Supporting Information

Table of Content

Supporting Texts

Text S1. Template ranking and modeling method in LOMETS2

For a given target, 220 templates are generated by 11 component threading programs, where the top 20 templates sorted in descending order of Z-scores are selected from each program. The Z-score of a template alignment is calculated by:

$$Z-score(i,j)=\frac{S(i,j)-\langle S(j)\rangle}{\sigma(j)} \quad (S1)$$

where $S(i,j)$ is the threading alignment score of the *i*-th template for the *j*-th server and ⟨S(j)⟩ and $\sigma(j)$ are the average alignment score and standard deviation across all templates for the *j*-th server, respectively.

The top 10 templates are selected from the 220 templates based on the following scoring function:

$$score(i,j)=conf(j)*\frac{Z-score(i,j)}{Z_0(j)}+seqid(i,j) \quad (S2)$$

where $seqid(i,j)$ is the sequence identity to query of the *i*-th template for the *j*-th server. Furthermore, $conf(j)$ is the confidence of the *j*-th server, which is equal to the average TM-score to the native structure for the first template calculated from a training set of 243 non-redundant proteins. $Z_0(j)$ is the Z-score cut-off for defining good/bad templates for the *j*-th server, which was decided by maximizing the Matthews correlation coefficient (MCC) for distinguishing a good template (with a TM-score ≥0.5) from a bad template (TM-score <0.5) on the same training set. As a result, the parameters $Z_0(j)$ (and $conf(j)$) are 5.6 (0.617), 83.0 (0.589), 6.9 (0.587), 33.0 (0.574), 8.7 (0.570), 6.1 (0.569), 10.0 (0.567), 7.0 (0.566), 7.6 (0.562), 3.2 (0.558), and 21.0 (0.536) for CEthreader, HHpred, SparksX, FFAS3D, Neff-MUSTER, MUSTER, HHsearch, SP3, PPAS, PROSPECT2, and PRC, respectively.

The five full-length models are constructed by MODELLER (1) based on the top 5 templates identified by LOMETS2. For each model, spatial restraints are collected from a set of similar templates, which have a close structural similarity (with a TM-score >0.5) to the target template, to assist the MODELLER structure modeling.

Text S2. The normalized number of effective sequences (Neff) in an MSA

The depth of a multiple sequence alignment (MSA) can be measured by the normalized number of effective sequences (*Neff*):

$$Neff=\frac{1}{\sqrt{L}}\sum_{n=1}^{N}\frac{1}{1+\sum_{m=1,m\neq n}^{N}I\left[S_{m,n}\geq 0.8\right]} \quad (S3)$$

where *L* is the length of a query protein, *N* is the number of sequences in the MSA, $S_{m,n}$ is the sequence identity between the *m*-th and *n*-th sequences. $I\left[S_{m,n}\geq 0.8\right]$ is equal to 1 if $S_{m,n}\geq 0.8$, or zero otherwise. Therefore, *Neff* is essentially equal to the number of non-redundant sequences (sequence identity<0.8) in the MSA normalized by the query length.

Text S3. ResPRE for contact-map prediction

ResPRE (https://zhanglab.ccmb.med.umich.edu/ResPRE/) is a newly developed method for contact-map prediction (2), which consists of two consecutive steps of precision matrix-based feature generation and deep residual neural network-based contact inference.

Precision matrix-based feature generation. Given an MSA with *N* sequences and *L* positions, the frequencies of observing residue type *a* at position *i* is denoted as $f_i(a)$, and the

co-occurrence of the two residue types *a* and *b* at positions *i* and *j* is denoted as $f_{i,j}(a,b)$. These can be estimated by

$$\begin{cases} f_i(a) = \dfrac{1}{\lambda + \sum_{n=1}^{N} 1/m_n}\left[\dfrac{\lambda}{q} + \sum_{n=1}^{N}\dfrac{1}{m_n}\delta_{a,a_i^n}\right] \\ f_{i,j}(a,b) = \dfrac{1}{\lambda + \sum_{n=1}^{N} 1/m_n}\left[\dfrac{\lambda}{q^2} + \sum_{n=1}^{N}\dfrac{1}{m_n}\delta_{a,a_i^n},\delta_{b,a_j^n}\right] \end{cases} \quad (S4)$$

where $\delta_{a,b^n} = 1$ if *a* and *b* are identical for the *n*-th sequence, or =0 otherwise; $\lambda = 1$ is the pseudocount to approximate the background observation; $1/m_n$ is used to reweight the *n*-th sequence, where m_n is the number of sequences in the MSA that have a sequence identity >80% to the *n*-th sequence; q is the number of possible residue types at one position and is set to 21 (i.e., 20 naturally-occurring residue types plus gap).

By extending each position of the MSA into a 21-dimensional vector using one-hot encoding, a 21×L by 21×L covariance matrix *S* can be computed by

$$S_{i,j}^{a,b} = f_{i,j}(a,b) - f_i(a)f_j(b) \quad (S5)$$

where all the parameters for marginal correlation between residue pair *i* and *j* is represented by a 21×21 submatrix $S_{i,j}$. From this covariance matrix, we can further obtain the inverse covariance matrix Θ (i.e., precision matrix) by minimizing the following objective function:

$$\sum_{i=1}^{L} S_{i,i}\cdot\Theta_{i,i} - \log\big(\det(\Theta)\big) + \rho\sum_{i=1}^{L}\sum_{j=i}^{L}\left\|\Theta_{i,j}\right\|_2^2 \quad (S6)$$

Here, the first term is the trace of $S\cdot\Theta$, the second term is the negative log determinant, and the last term is the L2 norm. The estimated precision matrix $\widehat{\Theta}$ that minimizes **Eq. (*S6*)** is therefore the inverse matrix of *S* under L2 regularization.

The precision matrix can be further split into $L \times L$ blocks, where each block represents a 21×21 matrix that indicates the direct coupling correlations for 21×21 residue type pairs at the corresponding position pair. The 441(=21×21) dimensional descriptors from the corresponding block will be utilized as the features and will be directly fed into the following deep neural network, without further pre-processing.

Deep residual neural network architecture. A protein contact-map can be treated as a two-dimensional image, where each pixel is one pair of residues. The contact prediction problem can thus be formulated as an image segmentation problem, i.e., a pixel-wise labeling problem, in computer vision. This makes the contact prediction problem naturally suitable for deep Convolutional Neural Networks (CNN), especially the recently proposed Residual Neural Network architecture.

In ResPRE, the fully residual networks (FRNs) for contact-map prediction take the L×L×441 precision matrix as input, where 441 is the number of input feature channels, to predict the L×L contact-map. The first layer employs a 1×1 convolutional kernel to reduce the feature dimension from 441 to 64. The reduced feature map output by this convolutional layer is then sequentially fed into 22 residual blocks where the kernel sizes are 3×3, the padding size is 1, and the activation function is ReLU. Finally, a convolutional layer with a 3×3 kernel size is used to obtain the final contact-map prediction with a sigmoid activation function.

Text S4. CEthreader for contact-map guided fold-recognition

CEthreader (https://zhanglab.ccmb.med.umich.edu/CEthreader/) is a fold-recognition algorithm to identify similar-fold structures from the PDB under the guidance of predicted contact-maps. The core part of the algorithm consists of contact-map prediction, eigen-decomposition of the contact matrix, and contact-guided template search and selection.

Contact-map prediction and the selection of contacts. The contact-map of a query sequence (with C_β distance <8 Å) is predicted using the ResPRE method (2) by coupling evolutionary precision matrices with deep residual neural networks (see **Text S3** for a detailed description). The top $N = \sum_{cr\in\{long,medium,short\}} \sigma_{cr} L$ predicted residue pairs, ranked by their confidence scores, are selected to form the final contact-map of the query, where cr refers to the long-, medium- and short-range contacts with sequence separation $|i-j| \geq 24$, $23 \geq |i-j| \geq 12$, and $|i-j| \leq 11$, respectively. The parameters σ_{cr} were determined by maximizing the TM-score of CEthreader on a set of 905 training protein-pairs.

Eigen-decomposition of contact-maps. The contact-map can be represented by an $L\times L$ symmetric binary matrix, M, in which residue pairs that form contacts are designated as 1 and non-contacting pairs are set to 0. Based on the eigen-decomposition theory, we can infer that

$$M = \sum_{k=1}^{L} \lambda_k \overrightarrow{V_k} * \overrightarrow{V_k}^T \quad (S7)$$

where λ_k represents the k-th eigenvalue of M, and $\overrightarrow{V_k} = \left(v_{1,k}, v_{2,k}, \cdots, v_{L,k}\right)^T$ is the corresponding eigenvector. Since the contribution of each eigenvector depends on the absolute magnitude of the eigenvalue in **Eq. (S7)**, the contact-map can be approximated by considering only the largest K positive eigenvalues and associated eigenvectors:

$$M \approx \sum_{k=1}^{K} \lambda_k \overrightarrow{V_k} * \overrightarrow{V_k}^T \quad (S8)$$

Here, we do not consider the negative eigenvalues because it introduces complex numbers to the following computation.

Based on **Eq. (S8)**, any pair of contacts between residues i and j can be written as

$$M_{i,j} \approx \left(\sqrt{\lambda_1} v_{i,1}, \sqrt{\lambda_2} v_{i,2}, \ldots, \sqrt{\lambda_K} v_{i,K}\right) * \left(\sqrt{\lambda_1} v_{j,1}, \sqrt{\lambda_2} v_{j,2}, \ldots, \sqrt{\lambda_K} v_{j,K}\right)^T \quad (S9)$$

In this way, the contact profiles for the i- and j-th residues are described by the vectors

$$\begin{cases} \overrightarrow{U_i} = \left(\sqrt{\lambda_1} v_{i,1}, \sqrt{\lambda_2} v_{i,2}, \ldots, \sqrt{\lambda_k} v_{i,K}\right) \\ \overrightarrow{U_j} = \left(\sqrt{\lambda_1} v_{j,1}, \sqrt{\lambda_2} v_{j,2}, \ldots, \sqrt{\lambda_k} v_{j,K}\right) \end{cases} \quad (S10)$$

This representation of contact-maps using single-body profiles allows the integration of the contact-map into dynamic program alignment for fold recognition.

Contact-guided fold recognition. The fold-recognition in CEthreader is performed by threading the query sequence through a non-redundant structure set collected from the PDB library. The contact-guided alignment score for aligning the i-th residue of the query to the j-th residue of the template protein is represented by

$$S_{cm+ss+prof}(i,j) = w_1 * S_{cm}(i,j) + w_2 * S_{prof}(i,j) + w_3 * S_{ss}(i,j) + w_4 \quad (S11)$$

Here, the first term accounts for the contact-map match between the query and template by:

$$S_{cm}(i,j) = \begin{cases} \dfrac{\overrightarrow{U_i} \cdot \overrightarrow{P_j}}{\max(|\overrightarrow{U_i}|, |\overrightarrow{P_j}|)^2} & \text{if } |\overrightarrow{U_i}| \neq \vec{0} \text{ and } |\overrightarrow{P_j}| \neq \vec{0} \\ 0 & \text{if } \left|\overrightarrow{U_i}\right| = |\overrightarrow{P_j}| = \vec{0} \end{cases} \quad (S12)$$

where $\overrightarrow{U_i}$ and $\overrightarrow{P_j}$ are the contact eigenvectors for the i-th residue of the query and the j-th residue of the template as defined in **Eq. (S10)**. The second and third terms, $S_{prof}(i,j)$ and $S_{ss}(i,j)$, account for the sequence profile-to-profile and secondary structure alignment scores between the i-th residue of the query and the j-th residue of the template. The weighting parameters are determined by maximizing the TM-score of the 905 training protein-pairs described above.

The Needleman-Wunsch dynamic programming algorithm is used to align two protein sequences, where an affine penalty scheme is utilized for an l residue gap where $G = g_o + g_e l$. g_o and g_e were determined using the 905 training protein pairs.

Text S5. Summary of the 11 component threading methods used in LOMETS2

The LOMETS2 server integrates predictions from 11 programs that represent a diverse set of state-of-the-art threading algorithms, including CEthreader, HHsearch (3), HHpred (4), SparksX (5), FFAS3D (6), Neff-MUSTER, MUSTER (7), SP3 (8), PPAS (9), PROSPECT2 (10), and PRC (11). All the programs are installed and run on our local supercomputer cluster and the template libraries are updated every week. Currently, the template library contains 74,336 domains/chains with a pairwise sequence identity <70%. For a protein chain that consists of multiple domains, both the whole chain and individual domain structures are included in the library. Below we give an overview of the threading programs that are used by the LOMETS2 server.

CEthreader is an in-house contact-based threading method. A detailed introduction of the algorithm is given in **Text S4**.

HHsearch (3) is based on profile hidden Markov models (HMMs), an extension of sequence profiles which also contain information on position-specific probabilities of amino acid matches, insertions, and deletions as well as the frequencies of emitting different amino acid types. By default, HHsearch builds an MSA for a query sequence using the HHblits (12) program from the HH-Suite package. From this alignment, a profile HMM is calculated. HHsearch searches the profile HMM for a query through a pre-built template database of profile HMMs, and outputs a ranked list of templates based on threading probability score as well as pairwise query-template alignments.

HHpred (4) is an extension of HHsearch. Starting from the template list generated by HHsearch, the templates are selected and re-ranked as follows: the first template is selected by a neural network, which uses 4 features to predict the template TM-score. Next, for each template in the template list, a score, which rewards homology quality and alignment quality, is calculated. The template with the highest score is iteratively added until no template has a positive score. Additionally, HHpred allows users to build a homology model using MODELLER (1) based on distance restraints collected from templates which follow multi-component Gaussian mixture distributions.

SP3 (8) detects templates by dynamic programming, where the alignment score contains structural fragment-derived sequence profiles, evolution-derived sequence profiles and secondary structure information.

SparksX (5) is an extension of the SP3 program, which calculates the alignment scores based on the estimated probability of a match between predicted and actual single-body structural properties, and incorporates the prediction of secondary structure, backbone torsional angles and solvent accessible surface area.

FFAS3D (6) implements local dynamic programming to perform profile-profile comparison based on structural features including sequence profiles, secondary structure, solvent accessibility, and residue depth. A template re-ranking strategy is proposed based on a neural network.

MUSTER (7) and Neff-MUSTER are two in-house programs built on Needleman-Wunsch dynamic programming. The alignment score contains terms from sequence profiles, secondary structures, structure fragment profiles, solvent accessibility, dihedral torsional angles, and hydrophobic scoring matrices. The major difference between MUSTER and Neff-MUSTER is that MUSTER uses fixed weights to combine the different terms but the weights in Neff-MUSTER are dynamically determined, in particular the weight for the sequence profile, by the effective number of sequences (*Neff*) in the MSA.

PPAS (9) is another in-house threading program based on profile-profile alignments in conjunction with secondary structure matches, where alignments are generated by a global dynamic programming algorithm.

PROSPECT2 (10) uses a scoring function that includes residue mutations, secondary structure propensity, solvent accessibility, and a generic pairwise contact potential. A divide-and-conquer searching approach is exploited to generate the global optimization of alignments.

PRC (11) is based on profile HMM–HMM alignments, which are computed by finding the Viterbi path that maximizes the sum of the forward–backward odds scores. The profile HMMs are generated by SAM-T2k (13).

Supporting Tables

Table S1. Comparison between the TM-scores for the first templates identified by different threading programs for the 614 test proteins using default profiles versus those using deep profiles built by the DeepMSA program (marked by 'D'). *P*-values are calculated between the TM-scores for the default and deep profile-based methods using one-sided Student's t-tests. Coverage is equal to the number of aligned residues divided by the length of the query sequence. N_{st} is the number of targets with a TM-score >0.5.

Type	Methods	TM-score	*p*-value	RMSD (Å)	Coverage	N_{st}
All	LOMETS2	0.6076	-	6.3127	0.8818	457
(614)	LOMETS2 (D)	0.6210	5.99E-08	6.3294	0.9057	478
	CEthreader	0.6010	-	6.6685	0.8957	448
	CEthreader (D)	0.6125	9.14E-08	6.3423	0.8977	467
	HHpred	0.5760	-	6.8778	0.8416	418
	HHpred (D)	0.5892	1.47E-05	6.5800	0.8463	429
	SparksX	0.5712	-	7.3927	0.8842	409
	SparksX (D)	0.5867	1.15E-06	7.1062	0.8864	429
	FFAS3D	0.5683	-	6.9449	0.8498	407
	FFAS3D (D)	0.5740	1.80E-01	6.8767	0.8574	413
	Neff-MUSTER	0.5566	-	7.8963	0.8956	397
	Neff-MUSTER (D)	0.5700	2.15E-08	7.4410	0.8926	414
	MUSTER	0.5532	-	7.9918	0.8929	383
	MUSTER (D)	0.5693	7.34E-10	7.6111	0.8853	408
	HHsearch	0.5579	-	6.8497	0.8084	404
	HHsearch (D)	0.5674	6.72E-06	6.6798	0.8160	414
	SP3	0.5164	-	9.5843	0.8918	346
	SP3 (D)	0.5656	1.49E-15	7.8658	0.8690	408
	PPAS	0.5408	-	8.3076	0.8707	381
	PPAS (D)	0.5618	4.33E-11	7.8848	0.8712	406
	PROSPECT2	0.5348	-	9.4497	0.9161	368
	PROSPECT2 (D)	0.5584	1.01E-12	8.5037	0.9074	395
	PRC	0.5005	-	7.2205	0.7530	352
	PRC (D)	0.5385	1.10E-20	6.0480	0.7626	384
Easy	LOMETS2	0.7018	-	4.5344	0.9057	380
(403)	LOMETS2 (D)	0.7055	4.32E-03	4.5332	0.9074	384
	CEthreader	0.6895	-	4.8330	0.9127	371
	CEthreader (D)	0.6977	1.03E-05	4.7727	0.9173	380
	HHpred	0.6963	-	4.6177	0.8967	375
	HHpred (D)	0.7018	3.70E-03	4.5753	0.9033	379
	SparksX	0.6920	-	4.7451	0.9023	370
	SparksX (D)	0.6959	1.60E-03	4.8226	0.9096	375
	FFAS3D	0.6840	-	4.4767	0.8857	373
	FFAS3D (D)	0.6810	9.04E-01	4.4807	0.8896	371
	Neff-MUSTER	0.6828	-	4.7875	0.9012	366
	Neff-MUSTER (D)	0.6886	2.10E-07	4.7537	0.9040	370
	MUSTER	0.6800	-	5.0450	0.9039	358
	MUSTER (D)	0.6869	4.04E-09	4.8593	0.9035	367
	HHsearch	0.6880	-	4.5878	0.8946	372
	HHsearch (D)	0.6882	1.84E-02	4.8170	0.8966	369

	SP3	0.6443	-	6.2238	0.9061	331
	SP3 (D)	0.6879	3.09E-09	5.1743	0.9091	368
	PPAS	0.6691	-	5.0719	0.8914	358
	PPAS (D)	0.6786	4.52E-08	4.8243	0.8923	363
	PROSPECT2	0.6663	-	5.9436	0.9175	347
	PROSPECT2 (D)	0.6821	2.35E-13	5.5249	0.9140	365
	PRC	0.6456	-	5.0052	0.8550	339
	PRC (D)	0.6678	7.55E-10	4.3943	0.8608	355
Hard (211)	LOMETS2	0.4277	-	9.7090	0.8360	77
	LOMETS2 (D)	0.4595	1.24E-06	9.7602	0.8836	94
	CEthreader	0.4319	-	10.1742	0.8633	77
	CEthreader (D)	0.4497	1.08E-03	9.3401	0.8604	87
	HHpred	0.3463	-	11.1946	0.7364	43
	HHpred (D)	0.3740	3.00E-04	10.4089	0.7374	50
	SparksX	0.3405	-	12.4496	0.8498	39
	SparksX (D)	0.3780	4.73E-05	11.4678	0.8419	54
	FFAS3D	0.3472	-	11.6590	0.7813	34
	FFAS3D (D)	0.3696	1.60E-03	11.4530	0.7960	42
	Neff-MUSTER	0.3156	-	13.8339	0.8849	31
	Neff-MUSTER (D)	0.3434	2.00E-03	12.5737	0.8708	44
	MUSTER	0.3110	-	13.6201	0.8718	25
	MUSTER (D)	0.3447	7.00E-04	12.8668	0.8506	41
	HHsearch	0.3095	-	11.1698	0.6438	32
	HHsearch (D)	0.3366	1.76E-05	10.2377	0.6619	45
	SP3	0.2721	-	16.0026	0.8646	15
	SP3 (D)	0.3320	2.54E-08	13.0066	0.7926	40
	PPAS	0.2956	-	14.4876	0.8311	23
	PPAS (D)	0.3387	1.05E-05	13.7302	0.8309	43
	PROSPECT2	0.2835	-	16.1461	0.9133	21
	PROSPECT2 (D)	0.3220	1.00E-03	14.1930	0.8948	30
	PRC	0.2234	-	11.4515	0.5581	13
	PRC (D)	0.2916	1.27E-13	9.2065	0.5749	29

Table S2. Comparison between the full-length models generated by LOMETS2 and its component threading programs for the 614 test proteins. *P*-values are calculated between the TM-scores of the LOMETS2 models and its component threading programs using one-sided Student's t-tests. N_{st} is the number of targets with a TM-score >0.5.

Type	Methods	TM-score	*p*-value	RMSD (Å)	N_{st}
All (614)	LOMETS2	0.6753	-	7.2050	523
	CEthreader	0.6391	2.94E-61	8.4379	496
	HHpred	0.6151	1.13E-77	9.8186	452
	SparksX	0.6057	1.12E-84	9.1840	442
	FFAS3D	0.5964	2.61E-87	10.2415	434
	HHsearch	0.5946	1.55E-82	11.1002	432
	MUSTER	0.5893	8.51E-86	9.9167	423
	Neff-MUSTER	0.5882	1.03E-87	9.8094	425
	SP3	0.5843	2.01E-89	10.4776	421
	PPAS	0.5836	1.74E-85	10.2869	422
	PROSPECT2	0.5763	3.32E-84	9.5524	404
	PRC	0.5707	1.11E-91	13.0354	404
Easy (403)	LOMETS2	0.7587	-	5.4001	398
	CEthreader	0.7218	3.64E-45	6.4460	386
	HHpred	0.7242	5.43E-52	6.4432	384
	SparksX	0.7157	8.80E-55	6.4732	383
	FFAS3D	0.7025	3.89E-60	7.3228	380
	HHsearch	0.7109	2.49E-54	6.9233	375
	MUSTER	0.7061	2.90E-54	6.9357	373
	Neff-MUSTER	0.7078	2.14E-55	6.8481	379
	SP3	0.7059	7.41E-57	6.9997	375
	PPAS	0.7002	2.78E-56	7.1106	373
	PROSPECT2	0.6987	5.12E-51	6.6738	366
	PRC	0.6966	2.52E-58	7.8082	363
Hard (211)	LOMETS2	0.5160	-	10.6521	125
	CEthreader	0.4811	1.10E-17	12.2422	110
	HHpred	0.4068	1.18E-31	16.2655	68
	SparksX	0.3957	4.18E-32	14.3614	59
	FFAS3D	0.3938	4.55E-31	15.8162	54
	HHsearch	0.3725	3.14E-31	19.0780	57
	MUSTER	0.3664	7.77E-34	15.6101	50
	Neff-MUSTER	0.3596	6.98E-34	15.4654	46
	SP3	0.3522	1.39E-34	17.1201	46
	PPAS	0.3608	1.91E-32	16.3534	49
	PROSPECT2	0.3425	3.76E-34	15.0504	38
	PRC	0.3301	8.76E-35	23.0192	41

Table S3. Model quality comparison between LOMETS2 and its component threading programs for 121 CASP13 domains. *P*-values are calculated between the TM-scores of the LOMETS2 models and its component threading programs using one-sided Student's t-tests.

Methods	**TM-score**	**p-value**	**RMSD (Å)**
LOMETS2	0.638	-	6.41
CEthreader	0.600	1.36E-08	6.73
HHpred	0.571	3.21E-06	14.95
SparksX	0.570	5.77E-07	10.49
FFAS3D	0.562	5.31E-08	10.84
HHsearch	0.561	3.57E-08	15.23
MUSTER	0.559	2.03E-08	10.94
SP3	0.555	5.21E-09	10.81
Neff-MUSTER	0.553	1.58E-08	11.21
PPAS	0.551	4.29E-09	11.81
PRC	0.548	6.32E-10	18.72
PROSPECT2	0.540	1.06E-10	10.65

Supporting Figures

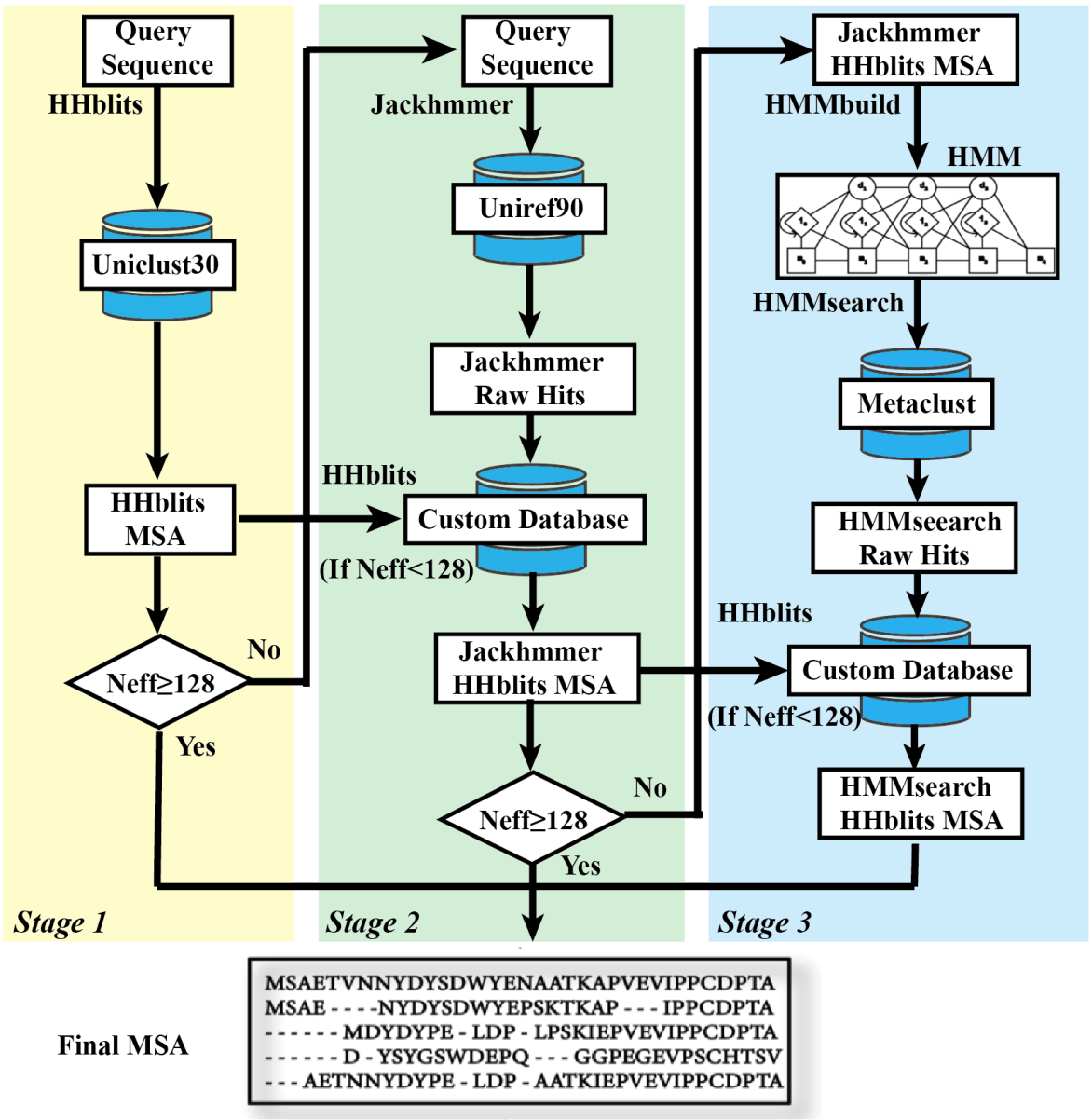

Figure S1. Flowchart of the DeepMSA algorithm, showing the three stages of MSA generation using sequences from HHblits search against Uniclust30 (first column in yellow), Jackhmmer search through UniRef (second column in green) and HMMsearch through Metaclust (third column in cyan).

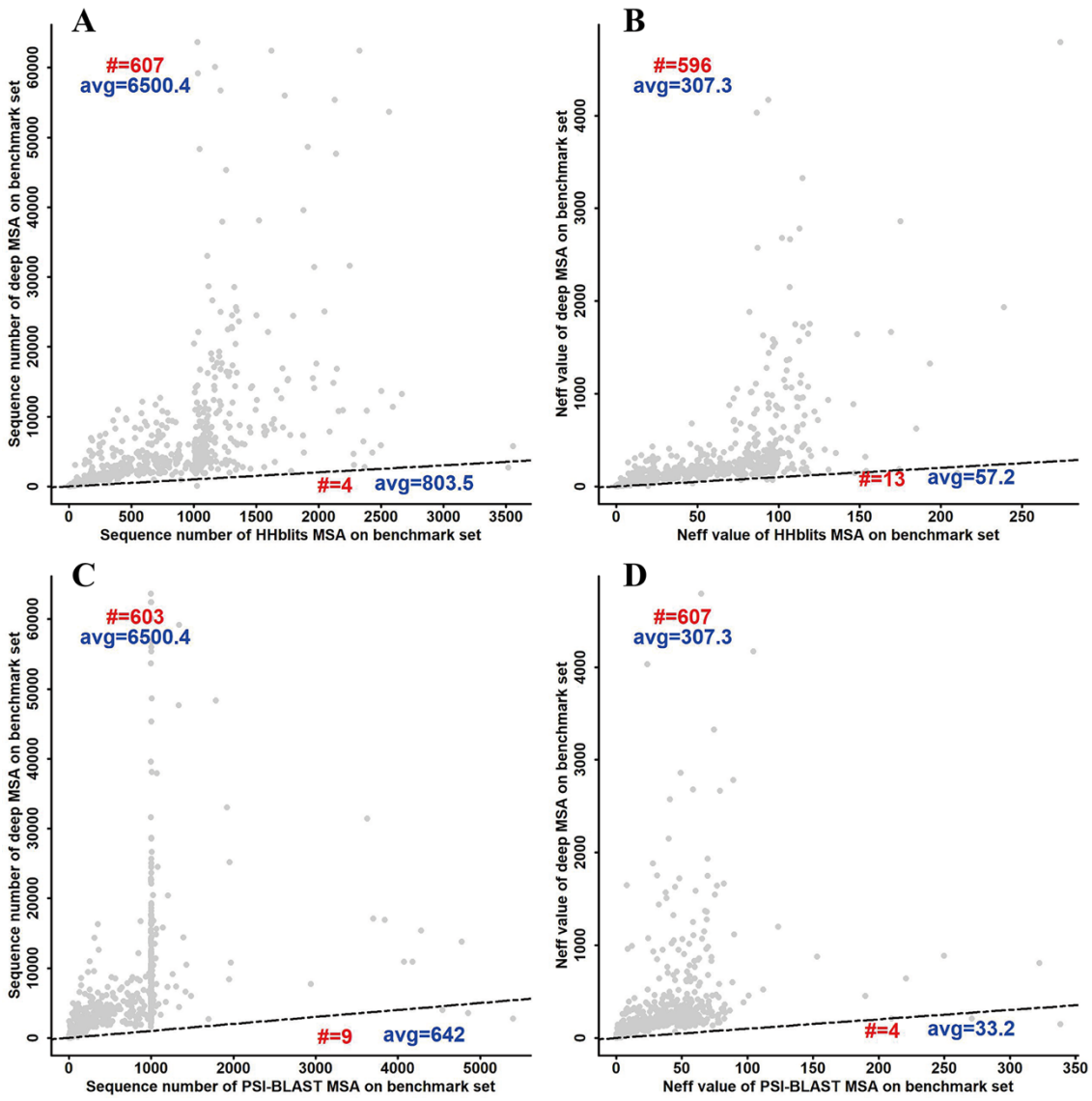

Figure S2. Comparison between the number of homologous sequences detected by our deep MSA generation program and the default programs that were used in the previous iteration of LOMETS. (A) Number of sequences in the deep MSAs and HHblits MSAs; (B) Number of effective sequences (*Neff*) in the deep MSAs and HHblits MSAs; (C) Number of sequences in the deep MSAs and PSI-BLAST MSAs; (D) *Neff* for the deep MSAs and PSI-BLAST MSAs. The black dashed-dotted line represents the diagonal line (i.e. $y = x$). The red numbers are the number of points above or below the diagonal line.

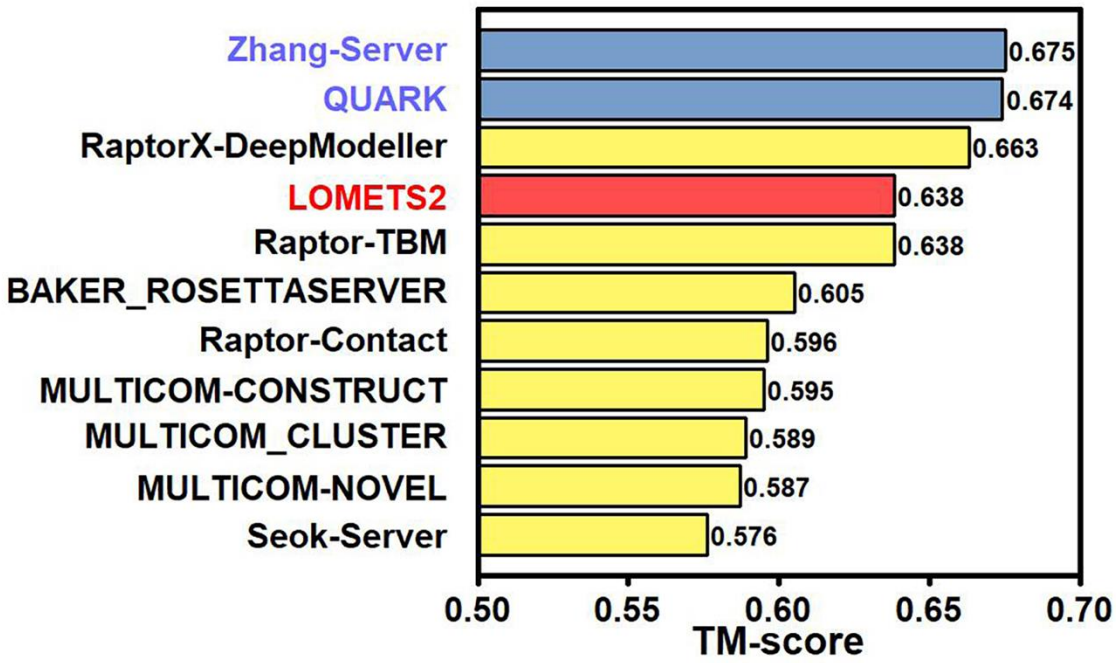

Figure S3. Comparison between the LOMETS2 modeling results and the top ten servers in CASP13 based on average TM-scores of the first models for the 121 released CASP13 domains. 'Zhang-Server' (blue) and 'QUARK' (blue) used 'LOMETS2' (red) as their template identification component in CASP13. Note that the CEthreader server also participated in CASP13 but it did not use the DeepMSA method to generate deep profiles and predict secondary structures at that time. Therefore, it had a moderate average TM-score of 0.566. When we re-ran the program with the deep MSAs, the average TM-score of CEthreader increased from 0.566 to 0.600 (see **Table S3**), which put it at the #6 position in the list. However, the difference between LOMETS2 and CEthreader for these domains is similar to the difference for the in-house benchmark test results (**Table S2**), showing the consistent improvement of LOMETS2 over its component programs.

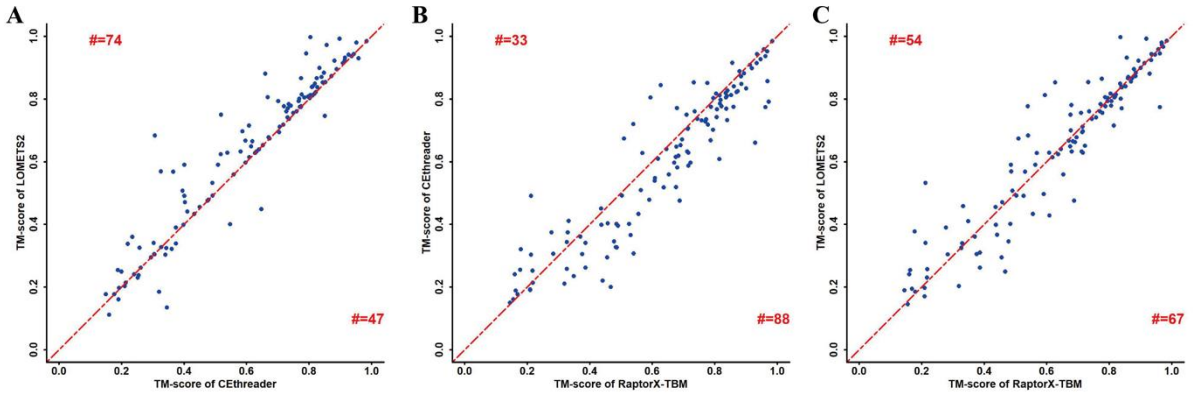

Figure S4. TM-score comparison between the first models generated by LOMETS2, CEthreader and RaptorX-TBM for the 121 released CASP13 domains. The RaptorX-TBM models were downloaded from the CASP13 webpage, and LOMETS2 and CEthreader were re-run with DeepMSA profiles after CASP13, but all templates generated after May 2018 were excluded. (A) LOMETS2 vs. CEthreader; (B) CEthreader vs. RaptorX-TBM; (C) LOMETS2 vs. RaptorX-TBM. The numbers in different sections indicate the number of proteins above or below the diagonal line.

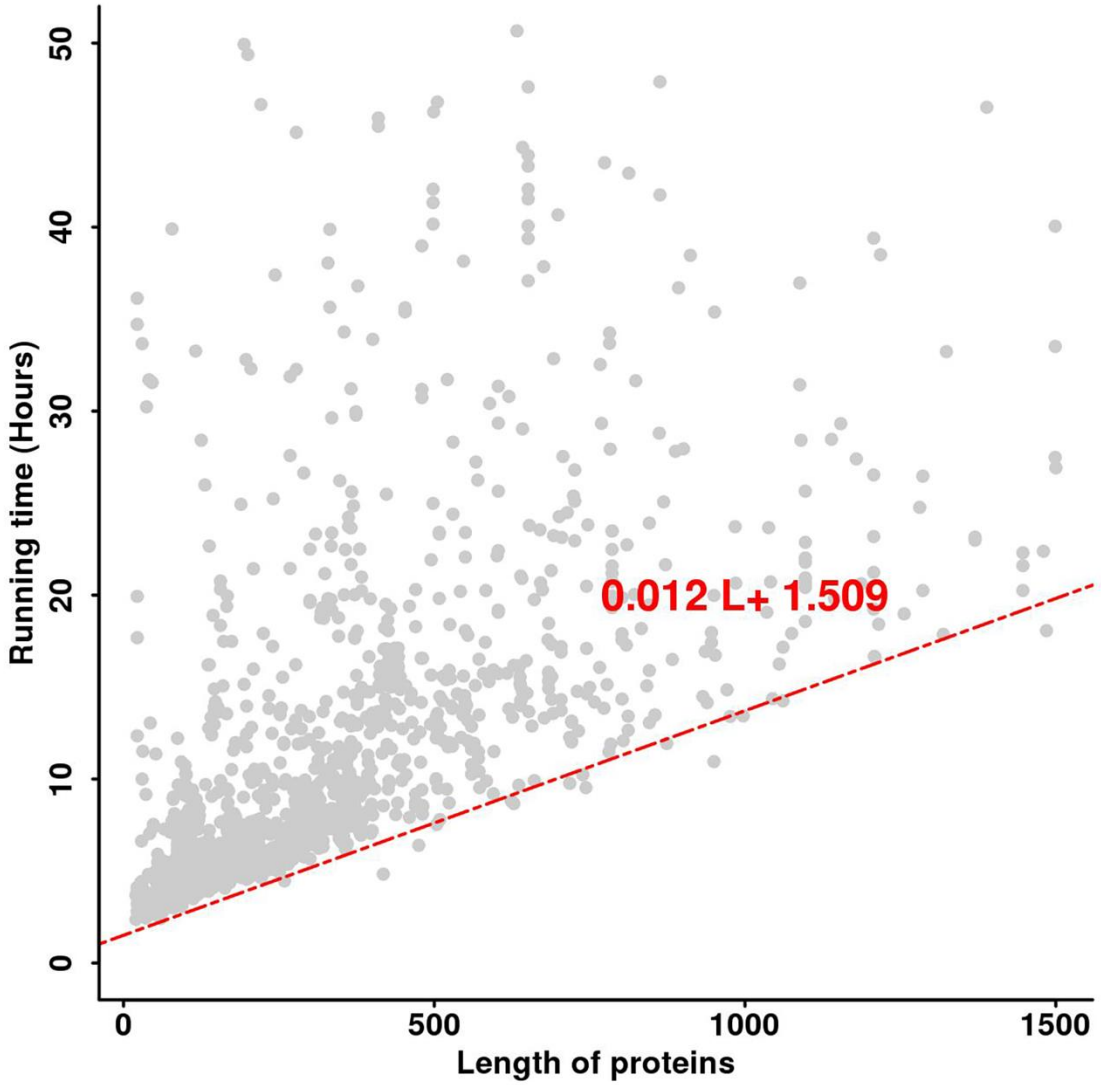

Figure S5. The actual response time versus protein size for the 1,433 jobs processed by the LOMETS2 server recently. In cases where many jobs are accumulated in the queue, it will take a longer time for the job to finish due to increased queue waiting times. The red line is fit to the targets with the quickest response time, which should correspond to the actual running time of the LOMETS2 programs when the job queue is clear.

Reference

1. Šali, A. and Blundell, T.L. (1993) Comparative protein modelling by satisfaction of spatial restraints. *Journal of molecular biology*, 234, 779-815.
2. Li, Y., Hu, J., Zhang, C., Yu, D. and Zhang, Y. (2019) ResPRE: high-accuracy protein contact prediction by coupling precision matrix with deep residual neural networks. *Bioinformatics (Oxford, England)*, in press.
3. Soding, J. (2005) Protein homology detection by HMM-HMM comparison. *Bioinformatics (Oxford, England)*, 21, 951-960.
4. Meier, A. and Söding, J. (2015) Automatic prediction of protein 3D structures by probabilistic multi-template homology modeling. *PLoS computational biology*, 11, e1004343.
5. Yang, Y., Faraggi, E., Zhao, H. and Zhou, Y. (2011) Improving protein fold recognition and template-based modeling by employing probabilistic-based matching between predicted one-dimensional structural properties of query and corresponding native properties of templates. *Bioinformatics (Oxford, England)*, 27, 2076-2082.
6. Xu, D., Jaroszewski, L., Li, Z. and Godzik, A. (2014) FFAS-3D: improving fold recognition by including optimized structural features and template re-ranking. *Bioinformatics (Oxford, England)*, 30, 660-667.
7. Wu, S. and Zhang, Y. (2008) MUSTER: Improving protein sequence profile-profile alignments by using multiple sources of structure information. *Proteins*, 72, 547-556.
8. Zhou, H. and Zhou, Y. (2005) Fold recognition by combining sequence profiles derived from evolution and from depth-dependent structural alignment of fragments. *Proteins*, 58, 321-328.
9. Wu, S. and Zhang, Y. (2007) LOMETS: a local meta-threading-server for protein structure prediction. *Nucleic acids research*, 35, 3375-3382.
10. Xu, Y. and Xu, D. (2000) Protein threading using PROSPECT: design and evaluation. *Proteins*, 40, 343-354.
11. Madera, M. (2008) Profile Comparer: a program for scoring and aligning profile hidden Markov models. *Bioinformatics (Oxford, England)*, 24, 2630-2631.
12. Remmert, M., Biegert, A., Hauser, A. and Soding, J. (2011) HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature methods*, 9, 173-175.
13. Karplus, K., Barrett, C. and Hughey, R. (1998) Hidden Markov models for detecting remote protein homologies. *Bioinformatics (Oxford, England)*, 14, 846-856.