The graduation of secondary structure elements

D.J. Thomas

European Molecular Biology Laboratory, Heidelberg, Germany

A method of graduating (i.e., least-squares fitting) a smooth polynomial curve through long elements of protein secondary structure is described. It uses the Chebyshev polynomials of a discrete (integer) variable with several restraints to prevent artifactual curvatures. A new recursion formula is given which allows the evaluation of the polynomials on rational-number points as well as on the integer points. High-order splines suitable for interpolation between integer points are also discussed. The new method finds applications in graphics and in structural analysis.

Keywords: polynomial, protein, secondary structure element

INTRODUCTION

There are occasions when it is considered desirable to fit a smooth curve through secondary structural elements of proteins. One obvious example is in a graphics program where small apparently random divergences from standard geometry may serve only to confuse the viewer and contribute little to the appreciation of the structure. A second example arises during simplified mechanical modeling, where effective points of contact must be found.[1] Analysis of super-helical twists and manipulation of trial structures are further examples.

It is not possible to define uniquely the best way to fit a secondary structure element because there is always a choice of what types of approximation are tolerable and which errors must be avoided. This paper describes how to fit smooth polynomial curves as free from artifacts as possible, with equal weight attaching to each Cα atom. This specification demands the use of the little known Chebyshev polynomials on the discrete domain;[2,11] the equivalent polynomials on the continuous domain are not one of those named after Chebyshev, but rather those named after Legendre. The discrete Chebyshev polynomials allow smooth interpolation between the integer points representing the numbered Cα atoms of the protein backbone, and they tend to spread the fitting errors uniformly over those points. A full-order expansion could fit the atomic positions exactly, but if the order of the fit is less than the number of atoms being fitted, then it is most likely that the fit will not be exact. It is, indeed, the inability of low-order polynomials to curve in a complicated way which makes them an attractive method for smoothing irregularities in atomic positions. Low order polynomial fits are, however, invariably subject to an artifact of overemphasizing curvature, particularly at the ends of the domain being fitted (see the stereo figure in Thomas[3]). This can be avoided by using a higher order polynomial expansion with an extra restraint trying to minimize the curvature of the fitted expansion.

Address reprint requests to Dr. Thomas at EMBL, Meyerhofstrasse 1, Postfach 10.2209, W-69012 Heidelberg, Germany.
Received 16 June 1993; revised 17 October 1993; accepted 22 October 1993

INTRODUCTION TO CHEBYSHEV POLYNOMIALS

When working with a discrete variable, it is usual to replace the differential operator of the calculus by a difference operator. The (first) forward difference operator is defined by

$$\Delta_g^1[F(g)] = F(g+1) - F(g) \tag{1}$$

where F is a function of g. The second forward difference is then

$$\begin{aligned}\Delta_g^2[F(g)] &= \Delta_g^1[F(g+1)] - \Delta_g^1[F(g)] \\ &= F(g+2) - 2F(g+1) + F(g)\end{aligned} \tag{2}$$

and so on, so that Δ_g^n is Δ_g applied n times, in which case we find that[4]

$$\Delta_g^n[F(g)] = \sum_{j=0}^{n} (-1)^j \binom{n}{j} F(g+n-j) \tag{3}$$

The Chebyshev polynomials orthogonal with uniform weighting on the discrete domain $[0, N-1]$ are defined by

$$T_n(g) = n!\,\Delta_g^n\left[\binom{g}{n}\binom{g-N}{n}\right]; \qquad \begin{cases} g \in \mathbb{Z}, & 0 \le g \le N-1 \\ n \in \mathbb{Z}, & 0 \le n \le N-1 \end{cases} \tag{4}$$

which is a discrete equivalent of Rodrigues' formula to generate polynomials on the continuous domain. This formula does not lend itself to particularly efficient computation, but shows clearly that $T_n(g)$ and all of its derivatives are integral when g is an integer. For practical computations recurrence relations are normally preferred. The relation is

given by Erdélyi et al.[2] as

$$(n + 1)T_{n+1}(g) - (2n + 1)(2g - N + 1)T_n(g) + n(N^2 - n^2)T_{n-1}(g) = 0 \quad (5)$$

which is rearranged for practical use as

$$T_n(g) = \frac{1}{n}[(2n - 1)(2g - N + 1)T_{n-1}(g) + (1 - n)(N - n + 1)(N + n - 1)t_{n-2}(g)] \quad (6)$$

This is primed by $T_{-1} = 0$ and $T_0 = 1$, and returns an integer so long as g is an integer. The polynomials for $N = 11$ are illustrated in Figure 1.

Relaxing the condition $g \in \mathbb{Z}$ to $g \in \mathbb{R}$, the derivatives of the Chebyshev polynomials with respect to g are also available by recursion, the formulas being

$$T'_n(g) = \frac{1}{n}[(2n - 1)(2T_{n-1}(g) + (2g - N + 1)T'_{n-1}(g)) + (1 - n)(N - n + 1)(N + n - 1)T'_{n-2}(g)] \quad (7)$$

for the first derivative, and

$$T''_n(g) = \frac{1}{n}[(2n - 1)[4T'_{n-1}(g) + (2g - N + 1)T''_{n-1}(g)] + (1 - n)(N - n + 1)(N + n - 1)T''_{n-2}(g)] \quad (8)$$

for the second. Higher derivatives follow the same pattern. These recursions are numerically unstable near to the two ends of the domain for large n and N because of the very steep gradients (see Figure 1). However, since the arithmetic involves only integers when g is itself integral, the polynomials can be evaluated exactly with extended integer arithmetic, when the numerical instability is not manifest. The evaluated polynomials are then best tabulated (i.e., stored on

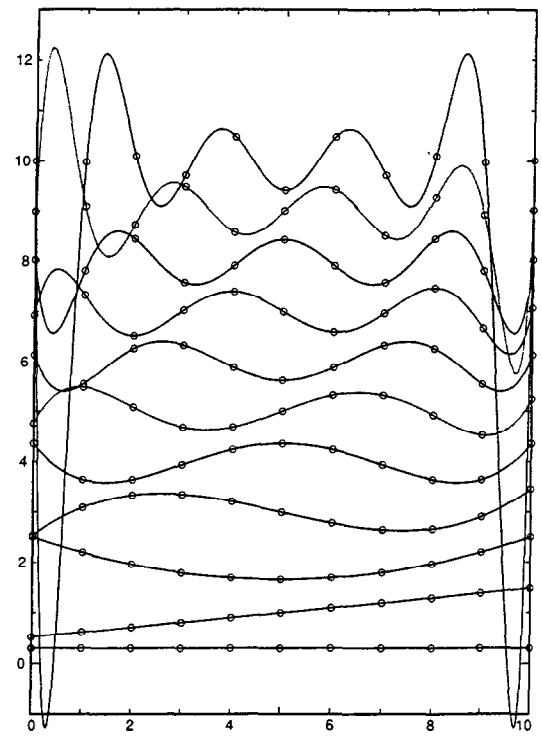

Figure 1. Normalized Chebyshev polynomials $\widehat{T_n}(g)$ *for* $N = 11$. *The abscissa labels mark* g, *while the ordinate labels denote both the order of the polynomials being plotted and the scale on which they are plotted. These polynomials are mutually orthogonal when sampled with equal weights on the integral points marked with circles. This figure shows how the gradients near the two ends of the domain rise rapidly for higher orders of polynomial. These very high gradients cause the recursion relation to be numerically unstable.*

NOTATION

c	Coefficients describing the cosinusoidal variation of the radial vector
e	Coefficients describing the mean central line
f	Fitted position of a Cα atom
g	Number of a residue (i.e., Cα atom)
i	x,y,z-axes
j	Dummy index
l	Mean central line of a secondary structure element
n	Order of a polynomial
p,q	Numerator and denominator of a rational number
r	Actual position of a Cα atom
s	Coefficients describing the sinusoidal variation of the radial vector
t	Fractional part of a residue number
u	Radial vector from the mean central line to the helicoidal line
ϵ	Error in fitting a Cα atom
ϕ	Change in azimuthal angle per residue
C	Normalized Chebyshev polynomial of a discrete variable times a cosine term
F	Generic function
L	Order of the highest polynomial used to fit the mean central line
N	Number of discrete points (e.g., Cα atoms) in the domain being graduated
P,Q	Numerator and denominator of Chebyshev polynomial of a rational argument
S	Normalized Chebyshev polynomial of a discrete variable times a sine term
T	Chebyshev polynomial of a discrete variable; $\hat{T}$ when normalized
U	Order of the highest polynomial used to fit the radial vector
Δ	Forward difference operator
$\mathbb{Q}$	Set of rational numbers
$\mathbb{R}$	Set of real numbers
$\mathbb{Z}$	Set of whole numbers (integers)
$\hat{\delta}$	Kronecker delta (1 if indices match, 0 otherwise)
$'$	Differentiation with respect to g; also marks a dummy index
$\binom{g}{n}$	Combinatoric ${}_gC_n = g!/[n!(g - n)!]$

disk) for future reference. A computationally stable recursion is not available for a general real number argument ($g \in \mathbb{R}$), but a workable modification of the recursion for rational numbers ($g \in \mathbb{Q}$) does exist. Writing

$$g = p/q; \quad p,q \in \mathbb{Z} \tag{9}$$

and

$$T_n(p/q) = \frac{P_n(p/q)}{Q_n(p/q)}; \quad P,Q \in \mathbb{Z} \tag{10}$$

is consistent with the integer denominator:

$$Q_n = q^n n! \tag{11}$$

and numerator:

$$P_n = (2n-1)[2p - q(N-1)]P_{n-1} - q^2(n-1)^2(N+n-1)(N-n+1)P_{n-2} \tag{12}$$

This is the most efficient form of the recursion on rationals known to the author. It should be noted that the recursion on integers (6) is less complicated than the recursion on rationals when the denominator $q = 1$ because the $n!$ divides out. Derivatives can also be calculated by recursion at rational points by using the relation $\partial p/\partial g = q$. The numerators of the first two are

$$P'_n = (2n-1)[2qP_{n-1} + [2p - q(N-1)]P'_{n-1}] - q^2(n-1)^2(N+n-1)(N-n+1)P'_{n-2} \tag{13}$$

and

$$P''_n = (2n-1)[4qP'_{n-1} + [2p - q(N-1)]P''_{n-1}] - q^2(n-1)^2(N+n-1)(N-n+1)P''_{n-2} \tag{14}$$

The derivatives take the same denominator, Q_n. Again, higher derivatives follow the same pattern. For most applications the lack of a stable recursion for general real number arguments is of no consequence because interpolating splines can be used between integral or rational pinning points. This topic is discussed later.

As defined, the polynomials are orthogonal but not orthonormal. The normalization factor is given implicitly[2] by the inverse of the square root of

$$\sum_{g=0}^{N-1} T_n^2(g) = \frac{1}{2n+1}\frac{(N+n)!}{(N-n-1)!} \tag{15}$$

which also requires extended range arithmetic when working to high orders. Thus the normalized polynomials are given explicitly by

$$\widehat{T_n}(g) = T_n(g)\sqrt{(2n+1)\frac{(N-n-1)!}{(N+n)!}} \tag{16}$$

so that

$$\sum_{g=0}^{N-1} \widehat{T_n}(g)\widehat{T_{n'}}(g) = \hat{\delta}_n^{n'} \tag{17}$$

This is known as the orthonormality condition.

DESCRIPTION OF SECONDARY STRUCTURE ELEMENTS BY CHEBYSHEV POLYNOMIALS

The mean line down the center of a helix can not be fit to the raw Cα positions directly with any hope of obtaining an accurate result; instead, a flexible helical path must be generated, orbiting the mean line with a variable radius and rate of change of azimuthal angle (see Figure 2).

The radius vector from a point on the mean central line to the corresponding point on the helicoid is also modeled with Chebyshev polynomials and could therefore be in any direction and have any magnitude. This necessary freedom can also cause artifactual results, so extra constraints must be applied. These try to set the radial vector to have a constant magnitude, to be perpendicular to the mean central line, and to vary smoothly along the length of the secondary structure element. These constraints act together to make the helical line behave rather like a right perpendicular coil spring with adjustable radius and rate of change of azimuthal angle.

As mentioned in the introduction, a smoothly curving central line is defined for a secondary structural element by an expansion in Chebyshev polynomials:

$$l_i(g) = \sum_{n=0}^{L} e_i^n \widehat{T_n}(g); \quad i = x,y,z \tag{18}$$

The terms e_i^n are the coefficients to be found (n being a label, not a power), and the highest order used, L, cannot exceed $N-1$, where N is the number of Cα atoms. In practice, $L \approx N/2$ is required for β-strands whilst $L \approx N/3$ is enough for helices.

The tangent to the central line is given by differentiation with respect to g, and is

$$l'_i(g) = \sum_{n=1}^{L} e_i^n \widehat{T'_n}(g) \tag{19}$$

This is useful for plotting using cubic splines between the integer (i.e., Cα) points, and for finding points of closest contact between helices or strands.[1] Higher splines require higher derivatives which follow the same pattern.

The helicoidal undulation from the central line is given by a radial vector:

$$u_i(g) = \sum_{n=0}^{U} s_i^n \widehat{T_n}(g)\sin\phi g + \sum_{n=0}^{U} c_i^n \widehat{T_n}(g)\cos\phi g = \sum_{n=0}^{U} s_i^n S_n(g) + \sum_{n=0}^{U} c_i^n C_n(g) \tag{20}$$

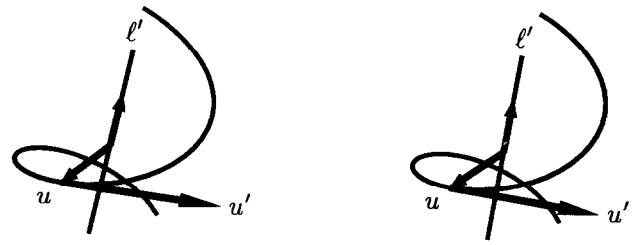

Figure 2. Fitting of a helix. The diagram shows the tangent to the mean central line l′, *the radial vector* u, *and its tangent* u′. *The length of* l′ *is the rise of the helix per residue while the length of* u′ *is the length of the radial vector* u *times the turn per residue measured in radians.*

where s_i^n and c_i^n are also to be found (again n is a label, not a power); U cannot exceed $N - 1$ and is generally smaller than L. The zeroth terms, s_i^0 and c_i^0, already express mean helicity because of the incorporation of trigonometric functions of the azimuthal angle ϕ and residue number g. The first-order terms, s_i^1 and c_i^1, express the error in ϕ, and are used to correct that error in an iterative procedure. (Iteration is necessary because of the nonlinearity in the equations.) Higher terms are used to accommodate distortions in the helicity, especially those consequent on curvature of the secondary structure element.

The change in azimuthal angle per residue must be primed to a value appropriate to the secondary structure being fitted, e.g., 102° for α-helices and 113° for 3_{10}-helices.[5] In the author's experience, the refined values tend to be a little smaller than these values when the mean line is correctly fitted as a curve. A slight difference is not surprising because the perceived mean twist angle depends critically on the positioning of the axis, and it would be expected that the values obtained with a well-fitting curved axis would be the more accurate estimators.

Suppose that the "real" positions of the Cα atoms are called $r_i(g)$, and the calculated positions are $f_i(g) = l_i(g) + u_i(g)$; an unconstrained fit would then be obtained by setting to zero the errors $\epsilon_i(g) = f_i(g) - r_i(g)$ by varying e_i^n, s_i^n and c_i^n. The natural way to do this is in a Newton–Raphson framework, using the derivatives

$$\frac{\partial \epsilon_i(g)}{\partial e_i^n} = \widehat{T_n}(g) \tag{21}$$

$$\frac{\partial \epsilon_i(g)}{\partial s_i^n} = S_n(g) \tag{22}$$

$$\frac{\partial \epsilon_i(g)}{\partial c_i^n} = C_n(g) \tag{23}$$

Although the fit to the data is good using this method, there is too much freedom in the determination of the radial vector; this manifests itself in an exaggeration of the curvature of the mean central line, particularly at the ends of the domain.[1,3] Several constraints need to be applied to avoid this artifact, as discussed in the introduction to this section.

The first derivative of the radial vector,

$$u_i'(g) = \sum_{n=0}^{U} s_i^n S_n'(g) + \sum_{n=0}^{U} c_i^n C_n'(g) \tag{24}$$

is used to restrain variations in the radius of the helicoid by driving to zero the dot product:

$$Q(g) = u_i(g) u_i'(g) \tag{25}$$

which is summed over $i = x,y,z$. This requires the derivatives

$$\frac{\partial Q(g)}{\partial s_i^n} = S_n'(g)\, u_i(g) + S_n(g)\, u_i'(g) \tag{26}$$

$$\frac{\partial Q(g)}{\partial c_i^n} = C_n'(g)\, u_i(g) + C_n(g)\, u_i'(g) \tag{27}$$

in the Newton–Raphson algorithm. The derivatives of $S_n(g) = \widehat{T_n}(g) \sin \phi g$ and of $C_n(g) = \widehat{T_n}(g) \cos \phi g$ are

$$S_n'(g) = \widehat{T_n'}(g) \sin \phi g + \widehat{T_n}(g) \phi \cos \phi g \tag{28}$$

and

$$C_n'(g) = \widehat{T_n'}(g) \cos \phi g - \widehat{T_n}(g) \phi \sin \phi g \tag{29}$$

Deviations of the helicoid from perpendicularity are minimized by setting to zero another dot product, this time between the local helix axis l_i' and the radial vector u_i:

$$R(g) = l_i'(g) u_i(g) \tag{30}$$

This minimization requires the following three derivatives:

$$\frac{\partial R(g)}{\partial e_i^n} = \widehat{T_n'}(g)\, u_i(g); \quad n > 0 \tag{31}$$

$$\frac{\partial R(g)}{\partial s_i^n} = S_n(g) l_i'(g) \tag{32}$$

$$\frac{\partial R(g)}{\partial c_i^n} = C_n(g) l_i'(g) \tag{33}$$

Curvature is constrained by trying to set to zero the second derivatives of the mean central line and the second derivatives of the effective helical radii in two perpendicular planes. The effective radii are obtained by replacing both $\sin \phi g$ and $\cos \phi g$ with unity. Therefore we have

$$l_i''(g) = \sum_{n=2}^{L} e_i^n \widehat{T_n''}(g) \tag{34}$$

$$\sum_{n=2}^{U} s_i^n \widehat{T_n''}(g) \tag{35}$$

$$\sum_{n=2}^{U} c_i^n \widehat{T_n''}(g) \tag{36}$$

which are driven to zero by varying the parameters e_i^n, s_i^n, and c_i^n. In all three cases, the appropriate derivative for the Newton–Raphson algorithm is just $\widehat{T_n''}(g)$.

An example of fitted helices is shown in Figure 3.

The fitting of β-strands is very like that for helices; the major difference being that the change in azimuthal angle per residue is set to exactly 180° per residue and the sine terms are not used. This makes the previously helicoidal fitting line flat and cosinusoidal, but it is still able to fit distorted β-strands because (as with helices) the radial vector is variable along the length of the strand. In addition, a higher order polynomial is used to accommodate the greater flexibility of β-strands. An example of a fitted strand is shown in Figure 4.

It might be thought that L and U can even be as high as $N - 1$, but there are two arguments against this. First, the higher order polynomials undulate rapidly, and act competitively against the cosinusoid; this can adversely affect the convergence to the desired result. To prevent this, L and U should not be much more than about $N/2$. Second, the computational time increases as the third power of the highest order used because the solution of a matrix equation is required for the Newton–Raphson iteration.

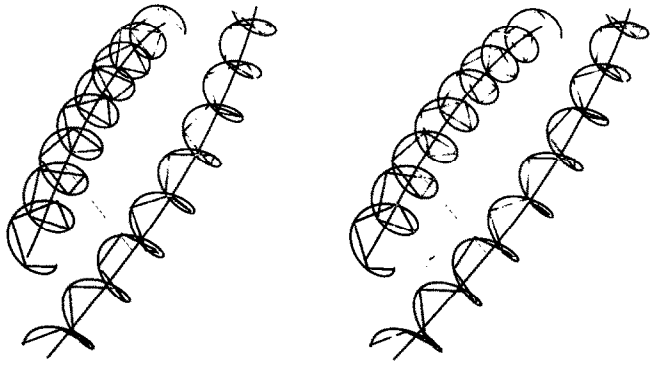

Figure 3. Example of fitted helices. These "parallel" helices form the dimerization domain of the yeast transcription factor GCN4.[12] It can be seen that the non-crystallographic dyad running up the superhelix axis is satisfied only approximately, which is thought to be a result of crystal packing effects. The inappropriateness of fitting helices with straight lines is obvious in this example. The helicoids in this picture are interpolated with quintic splines.

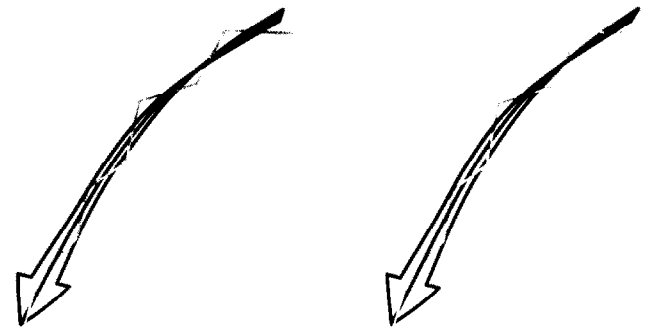

Figure 4. Example of a fitted strand. The strand is strand 4 of flavodoxin (4FXN), spanning residues 80 to 88. The sides of the arrow are interpolated with cubic splines in this picture; L = N − *1 and* U = N − *2 though usually* L *and* U *would be set to about* N/2.

It is important to realize that the extra conditions of constancy of radius, perpendicularity, and minimal curvature are not applied as constraints to be satisfied exactly within a framework of Lagrange undetermined multipliers, but as restraints in the least-squares sense of extra terms to be driven towards zero in a Newton–Raphson framework. It is probably possible to accommodate the restraints on radius and perpendicularity as constraints within a Lagrange formalism, but this has not been tried, and the author can see little advantage in doing so since the least-squares formalism works very well.

It will rarely be possible to satisfy exactly all of the restraints simultaneously, and this raises immediately the question of what relative weightings should be used. The author's experience is that for helices and strands the best results obtain when all terms have the same weighting, which must be a reflection of having a unified length scale based in the integer divisions of the line. Deviations in the relative weightings of a few tens of percent are tolerable, but deviations by a factor of 2 or more produce obvious degradations in the appearance or fit of the line. With a well-refined crystallographic structure it is found that helices and β-strands are fit within a typical root-mean-square error of the order of 0.1 Å, and that the fitted lines appear to be completely free from visible artifacts.

The fitting of loops is even simpler than strands; neither the sine nor the cosine terms are used (so that $U < 0$), and L can be set to $N - 1$. The curvature restraint is still used to advantage, but experience shows that it should be downweighted by a factor of about 10. Examples of fitted loops can be seen in Figure 6.

Exhaustive testing has shown that more than 99% of the secondary structural elements in the Protein Data Bank[6] are fitted correctly even if the Newton–Raphson iteration is primed with all e_i^n, s_i^n, $c_i^n = 0$, indicating a rather wide basin of convergence. The only cases of incorrect convergence were the very long doubly bent β-strands in some of the viral coat proteins (PDB files 4RHV, 2MEV, 2PLV) when L and U were also set too large ($N - 1$ and $N - 2$, respectively). This problem was cured by initializing the e_i^n according to

$$e_i^n = \sum_{g=0}^{N-1} r_i(g)\widehat{T_n}(g) \tag{37}$$

which exploits the properties of orthonormal polynomials directly to produce the coefficients appropriate to a fit without further restraints. It is easy to see, starting from Equation (18) and using the orthonormality condition (17), that if $L = N - 1$,

$$\begin{aligned} l_i(g) &= \sum_{n=0}^{N-1} e_i^n(g)\widehat{T_n}(g) \\ &= \sum_{n=0}^{N-1}\sum_{g'=0}^{N-1} r_i(g')\widehat{T_n}(g')\widehat{T_n}(g) \\ &= \sum_{g'=0}^{N-1} r_i(g')\hat{\delta}_g^{g'} \\ &= r_i(g) \end{aligned} \tag{38}$$

confirming Equation (37). Similar attempts to initialize s_i^n and c_i^n failed, perturbing the convergence badly; this was undoubtedly because the sets of functions S_n and C_n are neither orthogonal nor normalized.

SPLINE INTERPOLATION

It was pointed out above that the calculation of high order Chebyshev polynomials is inefficient and can pose problems when the residue number g is not a whole number. For this reason it is worthwhile (especially for graphical applications) to interpolate between the integer points of the optimally fitted polynomial curve using spline functions. Probably the most familiar spline is the cubic one, which will interpolate between endpoints with specified slopes (i.e., first derivatives). However, if a cubic spline is used to interpolate between Cα positions the result is not a faithful representation of the helicoidal fitting line, being rather angular and squared-off (see Figure 5). A quintic (fifth order) spline, on the other hand, having controllable curvature at the endpoints, appears to mimic the helicoid almost perfectly. The Bézier/Casteljau formulation of splines

makes higher order ones appear unreasonably complicated,[7] but the derivation given below for a heptic spline reveals a rather simple rule applicable to any odd order spline pinned only by its two endpoints and derivatives at those points.

Suppose the heptic spline be called $s(t)$, where t is a parameter in the range [0, 1] representing the fractional part of $g \in \mathbb{R}$; the two endpoints will therefore be $s(0)$ and $s(1)$. These endpoints will be required to match the boundary conditions $f(0)$, $f'(0)$, $f''(0)$, and $f'''(0)$ at the starting point, and $f(1)$, $f'(1)$, $f''(1)$, and $f'''(1)$ at the finishing point, f being the fit to the real position r, and the primes representing derivatives, as usual. This allows the spline to be written down immediately as a power series expansion about $t = 0$ with four known factors and four unknown:

$$s(t) = \frac{f(0)}{0!} + \frac{f'(0)}{1!}t + \frac{f''(0)}{2!}t^2 + \frac{f'''(0)}{3!}t^3 + at^4 + bt^5 + ct^6 + dt^7 \quad (39)$$

The four unknowns a, b, c, and d are fixed by asserting the boundary conditions for the point $t = 1$, which comprise the point itself and as many derivatives as are required to generate the necessary number of new equations:

$$s(1) = f(1) = \frac{f(0)}{0!} + \frac{f'(0)}{1!} + \frac{f''(0)}{2!} + \frac{f'''(0)}{3!} + a + b + c + d \quad (40)$$

$$s'(1) = f'(1) = \frac{f'(0)}{0!} + \frac{f''(0)}{1!} + \frac{f'''(0)}{2!} + 4a + 5b + 6c + 7d \quad (41)$$

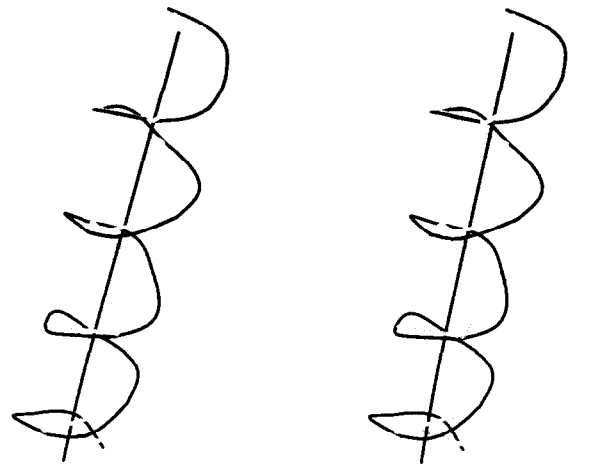

Figure 5. Inadequacy of cubic splines to interpolate between Cα positions. The helix shown is helix 1 of flavodoxin (3FXN), residues 10 to 25.

$$s''(1) = f''(1) = \frac{f''(0)}{0!} + \frac{f'''(0)}{1!} + 12a + 20b + 30c + 42d \quad (42)$$

$$s'''(1) = f'''(1) = \frac{f'''(0)}{0!} + 24a + 60b + 120c + 210d \quad (43)$$

Rewriting these four equations in matrix notation gives:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 4 & 5 & 6 & 7 \\ 12 & 20 & 30 & 42 \\ 24 & 60 & 120 & 210 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} -1 & -1 & -\frac{1}{2} & -\frac{1}{6} & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & -\frac{1}{2} & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(0) \\ f'(0) \\ f''(0) \\ f'''(0) \\ f(1) \\ f'(1) \\ f''(1) \\ f'''(1) \end{bmatrix} \quad (44)$$

whose solution is:

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 4 & 5 & 6 & 7 \\ 12 & 20 & 30 & 42 \\ 24 & 60 & 120 & 210 \end{bmatrix}^{-1} \begin{bmatrix} -1 & -1 & -\frac{1}{2} & -\frac{1}{6} & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & -\frac{1}{2} & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(0) \\ f'(0) \\ f''(0) \\ f'''(0) \\ f(1) \\ f'(1) \\ f''(1) \\ f'''(1) \end{bmatrix}$$

$$= \begin{bmatrix} -35 & -20 & -5 & -\frac{2}{3} & 35 & -15 & \frac{5}{2} & -\frac{1}{6} \\ 84 & 45 & 10 & 1 & -84 & 39 & -7 & \frac{1}{2} \\ -70 & -36 & -\frac{15}{2} & -\frac{2}{3} & 70 & -34 & \frac{13}{2} & -\frac{1}{2} \\ 20 & 10 & 2 & \frac{1}{6} & -20 & 10 & -2 & \frac{1}{6} \end{bmatrix} \begin{bmatrix} f(0) \\ f'(0) \\ f''(0) \\ f'''(0) \\ f(1) \\ f'(1) \\ f''(1) \\ f'''(1) \end{bmatrix} \quad (45)$$

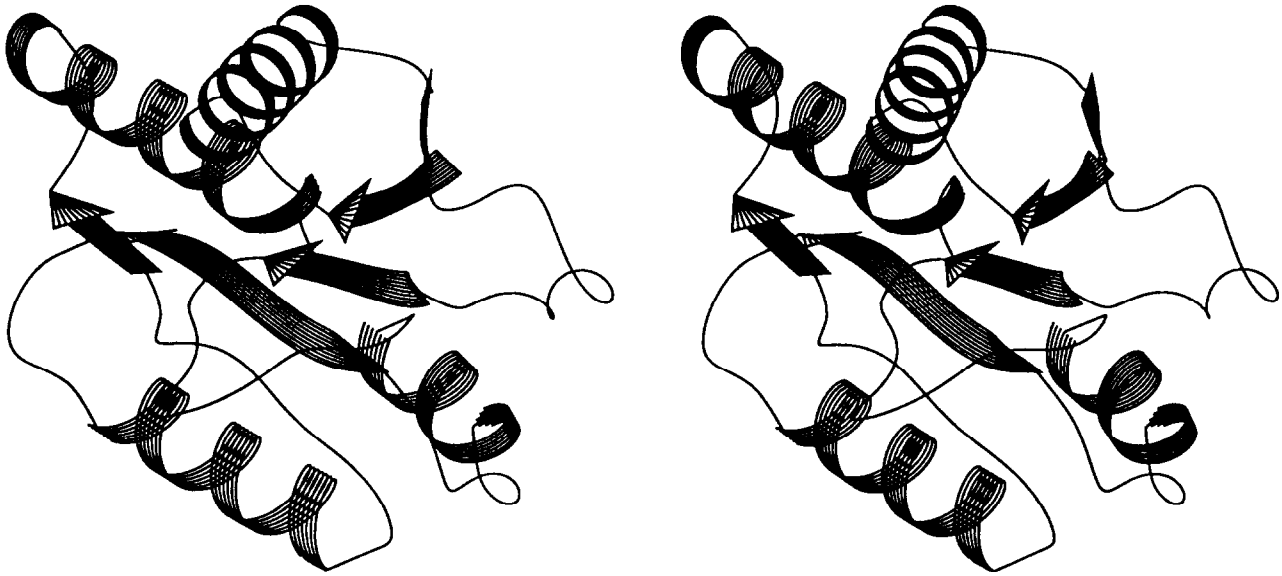

Figure 6. View of flavodoxin (from PDB file 4FXN) printed using the program WHAT IF of Vriend.[10] The helices, strands and loops are all fitted using the method described in this paper.

This method of evaluating splines pinned between two fixed endpoints works for any odd order. All that is necessary is to extend Equation (44) by following the obvious combinatoric patterns in the two matrices. Indeed, the matrix inversion required for nonic and even higher splines can be effected in a typical modern scientific pocket calculator. A useful check on the matrix inversion is that the sum of the columns of the inverse reproduces the top row of the matrix in the right-hand side of Equation (44).

CONCLUDING REMARKS

This paper describes a method of fitting long secondary structure elements with a polynomial curve. A major application is in the geometric analysis of protein structures, where a single analytic curve has enormous computational advantages for some elementary calculations, such as finding the points of closest approach of helices.[1] Another obvious example is in the characterization of coiled-coil structures.[8]

Another major application is graphics, where the method described presents the structural motifs largely without the random undulations which are so characteristic of direct splining methods.[9] Splines can be used, nonetheless, in this application, but they are used to interpolate the optimally fitted polynomial rather than the actual atomic positions. The paper includes for this purpose a simple derivation of splines of odd order pinned by their ends, which is much simpler than the Bézier/Casteljau formulation. The new methods have been incorporated into the program WHAT IF of Vriend.[10]

The recursion relation for Chebyshev polynomials on the rational points is thought to be novel.

ACKNOWLEDGMENTS

The author is grateful to Dr. David Wild and the two anonymous referees for helpful comments on the manuscript, and to Dr. Gert Vriend for extensive testing of the method as incorporated into his program WHAT IF.

REFERENCES

1 Thomas, D.J. A simplified mechanical model of proteins tested on the globin fold. *J. Mol. Biol.* 1991, **222,** 805–817

2 Erdélyi, A., Magnus, W., Oberhettinger, F., and Tricomi, F.G. Tchebichef's polynomials of a discrete variable and their generalizations. In *Higher Transcendental Functions II.* McGraw-Hill, New York, 1953, pp. 223–224

3 Thomas, D.J. Towards more reliable printed stereo. *J. Mol. Graphics.* 1993, **11,** 15–22, 42, 144

4 Davis, P.J. and Polonsky, I. Forward Differences. In *Handbook of Mathematical Functions.* (M. Abramowitz and I.A. Stegun, Eds.) Dover, New York, 1965

5 Barlow, D.J. and Thornton, J.M. Helix Geometry in Proteins. *J. Mol. Biol.* 1988, **201,** 601–619

6 Bernstein, F.C., Koetzle, T.F., Williams, G.J.B., Meyer, E.F., Jr, Brice, M.D., Rodgers, J.R., Kennard, O., Shimanouchi, T., and Tasumi, M. The Protein Data Bank: A Computer-based Archival File for Macromolecular Structures. *J. Mol. Biol.* 1977, **112,** 535–542

7 Burger, P. and Gillies, D. *Interactive Computer Graphics.* Addison-Wesley, Wokingham, 1989, Chapter 6

8 Seo, J. and Cohen, C. Pitch Diversity in α-Helical Coiled Coils. *PROTEINS: Struct. Funct. Genet.* 1993, **15,** 223–234

9 Carson, M. and Bugg, C.E. Algorithm for ribbon models of proteins. *J. Mol. Graphics.* 1986, **4,** 121–122, 107

10 Vriend, G. WHAT IF: A molecular modeling and drug design program. *J. Mol. Graphics.* 1990, **8,** 52–56

11 Hochstrasser, U.W. (1965). Orthogonal Polynomials of a Discrete Variable. In *Handbook of Mathematical Functions.* M. Abramowitz and I.A. Stegun, New York: Dover

12 O'Shea, E.K., Klemm, J.D., Kim, P.S. and Alber, T. X-ray Structure of the GCN4 Leucine Zipper, a Two-Stranded, Parallel Coiled Coil. *Science.* 1991, **254,** 539–544