Using Iterative Dynamic Programming to Obtain Accurate Pairwise and Multiple Alignments of Protein Structures

Mark Gerstein & Michael Levitt

Department of Structural Biology, Fairchild D109
Stanford University, Stanford, CA 94305
{mbg,levitt}@hyper.stanford.edu

Abstract

We show how a basic pairwise alignment procedure can be improved to more accurately align conserved structural regions, by using variable, position-dependent gap penalties that depend on secondary structural regions, by using variable, position-

A number of procedures for automatic structural alignment and comparison have been developed (Taylor & Orengo, 1989; Russell & Barton, 1993; Holm & Sander, 1993; Sali & Blundell, 1990; Godzik & Skolnick, 1994;...
Here we present two modifications our previously described alignment procedure (Subbiah et al., 1993; Laurents et al., 1994) to make it more accurate and better able to align conserved core regions: variable gap penalties and noisy, suboptimal alignment. These modifications, which are novel to structural alignment, are direct analogs of common techniques in sequence alignment — for instance, for a discussion of variable gap penalties see Lesk et al. (1986), Smith & Smith (1992), and Vingron & Waterman (1994), and for a discussion of suboptimal alignment, see Zuker (1991) and Waterman et al. (1992). They are feasible for our structural alignment procedure because it is so similar to normal sequence alignment, involving repetitive application of Needleman-Wunsch (1971) dynamic programming. In contrast, many of the other commonly used approaches to structural alignment, which involve comparing distance matrices for two structures (Taylor & Orengo, 1989; Holm & Sander, 1993) or looking for similarities in a graph (Artymiuk et al., 1989), would not be modifiable in this way. After describing how our alignment procedure can be made more accurate, we sketch how it can be extended in straightforward fashion to generate multiple structural alignments, based on aligning all structures to a central or median structure. Our results in the area of multiple structural alignment are only preliminary and will be

Pairwise Structural Alignment

The procedure we use for pairwise structural alignment, described in Subbiah et al. (1993) and Laurents et al. (1994), is based on iterative application of dynamic programming. As such it is a simple generalization of Needleman-Wunsch sequence alignment (Needleman & Wunsch, 1971). As shown in figure 2, one starts with two structures in an arbitrary orientation. Then one computes all pairwise distances between each atom in the first structure and every atom in the second structure. This results in an inter-protein distance matrix where each entry \( d_{ij} \) corresponds to the distance between atom \( i \) in the first structure and atom \( j \) in the second one. This distance matrix can be converted into a similarity matrix \( S_{ij} \), similar to the one used in sequence alignment, by application of the following formula:

\[
S_{ij} = \frac{M}{1 + \left( \frac{d_{ij}}{d_0} \right)^2}
\]

Here \( M \) is the maximum score of a match, which is arbitrarily chosen to be 20. \( d_0 \) is the distance at which the similarity falls to about half its maximum value (i.e. \( d_0 = 1.5d_0 \)).
Figure 2: Schematic showing how pairwise structural alignment works. TOP-LEFT shows two structures (abcde and αβγ) in a random initial orientation. All pairwise distances are calculated between atoms in abcde to those in αβγ. These are converted into similarities (see text) and put into a matrix (TOP-RIGHT). Normal dynamic programming is performed on this matrix to find equivalences between atoms in the two structures (TOP-MID-RIGHT). Unlike sequence alignment, these equivalences are not globally optimal. To refine them, they are used to fit αβγ onto abcde in a least-squares sense. This gives the structures a new relative orientation as shown in MID-LEFT. Then the procedure is repeated: all pairwise inter-molecular distances are calculated between the structures (MID-LEFT), a matrix of similarities is formed (BOT-MID-RIGHT), and dynamic programming is done (BOT-RIGHT). This gives a second set of equivalences. These are used to refit the structures (BOT-LEFT), and everything is repeated iteratively until the procedure converges — i.e. there is no change in the equivalences between iterations.
Using Cβ atoms

The simplest improvement was to use Cβ rather than Cα atoms for the computation of distances d_{ij}. Using Cβ atoms makes misalignments by one residue in helices and especially strands more difficult. Misalignments by a single residue are not serious in terms of matching the overall fold but give nonsensical alignments in detail. For instance, in the case of strands they often lead to mismatching of hydrophobic and hydrophilic residues.

Secondary Structure Dependent Gap Penalties

Because of the similarity between our structural alignment procedure and normal sequence alignment, it is possible to incorporate variable, position-dependent gap penalties into the alignment in a very straightforward fashion. Since we know the secondary structure of the two proteins we are aligning (e.g. from DSSP, Kabsch & Sander, 1983) we can make it more difficult to introduce a gap at a position in a secondary structure (i.e. strand or helix). This is similar to sequence alignment methods that make the penalty for opening a gap depend on where it starts (Lesk et al., 1986; Smith & Smith, 1992; Vingron & Waterman, 1994).

We derived specific values for the gap penalties by empirically testing them on a number of protein families. We found that as the gap opening penalty is decreased in secondary structure relative to that in loops and coils, one obviously increases the number of spurious gaps in strands and helices. This suggests that very high gap penalties in strands and helices might work well. However, we also found that such high gap penalties make it more difficult to align secondary structural elements (which often vary slightly in size); in fact, a penalty that is too high leads to completely mismatching secondary structures. (For instance, instead of aligning two helices of slightly different size through introducing a gap into the longer helix, the program might introduce many gaps into a loop preceding one helix and align this helix against a loop and the second against the introduced gaps). The specific values we chose are a compromise between these two competing effects. We always set the gap extension penalty to be a small constant value (0.025 M). We arranged the gap opening penalties for each structure into a vector α(k), indexed by the sequence position i or j. Initially, the α(k) values were set to 2 in sheets and helices and 1 otherwise. α(k) is then smoothed (by convolution with a gaussian) and rescaled so that the overall average gap penalty \( \overline{\alpha}(k) \) is half the maximum match score M.

As described in figure 3, the introduction of variable gap penalties makes the dynamic programming rather complex, though it is still possible to achieve in roughly \( N^2 \) operations (where N is the average size of the sequences being aligned).

![Figure 3: The Complexities Introduced by Variable Gap Penalties](image)
Figure 4: Suboptimal Paths. This figure illustrates the idea of possible suboptimal paths in tracing back through the sum matrix $S_{ij}$ (see figure 3). Here a sum matrix is shown for aligning ABCDE with AyBCDE with a match score of 2 and gap-opening penalty of -1. To get the optimum traceback (which is indicated by black boxes), one starts at the overall maximum and progressively finds each succeeding maximum in the matrix (e.g. 8 → 5 → 3 → 2). However, if one perturbs the values in the matrix by the addition of random noise (e.g. by adding a series of random numbers $R_i$, between -2 and 2, to each matrix element), one may find slightly suboptimal alignments (indicated by gray boxes) that now have favorable scores. That is, it is now possible that $2 + R_i > 3 + R_{i+1}$ for the highlighted alternates on the second row (2 and 3). (White boxes have much lower scores and will never be included, even with the addition of random noise.)

Noisy, Suboptimal Structural Alignment

One of the goals in accurate structural alignment is to separate out those regions that match really well from those that match only partially well. We achieve this by doing a number of noisy structural alignments and taking the consensus. What is meant by a noisy alignment is described below in detail.

In normal dynamic programming, one builds up a sum matrix $S_{ij}$ from the similarity matrix $s_{ij}$, where each entry in $S_{ij}$ represents the best possible score one would get by starting at the beginning of the alignment and creating an alignment that ends by equivalencing position i in the first sequence with position j in the second sequence. As shown in figure 4, to find the overall optimum path, one usually imagines tracing back through this sum matrix starting from the entry with the maximum score. At each aligned point (i, j), one selects as the next aligned point the entry in the previous part of the matrix with the highest score — i.e. the point $k,l$ such that $S_{kl}$ is maximum and $k < i$ and $l < j$. Consequently, at each step in the traceback one is in a sense optimizing a score. If one deviates off this optimal path, one gets a suboptimal path or suboptimal alignment. One way to systematically deviate off this optimal path is to do the traceback in a Monte-Carlo fashion, always choosing the next point if it is much higher than its neighbors, but sometimes choosing a non-optimal neighbor (in a Boltzmann fashion) if it has nearly the same score. If this is done one will get a variety of different suboptimal but still relatively high-scoring traceback through the matrix.
Figure 6: Two Sample Multiple Alignments. This figure (adapted from Gerstein & Levitt, submitted) shows sample multiple alignments for two protein families. The first is for the dihydrofolate reductase (DHFR) family, and the second, for the globin family. For each family, in turn, two separate multiple alignments are shown: the one marked "MANU" is a manually constructed "gold-standard" from Gerstein et al. (1994), and the one marked "AUTO" is automatically generated. The manually and automatically generated alignments have been aligned as blocks so that they have the fewest possible mismatches. Mismatches are scored only in the core alignable regions, marked by a character (e.g. "*") in the "CORE" row. They are flagged in the automatically generated alignment (by double underlining, changing case, and substituting "-" for "."). The DHFR alignment has 1 mismatch in total and has ldhf as the central structure to which everything is aligned. The globin alignment has 18 mismatches and has lmbd as the central structure.
## Globin alignment

<table>
<thead>
<tr>
<th>Core</th>
<th><strong>core</strong></th>
<th><strong>manu</strong></th>
<th><strong>auto</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>2hhb-A</td>
<td>VLSPADKTNVKAAGKVGAGLEYGAEMAILFLSFPTTKTYFPF</td>
<td>VHLTFEESNTVALWGVV-NVDEVGGEALRLLVYYWTQRFPPSF</td>
<td>HLTPEEEKSVTGALWGVV-NVEVGGEALRLLVYYWTQRFPPSF</td>
</tr>
<tr>
<td>2hhb-B</td>
<td>VLSEGEWQLVLHVAKVEA-DNGAVGKDCLIKFLSAHPQMAVFGF</td>
<td>GLSAARQGVIATWKLITAG--ADNGAVGKDCLKFLSAHPQMAVFGF</td>
<td>SLAAEADLAGKSWAPVFA-NKNALDFVALFEKFPSANFFADF</td>
</tr>
<tr>
<td>21hb</td>
<td>PIVDTGSVAPLSAAEKTIRSAWAPVSTTYTSGVILVFKFTSTPAAEFPFKA</td>
<td>VHLTPEEKSAVTALWGVV--NVEVGGEALRLLVYYWTQRFPPSF</td>
<td>VIDPEEKSVTGALWGVV-NVEVGGEALRLLVYYWTQRFPPSF</td>
</tr>
<tr>
<td>2md</td>
<td>VLSPADKTNVKAAGKVGAGLEYGAEMAILFLSFPTTKTYFPF</td>
<td>VHLTFEESNTVALWGVV-NVDEVGGEALRLLVYYWTQRFPPSF</td>
<td>HLTPEEKSVTGALWGVV-NVEVGGEALRLLVYYWTQRFPPSF</td>
</tr>
<tr>
<td>2bg</td>
<td>LSADQISTVQAGSFDFKVKG--DPVGILYAVFKADPSIMAKFTQF</td>
<td>GLSAARQGVIATWKLITAG--ADNGAVGKDCLKFLSAHPQMAVFGF</td>
<td>SLAAEADLAGKSWAPVFA-NKNALDFVALFEKFPSANFFADF</td>
</tr>
<tr>
<td>2ba</td>
<td>LSADQISTVQAGSFDFKVKG--DPVGILYAVFKADPSIMAKFTQF</td>
<td>GLSAARQGVIATWKLITAG--ADNGAVGKDCLKFLSAHPQMAVFGF</td>
<td>SLAAEADLAGKSWAPVFA-NKNALDFVALFEKFPSANFFADF</td>
</tr>
</tbody>
</table>

*Gerstein 65*
The same effect can be achieved in a somewhat simpler fashion by adding an element of random noise to both the match score \( s_{ij} \) (and the gap opening and extension penalty). Here we take the noise to be between \( \pm 7.5\% \) of the maximum match score \( M \).

To highlight the most accurately aligned regions of a structure, we can generate a number of these noisy suboptimal alignments. Then we can take only the part of the alignment that is the same for each. This is shown for one particular case in figure 4, where the 434 repressor protein is aligned with myoglobin. The most similar helices are clearly conserved in the different suboptimal alignments.

**Multiple Structural Alignment**

We found it possible to form a multiple structural alignment from evaluating the results of all pairwise alignments (Gerstein & Levitt, submitted). We have tried to do this in a fairly straightforward fashion. After doing all pairwise alignments, we have picked the structure that is on average closest to all other structures. This is in the sense the "median" structure in the "cluster" of all the structures. We then align everything to this.

This presents one obvious problem: If position \( i \) in the median structure (i-in-median) aligns with position \( j \) in a second structure (j-in-2) and with \( k \) in a third structure (k-in-3), we would align all three positions together. However, this is only really a true multiple alignment if \( k \)-in-3 aligns to \( j \)-in-2 (and the gap opening and extension penalties). Here we take the noise as a random number of mismatches. We only count mismatches in structurally conserved regions as certain regions of the protein structure, particularly some surface loops, are impossible to align correctly. As is evident our multiple alignments are correct. En figure 66.

**Availability of Results on the Internet**

We make available over the Internet supplementary material relevant to this paper (e.g. manual and automatically generated alignments). Go to the following URL:

http://hyper.stanford.edu/~mbg/Align/

**Acknowledgments**

MG is supported by a Damon-Runyon Walter-Winchell fellowship (DRG-1272). ML acknowledges support from the Department of Energy (grant 2HDZ-477).

**References**


