Supplementary Data to HH-MOTiF: *de novo* detection of short linear motifs in proteins by Hidden Markov Model comparisons

Authors: Roman Prytuliak[1] (prytuliak@biochem.mpg.de), Michael Volkmer[1] (volkmer@biochem.mpg.de), Markus Meier[2] (markus.meier@mpibpc.mpg.de), Bianca H. Habermann[1,3,*] (bianca.HABERMANN@univ-amu.fr ; habermann@biochem.mpg.de)

[1] Computational Biology Group, Max Planck Institute of Biochemistry, Martinsried, Germany
[2] Research Group Quantitative Biology and Bioinformatics, Max Planck Institute for Biophysical Chemistry, Göttingen, Germany
[3] Computational Biology Group, Developmental Biology Institute of Marseille (IBDM) UMR 7288, CNRS, Aix Marseille Université, Marseille 13288, France

1. Additional workflow details

This section does not contain full descriptions of the workflow steps, but rather relevant additions and explanations to the main text.

Residue masking: After the mask is obtained, it undergoes a consolidation procedure, which ensures that there is no consecutive stretch of either masked or unmasked residues shorter than 3 residues. This is equivalent to moving, if needed, the border between masked and unmasked stretches, by 1 residue. The implementation consists of two steps: first, all stretches of 3 or more residues with the same masking status are identified in a greedy manner; the masking status of these residues will remain unchanged. Second, the masking status of the remaining stretches is decided by the simple majority rule; stretches containing exactly half the number of masked residues become unmasked. The termini are treated as if infinite blocks of masked residues flanked the sequence.

Hidden Markov Model creation and comparison: HMMs in *.hhm format are generated from the aligned FASTA files with *hhmake* providing the following options: '`-id 100`' (which removes the upper cut-off for sequence identity; this offers the user the flexibility to submit his/her own orthologs in ***advanced mode***, allowing for orthologs with more than 95% sequence identity) and '`-M first`' (so that motifs are searched only in the query sequence and not the orthologs; query residues will not be masked even if there are only gaps in the aligned orthologs). For the comparisons, *hhalign* is used with the options '`-smin 0 -alt 100`' (reporting multiple, also suboptimal hits), '`-template_excl`' (to permit exclusion of masked residues in both HMMs of a pair), '`-norealign`' (which switches off additional realignments with the maximal accuracy algorithm) and '`-nocontxt`' (which switches off context-specific Viterbi score adjustment); the gap restriction option is set with the parameters '`-gaph 999 -gapi 999`' (which effectively sets the gap continuation penalties to infinity).

Motif tree assembly and evaluation: In the input set of N proteins submitted, the minimal number of leaves is N_{min}-1, where N_{min} is calculated according to the **False positive evaluation model** described in **section 2.**

In addition to the requirement that each alignment hit must have at least 3 aligned positions (hereafter referred to as columns), we also check that its sequence stretch in the motif root encompasses at least 4 residues. This is achieved through having either at least 4 columns or at least 1 gap in the root sequence stretch.

To calculate *S* for a leaf, its position scores are determined by the actual alignment sign as well as by the corresponding position score of the whole tree: the lowest of the two is always taken. That means that a leaf cannot improve its *S* by showing a high conservation in positions that are generally not or only moderately conserved in other leaves. Moreover, for a motif tree to persist, the average Viterbi score of the corresponding alignments and of each of its remaining leaves must be >=13.0 and >=11.0, respectively. These scores are calculated for the initial alignments and are not affected by the tree formation and trimming procedure, thus providing a means to incorporate conservation information from flanking amino acids in the overall motif tree evaluation pipeline.
To be processed further as a region of homology, an alignment hit must have a Viterbi score of at least 150.0 or cover at least 90% of the shorter sequence in the pair.

Regex generation and evaluation: The p-value of a regex with highly or moderately conserved columns C_{cons} and cumulative frequencies $freq_j$ of amino acids in the input dataset listed in the column j is calculated according to the formula:

$$p = \frac{\sum_{i=1}^{N_{corr}} \left(1 - \left(1 - \prod_{j=1}^{C_{cons}} freq_j\right)^M\right)}{N_{corr}}$$

This formula is the averaged version of the Šidák correction formula (1). N_{corr} is the effective number of non-homologous proteins for a given motif tree. The effective number of null hypotheses M (in the terms of Šidák) in this particular context is the number of all possible combinations to construct the given regex from the available sequence dimers and is calculated as the product of C_{cons}-1 cumulative counts of dimers D_j in all proteins participating in a regex along the regex:

$$M = \prod_{j=1}^{C_{cons}-1} D_j$$

The cumulative count D_j of dimers for the j-th conserved column is calculated as an average of dimer counts D_{ij} for each individual protein i that participates in the evaluated motif tree/regex. This average is further floored to ensure that the count remains integer. Each D_{ij} is the number of occurrences of a particular dimer (formed by j-th and (j+1)th columns of the regex) in a particular protein. The linker length (the number of wildcard positions between the columns) is also a part of a dimer. For example, if we have the dimer *[DE]..[ILV]*, we would be counting all occurrences of *DxxI, DxxL, DxxV, ExxI, ExxL*, and *ExxV* in the current protein sequence. Strictly speaking, for a pair of conserved columns j and $j+1$ separated by the linker of length m_j with $n(j)$ and $n(j+1)$ different amino acids encountered respectively in the regex

$$\ldots[X_{j,1}X_{j,2}\ldots X_{j,n(j)}].\{m_j\}[\ X_{j+1,1}X_{j+1,2}\ldots X_{j+1,n(j+1)}]\ldots$$

$D_{i,j}$ is the sum of counts $N(X_{j,1}.\{m_j\}X_{j+1,1})+N(X_{j,1}.\{m_j\}X_{j+1,2})+\ldots+N(X_{j,n(j)}.\{m_j\}X_{j+1,n(j+1)})$ in a given protein. D_j is then calculated as:

$$D_j = floor\left(\frac{\sum_{i=1}^{N_{tree}} D_{i,j}}{N_{tree}}\right)$$

N_{tree} is the number of proteins participating in the motif tree.

2. False positive evaluation model

One problem we had to address is the determination of N_{min}, i.e. the minimal number of proteins required to contain a motif. Providing a fixed number for N_{min} of at least 30% of submitted input sequences (rounded upwards, with a minimum of 3 sequences) turned out to produce too many false positive hits, which we learned by using negative data lacking a common motif, which we refer to as *negative datasets* (see columns 1-3 of Supplementary Table S9). To obtain these negative datasets, we have randomly selected N proteins that contain N different ELM motifs (therefore, each protein contains a distinct motif), constructing a negative dataset without any shared motif. All predictions returned by HH-MOTiF for these sets are consequently counted as false positives. For this and other tests, $FPR+Sp=1$, where Sp is the residue-wise specificity.

Next, we wanted to generate a mathematical link between a given N_{min} and its observed FPR. We assumed that we observe a false prediction, when several proteins just coincidentally have similar sequence stretches. Because observing a false positive is in this case a random event, it sounds reasonable that the more proteins are required to contain a motif, the lower the probability of this event will be (i.e., that all these proteins share a common SLiM-like sequence stretch). On the other hand, the more different ways (combinations) exist to select these proteins, the higher the probability that at least one combination will yield a false positive. Consequently selecting 3 proteins out of 10 leads to higher FPR values than selecting 3 proteins out of 3. The latter is, in fact, a multiple correction problem. Therefore, we can again use the Šidák correction formula:

$$FPR = 1 - (1 - X)^M$$

Here, X is the probability that a particular set of N_{min} proteins generates a false positive, and M is the evaluation for the number of ways to select N_{min} proteins out of N so that they all share the same false positive. Thus, M has posterior nature, i.e. it applies only when there is at least one set of N_{min} proteins that contains a false positive (and $M=0$ leads to $FPR=0$). Therefore, it can be calculated as the number of ways to substitute some of the N_{min} selected proteins with the equal number from the remaining N-N_{min} proteins; substituting 0 proteins also counts as a combination. If we assume that the probability of an average protein containing the same false positive is P, than we can calculate M as follows:

$$M = \sum_{k=0}^{\min(N_{min}, N-N_{min})} \frac{(N - N_{min})!}{k! * (N - N_{min} - k)!} * P^k$$

Given the way we defined P, it also can be used to evaluate X:

$$X = P^{N_{min}}$$

For a set size of 3, we select all the proteins, and there is only one way to do it, therefore we have M=1. Thus, for a set of 3 proteins:

$$FPR = 1 - (1 - X)^1 = X = P^{N_{min}} = P^3$$

As we observe an FPR=0.017 for a set of three proteins, we can estimate $P = 0.017^{1/3} = 0.26$.

As can be seen in Supplementary Table S9 (column 4), this model explains the observed FPR reasonably well. Therefore, we decided to use this formula for calculating N_{min} on the basis of a given N. In practice, we calculate the predicted FPR for incrementally increasing N_{min}, starting from 3 until the FPR exceeds 1%. Then we take the maximal N_{min}, for which the FPR is still below 1%. The only exception is the set size 4, for which the modeled $N_{min}=4$ would be too strict, suppressing virtually all predictions.

The described model will not be able to truly capture the complexity of false positive predictions in a SLiM search. For instance, we assume with this model that the FPR in a dataset is caused by a single false positive signal/motif, which might be reasonable for the majority of datasets, however must not be true for all cases. These calculations also require knowledge on the parameter *P*, which is estimated from the real observed FPR of a sufficiently large number of random 3-sets, and thus must be adjusted for newly implemented features in HH-MOTiF. In future releases of HH-MOTiF, *P* could also be dynamically assessed.

3. Reciprocal BLAST searches

HH-MOTiF includes reciprocal BLASTs. In the initial BLAST, each input query is identified in the NCBI nr-database. To be recognized, the input query must be identical in sequence and length to a top BLAST hit. Duplicated records of the same species are recorded, as are all possible isoforms of a protein. Sequences marked as partial, synthetic or originating from crossed organisms are excluded. HH-MOTiF assumes that the input queries are in the database (nr or RefSeq, if the input is found in RefSeq in the first BLAST search). Next, HH-MOTiF performs a reciprocal BLAST search against the original species for each hit that is above the threshold (minimum: 70% sequence identity; maximum: 95% sequence identity – the latter filters out too closely related orthologs). Only best-best relationships are considered orthologous.

4. Web-server implementation

The web-server is implemented in Python3 and Django web framework under Apache2. Results are presented as dynamic FASTA in the output page, containing the full-length sequences with highlights on the identified motifs. JavaScript with Bootstrap, JQuery and CSS3 are used to generate pop-up boxes with meta-information on motifs, as well as effects upon clicking and scrolling. WebLogo (2) is used to generate sequence logos of the motifs.

5. Tool optimization and comparison

Dataset. The performance of all tools was tested on the motifs from the ELM database ((3); as of 26.03.2016; the annotation file is available for download on the Tests page of the HH-MOTiF web-server), which occur in at least three proteins. There are a total of 176 motifs in 1,677 unique proteins, or 2,022 proteins gross, when counted separately for different motifs. There are a total of 1,452,618 residues gross, of which 17,909 are annotated as belonging to motifs – they make up the positive set. The remaining 1,434,709 residues are considered negative.

Performance measures. We compute the following performance measures: balanced F1-score (F1), which is the harmonic mean of recall and precision, performance coefficient (PC), and balanced accuracy (BA). As a proxy for a motif we define a ‘site’ as follows: a site is a continuous sequence stretch in a particular protein that corresponds either to a separate ELM instance or a predicted motif in the output of a tool. The performance measures are calculated by quite simple formulas on the basis of TP (true positives), FP (false positives), etc. However, as both annotated and predicted motif sites often overlap and duplicate, the terms 'true positive', etc. themselves become ambiguous. Therefore, we perform the calculations in three consecutive steps.

First, we calculate the following counts:

- $TP_{res.}$ (true positive residues): number of unique residues that belong both to at least one predicted and at least one annotated site
- $PA_{res.}$ (predicted and annotated residues): gross number of residues in the predicted sites that belong to at least one annotated site

- $PNA_{res.}$ (predicted and not annotated residues): gross number of residues in the predicted sites that do not belong to any annotated site
- $FP_{res.}$ (false positive residues): number of unique residues in the predicted sites that do not belong to any annotated site
- $FN_{res.}$ (false negative residues): number of unique residues in the annotated sites that do not belong to any predicted site
- $TN_{res.}$ (true negative residues): number of unique residues outside the annotated sites that do not belong to any predicted site
- AP_{site} (annotated and predicted sites): gross number of the annotated sites that have at least one common residue with at least one of the predicted sites
- PA_{site} (predicted and annotated sites): gross number of the predicted sites that have at least one common residue with at least one of the annotated sites
- PNA_{site} (false positive sites): gross number of the predicted sites that do not have common residues with any of the annotated sites
- ANP_{site} (annotated and not predicted sites): gross number of the annotated sites that do not have common residues with any of the predicted sites

Gross numbers imply that all the duplicates are counted separately. These counts are calculated for each sequence separately and then summed up for the whole protein set.

Second, we calculate for each motif separately the performance measures recall (*Rc*; a.k.a. sensitivity), precision (*Pr*), specificity (*Sp*), false positive rate (*FPR*), and PC. The formulas are as follows:

$$Rc_{res.} = \frac{TP_{res.}}{TP_{res.} + FN_{res.}}$$

$$Rc_{site} = \frac{AP_{site}}{AP_{site} + ANP_{site}}$$

$$Pr_{res.} = \frac{PA_{res.}}{PA_{res.} + PNA_{res.}}$$

$$Pr_{site} = \frac{PA_{site}}{PA_{site} + PNA_{site}}$$

$$Sp_{res.} = \frac{TN_{res.}}{TN_{res.} + FP_{res.}}$$

$$FPR_{res.} = \frac{FP_{res.}}{TN_{res.} + FP_{res.}}$$

$$PC_{res.} = \frac{PA_{res.}}{PA_{res.} + PNA_{res.} + FN_{res.}}$$

$$PC_{site} = \frac{PA_{site}}{PA_{site} + PNA_{site} + ANP_{site}}$$

These measures are calculated for each protein set (ELM class) independently. The values are available in Supplementary Tables S3-S6. A simple averaging is done over all the sets tested (for averages see Supplementary Table S2). All the sets have equal weights, regardless of number of instances and motif length; `nan` values are excluded from averaging.

Finally, we calculate F1 and BA on the basis of the corresponding averaged values according to the formulas:

$$F1 = \frac{2 * Pr * Rc}{Pr + Rc}$$

$$BA = 0.5 * (Rc + Sp)$$

Note that we calculate FPR, specificity and consequently BA only residue-wise.

Weighting performance measures with number of residues, number of sites and number of proteins. To calculate protein-, site-, and residue-weighted performance, we used the formula for the weighted average:

$$\bar{x} = \frac{\sum_{i=1}^{R} w_i x_i}{\sum_{i=1}^{R} w_i}$$

The averaging is done across all R non-`nan` values x_i (for averaging recall and specificity, as they are never `nan`, R equals the total number of data sets) with corresponding weights w_i taken from the Supplementary Table S10.

Optimization of HH-MOTiF. The optimization, which included development of concepts, debugging, and finally optimization of the input parameters was carried out on the limited training set consisting of the two ELM types CLV and DEG, with a total of 23 motifs. The goal was to maximize residue-wise F1. The test set consisted of the four remaining types (DOC, LIG, MOD, and TRG; total 153 motifs). The performance details can be seen in the Supplementary Table S2.

Testing HH-MOTiF on negative datasets. As the tests on ELM datasets always imply the presence of motifs, they cannot be used to estimate the performance of a method in case of the absence of any shared motif. Ideally, if no motifs are present, or they occur in an insufficient number of proteins, the desired behavior for a predictor would be to return no results. We have tested HH-MOTiF for its performance with negative datasets. To this end, for each set size N, we randomly selected 1 protein from N randomly selected, different ELM classes, prohibiting selection of one protein twice. In this test, we do not expect any shared motifs and therefore count all predictions as false positives. We repeated the procedure 100 times for each set size and conducted 2 independent runs with 100 repetitions for each set size. In Supplementary Table S1, we report averaged residue-wise FPRs (false positive rates) for each set size and the 2 independent runs. The data show that HH-MOTiF has a FPR < 1% for all set sizes except for the set size 4. In this set size, the minimal requirement of at least 3 proteins participating in a motif seems loose, but the only alternative requirement of the occurrence of a motif in all 4 proteins would suppress virtually all predictions.
When testing HH-MOTiF on the ELM dataset, it showed a residue-wise specificity of 99.3% (see Supplementary Tables S2, S3) or an FPR of 0.7%, which is – given the ELM dataset size distribution – somewhat higher than one could expect from our test on negative datasets. This can be explained by the fact that true positive motifs are usually predicted with some flanking residues in HH-MOTiF, which exceed the ELM annotations and therefore contribute to the FPR. This does not hold true for truly negative, random datasets. It should also be noted that a FPR of 0.7% might look like a very good number. In fact, it is quite high for the given task, as motifs in ELM occupy on average only 1.2% of the sequence. Thus, an FPR of 0.7% translates in a ~40% residue-wise precision (see Supplementary Tables S2, S3).

Performance comparison. The performance of the following three other tools was measured: MEME v. 4.0 (4), GLAM2 v. 4.11.1 (5), and SLiMFinder v. 5.2.3 (6). Several parameter combinations were tried for each tool, although no training procedure was done and tests were carried out on the full ELM dataset. Only the best combination, the one with the highest residue-wise F1, was reported for each tool in the main text. Intriguingly, our parameter choice worked substantially better than default settings for MEME and GLAM2. This is due to the fact that MEME by default reports only the best motif, while much more often, the right motif has rank 2 to 5. Both tools tend to output by default very long motifs, typically exceeding ELM

annotations to quite some extend. Our final combination for MEME is achieved by using the additional options '`-nmotifs 5 -minw 3 -maxw 15`'; for GLAM2 the options are '`-a 3 b 15`', which means that the output motif length in residues for both tools is kept in the range [3,15]. The final parameter settings we used for SLiMFinder were the following: '`dismask=T consmask=T probcut=1.0 topranks=5`'. These settings outperformed default settings in F1, at the cost of a significantly lower precision. Performance values on different settings of SLiMFinder and all other tools tested can be found in Supplementary Table S2; our observed results demonstrate the influence of different parameter settings on the performance measures recall, precision, as well as F1. Please note that as a user, one might prefer either higher recall or higher precision and therefore choose settings that favor one over the other.

Performance comparison to whmm. We want to note that a fair comparison to the tool whmm (7) cannot be provided, as the version of the ELM database has meanwhile changed and it is not clearly stated in the manuscript by (7), how overlapping residues are treated in the calculations of performance measures. Whenever comparing, we therefore refer to the original data published in (7), which suggest that whmm has a high site-based recall (0.615), however a very low precision (0.022). Residue-based recall is considerably lower (0.142), residue-based precision is as low as 0.026. Given these values, we assume that HH-MOTiF also outperforms whmm in F1 (see Supplementary Table S2).

6. Impact of tunable parameters on the performance of HH-MOTiF.

HH-MOTif is a quite complex tool and has a number of tunable parameters. We tested the robustness of HH-MOTiF to changes in settings. To this end, we plotted dependencies of residue-wise TPR (true positive rate; a.k.a. recall), FPR, F1 and PC (performance coefficient) on six parameters (see Supplementary Figure S2). General trends turned out to be as expected: more stringent values led to a drop in both, TPR and FPR. For most parameters, the performance remained stable for broad ranges of values. Thresholds and ranking filters in HH-MOTiF can explain the sometimes-observed slight increase of FPR and/or TPR.

From Supplementary Figure S2, it also becomes clear that the reported configuration of HH-MOTiF is not the optimal one. The performance could be improved, for example, by setting the number of hits to 9 instead of 4, or the minimal Viterbi score to 8.5 instead of 11.0. Eliminating surface accessibility prediction does also lead to an increased performance, however at the cost of a decreased precision. In addition, we tried to eliminate two further filters, namely the requirement of the average Viterbi score for a motif tree >=13.0 and the minimum number of positions, a motif leaf must span (>=4). This led again to a slightly better performance, at the cost of precision (see the 'simplified' configuration in the Supplementary Table S2). We also observed an increased F1 and PC when disabling homology filtering, due to a substantial rise in recall, but a drastic drop in precision (configuration 'disabled homology filters' in Supplementary Table S2). These facts can be explained by the following:

- Optimization of HH-MOTiF was conducted on a subset of ELM, and therefore it is not optimized for the whole ELM database.
- Only few values (in some cases only a single value) and not the complete ranges were tested during the optimization

However, we believe that our optimization approach is reasonable, and therefore we keep our original configuration. A more comprehensive optimization could yield an over-optimized configuration; consequently, the observed performance would not be sustainable for new datasets or for small changes in the configuration.

7. Real-case example (TRG_LysEnd_APsAcLL_3) of a motif search using the HH-MOTiF workflow

Some principal rules apply to all motif trees:
1. Motif roots are in specific proteins; different motif roots within one protein cannot overlap.
2. Motif instances from different motif trees (roots or leaves) can overlap. This means that a sequence stretch that is a leaf of a motif tree can at the same time be the root for another motif tree.

HH-MOTiF input sequences get number-coded with 4-digit sequences starting from '0000'. If a FASTA file is submitted, the order of sequences is preserved. If a ZIP archive is submitted, the alphabetic order of filenames is used to assign the numbers.

The workflow of HH-MOTiF begins with an orthology search using NCBI BLAST (8), HMM generation with *hhmake* (9), and surface accessibility prediction with NetSurfP (10). The raw masking with NetSurfP for this dataset is as follows (lowercase gray residues are masked, the sought-for motif is underlined):

```
>0000 (P11344|TYRO_MOUSE)
MFLAVlYcllWsfQIsDgHfpRAcASSKNLLAKEccpPWMGDGSPcGQLSGRgScQDILlSSAPSGPQfPFKGVDDRESwpSVfyNRTc
QcSGNfMgfNcGNcKFGFGGPNcTEKRVLirRNiFDlSVSekNKffSylTLaKHTISSVYVIPTGTYGQMNNGSTPMFNDINIyDLfvw
mhyyvsRDTLLGGSEIWRDIDfaheApgflPwhrlfllLweQeiRELtGDENfTvpywdwRDAENCDICTDEYLGGRHPENPNLLSPAS
FfSSWQIICSRSEEYNSHQVLCDGTPEGPlLRNpGNHDKAKTPRLPSSADvEFclSLTQyESGSMDRTANFSfRNTLEGfASPLTGIAD
PSQSsmhnalhifmngtmsqVQgsaNdpifllhhafvdSifEQwLRRHRPLLEVYPEANAPIGHNRDSYmVPfIPLYRNGDFfITSKDl
GyDySYLQESDPGFYRNYIEPYLEQASRIWPWLlgAALVGAVIAAALSGLsSRLCLQKKKKKKQPQEERQPlLMDKDDYHSLLYQSHL
>0001 (Q14108|SCRB2_HUMAN)
MGRCCFYTAGTLSLLLLVTSVTLLVARVfQKavDQsiEKKiVlRNGTEAfDSwEKPPLPvYtqfyffnvTnpEEiLRGETpRvEEVgPy
tyRELRNKANiQfGDNGTTiSaVSNKAyVfERDQsVGDPKIdLirtlnipvlTviEWSQVHflREiiEamlKAYQQKlfVTHTvDEllW
GyKDEilSLiHVFRPDISPYfglfyEKnGtNDGDyVfLtGEDSYLNftKiVEwNGKTSLDWwITDKcNMiNgTDGDSfHPLITKDEVlY
vfPSDfcRSvYiTfSDYESVQGlPafRyKvPAEIlANTSDNAgfcIPEGNcLGSgvlnvSIcKNGapiimsfphFYQaDERfVSAiEGM
HPNQEDhEtfvdinpLTGIIlKAAKRfqiniyvKKLDDfVEtGDiRTmvfpvmylNESVHiDKEtaSRlKSmiNtTLiiTNIPYIIMAL
GVFFGLVFtwlacKGQGSMDEGTADERAPLIRT
>0002 (Q90372|QNR71_COTJA)
MSQAHRhlalllpaeAvlcAAAMRFQDvLSNGRTAPVTNHKKIQGWSSDQNKWNEKLyPFwEDNDPRWKDcwKGGKVTTKlVTDSPAlV
GSNvTfVvTlQfPKCQKEDDDGNIIyQRNcTPDSPAAQDQYVYNWTEWIDNCGWENCTSNHSHNvfPDGKPfPHYPGWRRRnfvylfHt
vGQyyQTIGRSSaNFSVNTANITLGKhImAvsiYrRGHSTYVPIARASTTYVVTDKiPILvSMSQKHDRNISDSIFIKDSPiTfDvKiH
DPSYYLNDSAISyKwNfGDGSGLFVESGATtShTfSLQGNfTlNlTvQAIIPVPCKPVTPTPSLPTPAVTTDASSNSDPSAPNEMAEDN
PDGGcHIYRYGYYTAGiTiVEGILEVNIIQMTSiQMTESQAENPLvDfVvTcQGSFPTDvctAvSDPTCQVSQGMVCDPVVVTDECVLT
IRrAfDEPgTycinitlGDDTSQALASALiSvNGGSSSGTTKgvfifLgllAvfgaigafvlyKRYKQYKPIERSAGQAENQEGLsaYv
SNFKafFFPKSTERNPLlKSKPGIV
```

If the user has chosen several types of masking (e.g., for both buried and ordered regions), they are getting merged at this step with the OR logic: if a residue is masked in at least one masking program, it is processed further as masked.

The next step is the consolidation of the mask (algorithm description is in **1. Additional workflow details**). This step yields the following results (lowercase residues are finally masked; red residues changed their masking status in the course of this step):

```
>0000 (P11344|TYRO_MOUSE)
MFLAVLYcllWSFQISDGHFPRACASSKNLLAKEccpPWMGDGSPCGQLSGRgscQDILLSSAPSGPQFPFKGVDDRESwpsvfyNRTc
qcSGNfmgfncgncKFGFGGPNCTEKRVLIRRNIFDLSVSEKNKFFSYLTLAKHTISSVYVIPTGTYGQMNNGSTPMFNDINIYDLfvw
mhyyvsRDTLLGGSEIWRDIDfaheapgflpwhrlfllweqeiRELTGDENftvpywdwRDAENCDICTDEYLGGRHPENPNLLSPAS
FFSSWQIICSRSEEYNSHQVLCDGTPEGPLLRNPGNHDKAKTPRLPSSADvefclSLTQYESGSMDRTANFSFRNTLEGFASPLTGIAD
PSQSsmhnalhifmngtmsqvqgsandpifllhhafvdSIFEQWLRRHRPLLEVYPEANAPIGHNRDSYNVPFIPLYRNGDFFITSKDl
gydySYLQESDPGFYRNYIEPYLEQASRIWPWLLGAALVGAVIAAALSGLSSRLCLQKKKKKKQPQEERQPLLMDKDDYHSLLYQSHL
>0001 (Q14108|SCRB2_HUMAN)
MGRCCFYTAGTLSLLLLVTSVTLLVARVfqkavdqsiEKKivlRNGTEAFDSWEKPPLPVYtqfyffnvTNPEEILRGETprvEEVGPy
tyRELRNKANiqfGDNGTTisaVSNKAyvfERDQSVGDPKIDLirtlnipvltviEWSQVHflreiieamlKAYQQKLFVTHTvdellW
GYKDEilsliHVFRPDISPYfglfyEKNGTNDGDyvfltGEDSYLNftkivewNGKTSLDWWITDKCNMingTDGDSFHPLITKDEVly
vfPSDfcrsvyitfSDYESVQGlpafrykvPAEILANTSDNAgfcIPEGNCLGSgvlnvSICKNGapiimsfphFYQADERFVSAIEGM
HPNQEDHEtfvdinpLTGIILKAAKRfqiniyvKKLDDFVETGDIRTmvfpvmylNESVHIDKEtasrlksmintt liiTNIPYIIMAL
GVFFGLVFtwlacKGQGSMDEGTADERAPLIRT
>0002 (Q90372|QNR71_COTJA)
MSQAHRhlalllpaeavlcAAAMRFQDVLSNGRTAPVTNHKKIQGWSSDQNKWNEKLYPFWEDNDPRWKDCWKGGKVTTKLVTDSPALV
GSNvtfvvtlqfPKCQKEDDDGNIIYQRNCTPDSPAAQDQYVYNWTEWIDNCGWENCTSNHSHNVFPDGKPFPHYPGWRRRnfvylfht
vgqyyQTIGRSSANFSVNTANITLGKHIMAvsiYRRGHSTYVPIARASTTYVVTDKIPILVSMSQKHDRNISDSIFIKDSPitfdvkiH
DPSYYLNDSAISykwnfGDGSGLFVESGATtshtfSLQGNftlnltvQAIIPVPCKPVTPTPSLPTPAVTTDASSNSDPSAPNEMAEDN
```

```
PDGGCHIYRYGYYTAGitiVEGILEVNIIQMTSIQMTESQAENPLvdfvvtcQGSFPTDvctAVSDPTCQVSQGMVCDPVVVTDECVLT
IRrafDEPGTycinitlGDDTSQALASALisvNGGSSSGTTKgvfiflgllavfgaigafvlyKRYKQYKPIERSAGQAENQEGLsayv
SNFKAFFFPKSTERNPLLKSKPGIV
```

After the HMMs are generated and masking is finished, the main step (pairwise HMM-HMM comparison) starts. In this example, there are 3 sequences, which form 3 pairs. The comparison of the two first sequences, for example, is preformed through the following system call:

```
hhalign –i 0000.hhm -t 0001.hhm -o 0000_0001.hhr -smin 0 -alt 100 -gaph 999.0 -gapi 999.0
-norealign -nocontxt -excl 8-10,35-37,53-55,80-85,89-91,95-103,176-184,200-222,233-
238,318-322,361-394,445-449 -template_excl 29-37,41-43,62-69,81-83,89-91,100-102,109-
111,117-119,133-144,151-160,184-188,199-203,213-217,225-231,266-269,273-281,290-297,310-
312,322-326,333-341,365-371,383-389,404-411,421-435,454-458
```

This call writes the output file, from which the four best alignment hits under the described constraints (Viterbi score is >=11.0 and <=40.0, number of aligned columns is >=3 and <=30) are selected. In *hhalign*, 'Q' means 'query' and corresponds to the HMM submitted with the flag '`-i`', and with masking submitted with the flag '`-excl`'; while 'T' means 'template' and corresponds to the HMM submitted with the flag '`-t`', and with the masking submitted with the flag '`-template_excl`'; however, this difference is not important, as swapping of the files does not influence the alignments. The middle row of the alignments shown corresponds to the 'alignment signs', which are used to assess alignment quality at a later stage of the motif tree evaluation. For the first two sequences the alignment hits are following:

```
Probab=0.08  E-value=0.66  Score=13.74  Aligned_cols=4  Identities=100%  Similarity=1.481
Sum_probs=0.0  Template_Neff=1.400
Q 0000            330 GSMD  333 (533)
Q Consensus       330 gsMD  333 (533)
                      ||||
T Consensus       462 gsmd  465 (478)
T 0001            462 GSMD  465 (478)

Probab=0.08  E-value=0.66  Score=13.73  Aligned_cols=11  Identities=36%  Similarity=0.767
Sum_probs=0.0  Template_Neff=1.400
Q 0000            508 KQPQEERQPLL  518 (533)
Q Consensus       508 k~~~eE~qpLl  518 (533)
                      .++..|+.||+
T Consensus       466 egtaderapli  476 (478)
T 0001            466 EGTADERAPLI  476 (478)

Probab=0.06  E-value=0.78  Score=12.75  Aligned_cols=6  Identities=50%  Similarity=1.093
Sum_probs=0.0  Template_Neff=1.400
Q 0000            290 DGTPEG  295 (533)
Q Consensus       290 ngt~EG  295 (533)
                      |||..|
T Consensus       206 ngtndg  211 (478)
T 0001            206 NGTNDG  211 (478)

Probab=0.05  E-value=0.84  Score=12.18  Aligned_cols=5  Identities=60%  Similarity=1.703
Sum_probs=0.0  Template_Neff=1.400
Q 0000            269 FSSWQ  273 (533)
Q Consensus       269 FsSWQ  273 (533)
                      |.||.
T Consensus        50 fdsw~   54 (478)
T 0001             50 FDSWE   54 (478)
```

Similarly, for two other pairs we get the following hits:

```
Probab=0.11  E-value=0.51  Score=14.92  Aligned_cols=8  Identities=50%  Similarity=1.107
Sum_probs=0.0  Template_Neff=1.400
Q 0002            547 ERNPLLKS  554 (559)
Q Consensus       547 ernpllks  554 (559)
                      ||.||++.
T Consensus       471 eraplirt  478 (478)
T 0001            471 ERAPLIRT  478 (478)

Probab=0.06  E-value=0.77  Score=12.71  Aligned_cols=13  Identities=46%  Similarity=0.516
Sum_probs=0.0  Template_Neff=1.400
Q 0002             55 EKLYPFWEDNDPR   67 (559)
```

```
Q Consensus        55 eklypfweegdpr   67 (559)
                      .||--|-|.|+-|
T Consensus       390 ~klddf~etgnir  402 (478)
T 0001            390 KKLDDFVETGDIR  402 (478)

Probab=0.06  E-value=0.78  Score=12.62  Aligned_cols=5  Identities=60%  Similarity=1.916
Sum_probs=0.0  Template_Neff=1.400
Q 0002            357 PDGGC  361 (559)
Q Consensus       357 pdggc  361 (559)
                      |.|.|
T Consensus       314 p~gnc  318 (478)
T 0001            314 PEGNC  318 (478)

Probab=0.05  E-value=0.83  Score=12.20  Aligned_cols=4  Identities=75%  Similarity=1.655
Sum_probs=0.0  Template_Neff=1.400
Q 0002            439 TDEC  442 (559)
Q Consensus       439 tdec  442 (559)
                      ||+|
T Consensus       242 td~c  245 (478)
T 0001            242 TDKC  245 (478)

Probab=0.15  E-value=0.38  Score=16.41  Aligned_cols=11  Identities=55%  Similarity=0.818
Sum_probs=0.0  Template_Neff=1.500
Q 0002            545 STERNPLLKSK  555 (559)
Q Consensus       545 sternpllksk  555 (559)
                      +.|+.|||-.|
T Consensus       511 ~eE~qpLlmek  521 (533)
T 0000            511 QEERQPLLMDK  521 (533)

Probab=0.12  E-value=0.47  Score=15.61  Aligned_cols=5  Identities=60%  Similarity=1.365
Sum_probs=0.0  Template_Neff=1.500
Q 0002            363 IYRYG  367 (559)
Q Consensus       363 iyrng  367 (559)
                      .||||
T Consensus       432 lyrng  436 (533)
T 0000            432 LYRNG  436 (533)

Probab=0.11  E-value=0.5  Score=15.33  Aligned_cols=8  Identities=63%  Similarity=1.074
Sum_probs=0.0  Template_Neff=1.500
Q 0002            328 TPSLPTPA  335 (559)
Q Consensus       328 tap~ptsa  335 (559)
                      |+-||.||
T Consensus       309 TpRLPsSa  316 (533)
T 0000            309 TPRLPSSA  316 (533)

Probab=0.10  E-value=0.54  Score=14.91  Aligned_cols=9  Identities=44%  Similarity=0.671
Sum_probs=0.0  Template_Neff=1.500
Q 0002            198 ANITLGKHI  206 (559)
Q Consensus       198 anitlgkh~  206 (559)
                      |.++|.||.
T Consensus       136 AYL~LaK~t  144 (533)
T 0000            136 SYLTLAKHT  144 (533)
```

These pairwise alignment hits are further mapped to the sequences. Shown below is the result for the first sequence (orange: alignment hits with the second sequence; lime green: alignment hits with the third sequence, blue: alignment hits with both sequences; lowercase gray residues are masked, the sought-for motif is underlined):

```
>0000 (P11344|TYRO_MOUSE)
MFLAVLYcllWSFQISDGHFPRACASSKNLLAKEccpPWMGDGSPCGQLSGRgscQDILLSSAPSGPQFPFKGVDDRESwpsvfyNRTc
qcSGNfmgfncgncKFGFGGPNCTEKRVLIRRNIFDLSVSEKNKFFSYLTLAKHTISSVYVIPTGTYGQMNNGSTPMFNDINIYDLfvw
mhyyvsRDTLLGGSEIWRDIDfaheapgflpwhrlflllweqeiRELTGDENftvpywdwRDAENCDICTDEYLGGRHPENPNLLSPAS
FFSSWQIICSRSEEYNSHQVLCDGTPEGPLLRNPGNHDKAKTPRLPSSADvefclSLTQYESGSMDRTANFSFRNTLEGFASPLTGIAD
PSQSsmhnalhifmngtmsqvqgsandpifllhhafvdSIFEQWLRRHRPLLEVYPEANAPIGHNRDSYNVPFIPLYRNGDFFITSKDl
gydySYLQESDPGFYRNYIEPYLEQASRIWPWLLGAALVGAVIAAALSGLSSRLCLQKKKKKQPQEERQPLLMDKDDYHSLLYQSHL
```

For a set of 3 proteins, a motif is required to be in all 3 sequences (N_{min}=3). Thus, only sequence stretches having an alignment to N_{min}-1=2 other proteins will be further considered as motif roots. The only such stretch is '511 QEERQPLL 518'. It satisfies the requirement of being at least 3 residues long, and therefore the corresponding motif tree is formed:

```
0000   511 QEERQPLL 518  <- root (average score: 15.07)
0001   469 ADERAPLI 476  <- leaf (alignment score: 13.73)
```

```
0002   545 STERNPLL 552  <- leaf (alignment score: 16.41)
```

As the average alignment (Viterbi) score is no less than 13, the tree is retained and proceeds to the next step, at which the corresponding alignment signs are assessed (for details on the algorithm, see main text, as well as this supplement):

```
0000-0001   511/469 ..|+.||+ 518/476
0000-0002   511/545 +.|+.||| 518/552
column scores:      11221222
```

The sum of the column scores is 13, for the whole motif tree, as well as for each of the two leaves. This exceeds the minimal requirement of 6, and thus the tree gets finally validated and is reported to the user. Similarly, two more trees with roots in each of the other proteins corresponding to this motif are identified with the standard settings of HH-MOTiF.

8. Prediction results for the dataset TRG_LysEnd_APsAcLL_3 by the compared tool

original motif
true positive predictions
false positive predictions

ELM annotation:

```
Q90372|QNR71_COTJA          546 TERNPLL 552
Q14108|SCRB2_HUMAN          470 DERAPLI 476
P11344|TYRO_MOUSE           512 EERQPLL 518
```

HH-MOTiF:

```
Q90372|QNR71_COTJA          547 ERNPLLKS 554
Q14108|SCRB2_HUMAN          471 ERAPLIRT 478
P11344|TYRO_MOUSE           513 ERQPLLMD 520

Q90372|QNR71_COTJA          547 ERNPLL 552
Q14108|SCRB2_HUMAN          471 ERAPLI 476
P11344|TYRO_MOUSE           513 ERQPLL 518

Q90372|QNR71_COTJA          545 STERNPLL 552
Q14108|SCRB2_HUMAN          469 ADERAPLI 476
P11344|TYRO_MOUSE           511 QEERQPLL 518
```

MEME:

```
Q90372|QNR71_COTJA           21 AAMRFQ 26
Q14108|SCRB2_HUMAN          379 AAKRFQ 384

Q90372|QNR71_COTJA          167 WRRRN 171
P11344|TYRO_MOUSE           400 WLRRH 404

Q90372|QNR71_COTJA          509 KRYKQ 513
Q14108|SCRB2_HUMAN          161 KAYQQ 165
P11344|TYRO_MOUSE           505 KKKKQ 509

Q90372|QNR71_COTJA          115 YQRNC 119
```

```
Q14108|SCRB2_HUMAN              1 MGRCC 5
P11344|TYRO_MOUSE              85 YNRTC 89

Q90372|QNR71_COTJA              1 MSQAH 5
Q14108|SCRB2_HUMAN            146 WSQVH 150
P11344|TYRO_MOUSE             374 MSQVQ 378
```

GLAM2:

```
Q90372|QNR71_COTJA            547 ERNPLL 552
Q14108|SCRB2_HUMAN            471 ERAPLI 476
P11344|TYRO_MOUSE             513 ERQPLL 518

Q90372|QNR71_COTJA            546 TERNPLL 552
Q14108|SCRB2_HUMAN            470 DERAPLI 476
P11344|TYRO_MOUSE             512 EERQPLL 518

Q90372|QNR71_COTJA            547 ERNPLLKS 554
Q14108|SCRB2_HUMAN            471 ERAPLIRT 478
P11344|TYRO_MOUSE             513 ERQPLLMD 520

Q90372|QNR71_COTJA            196 NTANITLG 203
Q14108|SCRB2_HUMAN             96 NKANIQFG 103
P11344|TYRO_MOUSE              99 NCGNCKFG 106
```

SLiMFinder:

```
Q90372|QNR71_COTJA            547 ERNPL 551
Q14108|SCRB2_HUMAN            471 ERAPL 475
P11344|TYRO_MOUSE             513 ERQPL 517

Q90372|QNR71_COTJA            547 ERNPLL 552
Q14108|SCRB2_HUMAN            471 ERAPL- 475
P11344|TYRO_MOUSE             513 ERQPLL 518

Q90372|QNR71_COTJA            548 RNPL 551
Q14108|SCRB2_HUMAN            472 RAPL 475
P11344|TYRO_MOUSE             514 RQPL 517

Q90372|QNR71_COTJA            547 ERNP 550
Q14108|SCRB2_HUMAN            471 ERAP 474
P11344|TYRO_MOUSE             513 ERQP 516
```

9. Workflow of the HH-MOTiF proteome-wide search.

The proteome-wide search implements a much simpler workflow and is designed as an auxiliary tool to the *de novo* SLiM search.

The input of the proteome-wide search is an aligned FASTA file with all known instances of a motif. Only the motif itself and not the whole protein sequences should be included. The downloadable FASTA files from the results page of a HH-MOTiF *de novo* search are already in the correct format and can be used directly for a proteome-wide search.

The workflow begins by building an HMM from the input FASTA file using *hhmake*, with the option '`-M first`'. The HHM is further compared with our precompiled proteome databases (in form of concatenated HHMs) using *hhsearch* with the option '`-norealign`' (which prevents too rigorous output alignment checks for this task). The output is parsed, and the best hit for each sequence, if it has the Viterbi score of at

least 10.0, is retained. If the user activates the option 'full length only', the candidate hits will be further checked for the sufficient number of columns: it must match the number of columns in the input motif (i.e., number of the non-gap residues in the submitted FASTA record). The hits are reported to the user together with the corresponding full protein sequences and motif alignments.

10. Supplementary Tables

Supplementary Table S1: FPR values determined by testing HH-MOTiF against negative datasets containing only sequences without a shared SLiM.
Supplementary Table S2: Averaged performance measures for the tested methods.
Supplementary Table S3: Performance measures of HH-Motif for all 176 ELM-motifs selected for testing.
Supplementary Table S4: Performance measures of MEME for all 176 ELM-motifs selected for testing.
Supplementary Table S5: Performance measures of GLAM2 for all 176 ELM-motifs selected for testing.
Supplementary Table S6: Performance measures of SLiMFinder for all 176 ELM-motifs selected for testing.
Supplementary Table S7: Dependency on set sizes of tested methods. Performance measure: residue-wise, balanced F1-score.
Supplementary Table S8: weighted performance measures and differences to averaged measures for all tested methods. Weights are taken from Supplementary Table S10.
Supplementary Table S9: FPR values of HH-MOTiF searches for different negative test set sizes with a fixed N_{min}. Based on these results, calculating a dynamic N_{min} has been implemented.
Supplementary Table S10: Weights used for calculating weighted performance measures.

11. Supplementary Figure Legends

Supplementary Figure S1: Input pages of HH-MOTiF. (**A**) ***standard mode*** of a HH-MOTiF. A FASTA-formatted file with a minimum of three input queries has to be submitted. (**B**) ***advanced mode*** of an HH-MOTiF search. A .zip-archive of FASTA-formatted protein sequences is required as an input, which contains either a single FASTA-sequence, or a FASTA-formatted collection of orthologs of an input sequence. Again, a minimum of three input queries is required. A regions file containing information on the putative localization of a sought-after SLiM can be provided optionally and will enhance the chance of finding a functional SLiM. In case no orthologs are provided, the 'Search for orthologs' option should be activated. Additional optional parameters include the restriction of the gap size allowed within a motif (default: 1); the checks for surface accessibility and disordered regions; the maximal allowed regex p-value for reporting a SLiM (default: 0.3); the smart homology filtering. (**C**) Input mask for the ***proteome-wide motif search***. A FASTA-formatted set of short sequences representing a functional motif is required as an input. The species of the proteome to be searched has to be selected. As an optional parameter, search results can be restricted to full matches of the input motif.
In all three cases, an e-mail address can be optionally provided, which is highly recommended for the ***advanced mode***, as well as the proteome-wide searches, as both can take a long time (hours).

Supplementary Figure S2: Parameter-based performance plots for FPR, TPR, F1 and PC. We tested minimal and maximal Viterbi scores, the minimal length of a motif root, the maximal regex p-value, the number of hits considered for motif tree generation, as well as the RSA threshold given by NetSurfP. Scales on the left-hand side of the plots (from 0.00 to 0.30) relate to scales for TPR, F1 and PC; scales on the right-hand side of the plots are relevant for the FPR. Dashed vertical lines implicate default settings in HH-MOTiF.

12. Supplementary References

1. Wright, S.P. (1992) Adjusted P-Values for Simultaneous Inference. *Biometrics*, **48**, 1005-1013.
http://dx.doi.org/10.2307/2532694
http://www.ncbi.nlm.nih.gov/pmc/articles/Full publication date: Dec., 1992

2. Crooks, G.E., Hon, G., Chandonia, J.M. and Brenner, S.E. (2004) WebLogo: a sequence logo generator. *Genome Res*, **14**, 1188-1190.
http://www.ncbi.nlm.nih.gov/pubmed/15173120
http://dx.doi.org/10.1101/gr.849004

3. Dinkel, H., Michael, S., Weatheritt, R.J., Davey, N.E., Van Roey, K., Altenberg, B., Toedt, G., Uyar, B., Seiler, M., Budd, A. *et al*. (2012) ELM--the database of eukaryotic linear motifs. *Nucleic acids research*, **40**, D242-251.
http://www.ncbi.nlm.nih.gov/pubmed/22110040
http://dx.doi.org/10.1093/nar/gkr1064
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3245074

4. Bailey, T.L., Johnson, J., Grant, C.E. and Noble, W.S. (2015) The MEME Suite. *Nucleic acids research*, **43**, W39-49.
http://www.ncbi.nlm.nih.gov/pubmed/25953851
http://dx.doi.org/10.1093/nar/gkv416
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4489269

5. Frith, M.C., Saunders, N.F., Kobe, B. and Bailey, T.L. (2008) Discovering sequence motifs with arbitrary insertions and deletions. *PLoS computational biology*, **4**, e1000071.
http://www.ncbi.nlm.nih.gov/pubmed/18437229
http://dx.doi.org/10.1371/journal.pcbi.1000071
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2323616

6. Edwards, R.J., Davey, N.E. and Shields, D.C. (2007) SLiMFinder: a probabilistic method for identifying over-represented, convergently evolved, short linear motifs in proteins. *PloS one*, **2**, e967.
http://www.ncbi.nlm.nih.gov/pubmed/17912346
http://dx.doi.org/10.1371/journal.pone.0000967
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1989135

7. Song, T. and Gu, H. (2015) Discovering short linear protein motif based on selective training of profile hidden Markov models. *Journal of theoretical biology*, **377**, 75-84.
http://www.ncbi.nlm.nih.gov/pubmed/25791288
http://dx.doi.org/10.1016/j.jtbi.2015.03.010

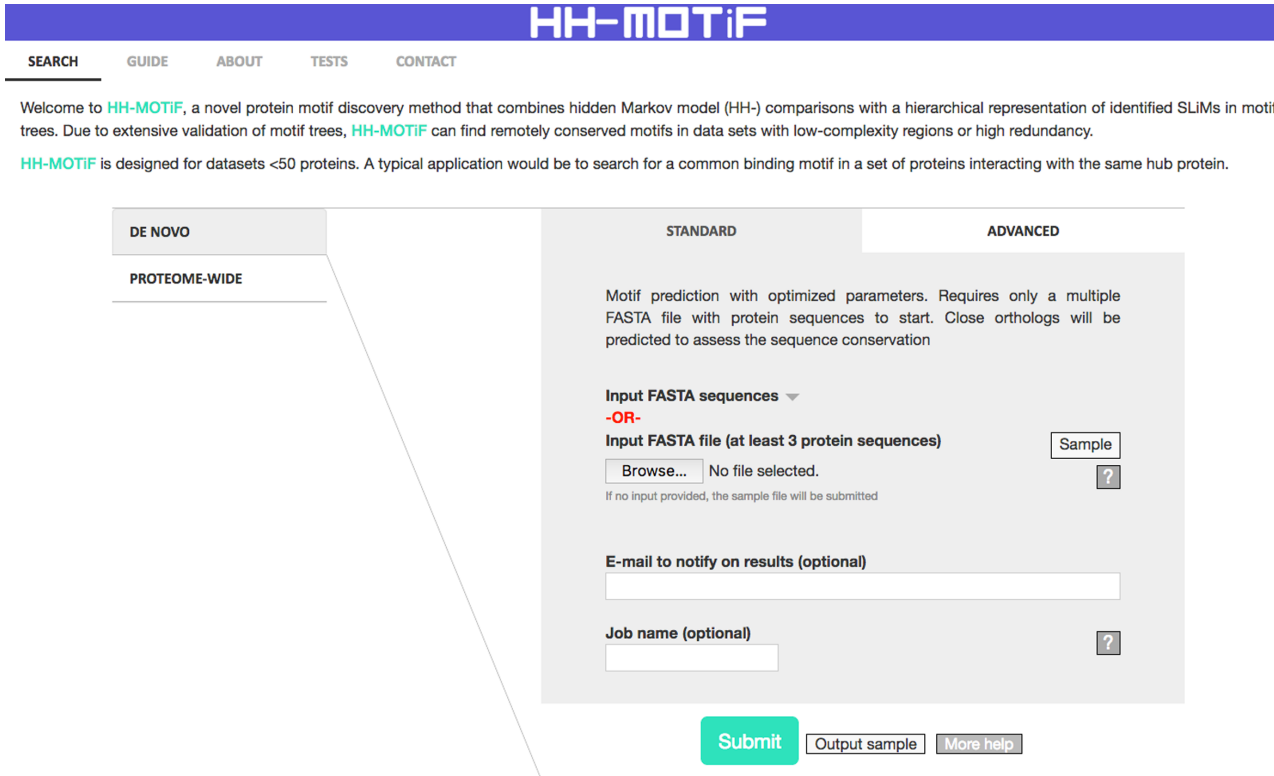
8. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic acids research*, **25**, 3389-3402.
http://www.ncbi.nlm.nih.gov/pubmed/9254694
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC146917

9. Soding, J. (2005) Protein homology detection by HMM-HMM comparison. *Bioinformatics (Oxford, England)*, **21**, 951-960.
http://www.ncbi.nlm.nih.gov/pubmed/15531603
http://dx.doi.org/10.1093/bioinformatics/bti125

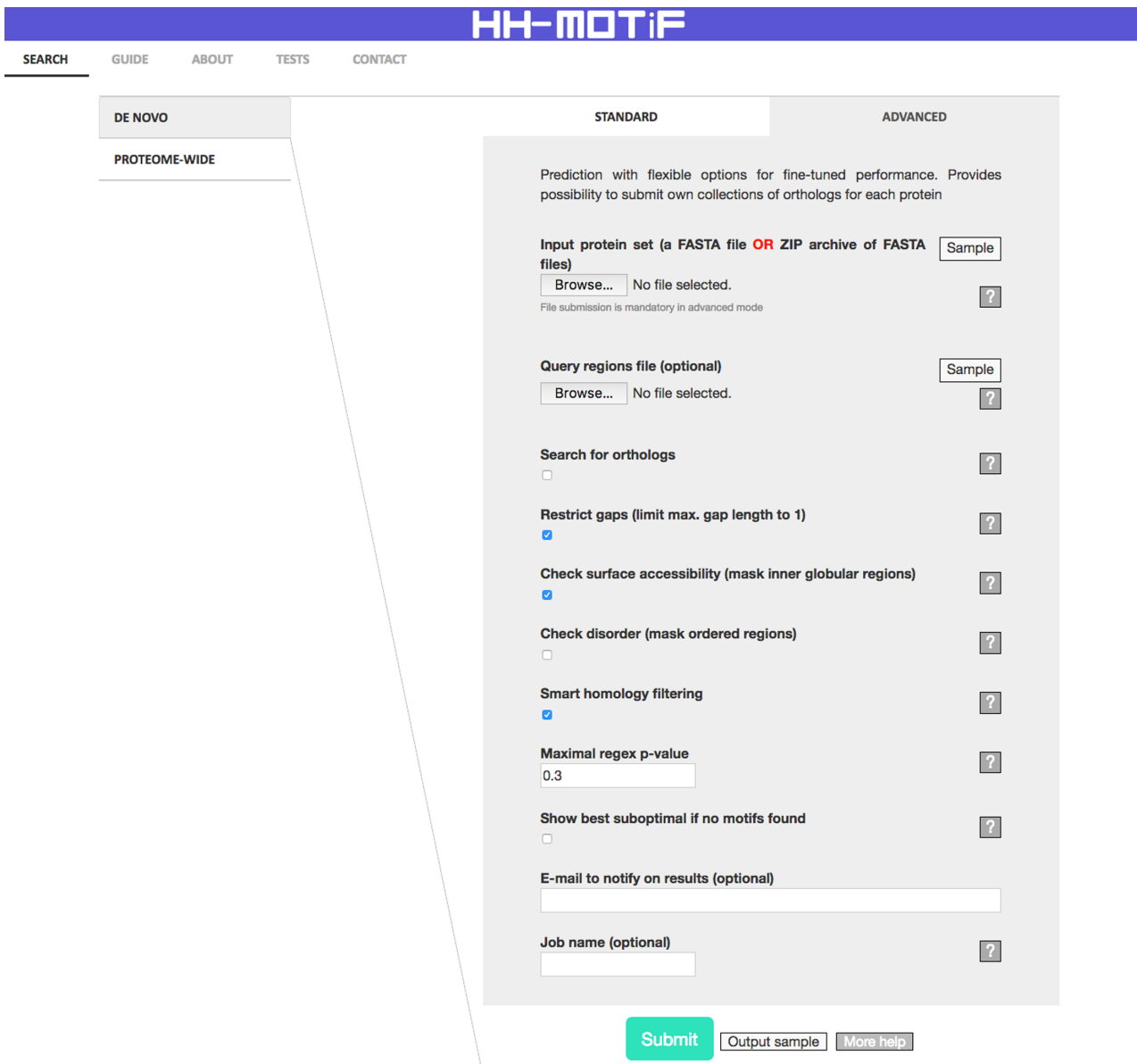
10. Petersen, B., Petersen, T.N., Andersen, P., Nielsen, M. and Lundegaard, C. (2009) A generic method for assignment of reliability scores applied to solvent accessibility predictions. *BMC structural biology*, **9**, 51.
http://www.ncbi.nlm.nih.gov/pubmed/19646261
http://dx.doi.org/10.1186/1472-6807-9-51
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2725087

Supplementary Figure S1

A

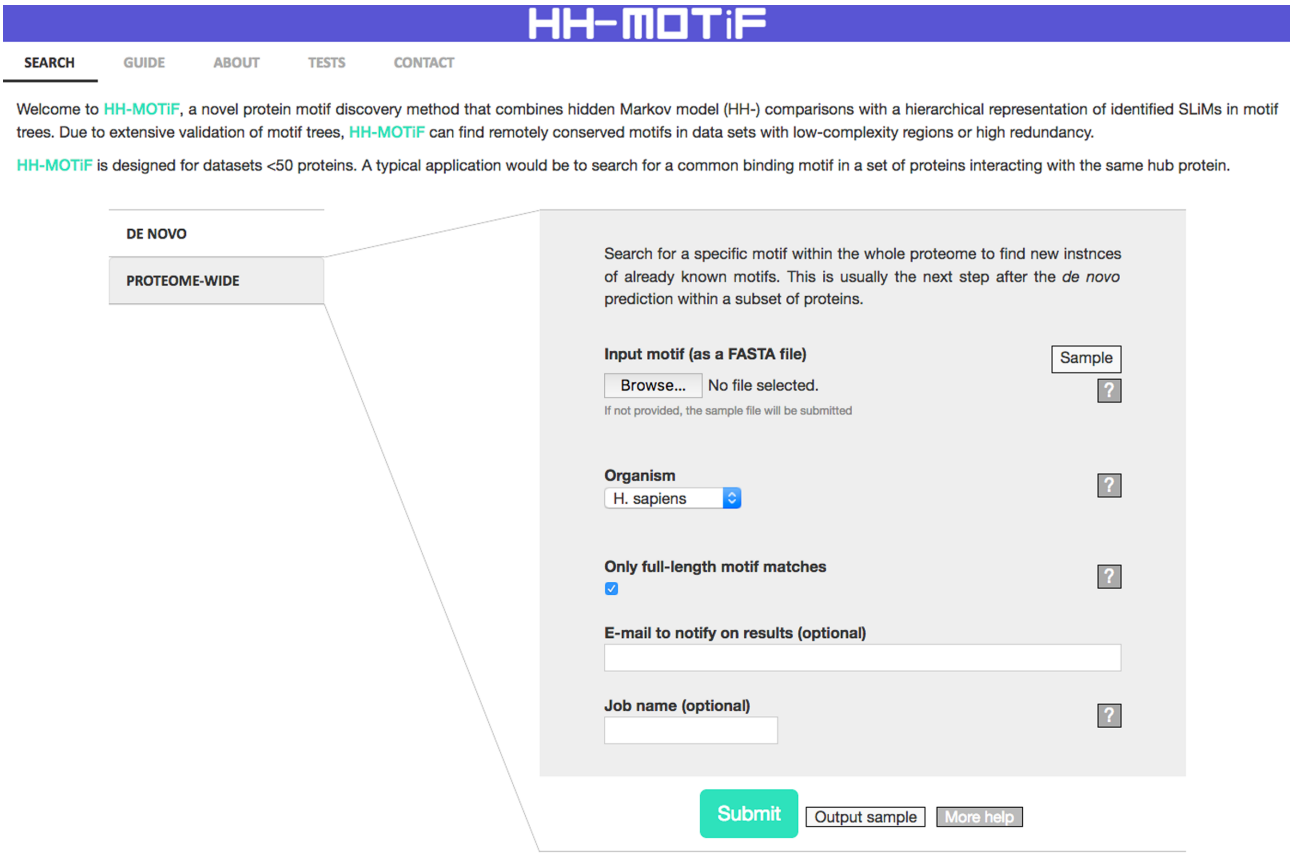


B



Supplementary Figure S1

C



Supplementary Figure S2