The method based on symmetric matrix factorization (SMF)

SMF factorizes the *N*-matrix, as:

$$N = H * D * H', \quad (1)$$

where H^T is the transpose of *H*. *H* is an *m*-by-*nd* binary matrix, where *nd* is the number of putative domains. This is the domain definition matrix: *H(i,j)* is 1 if domain *j* includes residue *i* or 0 otherwise. We assume that a residue cannot belong to more than one domain, in which case *H* is column-wise orthogonal, i.e. its column vectors are orthogonal to each other:

$$\sum H(i,a) * H(i,b) = nr(a) * \delta(a,b), \quad (2)$$

where *nr(a)* is the number of residues in the putative domain *a* and $\delta(a,b) = 1$ if *a*=*b* and 0 otherwise. The *H* matrix is obtained by clustering the rows of *N* using a clustering algorithm described later.

D is an *nd*-by-*nd* symmetric square matrix. Once *H* is known, *D* is obtained from *N* by inverting eq. (1):

$$D = D_h^{-1} * H^T * N * H * D_h^{-1}, \quad (3)$$

where $D_h = H^T * H$ is an *nd*-by-*nd* diagonal matrix (see eq. (2)). $N_b = H^T * N * H$ is also an *nd*-by-*nd* square matrix. Suppose *N* is divided into *nd* by *nd* blocks, of which each block *(a,b)* is made of *nr(a)* by *nr(b)* residues. Then $N_b(a,b)$ is the sum of all the elements of *N* in block *(a,b)*. Equation (3) then becomes

$$D(a,b) = N_b(a,b)/(nr(a) * nr(b)), \quad (4)$$

which shows that *D(a,b)* is the average value of *N* over the block (*a,b),* or the density of *N* matrix in block *(a,b).*

The rows of the *N*-matrix were first hierarchically clustered into 12 (maximum number of domains considered) clusters using MATLAB (http://www.mathworks.com/) with ‘cosine’ for distance and ‘ward’ for linkage. The result of this clustering was used as the seed for a

'kmeans' clustering, again using ‘cosine’ as the distance, for 12 clusters. These clusters were then joined, a pair at a time, to produce successive trial solutions with number of domains from 12 to 1. The two clusters, *a* and *b*, to be joined were selected as the pair for which the ratio *D(a,b)/min(D(a,a),D(b,b))* was the largest.

The final solution was the one among these 12 putative solutions that had the maximum *Q* score, which was defined as

$$Q = Q_3 * Q_4 * Q_5 \quad (5)$$

where

$$Q_3 = 0.5 * erfc\ ((Q_1 - c_1)/s_1), \quad (6)$$

$$Q_4 = 1 - 0.5 * erfc\ ((Q_2 - c_2)/s_2), \quad (7)$$

$$Q_1 = \max(\ D(a,b)/\min(\ D(a,a), D(b,b))), \quad (8)$$

$$Q_2 = \min(\ D(a,a))/D_1, \quad (9)$$

and

$$Q_5(nd) = Q_1(nd+1) - Q_1(nd). \quad (10)$$

Q_3 is a smoothened switch function, which varies from 1 to 0. The midpoint of the switch occurs when $Q_1=c_1$ and the sharpness of the transition is controlled by s_1. Q_4 is also a smoothened switch function, but it varies from 0 to 1 according to Q_2. D_1 is the value of the 1-by-1 *D* matrix for one-domain solution, which is equal to the overall density of points in the *N*-matrix. The values of the four parameters, *c1, s1, c2,* and *s2* were chosen as 0.3, 0.05, 1.0, and 0.5 after some trials.

Q_5 measures the increase in Q_1 when the number of domains (*nd*) is increased by 1. While other functions (Q_1 through Q_4) measure the intrinsic property of an *nd*-domain solution, Q_5 measures the merit (Q_1) of a solution relative to the (*nd+1)*-domain solution in which a domain is split into two. Curiously, we observed that the score function that included this factor performed better on average than those that did not include this factor.

One can obtain a ‘calculated’ *N*-matrix by

$$N_c = H * D' * H^T \qquad (11)$$

D' is matrix *D* with all off-diagonal terms set to zero. N_c is like the equivalence relation matrix describe in the next section, except that the non-zero blocks have the average density of the original *N*-matrix rather than unity as in a true equivalence relation matrix. A score function based on the norm of the difference between *N* and N_c did not perform as well as that given by eq. (5), presumably because of the large difference between the actual and the average number of pLSSPs within each domain even when the domain assignment is perfect.