino-acids that do not have rotamers: prolines and glycines are used where they appear in the query sequence, and all other sidechains are replaced with alanines (A/G/P representation). Because the inserts will not exactly fit the backbone, they are next allowed to relax in the A/G/P representation via conjugate gradient energy minimization. The minimization is carried out using either the PRESTO ver 3 [16] molecular dynamics package with AMBER force field parameters (default) or Gromacs [17]. To close the gaps between the insert ends and the template backbone, *Spanner* freezes all residues in the model except the inserted fragments, and runs the minimization for 1000 steps. Once the inserted backbones have been

positioned, the A/G/P representation is replaced with the actual query sidechains and their rotamers chosen using DEE or SCWRL4. Next, the entire structure is optimized in a three-step energy minimization procedure. The first step is similar to the backbone-only step above, and aims to relax the inserted fragments with the added sidechains: all residues except the inserts are frozen and the inserted fragments allowed to relax by conjugate gradient minimization. In the second step, the non-insert residues are allowed to move but their positions are restrained to their initial positions by a harmonic potential. In the third step, only the template backbone atoms are restrained and all sidechains are allowed to relax, producing the final model structure.

Web interface

Spanner is available through the web at
http://sysimm.ifrec.osaka-u.ac.jp/spanner/
The web server has the following functionalities.

1. Job scheduling. Jobs are run on a 200 core PC cluster, so multiple submissions without need for logging in are allowed.
2. Progress of each job can be monitored, and email notification is available but not required.
3. Users may select the minimization engine (Gromacs or Presto) as well as key parameters (margin and maximum anchor tolerance).
4. Structures can be visualized in 3D using the *jV* molecular viewer applet (http://www.pdbj.org/jv/index.html).
5. *Spanner* results can be exported to SeSAW, a functional annotation tool that uses sequence-weighted structural alignments to identify similar motifs in PDB entries [5].

In addition, the source code for building a local copy of *Spanner* can be downloaded from the above address.

Benchmark sets

Spanner requires a template and a pair-wise query-template alignment. To test *Spanner* three benchmark sets were assembled as follows.

Ig Set

In order to test the fragment retrieval function of *Spanner*, we selected third complementary determining regions of immunoglobulin heavy chains (CDR-H3s) from a representative set of antibodies. The selection of antibody structures were as described previously [18] and supplemented with recently registered entries in the PDB as of Mar. 2010. Briefly, all antibodies in the PDB with resolutions of 2.80 Å or better were extracted yielding a total of 776 structures having heavy and light chains. Then, structures with the highest resolution for each antibody were selected as representatives of free and complex structures, respectively, from the 776 structures. When more than one structure with the same high resolution was available, the structure with the best R-factor was selected. Consequently, we obtained 367 non-redundant antibody structures with CDR-H3 loop lengths from 5 to 22 (Table S1).

CASP Set

206 query template pairs were taken from historical results from the Critical Assessment of protein Structure Prediction (CASP) experiment in cases where the query has been solved and deposited in the PDB. In cases where alignments were deposited by the authors, these were used directly; in cases where only a 3D model was deposited and a single template was used, the alignment was estimated by structural alignment of the model onto the template using the program ASH [11]. These alignments represent a range of methods and naturally include a realistic level of noise. All query-template pairs are listed in Table S2.

ASH Set

Structural alignments, which represent essentially perfect input, were computed between 217 low homology query-template pairs using the program ASH. All query-template pairs are listed in Table S3.

Accuracy assessment

We assessed errors in the structural models using the root mean square deviation (RMSD) from the native structure coordinates.

Authors' contributions

ML wrote most of the original source code, TK developed ASH and CASP benchmarks, JJS co-developed source code, DK developed antibody benchmark set, HVD designed and developed web interface, ARK designed backend database, KM co-developed web interface, SD supervised code development, HN conceived of original methodology and co-authored the manuscript, and DMS managed project co-authored source code, and drafted the manuscript.

Acknowledgements

Funding: DMS was supported by a Grant-in-Aid for Scientific Research from the Japan Society for Promotion of Science. DK was a research fellow of Japan Society for the Promotion of Science and was supported by an Excellent Young Researcher Overseas Visit Program.

References

1. Matsushita K, Takeuchi O, Standley DM, Kumagai Y, Kawagoe T, Miyake T, Satoh T, Kato H, Tsujimura T, Nakamura H, Akira S: **Zc3h12a is an RNase essential for controlling immune responses by regulating mRNA decay.** *Nature* 2009, **458:**1185-1190.
2. Simons KT, Bonneau R, Ruczinski I, Baker D: **Ab initio protein structure prediction of CASP III targets using ROSETTA.** *Proteins-Structure Function and Genetics* 1999**:**171-176.
3. Wu S, Skolnick J, Zhang Y: **Ab initio modeling of small proteins by iterative TASSER simulations.** *BMC Biol* 2007, **5:**17.
4. Sali A, Blundell TL: **Comparative protein modelling by satisfaction of spatial restraints.** *J Mol Biol* 1993, **234:**779-815.
5. Standley DM, Yamashita R, Kinjo AR, Toh H, Nakamura H: **SeSAW: balancing sequence and structural information in protein functional mapping.** *Bioinformatics* 2009, **26:**1258-1259.
6. Sivasubramanian A, Sircar A, Chaudhury S, Gray JJ: **Toward high-resolution homology modeling of antibody Fv regions and application to antibody-antigen docking.** *Proteins* 2009, **74:**497-514.
7. Choi Y, Deane CM: **FREAD revisited: Accurate loop structure prediction using a database search algorithm.** *Proteins* 2009, **78:**1431-1440.
8. Jones DT: **Protein secondary structure prediction based on position-specific scoring matrices.** *J Mol Biol* 1999, **292:**195-202.
9. Arnold K, Bordoli L, Kopp J, Schwede T: **The SWISS-MODEL workspace: a web-based environment for protein structure homology modelling.** *Bioinformatics* 2006, **22:**195-201.
10. Frishman D, Argos P: **Knowledge-based protein secondary structure assignment.** *Proteins* 1995, **23:**566-579.
11. Standley DM, Toh H, Nakamura H: **ASH structure alignment package: Sensitivity and selectivity in domain classification.** *BMC Bioinformatics* 2007, **8:**-.
12. Kawabata T, Nishikawa K: **Protein structure comparison using the Markov transition model of evolution.** *Proteins-Structure Function and Genetics* 2000, **41:**108-122.
13. Desmet J, Demaeyer M, Hazes B, Lasters I: **The Dead-End Elimination Theorem and Its Use in Protein Side-Chain Positioning.** *Nature* 1992, **356:**539-542.
14. Tanimura R, Kidera A, Nakamura H: **Determinants of Protein Side-Chain Packing.** *Protein Science* 1994, **3:**2358-2365.
15. Krivov GG, Shapovalov MV, Dunbrack RL, Jr.: **Improved prediction of protein side-chain conformations with SCWRL4.** *Proteins* 2009.
16. Morikami K, Nakai T, Kidera A, Saito M, Nakamura H: **Presto(Protein Engineering Simulator) - a Vectorized Molecular Mechanics Program for Biopolymers.** *Computers & Chemistry* 1992, **16:**243-248.

17. Van der Spoel D, Lindahl E, Hess B, Groenhof G, Mark AE, Berendsen HJC: **Gromacs: Fast, Flexible, and Free.** *Journal of Computational Chemistry* 2005, **26:**1701-1718.
18. Kuroda D, Shirai H, Kobori M, Nakamura H: **Structural classification of CDR-H3 revisited: a lesson in antibody modeling.** *Proteins* 2008, **73:**608-620.