Spanner

OVERVIEW

Spanner is a structural homology modeling program that threads a query amino-acid sequence onto a template protein structure. Spanner is unique in that it handles gaps by spanning the region of interest using fragments of known structures.

Methods

Spanner Spanner consists of several modules that are managed by a Python driver script. The first step involves defining the start and end points of fragments corresponding to insertions or deletions. The start and end points are referred to as *anchors* because they must be equivalent in both the template and any candidate fragment. The *margin* parameter determines how far from the edge of a gap the fragment begins or ends. For example a margin of 0 would mean that the anchor begins at the very edge of a gap. This is usually not a good idea, and the default margin is set to 1.

A representative set of protein chains was prepared using the cd-hit program (Li and Godzik, 2006) at 100% sequence identity. All continuous fragments were then extracted from this set of chains and stored in a relational database, indexed by the internal coordinates of the fragment endpoints (figure 2). A separate database is prepared for each fragment length. Currently, fragments of length 4-40, including the 4 anchor residues, are stored in the database.

Spanner considers each fragment in order of it position in the query sequence. For a given fragment, a fragment index is generated from the anchor residues. Fragments are retrieved based on the anchor coordinate, fragment length, and predicted secondary structure of the query in the corresponding region. A specified *tolerance* in the fit to the anchor residues corresponds to a range of index values. The index range is sent to the database and all fragments satisfying the indices are returned. These fragments are then scored by a empirical score that considers sequence, secondary structure, the RMSD of the fitted anchor residues, and the number of backbone clashes between the fragment and the rest of the template.

$$S_{frag} = \frac{S_{seq} + S_{sec} - w_{clash} * N_{clash}}{RMSD}$$

where S_{seq} is a log-odds sequence substitution matrix score derived from a large number of structure alignments (Standley, et al., 2007), S_{sec} is a secondary structure (Kawabata and Nishikawa, 2000), w_{clash} is an adjustable parameter set to 0.05 by default, N_{clash} is the sum of clashes between the fragment backbone and the template structure excluding residues that are to be replaced by the fragment, and RMSD is the root-mean square deviation of the fitted anchor residues. The specified number of top-scoring fragments is output, and the best fragment selected by the above score in addition to the minimized energy.

Two options are available for replacing and optimizing their the conformations of side-chains. The first is to use the dead-end elimination method (Hess, et al., 2008). The second is to use the

SCWRL4 program (Krivov, et al., 2009). The complete model is then refined by energy minimization using either Presto (Morikami, et al., 1992) or Gromacs (Hess, et al., 2008).

USAGE

To create a model, you must provide a template structure, as well as an alignment of the query sequence you wish to model onto the template sequence. Spanner will replace matching residues, fill any gaps caused by insertions or deletions residues, and minimize the energy of the structure.

The resulting PDB-formatted model, as well as a log file, will be emailed to you when spanner is finished. If an error prevented a complete homology model from being generated (either due to an internal error or when the input can not be processed), the log file will explain which part of the modeling sequence failed. Alternatively, we provide an alignment 'sanity check' that will validate, and, if necessary and feasible, correct the input.

References

1. Hess, B., Kutzner, C., van der Spoel, D. and Lindahl, E. (2008) GROMACS 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation, Journal of Chemical Theory and Computation, 4, 435-447.

2.Kawabata, T. and Nishikawa, K. (2000) Protein structure comparison using the Markov transition model of evolution, Proteins-Structure Function and Genetics, 41, 108-122.

3.Krivov, G.G., Shapovalov, M.V. and Dunbrack, R.L., Jr. (2009) Improved prediction of protein side-chain conformations with SCWRL4, Proteins.

4.Li, W.Z. and Godzik, A. (2006) Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences, Bioinformatics, 22, 1658-1659.

5.Morikami, K., Nakai, T., Kidera, A., Saito, M. and Nakamura, H. (1992) Presto(Protein Engineering Simulator) - a Vectorized Molecular Mechanics Program for Biopolymers, Computers & Chemistry, 16, 243-248.

6.Standley, D.M., Toh, H. and Nakamura, H. (2007) ASH structure alignment package: Sensitivity and selectivity in domain classification, Bmc Bioinformatics, 8, -.