Acta Crystallographica Section D
Biological Crystallography
ISSN 0907-4449

Automatic modeling of protein backbones in electron-density maps *via* prediction of C^{α} coordinates

Thomas R. Ioerger[a]* and James C. Sacchettini[b]

[a]Department of Computer Science, Texas A&M University, College Station, TX 77843-3112, USA, and [b]Department of Biochemistry and Biophysics, Texas A&M University, College Station, TX 77843-311, USA

Correspondence e-mail: ioerger@cs.tamu.edu

Received 31 May 2002
Accepted 13 September 2002

Most crystallographers today solve protein structures by first building as much of the protein backbone as possible and then modeling the side chains. Automating the determination of backbone coordinates by computer-based interpretation of the electron density would enhance the speed and possibly improve the accuracy of the structure-solution process. In this paper, a new computational procedure called *CAPRA* is described that predicts coordinates of C^{α} atoms in density maps and outputs chains of C^{α} atoms representing the backbone of the protein. The result constitutes a significant step beyond tracing the density, because there is ideally a one-to-one correspondence between atoms predicted in the chains output by *CAPRA* and C^{α} atoms in the true structure (refined model). *CAPRA* is based on pattern-recognition techniques, including extraction of rotation-invariant numeric features to represent patterns in the density and use of a neural network to predict which pseudo-atoms in the trace are closest to true C^{α} atoms. Experiments with several MAD and MIR electron-density maps of 2.4–2.8 Å resolution reveal that *CAPRA* is capable of building ∼90% of the backbone of a protein molecule, with an r.m.s. error for C^{α} coordinates of around 0.9 Å.

1. Introduction

Most crystallographers today solve protein structures by first building as much of the protein backbone as possible and then modeling the side chains. Automating the determination of backbone coordinates by computer-based interpretation of the electron density would enhance the speed and possibly improve the accuracy of the structure-solution process. In particular, once the coordinates of C^{α} atoms are known, a variety of automated techniques can be used to build in side-chain atoms, such as fragment-library approaches (Holm & Sander, 1991; Levitt, 1992) or the shape analysis of local density patterns (Oldfield, 1996; Whelan & Glasgow, 2000; Holton *et al.*, 2000). However, the most frequent approach used in practice today consists of skeletonizing a map to produce pseudo-atoms (*e.g.* using a tool such as *BONES*; Jones *et al.*, 1991) and then manually specifying the desired position of C^{α} atoms using interactive graphics software such as *O* (Jones *et al.*, 1991). It should be emphasized that while a skeleton (or 'trace') provides a very useful and compact representation of the density in complex maps, most skeletonization programs do not actually determine C^{α} coordinates (Greer, 1985; Swanson, 1994). Though C^{α} atoms are often found near branch points, there are many exceptions, such as breaks in continuity, false connections between side chains, missing branches owing to weak side-chain density or

extra branches in the side chains themselves. Thus, a great deal of human judgement is currently required to interpret electron-density maps.

In this paper, we introduce a new method for predicting chains of C^{α} atoms in electron-density maps, called *CAPRA* (C-Alpha Pattern Recognition Algorithm). The two essential features of *CAPRA* are (i) use of pattern-recognition techniques to recognize regions in a map that are likely to be centered on true C^{α} atoms and (ii) use of a variety of heuristics, *e.g.* based on geometry and connectivity, to decide how to link the predicted C^{α} atoms together into linear chains representing reasonable secondary structures. *CAPRA* was developed as the first stage of an automated model-building procedure in *TEXTAL* (Holton *et al.*, 2000). It has been used to automatically build models for a variety of experimental (MAD- and MIR-phased) electron-density maps in the 2.0–3.0 Å resolution range.

2. Background

Currently, the most common practice among crystallographers in solving protein structures is to identify C^{α} coordinates by hand, build the backbone into the model and then try to manipulate various side-chain rotamers to fit the local patterns in the electron-density map. However, there have been a number of efforts to incorporate automated methods into the process, ranging from skeletonization through attempts to build complete protein models in an entirely automatic way (Perrakis *et al.*, 1997). Some methods start by building a trace, choosing candidate C^{α} locations and then refining them based on geometric constraints. For example, in the X-Powerfit routine (Oldfield, 1996) of the *QUANTA* package, a chain of C^{α} atoms is extended by finding a position in the density that is 3.8 Å from the end of the chain. It balances a number of criteria, such as proximity to a branch point in the trace and acceptability of bond angles and torsions, based on probability distributions for typical C^{α} chains (Oldfield & Hubbard, 1994). Real-space refinement against these constraints is then applied to each chain built.

Another approach to identifying C^{α} locations is 'critical point analysis' (Leherte *et al.*, 1994), which involves analyzing peaks in the density map. In critical point analysis, gradients in the density are determined by computing the Hessian and this is used to search for a ridge (or linear sequence of high-density grid points) consisting of alternating peaks and saddle points, assumed to represent the protein backbone. These peaks are then connected together using minimum-spanning tree graph algorithms to generate the protein backbone.

A different approach is based on searching a map for regions of density that resemble prototypical secondary-structural elements (α-helices and β-sheets) and then utilizing this interpretation of the density to fit atoms in expected positions. This type of approach is generally known as *template convolution*. The 'templates' are fragments of prototypical secondary structure. Their fit to different positions in the map is calculated by evaluating similarity between the expected and actual density *via* convolution. One of the earliest versions of this approach, *ESSENS* (Kleywegt & Jones, 1997), performed the search in a straightforward way. However, the program is inefficient because it requires a six-dimensional search (three real-space dimensions plus three rotational dimensions for orientation of the fragment). A more computationally efficient version of template convolution was implemented in *FFFEAR* (Cowtan, 1998). *FFFEAR* converts fragment templates into their reciprocal-space coefficients by fast Fourier transform (FFT) and then searches the map for locations where the profiles match by simple multiplication and peak search. A real-space correlation search using Monte Carlo methods has also been used for rigid-body positioning of whole domains in maps (Diller *et al.*, 1999).

While template-convolution methods are capable of identifying C^{α} locations by exploiting the regularity of α-helices and β-strands, they must be complemented with other techniques to complete the structure by connecting these regular fragments through loops and random coil. *MAID* (Levitt, 2001) is an example of a program that does this. Prototypical helices and strands are shifted along the trace until a good fit to the density is detected. These core fragments are then incrementally extended by adding atoms in the density until the fragments connect. Each step is interleaved with real-space refinement to enhance the fit, while simultaneously enforcing sterochemical constraints on bond distances and angles.

A recent development has been to integrate building of partial models with phase improvement. Firstly, a map is searched for regions that look like prototypical structures (for example using template convolution), a partial model is constructed based on this and then the model is back-transformed to generate calculated phases which are combined with the original phases and experimental amplitudes to produce more accurate maps. This approach is the basis of *RESOLVE* (Terwilliger, 1999) and its recent model-building extension (Terwilliger, 2002) which implements a statistical form of density modification based on Bayesian probability. This formalism helps to address the problem of model bias by controlling the relative contribution to phase updating, based on the relative degree of belief in the initial phase estimates *versus* the quality of the fit of a fragment to the density. *MAIN* (Turk, 2001) also iterates model building with refinement. The unique aspect of *MAIN* is that it exploits the chirality of C^{α} atoms found in L-amino acids, which helps produce more accurate backbone geometry. As the cycles of refinement iterate, *MAIN* is able to produce increasingly interpretable maps and concomitantly more accurate and complete models.

Finally, C^{α} coordinates may be predicted in the process of building complete models, such as by *ARP*/*wARP* (Perrakis *et al.*, 1997). *ARP*/*wARP* uses the 'free-atom insertion' method to incrementally extend partial models by adding a few pseudo-atoms (scatterers) at the periphery of the model and refining their fit to the density through consistency with observed amplitudes in reciprocal space. These pseudo-atoms are then interpreted as side-chain and backbone atoms and included in the model and the whole process iterates. *ARP*/

wARP has been used to build very accurate models, including accurate coordinates for C^{α} locations. However, the success of *ARP*/*wARP*, as with several of the other approaches, seems to be limited to maps of relatively high resolution.

3. Methods

Given an electron-density map as input, *CAPRA* ultimately generates a set of C^{α} chains that characterize the protein backbone. The chains are represented in the form of a list of ATOM records (atomic coordinates) in PDB format. Although ideally *CAPRA* would output a single chain of length equal to the number of residues expected in the protein, it often outputs a set of smaller chains of varying length, depending on the quality of map.

CAPRA is based on the principles of pattern recognition. The goal of the software is to recognize when regions of density are located at or near a true C^{α} atom in a way that mimics the crystallographer's ability to recognize C^{α} locations visually. *CAPRA* uses a neural network which has been trained on prior examples of regions of density whose distance to true C^{α} atoms is known to make these predictions. Feature extraction is used to characterize patterns in the density and these features are provided as input to the neural network.

The *CAPRA* method consists of a sequence of eight major steps (Fig. 1). Firstly, the electron density is scaled to a uniform level, which is similar to map normalization. The map is then traced (or skeletonized) to produce a set of pseudo-atoms that lie along the centers of contours of density in the map. Each of these pseudo-atoms is considered to be a candidate for a predicted C^{α} atom. Next, numeric features are calculated based on patterns of the density in the region around each pseudo-atom. These features are input to a neural network that makes a prediction about the likely distance from each pseudo-atom to a true C^{α} atom.

Given this predicted distance for each pseudo-atom, a subset of the pseudo-atoms called 'way-points' is selected. The way-points are the pseudo-atoms predicted to be closest to true C^{α} atoms among their neighbors. The connectivity among way-points in initially determined by the trace. However, this produces an over-determined graph with excess links, including many branches and cycles. Hence, the next step is to reduce this graph to a subset of linear chains by making choices about which connections are most likely to represent legitimate backbone connections, as opposed to spurious connections through side chains, solvent, noise in the density *etc.* Finally, a simple refinement procedure is applied to the C^{α} chains to adjust the inter-C^{α} distance to be about 3.8 Å and to smooth out other imperfections, such as implausible angles.

CAPRA can work with maps in any space group, based on any unit-cell parameters; the unit-cell axes do not have to be orthogonal. No specific grid spacing is required. However, it is important that the map covers at least one entire molecule; cases where *CAPRA* is run on an asymmetric unit in which the borders of the map cut the molecule into pieces will not produce good results because *CAPRA* will end up identifying many short disconnected fragments that appear to terminate at the edge of the map.

3.1. Map scaling and tracing

The first steps in *CAPRA* are to scale the density and trace the map. Firstly, the map is scaled in order to make the magnitude of the density patterns roughly comparable among maps. The density-scaling process involves collecting statistics on 1000 randomly sampled 5 Å spheres throughout the map, restricting attention to the top 10% with the highest variance, which are most likely to represent protein rather than solvent. The threshold levels are determined above which 20% of the highest density points lie (θ_1) and below which 20% of the lowest density points lie (θ_2). All density values ρ_{ijk} are then scaled linearly so that these thresholds are mapped to ± 1, [*i.e.* $\theta_1 = 1, \theta_2 = -1$ and the average of the two thresholds is set to 0, $(\theta_1 + \theta_2)/2 = 0$],

$$\rho'_{ijk} = [\rho_{ijk} - (\theta_1 + \theta_2)/2]/[(\theta_1 - \theta_2)/2].$$

The result is similar to normalizing the map (so that it has a mean of zero and a standard deviation of 1.0), though the *CAPRA* scaling routine is less sensitive to varying proportions of solvent content.

Next, we trace the map. Our map-tracing routine, 'Tracer', is a variant of other standard skeletonization routines that have been described in the literature (Greer, 1985; Swanson, 1994; Jones *et al.*, 1991). An orthogonal 0.5 Å grid (with 0.5 Å spacing in x, y and z directions) is constructed over the entire map, densities are interpolated at each grid point and all the grid points that fall within a contour level of 1.0 (that is, with density > 1.0) are collected into a list. The 0.5 Å grid spacing was chosen as a compromise between the desire to provide initial candidate pseudo-atoms that are sufficiently close to true C^{α} atoms (maximum distance from any coordinate to the closest grid point is <0.877 Å) and the desire to minimize computation.

The initial list of grid points, called candidates, that fall inside the 1.0 scaled contour is sorted by density. Points in this list are then considered for deletion in order from the lowest density to the highest. A candidate grid point is deleted

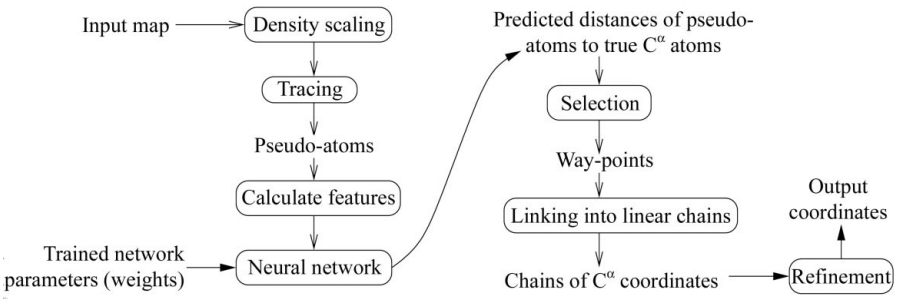

Figure 1
Steps in the *CAPRA* method.

unless doing so would create a discontinuity in the surrounding 3 × 3 × 3 box of grid points. A discontinuity occurs when, by removing the grid point in the middle, the remaining candidates in the surrounding box are broken into two or more separate clusters (see Fig. 2). Since this algorithm is applied to candidates in order from lowest density first, it has a tendency to incrementally remove grid points at the periphery of the regions inside the contours and these regions shrink toward the medial axis (ridge of highest density, along the center of the contoured regions) until a critical point is reached at which removing any more points would disconnect the trace. What remains is the skeleton of the density: chains of grid points within the map that represent the shape of the density contours in a compact form.

3.2. Prediction using a neural network

Given that the pseudo-atoms in the trace initially come from a 0.5 Å grid, our experiments have shown that there are typically around ten pseudo-atoms per residue, including 4–5 along the backbone as expected. Hence, one of the major challenges of *CAPRA* is deciding which subset of the pseudo-atoms represent C^{α} atoms in the true structure. The approach taken by *CAPRA* is to use pattern-recognition techniques to recognize which pseudo-atoms look more like C^{α} atoms, based on their surrounding patterns of density. This is accomplished in two steps: feature extraction and use of a neural network. In particular, the calculated features are used as input to the neural network, which predicts, for each pseudo-atom, a numeric estimate of how far away it appears to be from a true C^{α} atom.

Feature extraction involves calculation of various numeric values that characterize the patterns of density in regions around each pseudo-atom. In *CAPRA*, we define regions as spheres of up to 4 Å in radius. Since we ultimately want to recognize when a region is centered on a C^{α} atom regardless of the direction of the backbone or orientation of the side chain, it is important for the features to be *rotation invariant*, such that their scalar values do not change when an arbitrary rotation transform is applied. We use the same rotation-invariant features in *CAPRA* as are used in *TEXTAL*. Examples include mean and standard deviation of density in region, higher-order statistics, distance to center of mass, moments of inertia, ratios of moments of inertia and some specialized geometric features (see Holton *et al.*, 2000, for details of feature definitions). These feature values are all independent of orientation (*i.e.* they remain constant even if the region is rotated) and represent aspects of the shape and symmetry of the density patterns in a quantitative way. In *CAPRA*, these features are calculated over spheres of both 3 and 4 Å radius. Counting both radii, there are a total of 38 features.

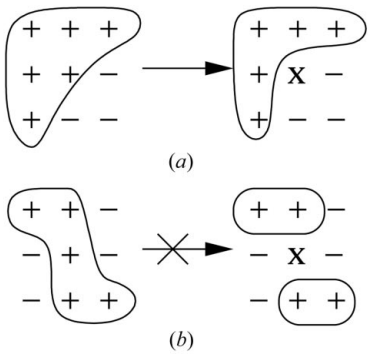

Figure 2
Illustration of how Tracer determines local connectivity for decision-making during the reduction of grid points. Initially, all grid points in the contour are included. They are then incrementally dropped, in order of lowest density first, as long as this does not break local connectivity. Local connectivity is determined by examining the points in a surrounding 3 × 3 × 3 box (shown in two dimensions here); + signs are points with high density, while − signs are points outside the contour. Panel (*a*) shows a case where dropping the point in the middle will not create disconnected components, so the point may be eliminated. Panel (*b*) shows a case where dropping the point in the middle would create two separate connected components. Hence, this point would not be eliminated because it is critical to maintaining the connectivity of the trace.

Once these features are calculated, they are fed into a neural network to predict how far away the center of the region is from a C^{α} atom in the true structure. The neural network is a standard feed-forward network (Hinton, 1989) with one input layer consisting of the 38 features, one layer of hidden units with sigmoid thresholds, and one output node: the predicted distance (unthresholded). The hidden layer has 20 nodes and the network is fully interconnected between layers (Fig. 3). The size of the hidden layer was chosen to be 20 nodes because experiments showed that fewer nodes (10) resulted in convergence to a higher mean-squared error, while more nodes (30) did not significantly improve the accuracy of the network. Each link between nodes a and b has a real-valued weight $w_{a,b}$. Each node, j, has an internal activation level, act_j, and an output, out_j, which is the thresholded version of the activation. In general, the activation levels of non-input nodes, j, in the network are calculated as weighted linear sums over their inputs, i, plus a tunable bias parameter (Fig. 4),

$$\mathrm{act}_j = \sum_i w_{i,j} \cdot \mathrm{out}_i + \mathrm{bias}_j.$$

For the single-output node the output level is equal to its activation, whereas for the hidden nodes the output is the

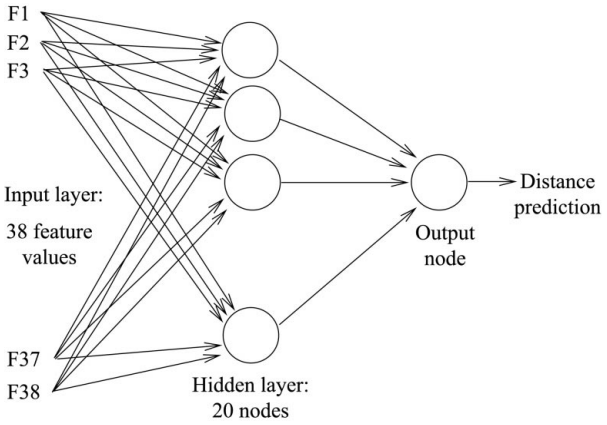

Figure 3
Structure of neural network used in *CAPRA*.

thresholded version of the activation using the sigmoid squashing function,

$$\mathrm{out}_j = \frac{1}{1+\exp(-\mathrm{act}_j)}.$$

Feed-forward neural networks of this type can be trained using the well known back-propagation algorithm (Hinton, 1989). Back-propagation is based on running the neural network forward on training data, for which the target output values are known, and then distributing the blame for discrepancies backward through the network in proportion to the contribution of the nodes, in order to update the weights on the links (the free parameters of the model). If the relative errors on the successors of node j are δk, then the error on node j can be calculated as

$$\delta_j = \mathrm{out}_j(1-\mathrm{out}_j)\sum_k w_{j,k}\cdot\delta_k.$$

This formula is determined from the partial derivative of the error of the whole network as a function of the output of node i using the chain rule and exploiting the differentiability of the sigmoid function. The weights are updated in proportion to these internal distributed errors,

$$\Delta w_{j,k} = -\eta \mathrm{out}_j \delta_k,$$

where η is a learning-rate constant. The bias parameter for each node can be updated similarly by treating it in a uniform way as a weight on a link of constant input value, 1.

This back-propagation process is iterated over many training examples. The training examples consisted of a sample of grid points in a calculated (F_c) map at 2.8 Å resolution. The map was generated using the coordinates from the large α/β protein 1fdi, with 715 residues. The structure factors were calculated from the model, using B factors as given in the original PDB file, and a map was then calculated with reflections from 20 down to 2.8 Å by FFT using *X-PLOR* (Brünger, 1992). After scaling the map, a random subset of 10 000 grid points that occurred inside the 1.0 contour level was selected. For each point, the feature vectors were calculated and the true distance to the closest C^α atom was measured, which ranged from almost 0 up to 6 Å (but still inside the 1.0 contour). The measured distances were used as the target values on the output node for back-propagation. The network was trained for over 100 epochs, using an adaptively decreasing learning-rate parameter, until the network error converged. The final mean-squared error of the distance predictions was 0.592.

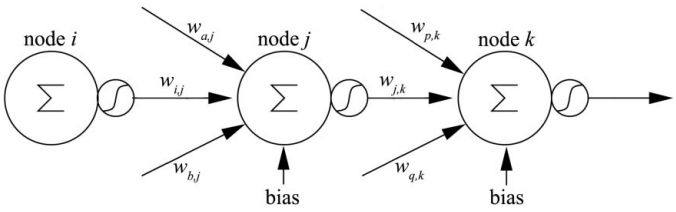

Figure 4
Details of a typical node in a neural network. The central node, j, first takes a weighted sum over its inputs (large circle with a Σ) and then thresholds this value to output a signal between 0 and 1 (small circle with an S curve). Each node also has a bias input, which, along with all the weights in the network, is trained *via* back-propagation. In a network with a single hidden layer, as is used in *CAPRA*, the input layer (containing node i) would contain the input feature values and the output layer (node k) would have a single node without a threshold.

While the map of a large protein with a mixture of secondary-structure types was intentionally chosen for training to minimize bias (*i.e.* to ensure a representative distribution of patterns from helical, extended and coiled regions), it should be acknowledged that this single map calculated with model phases does not represent the full spectrum of types of noise and errors found in real maps, which can be derived from a wide range of factors affecting data measurement. Nonetheless, our results show that use of patterns from this idealized map to train the neural network enables it to recognize C^α locations with reasonable accuracy in real maps, at least at resolutions around 2.8 Å. The resolution limit of 2.8 Å was chosen for the training set because this is in the middle of the typical range of resolutions that can be expected to come from data sets collected at synchrotrons *via* MAD.

3.3. Selection of way-points

Prediction of the approximate distance from each pseudo-atom in the trace to a true C^α atom is used to help select the subset of the trace atoms that most closely represent C^α atoms. These are called way-points. Way-points are selected in the following manner. Firstly, all the pseudo-atoms are ranked by their predicted distance to a C^α atom, from closest to farthest. Pseudo-atoms with predicted distances to C^α atoms of >3 Å are discarded, as they are very unlikely to represent true C^α atoms and are more often associated with side chains. The top atom in the list is then selected as a way-point, or candidate C^α atom, and all the remaining atoms in the list that are within 2.5 Å of this atom, which have worse predictions by definition, are deleted. This process is repeated, picking the next best pseudo-atom, removing its neighbors and so on. Candidate C^α atoms are thus picked in a random order throughout the map, giving preference to the best ones first according to the neural net predictions.

After picking the way-points, their connectivity is established based on the trace. From each way-point, contiguous sequences of adjacent pseudo-atoms are followed from the trace radially outward until either another way-point is encountered or a 5 Å cutoff is reached. While there is no enforcement or constraint on the distance between neighboring way-points (only that they are no closer than 2.5 Å and no farther than 5 Å), they are on average spaced about 3.8 Å apart, as expected; furthermore, there is typically only one way-point in the proximity of each true C^α atom (*i.e.* a one-to-one correspondence).

3.4. Extraction of linear chains

Since the connectivity among selected way-points, or candidate C^α atoms, is determined by the trace of the density, this often produces an over-connected graph with many

extraneous edges ($C^\alpha-C^\alpha$ links). For example, there may be branches (*e.g.* into side chains) or loops owing to disulfide bridges, side-chain or backbone contacts, or noise in the density. This graph must ultimately be reduced to a sub-graph of linear chains representing only the protein backbone.

The process of extracting linear chains of C^α atoms starts by computing connected components of the graph. This is accomplished by partitioning them according to the connectivity relation, *i.e.* iteratively merging groups of nodes whenever a connection between them exists until only separated clusters remain. Chains are then extracted using one of two strategies depending on the size of the connected component. Both strategies attempt to identify long chains that contain way-points with good neural net scores; that is, low predicted distances to true C^α atoms. Both strategies also try to follow plausible secondary structures recognized in the map. Putative secondary-structure elements are identified by looking for connected fragments of seven consecutive C^α atoms that are either relatively straight or helical, according to geometric constraints, as described below and links between C^α atoms in such fragments are given preference. For smaller components (with <20 way-points), an exhaustive search is performed over all pathways between any pair of atoms that can be connected to identify the chain that maximizes a score (see below) based on these criteria. For larger connected components (with $\geq$20 way-points), a strategy is used in which branch-points (way-points with connections to three or more neighbors) are individually considered for reduction to a linear path by clipping one of the extra branches. A scoring function is described below that is again designed to reflect the preference criteria above. The most obvious branch links are dropped first and this process is repeated until there are no more branch-points remaining, leaving a sub-graph consisting of typically several linear chains.

3.4.1. Secondary-structure analysis. To formalize this, we first need to describe how the putative secondary-structure elements in the map are analyzed. An exhaustive list of all heptamers of connected C^α atoms (fragments of length seven) throughout the entire map is formed. Each fragment is then evaluated geometrically for linearity or helicity. The linearity of a fragment is determined by the ratio between the overall end-to-end distance and the sum of the distances of consecutive pairs of atoms in the fragment,

$$\text{linearity}(p_1 \ldots p_7) = \frac{\text{dist}(p_1, p_7)}{\sum_{i=1}^{6} \text{dist}(p_i, p_{i+1})}.$$

A connection between way-points is identified as a potential link within a β-strand if it belongs to a fragment whose linearity score is >0.8. The helicity of a fragment is determined by average absolute deviation of C^α bond angles from 95° and torsion angles from 50° among consecutive way-points along the fragment (these angles are the ideal parameters for a helix; Oldfield & Hubbard, 1994),

$$\begin{aligned}\text{helicity}(p_1 \ldots p_7) = {} & \frac{1}{5}\sum_{i=1}^{5} \text{abs}[95 - \text{angle}(p_i, p_{i+1}, p_{i+2})] \\ & + \frac{1}{4}\sum_{i=1}^{4} \text{abs}[50 - \text{torsion}(p_i, p_{i+1}, p_{i+2}, p_{i+3})].\end{aligned}$$

A connection between any pair of way-points in the graph is identified as a potential link within an α-helix if it belongs to a fragment whose helicity score is <20.0 (representing mean absolute angle deviation).

3.4.2. Exhaustive search for chains in small connected components. This secondary-structure analysis is used to extract chains from both small and large connected components. For small connected components, with fewer than 20 way-points, an exhaustive depth-first search is used to enumerate all possible paths between each pair of way-points. Each path is scored according to the function

$$\text{score}_{\text{short}}(p_1 \ldots p_n) = n - \sum_{i=1}^{n} \text{pred}_i + \sum_{i=1}^{n-1} \text{ss}(p_i, p_{i+1}),$$

where n is the length of the chain, pred_i is the neural net prediction of distance from way-point i in the chain to the closest C^α atoms and $\text{ss}(i, i+1)$ is a bonus for links along secondary structures,

$$\begin{aligned}\text{ss}(p, q) = 0.8 \text{ if } (p, q) \in \text{helices} \\ \text{or } 0.6 \text{ if } (p, q) \in \text{strands, else } 0.\end{aligned}$$

Slight preference is given to helices since they are recognized with greater specificity in maps by the procedure; straight fragments of length seven are sometimes observed traversing perpendicularly across the main chain along spurious side-chain connections. The chain in the connected component that maximizes this score is returned. Hence longer chains are preferred, especially if they follow secondary structures, as long as they do not contain way-points with poor (high) predicted distances to C^α atoms. Chains of length less than six are filtered out and dropped from the output.

3.4.3. Clipping branch points in large connected components. For larger connected components, with 20 or more way-points, this exhaustive enumeration of chains is too inefficient. Instead, a heuristic approach is taken in which branch points are identified and incrementally reduced by clipping one or more incident edges until the resulting sub-graph is completely linear. Firstly, cycles in the connected component are eliminated by removing all $C^\alpha-C^\alpha$ links and then adding them back one at a time in order from best link to worst (the quality of a link is defined to be the maximum of the predicted distances to a true C^α between the two way-points that are connected by the link), while rejecting those edges that would create a cycle. Hence, the worst links, *i.e.* those adjacent to way-points with high predicted distances to C^α atoms, tend to get dropped to break cycles. Next, all branch-points are reduced to at most three-way connections by dropping whichever additional links connect the branch-point to the fewest atoms (smallest sub-component) of the graph (recall that the graph is acyclic now, so each link at a branch-point connects to a disjoint subset of atoms in the graph).

At this point, we can assume the large connected component is acyclic and contains only nodes with at most three connections (degree ≤ 3). The goal is to produce a linear sub-graph that contains only nodes with degree at most two. For each branch point, there are three possible actions, corresponding to clipping one of the three links. Hence, we use an evaluation function to determine which link of which branch point is best to clip. Note that clipping any such link at a branch point will produce two disconnected components owing to the acyclicity.

Suppose a way-point w is a branch point connected to three other atoms called p, q and r. The score of clipping a particular link, say $w-p$, is computed by (i) rewarding it if $w-q$ and $w-r$ are members of a putative helix or strand, (ii) penalizing it if $w-p$ itself is a member of an apparent secondary structure and hence bad to clip, (iii) rewarding it if the set of atoms that would be clipped off is small, (iv) rewarding it if the sets of atoms that remain connected are both large, (v) rewarding it if there is a nearby way-point, up to three steps away, with a poor (high) predicted C^{α} distance in the resulting clipped components and (vi) penalizing it if a nearby bad atom remains in the non-clipped portion.

Formally, the score is calculated in two phases: firstly, the atoms are ranked by consensus votes and then a quantitative score is computed to break ties and refine the ranking. According to the criteria above, clipping a given link at a given branch point might satisfy or violate a number of rules, both for the link to be clipped, say $w-p$, and for the two that remain, $w-q$ and $w-r$, forming the sequence $q-w-r$. However, for example, if the two links that remain connected are part of a common secondary structure and the link that is clipped is not part of any secondary structure, or leads off into a sub-graph of only a few atoms, or is connected to an atom with a very poor neural net score, then these criteria reinforce each other and the decision is relatively obvious that this is a good link to clip.

Thus, each link is analyzed by the above criteria and points, or votes, are accumulated. A link to be clipped receives two points if the non-clipped links at the same branch point are part of a putative helix and one point for a strand. It receives an extra vote if it connects to a group of atoms of size < 10 but the other links connect to groups of size > 20, so that preference is given for keeping links that maintain connectivity of long chains. Finally, if a link at a branch point leads to a small group of atoms (size < 6) that contains a way-point whose predicted distance to a C^{α} is >2.0 Å, it is given another vote for being clipped.

If only one link at a given branch point received any votes, then it may be clipped and the degree of priority is proportional to the number of votes it received. However, there are also frequently ambiguous cases, where there are votes for clipping more than one of the links at a given branch-point, making the decision less obvious. To resolve these ambiguities, a refinement of the score is computed. It is the number of votes, minus 0.04 if the link itself is in a helix, minus 0.02 if the link is in a strand, plus a weighted function of the size of the clipped and non-clipped fragments. If the clipped fragment is small (size ≤ 6), then 0.04 is added for each atom less than six; the same points are subtracted if the size of the non-clipped fragment is this small. If the clipped or non-clipped fragment is of medium size ($6 < n \leq 20$), a penalty of 0.0015 is subtracted for each atom less than 20 (but >6); this is because, all things considered, clipping off groups of atoms of this intermediate size (*e.g.* 6–20) tends to fragment the resulting structure excessively. These microscores typically cannot override the number of votes received by a link, but they can be used to refine the ranking and typically play more of a role in making decisions when it is not obvious which link at a branch-point to clip.

While the complex scoring functions used at this stage may seem simplistic (in comparison to the neural network), they reflect a knowledge-based approach in which expertise about how to distinguish the true backbone pathway from among a set of pseudo-atoms is encoded in weights that are fine-tuned for making the correct decisions.

3.4.4. Refinement. The final step in the *CAPRA* process is a modest form of refinement of the C^{α} coordinates. Owing to the manner in which way-points are selected, adjacent C^{α} atoms do not always have exactly a 3.8 Å distance. The way-points are initially selected from pseudo-atoms in the trace, which are restricted to a 0.5 Å grid, and they are chosen in random order throughout the map based on the neural net predictions, without regard to the distance to neighbors, except that they are constrained to not be closer than 2.5 Å. Hence, the distance between consecutive C^{α} atoms tends to vary widely between 2.5 and 5 Å. Also, there can be implausible angles, such as kinks in the chain with angles $<70^{\circ}$.

There are many possible approaches to refinement of the C^{α} chains. One approach would be to build and refine a poly-alanine backbone (similar to *MAIN*; Turk, 2001) and then apply standard real- or reciprocal-space refinement. The constraints on chirality of C^{α} atoms, commitment to directionality of backbone and planarity of the peptide bond can help place the C^{α} atoms in more plausible positions. A similar approach would be to actually build a complete model, with all the additional backbone and side-chain atoms (*e.g.* using the *TEXTAL* program; Holton *et al.*, 2000) and then apply real-space refinement to that model. A very different approach would be to clean up the geometry by recognizing secondary-structure elements in the map and replacing them with 'idealized' templates [see *MAID* (Levitt, 2001) and *RESOLVE* (Terwilliger, 2002)].

In contrast, the approach used in *CAPRA* is a simplified refinement of pure C^{α} chains without adding additional backbone atoms. First, a pair of adjacent C^{α} atoms is randomly picked and their coordinates are each perturbed randomly and independently by up to 0.3 Å in all three directions. An objective function is then evaluated and if the energy has gone down the perturbations are kept, otherwise the perturbations are discarded. This is repeated for each chain until no decreases in energy are observed for 1000 iterations. The objective function has the following terms: (i) a quadratic term for deviation of distance between consecutive C^{α} atoms from 3.8 Å, (ii) a quadratic term for deviation of distance of each C^{α}

Table 1
Proteins used in this study.

Protein	PDB code	Final resolution (Å)	Method	Resolution used in this study (Å)	Secondary structure	Size (No. of amino acids) refined/total	Phase error† (°)
CzrA		2.3	MAD/MR	2.8	α	94/104	46
IF-5a	1bkb	1.75	MAD	2.8	β	136/139	30
MVK	1kkh	2.4	MAD	2.4	α/β	317/317	46
PCA	1l1e	2.0	MAD	2.8	α/β	262/287	54
P2 myelin	1pmp	2.7	MIR	2.7	β	131/131	n.a.

† Phase error was calculated for each map as the mean absolute difference between the experimental phases for the map used (not available for P2) and ideal phases calculated from the refined model by FFT.

from its original position, (iii) a quadratic term for angles of $<90^\circ$ among three consecutive atoms and (iv) the magnitude of the density. Torsion angles are not restrained.

This simple refinement procedure tends to regularize the backbone so that the $C^\alpha-C^\alpha$ distances are more reasonable, typically in the range of 3.6–4.0 Å. This distance improvement also improves the visual appearance of helices and strands slightly. However, the overall r.m.s.d. of predicted C^α coordinates to those in a refined model is reduced by only a tenth of an ångstrom or so.

4. Results

To evaluate *CAPRA*, we ran it on several real electron-density maps generated from MAD or MIR experimental data. The maps cover a range of medium resolutions (2–3 Å) and include a variety of α-helical and β-sheet structures. Table 1 summarizes the details of these test cases. All of these maps have had some form of density-modification applied, such as solvent flattening (Cowtan, 1998).

CzrA (chromosome-determined zinc-responsible operon A) is a metal-ion regulatory protein from *Staphalococcus aureus* (Christoph Eicken, manuscript in preparation). It consists of four α-helices and is a dimer in the asymmetric unit. CzrA was originally solved by collecting a 2.8 Å MAD data set, building a model and then performing molecular replacement with a similar protein recognized in the PDB, with phase extension to 2.3 Å. In these experiments, we used the earlier 2.8 Å map.

IF-5a (translation initiation factor 5a) is involved in protein synthesis and cell-cycle regulation in *Pyrobaculum aerophilum* (Peat *et al.*, 1998). It consists of a pair of SH3-like β-barrel domains and has a disordered N-terminal tail of about eight amino acids. IF-5a was initially solved at 1.75 Å using *RESOLVE* (Terwilliger, 1999). In our experiments, we used a 2.8 Å map generated by limiting the structure factors used in the FFT, since *CAPRA* works best at this resolution (see below).

MVK (mevalonate kinase) is a metabolic protein from *Methanococcus jannaschii* (Yang *et al.*, 2002). MVK is a medium-sized protein with 317 amino acids, including both α and β secondary structures. The map for MVK was generated from 2.4 Å MAD data.

PCA (mycolic acid cyclopropane synthase) from *Mycobacterium tubeculosis* is involved in the fatty-acid biosynthesis pathway I (Huang *et al.*, 2002). PCA has a deep hydrophobic pocket in which a long-chain fatty acid is inserted with the help of ACP (acyl-carrier protein). At the base of the pocket, a ligand, *S*-adenosyl homocysteine (SAH), sits ready to transfer a methyl group to reduce a double bond in the mostly saturated aliphatic chain. Density for the SAH molecule is visible in the map. However, the density for a large 12-residue loop which is purported to be the docking location for ACP is very weak and was not built in the original model. The original map was generated at 2.8 Å, after which a second round of data collection was performed and the phases were extended to 2.0 Å to build the refined structure. The earlier 2.8 Å map is the one used in this study.

P2 myelin is a retinol-binding protein that performs fatty-acid transport (Cowan *et al.*, 1993). This protein consists of a β-barrel fold and was crystalized with a molecule of oleic acid bound. P2 myelin crystallizes as a trimer in the asymmetric unit and the structure was originally solved (built manually) from an MIR map calculated at 2.7 Å resolution. In the map used in our experiments, ten rounds of symmetry-averaging had been applied using *RAVE* (Kleywegt & Read, 1997), producing very high-quality density.

Fig. 5 shows an example of some of the main steps in *CAPRA* for a portion of CzrA. The four panels show the trace of the initial density (contoured at a threshold of 1.0 in the scaled map), choice of way-points (subset that are predicted to be locally closest to true C^α atoms), connectivity of way-points (based on the trace) and finally selection of a subset of the links that form linear chains (*i.e.* eliminating cycles and clipping branches).

One of the key computational steps in *CAPRA* is estimation of the distance between pseudo-atoms in the trace and true C^α atoms in the map, which is predicted using the neural network. Fig. 6 shows a histogram of the errors in these predictions for pseudo-atoms in the trace of MVK that are between 0 and 5 Å away from a true C^α atom in the refined structure. The errors are calculated as absolute value of the difference between the predicted and actual measured distances. The distances predicted by the neural network are fairly well correlated with the true distances ($r = 0.56$) and the vast majority of points are predicted to within 1 Å of their true distance. Hence, the neural network predicts closer atoms to have smaller distance values and atoms with larger predicted distances are less likely to be near true C^α atoms. The method by which way-points are selected exploits this effect by choosing pseudo-atoms in the trace that are predicted to be closest to true C^α atoms in each local area, provided they are at least 2.5 Å apart.

The results of running *CAPRA* on these five models are summarized in Table 2. The portion of the molecule that was interpretable by *CAPRA*, relative to the portion of the map

Table 2
Results of *CAPRA*.

Protein	Resolution used† (Å)	Portion of structure built‡ (%)	R.m.s. error§ (Å)	No. of chains output	Length of longest chain	No. of insertions and deletions¶	No. of atoms under 1 Å††	Cross-overs‡‡
CzrA	2.8	81 (84/104)	1.08	5	53	2, 2	41 (49%)	0
CzrA/MR§§	2.3	80 (83/104)	0.69	2	64	0, 1	76 (92%)	0
IF-5a	2.8	93 (127/136)	0.78	4	52	3, 1	108 (85%)	0
MVK	2.4	95 (298/317)	0.83	6	101	4, 6	236 (79%)	0
PCA	2.8	81 (212/262)	0.89	11	50	2, 4	150 (71%)	1
P2 myelin	2.7	85 (111/131)	0.91	6	63	3, 2	84 (76%)	2

† The resolution of the map given as input to *CAPRA*. ‡ The ratio and proportion of the structure built, compared with the manually built and refined model. § The r.m.s. error of the C^{α} predictions relative to the refined model. ¶ The number of insertions and deletions, determined by one-to-one matching to closest C^{α} atoms in the refined structure. †† The number of atoms in the *CAPRA* model that had less than 1 Å r.m.s. error. ‡‡ The number of spurious connections, or crossovers, between non-contiguous parts of the molecule. §§ For comparison, the results of *CAPRA* on the 2.3 Å molecular-replacement map for CzrA are included in this table, though our conclusions in this paper are based only on the MAD and MIR maps.

interpretable *via* manual model-building, ranges from 80–95%. The r.m.s. error of the predicted C^{α} coordinates, relative to their true coordinates in the refined model, is 0.8–1.1 Å for *CAPRA* on these maps. (A breakdown of these r.m.s. scores in terms of secondary structure is presented later.) Typically, *CAPRA* produces several (4–11) chains for these structures, with at least one chain of length greater than 50 residues in each case. However, the number of insertions and deletions was low; only a few (1–6) atoms were not matched one-to-one in each structure. On average, more than half of the molecule was fitted with coordinates that had less than 1.0 Å r.m.s. error. The *CAPRA* chains generally followed the true connectivity of the backbone and only demonstrated cross-overs between non-contiguous portions of the molecule in three cases.

In CzrA, *CAPRA* built five chains, of which one covered almost half of the molecule (53/104 residues). In addition, there were several 'tails' that crossed over into the other molecule in the asymmetric unit, owing to close contacts within the dimer (these were manually clipped off for the analysis above). *CAPRA* did not build parts of the N- and C-terminal ends, as well as a β-hairpin loop, where density was quite weak owing to disorder. The *CAPRA* model for CzrA had the highest r.m.s. error (1.08 Å) of all five test cases. However, after an initial model had been manually built with this 2.8 Å map, a structural homolog was discovered in the PDB and the final structure was solved at 2.3 Å by molecular replacement (MR). In this 2.3 Å MR map, *CAPRA* was able to fit the same portion of the structure (about 80%) with an r.m.s. error of only 0.69 Å, including two long chains of 64 and 19 residues interrupted only by three disordere residues in the β-hairpin region.

In IF-5a, *CAPRA* built four chains, including chains with 52 and 46 residues. The density was of high quality, owing to refinement with *RESOLVE*, and the r.m.s. error for the C^{α} atoms predicted by *CAPRA* was 0.78 Å, covering 93% of the molecule. In one experiment, the resolution of the map for IF-5a was varied by limiting the structure factors used in the FFT. For a 2.1 Å version of the map, the accuracy of *CAPRA* was found to be only 1.23 Å, while for a 2.8 Å version of the same map the accuracy of *CAPRA* was 0.86 Å (before running the final C^{α} refinement step). We conclude from this experiment that *CAPRA* probably works best at 2.8 Å, because that is the resolution of the map on which the neural network was trained. Further evidence for this is given below.

MVK is a medium-sized protein with 317 residues and a mixture of α-helices and β-sheets. *CAPRA* fitted 95% of the molecule with an r.m.s. error of 0.83 Å. Fig. 7 shows some examples of the chains of C^{α} atoms built by *CAPRA* for MVK. Even the pleats in the β-sheets are modeled correctly.

PCA was more fragmented than the other models, with seven of the 11 chains having a length less than 20. This map was only of medium quality, based on the density generated from an initial MAD data set of 2.8 Å resolution that had been collected, although the final refined structure was solved at 2.0 Å using a second round of data collection. In addition to 15 residues on the N-terminus, there are residues forming several large helices that project out from the molecule which could not be built (manually or by *CAPRA*) in the initial map. However, there were also two small loops (four and seven residues) and a buried β-strand (eight residues long) that were not built by *CAPRA*, even though they did have well defined density. Furthermore, there was a spurious connection between residues 227 and 280 through some noisy density in the core of the molecule. Finally, there was one span of three residues (33–35) in the core which *CAPRA* bypassed with three 'false' C^{α} atoms through a patch of density arising from a glutamine side chain from a non-contiguous part of the molecule, giving the appearance of a brief separation from the main path of the backbone. Correctly, *CAPRA* did not fit the density for the SAH ligand buried in the core of the molecule, probably because the corresponding region of density was small and not connected to anything else.

Hence, while *CAPRA* built most of the PCA molecule (81%) with high accuracy (0.89 Å), it made several errors, some of which were genuine mistakes and others of which arose from either absent or noisy density. For a comparison, a $2F_o - F_c$ map (with calculated rather than experimental phases) was constructed for PCA from structure factors deposited in the PDB (PDB code 1l1e). Two maps at different resolutions were generated. For one map at 2.0 Å resolution, the accuracy of the C^{α} atoms predicted by *CAPRA* was found to be 1.10 Å. For a 2.8 Å map generated from the same data, the r.m.s. error was found to be 0.86 Å. These observations, along with those for IF-5a above, suggest that either (i) higher resolution maps should be systematically reduced in resolution before running *CAPRA* on them or (ii) different parameter sets for the neural network should be generated by training them on different resolution maps.

In the 2.7 Å MIR map for P2 myelin, *CAPRA* produced six chains of lengths ranging from eight to 63 residues. These

Table 3
Analysis of r.m.s. scores by secondary structure based on *DSSP*.

	CzrA	IF-5a	MVK	PCA	P2 myelin
R.m.s. for α (Å)	1.03 (n = 58)	0.57 (n = 4)	0.76 (n = 128)	0.87 (n = 99)	1.15 (n = 16)
R.m.s. for β (Å)	1.25 (n = 11)	0.78(n = 75)	0.78 (n = 76)	0.90 (n = 36)	0.79 (n = 67)
R.m.s. for coil (Å)	1.15 n = 15)	0.79 (n = 48)	0.95 (n = 94)	0.92 (n = 77)	1.03 (n = 28)
R.m.s. combined (Å)	1.08 (n = 84)	0.78 (n = 127)	0.83 (n = 298)	0.89 (n = 212)	0.91 (n = 111)

included two crossovers between discontiguous regions. There were three short stretches of five residues or less that were not built by *CAPRA*. Among the chains that were built, *CAPRA* only made two deletions, one short-cut across the two distal residues of a β-hairpin turn and three separate insertions. The overall r.m.s. error was 0.91 Å and 84 of 111 predicted C^{α} atoms were within 1 Å of their correct positions relative to the refined model. Interestingly, one of the C^{α} chains extended into the interior of the molecule, tracing the path of the bound oleic acid, a long-chain fatty acid that looks somewhat like protein backbone.

In order to evaluate whether the accuracy of *CAPRA* is biased toward any particular type of secondary structure, a detailed analysis of the r.m.s. scores above was carried out using secondary-structure annotations by the *DSSP* program (Kabsch & Sander, 1983). It could be hypothesized that *CAPRA* might be more accurate in well defined secondary structure regions (α-helices and β-sheets) in comparison to random coil, especially because the regular secondary structures are most often found buried and rigid, whereas coil (which includes turns) more often occurs at the surface, where density is often weaker. Table 3 shows the r.m.s. scores for the various secondary-structure components in each of the five proteins used in this study. The number of instances of each secondary-structure type are indicated in the table and it should be noted that some proteins contain very little of certain types of secondary structure (for example, P2 myelin is mostly β-sheet and contains very little α-helix, whereas the

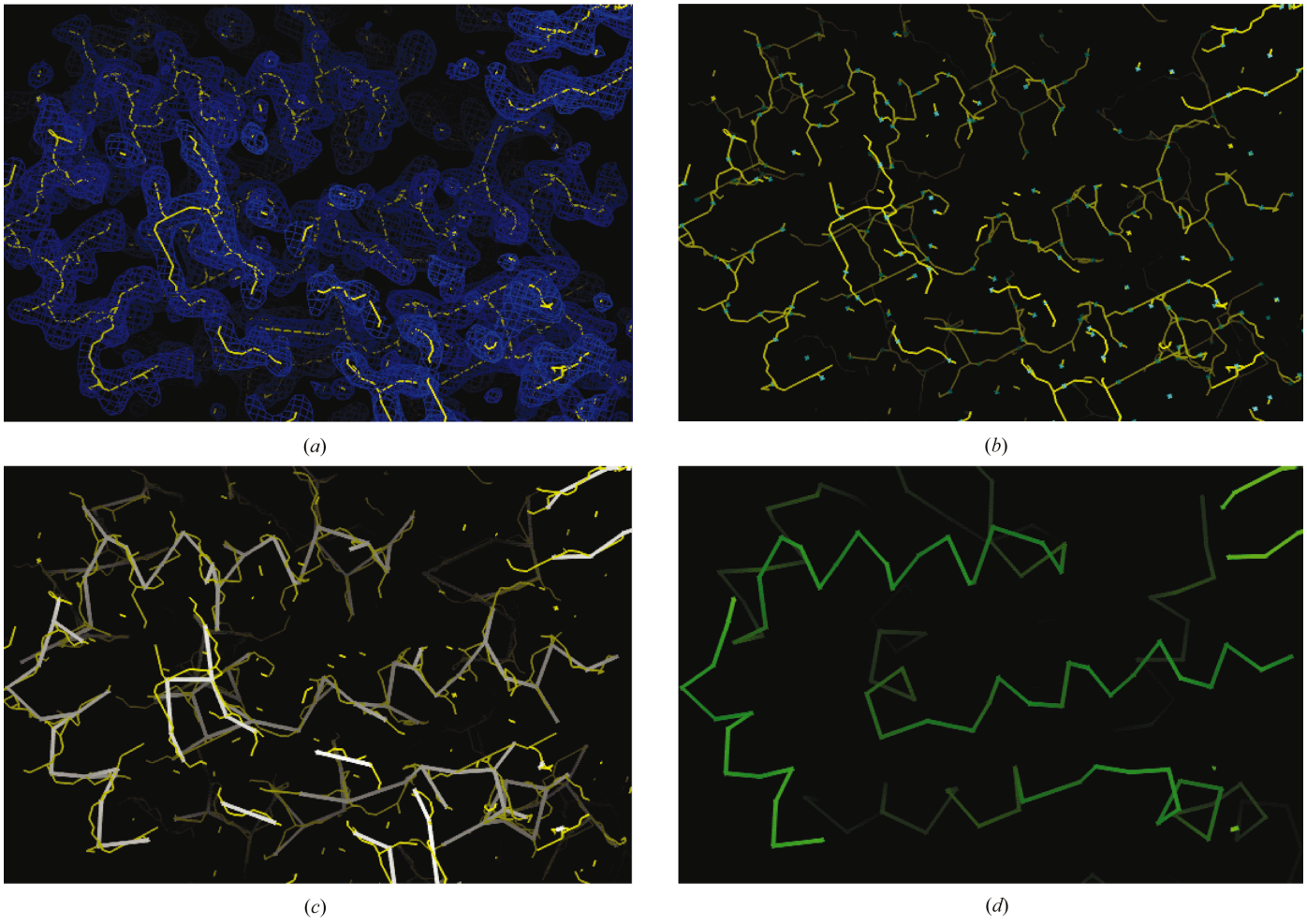

Figure 5
Illustration of the major steps in *CAPRA* in a map for CzrA. (*a*) Trace atoms in yellow, with density contour in blue. (*b*) Light-blue atoms are selected way-points (candidate C^{α} atoms, those with locally best neural net scores). (*c*) White links represent connectivity among way-points based on trace. (*d*) Green links represent linear chains of predicted C^{α} atoms [a subset of those in (*c*) by clipping cycles and branches]. (All images in this paper were created with *SPOCK*, a molecular-graphics program written by Dr Jon A. Christopher; http://quorum.tamu.edu/spock.)

opposite is true for CzrA). While the r.m.s. scores are slightly higher for coil *versus* the predominant secondary structure in some instances, *CAPRA* generally fits all three types of regions with similar accuracy.

In terms of run-time, on average-sized maps (*e.g.* an asymmetric unit with one protein of 100–300 residues), *CAPRA* typically takes less than 1 h to run (our experiments were run on a 400 MHz R12000 processor of an SGI Origin 2000). The majority of the CPU time is spent in calculating features for regions of density around all the pseudo-atoms in the trace.

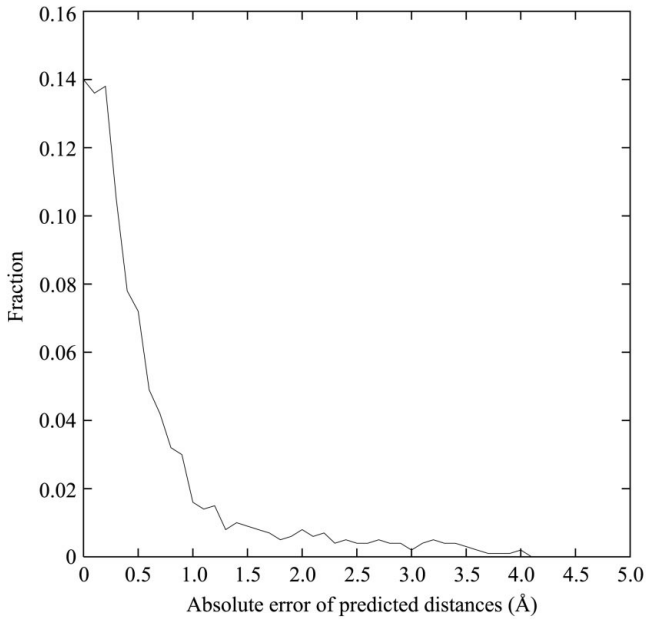

Figure 6
Histogram of absolute value of errors between predicted and actual distances between pseudo-atoms in the trace of MVK and true C^{α} atoms in the refined structure, in increments of 0.1.

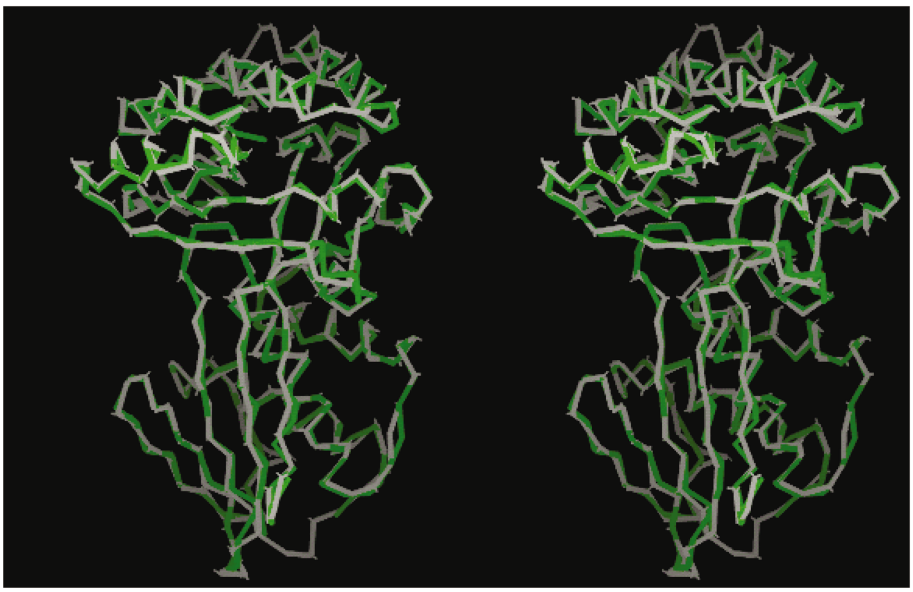

Figure 7
Stereoview of predicted C^{α} chains (in green) for MVK (in white).

5. Discussion

In this paper, we have presented a new method for modeling protein backbones by predicting the coordinates of C^{α} atoms. The method goes beyond basic tracing of a map by using a neural network to recognize which locations in a map look most like typical C^{α} atoms, based on local patterns in the density. Specifically, rotation-invariant numeric features that characterize density patterns are extracted around each point (pseudo-atom in the trace) and the neural network outputs a scalar value that represents an estimate of the distance to the nearest true C^{α} atom. *CAPRA* then uses a number of heuristics to link candidate C^{α} atoms together into linear chains, including analysis of connectivity and plausible secondary structures. In the experimental (MAD and MIR) maps of 2.4–2.8 Å resolution examined in this paper, *CAPRA* was able to fit approximately 90% of the backbone structures with around 0.9 Å r.m.s. error.

There are a number of ways in which an accurate method for recognizing C^{α} coordinates in a map could be used. For example, these coordinates could be input to a fragment-based modeling program, such as Levitt's segment match modeling algorithm (Levitt, 1992) to automatically generate coordinates for other backbone and side-chain algorithms. Prior to now, C^{α} coordinates had to be picked manually, perhaps as a subset of the *BONES* pseudo-atoms; *CAPRA* enables the automation of this step. In a similar way, *CAPRA* is being used as the first stage in *TEXTAL* (Holton *et al.*, 2000), which also uses feature extraction and pattern recognition to build in side-chain and other backbone atoms.

Automation of model building will become increasingly important as structural genomics projects gain momentum (Burley *et al.*, 1999). While advances have been made in nearly every other step of protein crystallography, from robotic crystallization to the use of synchrotrons for rapid data collection and the development of improved computational methods for heavy-atom searches, model building has remained a relatively manual procedure, requiring the skill and time of a human crystallographer. *CAPRA* and *TEXTAL* as a whole, have the potential to reduce the time and effort spent by expert crystallographers in building models for more straightforward cases (or the easier parts of maps), allowing crystallographers to focus their attention on addressing the more problematic cases. *CAPRA* and *TEXTAL* are being incorporated as modules into the *PHENIX* software system, an integrated Python-based crystallographic

computing environment (Adams *et al.*, 2002).

In fact, *CAPRA* could potentially be used on synchrotron beamlines themselves to quickly evaluate the quality of data being collected, based on the number and lengths of chains extracted, in order to make judicious decisions about how best to use remaining beam time. The C^{α} chains output by *CAPRA* could also be used for automated density-based fold recognition (Diller *et al.*, 1999; Oldfield, 2002), *e.g.* to pull similar structures out of the PDB for attempting molecular replacement. Importantly, the pattern-recognition routines in *CAPRA* are tuned for medium-resolution maps (around 2.8 Å), which are predominant among MAD data sets currently being collected at synchrotrons; *CAPRA* does not require higher resolution data to produce accurate results.

One of the limitations of *CAPRA* is that it only generates chains of C^{α} atoms, not complete backbones containing N atoms and carbonyl C atoms and O atoms. In fact, *CAPRA* does not even attempt to determine the directionality of the backbone and the order in which the chains are output is random. If directionality could be robustly determined, then a simple solution would be to build in extra atoms to form a polyalanine backbone and then refine it by real-space refinement. Instead, *CAPRA* relies on the subsequent model-building stages of *TEXTAL* to build-in both backbone and side-chain atoms using pattern recognition. *TEXTAL* often chooses the correct directionality of the backbone by exploiting the fact that certain side-chain conformations constrain the location of the carbonyl O atom to be on one side of the C^{α} or the other.

An interesting extension of *CAPRA* would be to use the C^{α} chains to perform some kind of phase improvement. For example, *CAPRA* could be iterated with phase recombination and map generation to try to incrementally improve the quality of the density and produce better refined models. *CAPRA* does not currently do this, although this strategy is explicitly used in some other model-building methods (Terwilliger, 2002; Turk, 2001). However, we have run a preliminary experiment in which the small clusters of tracer atoms which are often found in spurious patches of density in solvent are eliminated and new phases are calculated from the reduced trace, treating each trace atom as having one-quarter of the scattering power of a carbon, since they are spaced 0.5–1.0 Å apart. When recombined with the experimental phases using *SIGMAA* (Read, 1986), the quality of the density improved slightly. Still, it is anticipated that building more complete models, with all the additional side-chain and backbone atoms, might be more successful.

This work was supported in part by grants R21-GM-59398 and P01-GM-63210 from the National Institutes of Health. Additional support was provided by the Welch Foundation (JCS). The authors would like to thank Drs Marjorie E. Wilke and Ralf W. Grosse-Kunstleve for helpful comments and suggestions on the manuscript. We would also like to thank the members of the PHENIX group, especially Drs Paul Adams, Randy Read and Tom Terwilliger, for previous discussions of the work presented in this paper.

References

Adams, P. D., Grosse-Kunstleve, R. W., Hung, L.-W., Ioerger, T. R., McCoy, A. J., Moriarty, N. W., Read, R. J., Sacchettini, J. C., Sauter, N. K. &Terwilliger, T. C. (2002). *Acta Cryst.* D**58**, 1948–1954.

Brünger, A. T. (1992). *X-PLOR Version* 3.1. *A System for X-ray Crystallography and NMR*. New Haven, CT, USA: Yale University Press.

Burley, S. K., Almo, S. C., Bonanno, J. B., Capel, M., Chance, M. R., Gaasterland, T., Lin, D. W., Sali, A., Studier, F. W. & Swaminathan, S. (1999). *Nature Genet.* **23**, 151–157.

Cowan, S. W., Newcomer, M. E. & Jones, T. A. (1993). *J. Mol. Biol.* **230**, 1225–1246.

Cowtan, K. (1998). *Acta Cryst.* D**54**, 750–756.

Diller, D. J., Redinbo, M. R., Pohl, E. & Hol, W. G. J. (1999). *Proteins*, **36**, 526–541.

Greer, J. (1985). *Methods Enzymol.* **115**, 206–224.

Hinton, G. E. (1989). *Artif. Intell.* **40**, 185–234.

Holm, L. & Sander, C. (1991). *J. Mol. Biol.* **218**, 183–194.

Holton, T. R., Christopher, J. A., Ioerger, T. R. & Sacchettini, J. C. (2000). *Acta Cryst.* D**56**, 722–734.

Huang, C.-C., Smith, C. V., Glickman, M. S., Jacobs, W. R. & Sacchettini, J. C. (2002). *J. Biol. Chem.* **277**, 11559–11569.

Jones, T. A., Zou, J.-Y. & Cowtan, S. W. (1991). *Acta Cryst.* A**47**, 110–119.

Kabsch, W. & Sander, C. (1983). *Biopolymers*, **22**, 2577–2637.

Kleywegt, G. J. & Jones, T. A. (1997). *Acta Cryst.* D**53**, 179–185.

Kleywegt, G. J. & Read, R. J. (1997). *Structure*, **5**, 1557–1569.

Leherte, L., Fortier, S., Glasgow, J. & Allen, F. H. (1994). *Acta Cryst.* D**50**, 155–166.

Levitt, D. G. (2001). *Acta Cryst.* D**57**, 1013–1019.

Levitt, M. (1992). *J. Mol. Biol.*, **226**, 507–533.

Oldfield, T. J. (1996). In *Crystallographic Computing 7: Proceedings of the Macromolecular Crystallography Computing School*, edited by P. Bourne & E. Watenpaugh. Oxford University Press.

Oldfield, T. (2002). *Acta Cryst.* D**58**, 487–493.

Oldfield, T. & Hubbard, R. E. (1994). *Proteins*, **18**, 324–337.

Peat, T. S., Newman, J., Waldo, G. S., Berendzen, J. & Terwilliger, T. C. (1998). *Structure*, **6**, 1207–1214.

Perrakis, A., Sixma, T. K., Wilson, K. S. & Lamzin, V. S. (1997). *Acta Cryst.* D**53**, 448–455.

Read, R. J. (1986). *Acta Cryst.* A**42**, 140–149.

Swanson, S. M. (1994). *Acta Cryst.* D**50**, 695–708.

Terwilliger, T. (1999). *Acta Cryst.* D**55**, 1863–1871.

Terwilliger, T. (2002). *Interdisciplinary Workshop Promoting Collaboration in High-Throughput X-ray Structure Determination*. http://www.solve.lanl.gov/santa_fe_workshop_summary.html.

Turk, D. (2001). *Methods in Macromolecular Crystallography*, edited by D. Turk & L. Johnson, pp. 148–155. Amsterdam: IOS Press.

Whelan, K. & Glasgow, J. (2000). *Pacific Symp. Biocomp.* **5**, 401–412.

Yang, D., Shipman, L. W., Roessner, C. A., Scott, I. A. & Sacchettini, J. C. (2002). *J. Biol. Chem.* **277**, 9462–9467.