PAIRING INTERACTING PROTEIN SEQUENCES USING MASKED LANGUAGE MODELING

Damiano Sgarbossa[†], **Umberto Lupo**[†], **Anne-Florence Bitbol** [*]
Institute of Bioengineering, School of Life Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, CH-1015, Switzerland
{damiano.sgarbossa,umberto.lupo,anne-florence.bitbol}@epfl.ch

ABSTRACT

Predicting which proteins interact together from amino-acid sequences is an important task. We develop a method to pair interacting protein sequences which leverages the power of protein language models trained on multiple sequence alignments, such as MSA Transformer and the EvoFormer module of AlphaFold. We introduce a method called DiffPALM that solves it by exploiting the ability of MSA Transformer to fill in masked amino acids in multiple sequence alignments using the surrounding context. Relying on MSA Transformer without fine-tuning, DiffPALM outperforms existing coevolution-based pairing methods on difficult benchmarks of shallow multiple sequence alignments extracted from ubiquitous prokaryotic protein datasets. Paired alignments of interacting protein sequences are a crucial ingredient of supervised deep learning methods to predict the three-dimensional structure of protein complexes. Starting from sequences paired by DiffPALM substantially improves the structure prediction of some eukaryotic protein complexes by AlphaFold-Multimer.

1 INTRODUCTION

Interacting proteins play key roles in cells, ensuring the specificity of signaling pathways and forming multi-protein complexes that act e.g. as molecular motors or receptors. Predicting protein-protein interactions and structure of complexes is challenging (Rajagopala et al., 2014), even with high-throughput experiments.

A major advance in protein structure prediction was achieved by AlphaFold (Jumper et al., 2021) and other deep learning approaches (Baek et al., 2021; Lin et al., 2023). Extensions to protein complexes have been proposed (Humphreys et al., 2021; Mirdita et al., 2022; Bryant et al., 2022), including AlphaFold-Multimer (AFM) (Evans et al., 2021), but their performance is heterogeneous and less impressive than for monomers (Schweke et al., 2023).

The results of CASP15 demonstrated that the quality of multiple-sequence alignments (MSAs) is crucial (Elofsson, 2023; Alexander et al., 2023), and paired MSAs of interacting chains can offer coevolutionary signals to predict inter-chain contacts (Weigt et al., 2009; Bitbol et al., 2016; Gueudre et al., 2016; Szurmant & Weigt, 2018). However, constructing paired MSAs poses the challenge of properly pairing sequences. Accordingly, the quality of pairings strongly impacts the accuracy of heteromer structure prediction (Bryant et al., 2022; Si & Yan, 2022; Chen et al., 2023). Pairing interaction partners is difficult because many protein families contain several paralogous proteins encoded within the same genome. This problem is known as paralog matching. In prokaryotes, genomic proximity can often be used to solve it, since most interaction partners are encoded in close genomic locations (Orchard et al., 2013; Peters et al., 2016). However, this is not the case in eukaryotes. Large-scale coevolution studies of protein complexes (Ovchinnikov et al., 2014; Green et al., 2021) and deep learning approaches (Zeng et al., 2018; Evans et al., 2021; Humphreys et al., 2021; Bryant et al., 2022; Mirdita et al., 2022) have paired sequences by using genomic proximity

[*][†] These authors contributed equally to this work.

when possible (Ovchinnikov et al., 2014; Zeng et al., 2018; Mirdita et al., 2022), and/or by pairing together the closest, or equally ranked, hits to the query sequences, i.e. relying on approximate orthology (Zeng et al., 2018; Green et al., 2021; Mirdita et al., 2022).

However, coevolution-based approaches are data-thirsty and need large and diverse MSAs to perform well. This limits their applicability, especially to eukaryotic complex structure prediction. Nevertheless, the core idea of finding pairings that maximise coevolutionary signal holds promise for paralog matching and complex structure prediction.

We develop a new coevolution-based method for paralog matching that uses MSA Transformer (Rao et al., 2021b), a protein language model which was trained on MSAs using the masked language modeling (MLM) objective in a self-supervised way. We introduce DiffPALM, a differentiable method for predicting paralog matchings using MLM. DiffPALM excels in paralog matching on prokaryotic datasets and significantly improves AlphaFold's structure prediction for eukaryotic protein complexes in some cases, without yielding any significant deterioration. It also remains competitive with orthology-based pairing.

2 Results

2.1 Leveraging MSA-based protein language models for paralog matching

MSA-based protein language models, which include MSA Transformer (Rao et al., 2021b) and the EvoFormer module of AlphaFold (Jumper et al., 2021), are trained to correctly fill in masked amino acids in MSAs with the MLM loss (see B.5 and (Rao et al., 2021b; Lupo et al., 2022) for details). To this end, they use the rest of the MSA as context, which allows them to capture coevolution. Indeed, MSA Transformer achieves state-of-the-art performance at unsupervised structural contact prediction (Rao et al., 2021b), captures pairwise phylogenetic relationships between sequences (Lupo et al., 2022), and can be used to generate new sequences from given protein families (Sgarbossa et al., 2023). While MSA Transformer was only trained on MSAs corresponding to single chains, inter-chain coevolutionary signal has strong similarities with intra-chain signal (Weigt et al., 2009; Bitbol et al., 2016; Gueudre et al., 2016). As illustrated by Fig. 3, MSA Transformer is able to detect inter-chain contacts from a properly paired MSA (Xie & Xu, 2022). Fig. 3 further shows that it cannot do so from a wrongly paired MSA. Moreover, we find that the MLM loss (used for the pre-training of MSA Transformer) decreases as the fraction of correctly matched sequences increases, see Fig. 4. These results demonstrate that MSA Transformer captures inter-chain coevolutionary signal.

In this context, we ask the following question: Can we exploit MSA-based protein language models to address the paralog matching problem? Let us focus on the case where two MSAs have to be paired, which is the relevant one for heterodimers. Paralog matching amounts to pairing these two MSAs, each corresponding to one of two interacting protein families, so that correct interaction partners are placed on the same row of the paired MSA. Throughout, we will assume that interactions are one-to-one, excluding cross-talk, which is valid for proteins that interact specifically (Laub & Goulian, 2007). Thus, within each species, assuming that there is the same number of sequences from both families, we aim to find the correct one-to-one matching that associates one protein from the first family to one protein from the second family. Motivated by our finding that the MLM loss is lower for correctly paired MSAs than for incorrectly paired ones, we address the paralog matching problem by looking for pairings that minimise an MLM loss. A challenge is that the number of possible such one-to-one matchings scales factorially with the number of sequences in the species, making it difficult to find the permutation that minimises the loss by a brute-force search. We address this challenge by formulating a differentiable optimization problem that can be solved using gradient methods, to yield configurations minimizing our MLM loss, see B. We call our method **DiffPALM**, short for **Diff**erentiable **P**airing using **A**lignment-based **L**anguage **M**odels.

2.2 DiffPALM outperforms other coevolution methods on small MSAs

We start out by considering a well-controlled benchmark dataset composed of ubiquitous prokaryotic proteins from two interacting families, namely histidine kinases (HKs) and response regulators (RRs) (Barakat et al., 2009; 2011), see B.7. Because most cognate HK-RR pairs are encoded in the same operon, many interaction partners are known from genome proximity, which enables us to assess performance. In addition, earlier coevolution methods for paralog matching were tested on this dataset, allowing rigorous comparison (Bitbol et al., 2016; Bitbol, 2018; Gandarilla-Perez et al., 2023). Here, we focus on datasets comprising about 50 cognate HK-RR pairs. Indeed, this small data regime is problematic for existing coevolution methods, which require considerably deeper alignments to achieve good performance (Bitbol et al., 2016; Gueudre et al., 2016; Bitbol, 2018; Gandarilla-Perez et al., 2023). Furthermore, this regime is highly relevant for eukaryotic complexes, because their homologs have relatively low sequence diversity. While prokaryotic proteins such as HKs and RRs feature high diversity, focusing on small datasets allows us to address the relevant regime of low diversity in this well-controlled benchmark case. We test two variants of our DiffPALM method (see B) on 40 MSAs from the HK-RR dataset comprising about 50 HK-RR pairs each (see B.7). We first address the *de novo* pairing prediction task, starting from no known HK-RR pair, and then we study the impact of starting from known pairs.

Fig. 1 shows that DiffPALM performs better than the chance expectation, obtained for random within-species matching. Moreover, it outperforms other coevolution-based methods, namely DCA-IPA (Bitbol et al., 2016), MI-IPA (Bitbol, 2018) and GA-IPA (Gandarilla-Perez et al., 2023). Importantly, these results are obtained without giving any paired sequences as input to the algorithm. The performance of DiffPALM is particularly good for pairs with high confidence score (see B.2), as shown by the "precision-10" curve, which focuses on top 10% predicted pairs, when ranked by predicted confidence (see Fig. 1). We also propose a method based on a protein language model trained on a large ensemble of single sequences, ESM-2 (650M) (Lin et al., 2023), see B.3. DiffPALM also outperforms this method. This confirms that the coevolution information contained in the MSA plays a key role in the performance of DiffPALM, which is based on MSA Transformer. Fig. 1 shows that both variants of DiffPALM, namely Multi-Run Aggregation (MRA) and Iterative Pairing Algorithm (IPA), outperform all baselines, and that precision of MRA increases with the number of runs used. In MRA, we aggregate pairs predicted via independent optimization runs, while in IPA, we gradually add pairs with high confidence as positive examples (see B). DiffPALM-IPA thus exploits the good results obtained for high-confidence pairs, evidenced by the high precision-10 scores in Fig. 1.

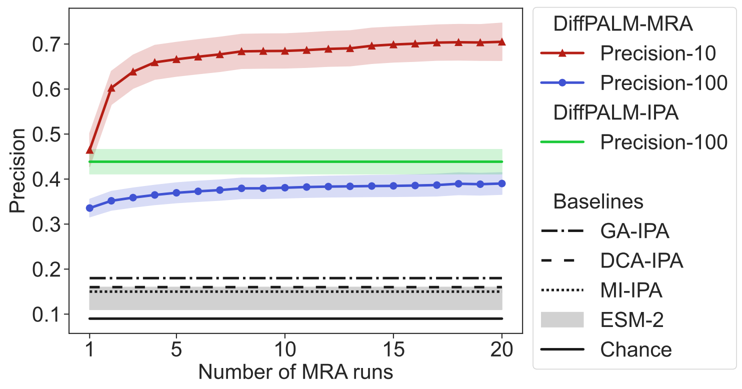

Figure 1: **Performance of DiffPALM on small HK-RR MSAs.** The performance of two variants of DiffPALM (MRA and IPA, see B.2) is shown versus the number of runs used for the MRA variant, for 40 MSAs comprising about 50 HK-RR pairs. The chance expectation, and the performance of various other methods, are reported as baselines. Three existing coevolution-based methods are considered: DCA-IPA (Bitbol et al., 2016), MI-IPA (Bitbol, 2018), and GA-IPA (Gandarilla-Perez et al., 2023). We also consider a pairing method based on the scores given by the ESM-2 (650M) single-sequence protein language model (Lin et al., 2023), see B.3. With all methods, a full one-to-one within-species pairing is produced, and performance is measured by precision (also called positive predictive value or PPV), namely, the fraction of correct pairs among predicted pairs. The default score is "precision-100", where this fraction is computed over all predicted pairs (100% of them). For DiffPALM-MRA, we also report "precision-10", which is calculated over the top 10% predicted pairs (this is not defined for the DiffPALM-IPA method so we do not report it), when ranked by predicted confidence within each MSA (see B).

So far, we addressed *de novo* pairing prediction, where no known HK-RR pair is given as input. Can DiffPALM precision increase by exploiting "positive examples" of known interacting partners? This is an important question, since experiments on model species may for instance give some positive examples (see B.1). To address it, we included different numbers of positive examples, by using the corresponding non-masked interacting pairs as context (see B.2). We observe that the performance of DiffPALM significantly increases with the number of positive examples used, reaching almost perfect performance for the highest-confidence pairs (precision-10), see left panel of Fig. 5.

Another important parameter is MSA depth. Indeed, deeper MSAs yield better performance for traditional coevolution methods (Bitbol et al., 2016; Bitbol, 2018; Gandarilla-Perez et al., 2023). DiffPALM performance also increases with MSA depth, while already reaching good performance for shallow MSAs (see middle panel of Fig. 5). We extended this analysis also to other cases obtaining similar results, we considered an alignment of homologs of the *E. coli* proteins MALG-MALK, which are involved in ABC transporter complexes.

2.3 USING DIFFPALM FOR EUKARYOTIC COMPLEX STRUCTURE PREDICTION BY AFM

An important and more challenging application of DiffPALM is predicting interacting partners among the paralogs of two families in eukaryotic species. Indeed, eukaryotes often have many paralogs per species (Makarova et al., 2005) but eukaryotic-specific protein families generally have fewer total homologs and smaller diversity than prokaryotes. Moreover, most interacting proteins are not encoded in close proximity in eukaryotic genomes. Paired MSAs are a key ingredient of protein complex structure prediction by AFM (Evans et al., 2021; Bryant et al., 2022). When presented with query sequences, the default AFM pipeline (Evans et al., 2021) retrieves homologs of each of the chains. Within each species, homologs of different chains are ranked according to Hamming distance to the corresponding query sequence. Then, equal-rank sequences are paired. Can DiffPALM improve complex structure prediction by AFM? To address this question, we consider 15 complexes, listed in Tab. 1, whose structures are not included in the training set of the AFM release we used unless specified otherwise (v2, see B.8), and for which the default AFM complex prediction was previously reported to perform poorly (Chen et al., 2023; Evans et al., 2021) (see B.7).

Fig. 2 (top panels) shows that DiffPALM can improve complex structure prediction by AFM. This suggests that it is able to produce better paired MSAs than those from the default AFM pipeline. In particular, substantial improvements are obtained for the complexes with PDB identifiers 6L5K and 6FYH. We found no cases in which using DiffPALM improved AFM confidence scores but deteriorated structure predictions. In most cases, the quality of structures predicted using DiffPALM pairing is comparable to that obtained using the pairing method adopted e.g. by ColabFold (Mirdita et al., 2022), where only the orthologs of the two query sequences, identified as their best hits, are paired in each species (resulting in at most one pair per species) (Mirdita et al., 2022), see Fig. 2. Note however that, for 6PNQ, the ortholog-only pairing method is outperformed both by DiffPALM and by the default AFM pairing. Indeed, the raw and effective MSA depths are smaller for this structure than e.g. for 6L5K and 6FYH (see Tab. 1). Thus, further reducing diversity by restricting to paired orthologs may be negatively impacting structure prediction in this case.

Although DiffPALM achieves similar performance on these structure prediction tasks as using orthology, it predicts some pairs that are quite different from orthology-based pairs. Indeed, Fig. 6 shows that the fraction of pairs identically matched by DiffPALM and by orthology is often smaller than 0.5. Fig. 7 further shows that, for the sequences that are paired differently by DiffPALM and by orthology, the Hamming distances between the two predicted partners is often above 0.5. Nevertheless, most of the pairs that are predicted both by DiffPALM and by using orthology have high DiffPALM confidence (see Fig. 6), confirming the importance of these pairs. Finally, Fig. 8 compares the performance of DiffPALM to the AFM default and the ortholog-only pairing methods using the latest AFM release (v3, see B.7). Since all the structures considered here are included in the training set of this release, we expect the standard AFM pipeline to perform well. However, we observe that using DiffPALM yields similar or better performance than using the AFM default or ortholog-only pairing methods. In particular, the structure prediction of 6L5K remains substantially improved by using DiffPALM rather than AFM default pairing. Moreover, for 6THL, which was poorly predicted by all methods with the older AFM release (DockQ < 0.1), DiffPALM yields a substantial improvement compared to both other pairing methods with the latest AFM release.

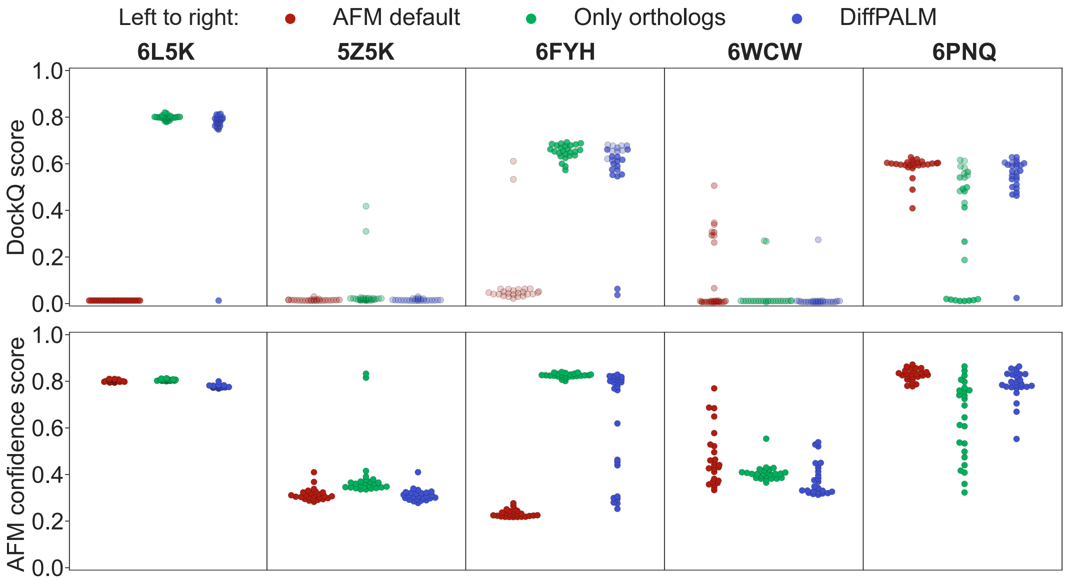

Figure 2: **Performance of AFM using different pairing methods.** We use AFM to predict the structure of protein complexes starting from differently paired MSAs, each of them constructed from the same initial unpaired MSAs. Three pairing methods are considered: the default one of AFM, only pairing orthologs to the two query sequences, and a single run of DiffPALM (equivalent to one MRA run). We used a single run for computational time reasons. Performance is evaluated using DockQ scores (top panels), a widely used measure of quality for protein-protein docking (Basu & Wallner, 2016), and the AFM confidence scores (bottom panels), see B.8. The latter are also used as transparency levels in the top panels, where more transparent markers denote predicted structures with low AFM confidence. For each query complex, AFM is run five times. Each run yields 25 predictions which are ranked by AFM confidence score. The top five predicted structures are selected from each run, giving 25 predicted structures in total for each complex. Out of the 15 complexes listed in Tab. 1, we restrict to those where any two of these three pairing methods yield a significant difference (> 0.1) in average DockQ scores for at least one set of predictions coming from different runs but with the same within-run rank according to AFM confidence. Panels are ordered by increasing mean DockQ score for the AFM default method.

3 DISCUSSION

DiffPALM outperforms existing coevolution-based methods and a method based on a state-of-the-art language model trained on single sequences. Its performance quickly increases when adding examples of known interacting sequences. It also increases with MSA depth, as for traditional coevolution methods. Paired MSAs of interacting partners are a key ingredient to complex structure prediction by AFM. We found that using DiffPALM can improve the performance of AFM, and achieves competitive performance with orthology-based pairing.

DiffPALM illustrates the power of neural protein language models trained on MSAs, and their ability to capture the rich structure of biological sequence data. The fact that these models encode inter-chain coevolution, while they are trained on single-chain data, suggests that they are able to generalize. We used MSA Transformer in a zero-shot setting, without fine-tuning it to the task of interaction partner prediction. Such fine-tuning could yield further performance gains (Hawkins-Hooker et al., 2022).

Like MSA Transformer, AlphaFold's EvoFormer (Jumper et al., 2021) also processes MSA input. Thus, one could develop a paralog matching method based on EvoFormer. However, the substantially larger pre-training dataset of MSA Transformer should make DiffPALM more broadly applicable. Nevertheless, an interesting perspective would be to integrate paralog matching into an end-to-end retraining of AlphaFold Multimer.

REFERENCES

L. T. Alexander, J. Durairaj, A. Kryshtafovych, L. A. Abriata, Y. Bayo, G. Bhabha, C. Breyton, S. G. Caulton, J. Chen, S. Degroux, D. C. Ekiert, B. S. Erlandsen, P. L. Freddolino, D. Gilzer, C. Greening, J. M. Grimes, R. Grinter, M. Gurusaran, M. D. Hartmann, C. J. Hitchman, J. R. Keown, A. Kropp, P. Kursula, A. L. Lovering, B. Lemaitre, A. Lia, S. Liu, M. Logotheti, S. Lu, S. sson, M. D. Miller, G. Minasov, H. H. Niemann, F. Opazo, G. N. Phillips, O. R. Davies, S. Rommelaere, M. Rosas-Lemus, P. Roversi, K. Satchell, N. Smith, M. A. Wilson, K. L. Wu, X. Xia, H. Xiao, W. Zhang, Z. H. Zhou, K. Fidelis, M. Topf, J. Moult, and T. Schwede. Protein target highlights in CASP15: Analysis of models by structure providers. *Proteins*, pp. 1–29, 2023. doi: 10.1002/prot.26545.

Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy DeGiovanni, Jose H. Pereira, Andria V. Rodrigues, Alberdina A. van Dijk, Ana C. Ebrecht, Diederik J. Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K. Rathinaswamy, Udit Dalwadi, Calvin K. Yip, John E. Burke, K. Christopher Garcia, Nick V. Grishin, Paul D. Adams, Randy J. Read, and David Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021. doi: 10.1126/science.abj8754.

M. Barakat, P. Ortet, C. Jourlin-Castelli, M. Ansaldi, V. Mejean, and D. E. Whitworth. P2CS: a two-component system resource for prokaryotic signal transduction research. *BMC Genomics*, 10:315, 2009. doi: 10.1186/1471-2164-10-315.

M. Barakat, P. Ortet, and D. E. Whitworth. P2CS: a database of prokaryotic two-component systems. *Nucleic Acids Research*, 39(Database issue):D771–776, 2011. doi: 10.1093/nar/gkq1023.

Sankar Basu and Björn Wallner. DockQ: A quality measure for protein-protein docking models. *PLoS ONE*, 11(8):1–9, 2016. doi: 10.1371/journal.pone.0161879.

Anne-Florence Bitbol. Inferring interaction partners from protein sequences using mutual information. *PLoS Comput. Biol.*, 14(11):e1006401, 2018. doi: 10.1371/journal.pcbi.1006401.

Anne-Florence Bitbol, Robert S Dwyer, Lucy J Colwell, and Ned S Wingreen. Inferring interaction partners from protein sequences. *Proc. Natl. Acad. Sci. U.S.A.*, 113(43):12180–12185, 2016. doi: 10.1073/pnas.1606762113.

Patrick Bryant, Gabriele Pozzati, and Arne Elofsson. Improved prediction of protein-protein interactions using AlphaFold2. *Nat Commun*, 13(1):1265, 2022. doi: 10.1038/s41467-022-28865-w.

Bo Chen, Ziwei Xie, Jiezhong Qiu, Zhaofeng Ye, Jinbo Xu, and Jie Tang. Improved the heterodimer protein complex prediction with protein language models. *Briefings in Bioinformatics*, 24(4): bbad221, 2023. doi: 10.1093/bib/bbad221.

Qian Cong, Ivan Anishchenko, Sergey Ovchinnikov, and David Baker. Protein interaction networks revealed by proteome coevolution. *Science*, 365(6449):185–189, 2019. doi: 10.1126/science.aaw6718.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

A. Elofsson. Progress at protein structure prediction, as seen in CASP15. *Current Opinion in Structural Biology*, 80:102594, 2023. doi: 10.1016/j.sbi.2023.102594.

Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer. *bioRxiv*, 2021. doi: 10.1101/2021.10.04.463034.

Carlos A. Gandarilla-Perez, Sergio Pinilla, Anne-Florence Bitbol, and Martin Weigt. Combining phylogeny and coevolution improves the inference of interaction partners among paralogous proteins. *PLoS Comput. Biol.*, 19(3):e1011010, 2023. doi: 10.1371/journal.pcbi.1011010.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Exposing the implicit energy networks behind masked language models via Metropolis–Hastings. *arXiv*, 2021. doi: 10.48550/arXiv.2106.02736.

A. G. Green, H. Elhabashy, K. P. Brock, R. Maddamsetti, O. Kohlbacher, and D. S. Marks. Large-scale discovery of protein interactions at residue resolution using co-evolution calculated from genomic sequences. *Nat Commun*, 12(1):1396, 2021. doi: 10.1038/s41467-021-21636-z.

T. Gueudre, C. Baldassi, M. Zamparo, M. Weigt, and A. Pagnani. Simultaneous identification of specifically interacting paralogs and interprotein contacts by direct coupling analysis. *Proc. Natl. Acad. Sci. U.S.A.*, 113(43):12186–12191, 2016. doi: 10.1073/pnas.1607570113.

Alex Hawkins-Hooker, David T Jones, and Brooks Paige. Using domain-domain interactions to probe the limitations of MSA pairing strategies. In *Machine Learning for Structural Biology Workshop, NeurIPS*, 2022. URL `https://www.mlsb.io/papers_2022/Using_domain_domain_interactions_to_probe_the_limitations_of_MSA_pairing_strategies.pdf`.

Ian Humphreys, Jimin Pei, Minkyung Baek, Aditya Krishnakumar, Ivan Anishchenko, Sergey Ovchinnikov, Jing Zhang, Travis J. Ness, Sudeep Banjade, Saket R. Bagde, Viktoriya G. Stancheva, Xiao Han Li, Kaixian Liu, Zhi Zheng, Daniel J. Barrero, Upasana Roy, Jochen Kuper, Israel S. Fernández, Barnabas Szakal, Dana Branzei, Josep Rizo, Caroline Kisker, Eric C. Greene, Sue Biggins, Scott Keeney, Elizabeth A. Miller, J. Christopher Fromme, Tamara L. Hendrickson, Qian Cong, and David Baker. Computed structures of core eukaryotic protein complexes. *Science*, 374(6573), 2021. ISSN 10959203. doi: 10.1126/science.abm4805.

L Steven Johnson, Sean R Eddy, and Elon Portugaly. Hidden Markov model speed heuristic and iterative HMM search procedure. *BMC bioinformatics*, 11:1–8, 2010. doi: 10.1186/1471-2105-11-431.

J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955. doi: 10.1002/nav.3800020109.

M. T. Laub and M. Goulian. Specificity in two-component signal transduction pathways. *Annu. Rev. Genet.*, 41:121–145, 2007. doi: 10.1146/annurev.genet.41.042007.170548.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

Umberto Lupo, Damiano Sgarbossa, and Anne-Florence Bitbol. Protein language models trained on multiple sequence alignments learn phylogenetic relationships. *Nat Commun*, 13(6298), 2022. doi: 10.1038/s41467-022-34032-y.

Kira S. Makarova, Yuri I. Wolf, Sergey L. Mekhedov, Boris G. Mirkin, and Eugene V. Koonin. Ancestral paralogs and pseudoparalogs and their role in the emergence of the eukaryotic cell. *Nucleic Acids Research*, 33(14):4626–4638, 2005. ISSN 0305-1048. doi: 10.1093/nar/gki775.

Gonzalo E. Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with Gumbel-Sinkhorn networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, pp. 1–22, 2018. URL https://openreview.net/forum?id=Byt3oJ-0W.

Milot Mirdita, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. ColabFold: making protein folding accessible to all. *Nat Methods*, 19(6):679–682, 2022. doi: 10.1038/s41592-022-01488-1.

Christoffer Norn, Basile I. M. Wicky, David Juergens, Sirui Liu, David Kim, Doug Tischer, Brian Koepnick, Ivan Anishchenko, Foldit Players, David Baker, and Sergey Ovchinnikov. Protein sequence design by conformational landscape optimization. *Proc. Natl. Acad. Sci. U.S.A.*, 118 (11):e2017228118, 2021. doi: 10.1073/pnas.2017228118.

Sandra Orchard, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H. Campbell, Gayatri Chavali, Carol Chen, Noemi del Toro, Margaret Duesbury, Marine Dumousseau, Eugenia Galeota, Ursula Hinz, Marta Iannuccelli, Sruthi Jagannathan, Rafael Jimenez, Jyoti Khadake, Astrid Lagreid, Luana Licata, Ruth C. Lovering, Birgit Meldal, Anna N. Melidoni, Mila Milagros, Daniele Peluso, Livia Perfetto, Pablo Porras, Arathi Raghunath, Sylvie Ricard-Blum, Bernd Roechert, Andre Stutz, Michael Tognolli, Kim van Roey, Gianni Cesareni, and Henning Hermjakob. The MIntAct project—IntAct as a common curation platform for 11 molecular interaction databases. *Nucleic Acids Research*, 42(D1):D358–D363, 2013. ISSN 0305-1048. doi: 10.1093/nar/gkt1115.

S. Ovchinnikov, H. Kamisetty, and D. Baker. Robust and accurate prediction of residue-residue interactions across protein interfaces using evolutionary information. *eLife*, 3:e02030, 2014. doi: 10.7554/eLife.02030.

Jason M. Peters, Alexandre Colavin, Handuo Shi, Tomasz L. Czarny, Matthew H. Larson, Spencer Wong, John S. Hawkins, Candy H.S. Lu, Byoung-Mo Koo, Elizabeth Marta, Anthony L. Shiver, Evan H. Whitehead, Jonathan S. Weissman, Eric D. Brown, Lei S. Qi, Kerwyn Casey Huang, and Carol A. Gross. A comprehensive, CRISPR-based functional analysis of essential genes in bacteria. *Cell*, 165(6):1493–1506, 2016. ISSN 0092-8674. doi: 10.1016/j.cell.2016.05.003.

Gabriele Pozzati, Wensi Zhu, Claudio Bassot, John Lamb, Petras Kundrotas, and Arne Elofsson. Limits and potential of combined folding and docking. *Bioinformatics*, 38(4):954–961, 2021. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab760.

S. V. Rajagopala, P. Sikorski, A. Kumar, R. Mosca, J. Vlasblom, R. Arnold, J. Franca-Koh, S. B. Pakala, S. Phanse, A. Ceol, R. Hauser, G. Siszler, S. Wuchty, A. Emili, M. Babu, P. Aloy, R. Pieper, and P. Uetz. The binary protein-protein interaction landscape of Escherichia coli. *Nat Biotechnol*, 32(3):285–290, 2014. doi: 10.1038/nbt.2831.

Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. Transformer protein language models are unsupervised structure learners. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=fylclEqgvgd.

Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. MSA Transformer. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 8844–8856. PMLR, 2021b. URL https://proceedings.mlr.press/v139/rao21a.html.

Hugo Schweke, Tal Levin, Martin Pacesa, Casper A. Goverde, Prasun Kumar, Yoan Duhoo, Lars J. Dornfeld, Benjamin Dubreuil, Sandrine Georgeon, Sergey Ovchinnikov, Derek N. Woolfson, Bruno E. Correia, Sucharita Dey, and Emmanuel D. Levy. An atlas of protein homo-oligomerization across domains of life. *bioRxiv*, 2023. doi: 10.1101/2023.06.09.544317.

D. Sgarbossa, U. Lupo, and A.-F. Bitbol. Generative power of a protein language model trained on multiple sequence alignments. *Elife*, 12:e79854, 2023. doi: 10.7554/eLife.79854.

Yunda Si and Chengfei Yan. Protein complex structure prediction powered by multiple sequence alignments of interologs from multiple taxonomic ranks and AlphaFold2. *Briefings in Bioinformatics*, 23(4):bbac208, 2022. doi: 10.1093/bib/bbac208.

Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat Biotechnol*, 35(11):1026–1028, 2017. ISSN 1087-0156. doi: 10.1038/nbt.3988.

Hendrik Szurmant and Martin Weigt. Inter-residue, inter-protein and inter-family coevolution: bridging the scales. *Current Opinion in Structural Biology*, 50:26–32, 2018. doi: 10.1016/j.sbi.2017.10.014.

Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a Markov random field language model. *arXiv*, 2019. doi: 10.48550/arXiv.1902.04094.

M. Weigt, R. A. White, H. Szurmant, J. A. Hoch, and T. Hwa. Identification of direct residue contacts in protein-protein interaction by message passing. *Proc. Natl. Acad. Sci. U.S.A.*, 106(1): 67–72, 2009. doi: 10.1073/pnas.0805923106.

Ziwei Xie and Jinbo Xu. Deep graph learning of inter-protein contacts. *Bioinformatics*, 38(4): 947–953, 2022. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab761.

Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *arXiv*, 2012. doi: 10.48550/arXiv.1212.5701.

Hong Zeng, Sheng Wang, Tianming Zhou, Feifeng Zhao, Xiufeng Li, Qing Wu, and Jinbo Xu. ComplexContact: a web server for inter-protein contact prediction using deep learning. *Nucleic Acids Research*, 46(W1):W432–W437, 2018. ISSN 0305-1048. doi: 10.1093/nar/gky420.

A Supplementary figures and tables

PDB ID	L_A	L_B	D	F_{paired}	$\langle d_p \rangle$	MSR	F_{same}	F_{pred}	D_{DiffPALM}	D_{eff}^A	D_{eff}^B
6QU1	322	48	9267	0.02	3.4	5	0.37	0.45	76	20	11
6POG	114	249	18390	0.20	30.1	3	0.03	0.22	821	114	93
6THL	240	185	3515	0.03	2.2	5	0.40	0.46	53	28	26
6L5K	98	113	15857	0.29	22.9	3	0.05	0.75	3478	402	644
6A6I	98	76	4793	0.11	3.1	5	0.30	0.48	259	57	87
5Z5K	380	66	6349	0.03	12	10	0	0.02	3	3	3
6FYH	124	76	15285	0.51	7.1	3	0.14	0.72	5550	1703	1640
6WCW	184	254	20435	0.21	6.7	3	0.18	0.57	2509	263	402
5XLN	190	45	9434	0.04	3.4	5	0.30	0.65	228	30	31
7BQU	114	28	5915	0.04	2.9	5	0.44	0.65	152	67	61
6ABO	227	82	5058	0.10	4.5	3	0.32	0.56	310	65	25
6INE	267	177	19227	0.18	4	3	0.26	0.39	1334	402	480
6IRE	234	194	7599	0.12	4.2	3	0.21	0.63	591	120	210
6PNQ	202	168	5694	0.20	8.7	5	0.13	0.25	282	117	23
6GK2	106	92	3062	0.08	2.4	5	0.35	0.59	145	20	40

Table 1: **Dataset of eukaryotic complexes.** All 15 eukaryotic protein complexes considered here are listed by their PDB ID, and various MSA properties are given. L_A and L_B are the lengths of the aligned amino acid sequences of the two chains A and B considered. D denotes the depth of the full MSA built by AFM, consisting of the paired MSA and the block MSAs (see B.4). F_{paired} is the fraction of sequences that are paired by AFM, i.e. the depth of the paired MSA divided by D. $\langle d_p \rangle$ is the average depth of the MSAs of the single species in the AFM-paired MSA. Thus it is the average of the largest depth among the two chains, since the other one is completed by padding sequences of gaps. MSR stands for "maximum size ratio": if the ratio of the larger to the smaller of the depths of MSAs A and B is larger than MSR, species are not paired by DiffPALM. F_{same} is the fraction of pairs predicted by DiffPALM that is identical to the pairs predicted by the default AFM pairing method. F_{pred} is the ratio of the number of pairs predicted with DiffPALM to the number predicted using the default AFM pairing method. $D_{\mathrm{DiffPALM}} = D \times F_{\mathrm{paired}} \times F_{\mathrm{pred}}$ denotes the number of pairs output by DiffPALM. D_{eff}^A (resp. D_{eff}^B) is the effective depth corrected with phylogenetic weights (with Hamming distance threshold 0.2) (Weigt et al., 2009) of the MSA associated to chain A (resp. chain B) in the MSAs which are paired by DiffPALM. These effective depths quantify MSA diversity. Rows are ordered by increasing mean DockQ score for the default AFM pairing method.

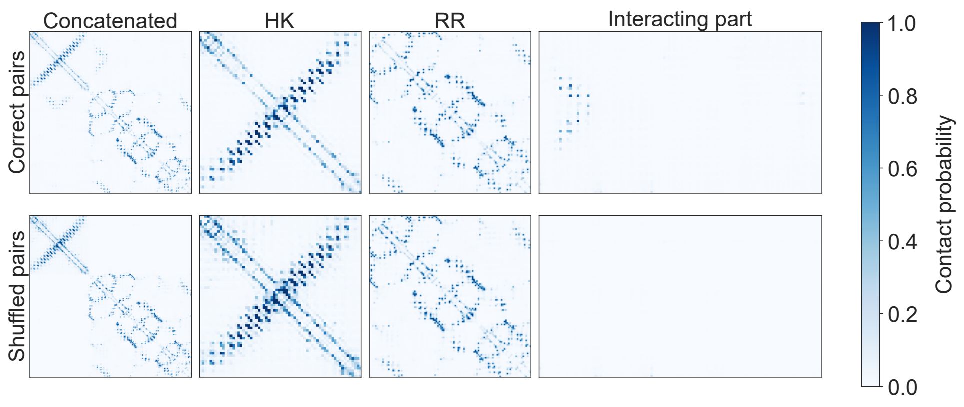

Figure 3: **Comparison of contact maps predicted by MSA Transformer for the correct pairing of an HK MSA and an RR MSA ("Correct pairs"), and for an incorrect pairing ("Shuffled pairs").** We observe that MSA Transformer is able to correctly predict the inter-protein contacts when given as input a paired MSA made of correctly matched sequences. Conversely, if the model is given as input a paired MSAs where rows have been shuffled before pairing, it is not able to recover the inter-protein contact map (even though it correctly recovers correctly the intra-protein contact maps). These results suggests that MSA Transformer can distinguish between interacting and non-interacting pairs of protein sequences, despite the fact that dimers or paired MSAs were not in the training set used for its MLM pre-training (Rao et al., 2021b).

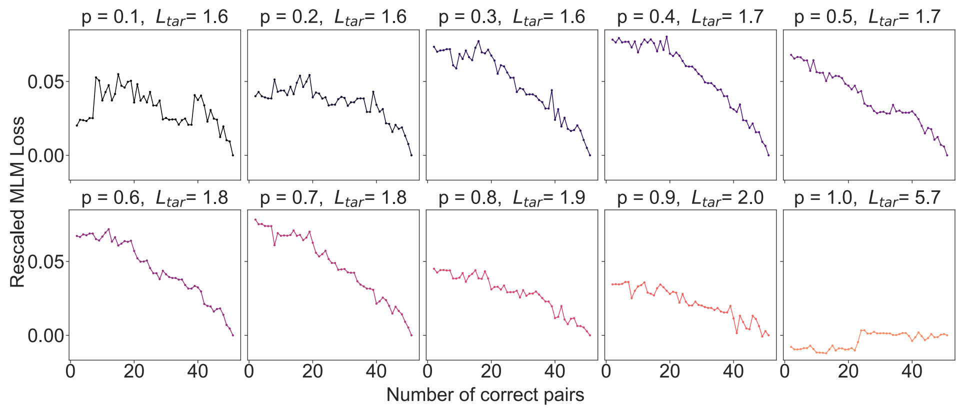

Figure 4: **MLM loss vs. number of correct pairs for different masking probabilities.** We use an MSA of 50 correctly paired sequences comprising 5 different species from the HK-RR dataset. To estimate the expected loss accurately, we used 20 different masks at each step. L_{tar} denotes the expected loss when all pairs are correctly matched. For visualization purposes, in every plot we rescale the loss by shifting it by L_{tar}. We find that our MLM loss in 1 decreases for increasing numbers of correctly matched sequences in the MSA. We see that the sweet spot of the masking probability p (i.e. the value that gives steeper and smoother loss curves) is at moderately high values $(0.4 \leq p \leq 0.8)$. As explained in B, high masking probabilities make it more challenging for the model to predict the masked amino acids using only information coming from the masked MSA, thus encouraging it to use, instead, information coming from the matched MSA. This motivates our choice of a masking probability of $p \geq 0.7$.

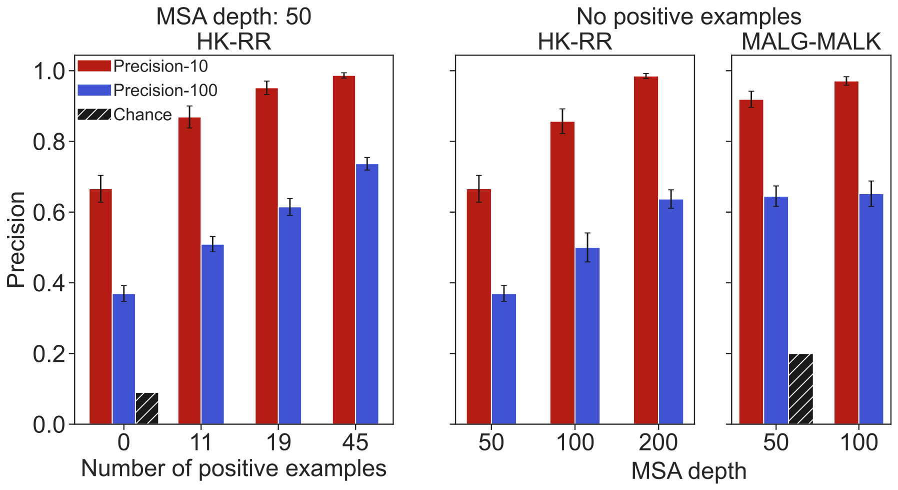

Figure 5: **Impact of positive examples, MSA depth, and extension to another pair of protein families.** We report the performance of DiffPALM with 5 MRA runs (measured as precision-100 and precision-10, see Fig. 1), for various numbers of positive examples, on the same HK-RR MSAs as in Fig. 1 (left panel). We also report the performance of DiffPALM (using no positive examples) versus MSA depth for both HK-RR and MALG-MALK pairs (middle and right panel). In all cases, we show the mean value over different MSAs and its standard error, and we plot the chance expectation for reference.

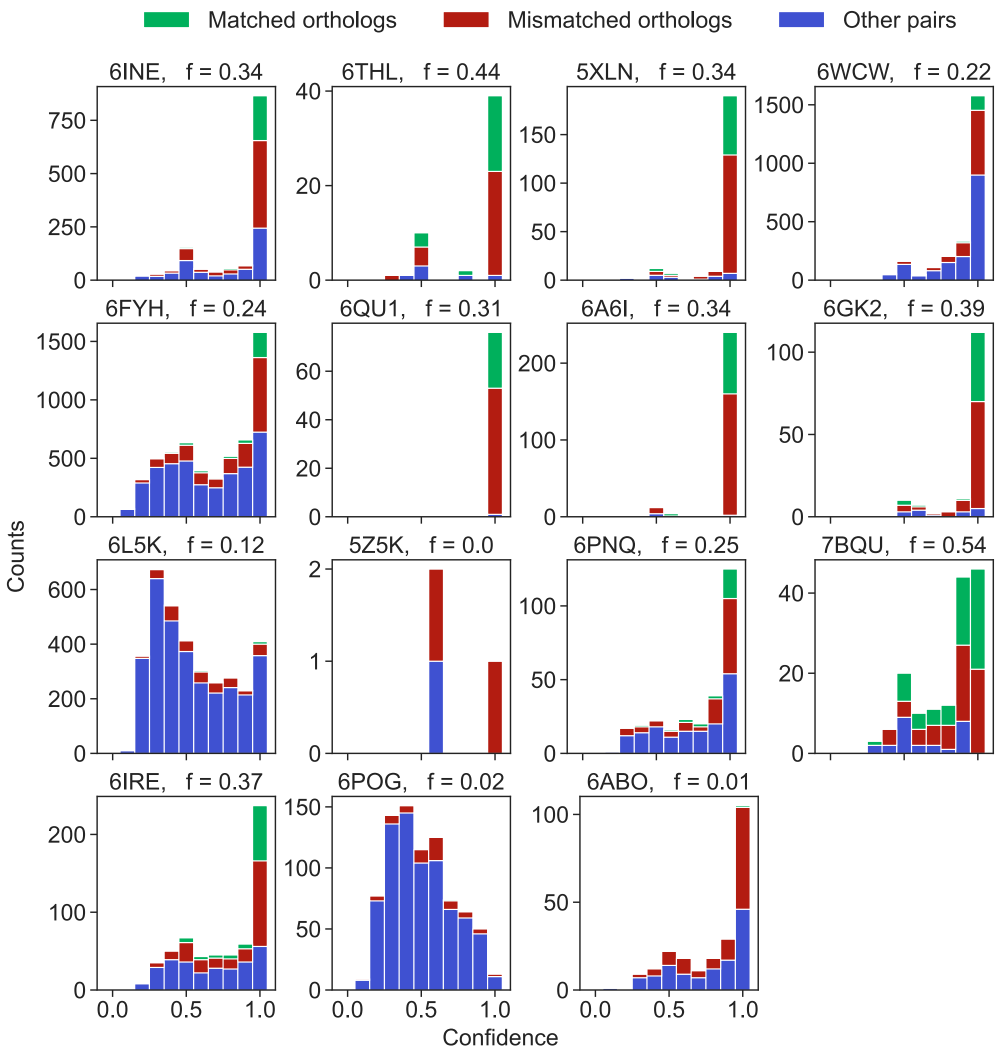

Figure 6: **Confidence of DiffPALM predictions.** We show, for the 15 complexes listed in Tab. 1, histograms of the DiffPALM confidence values (see B.2). We distinguish the orthology-based pairs that are recovered by DiffPALM, the otherwise paired orthologs, and all the other paired sequences. We indicate in panel titles the value of the fraction f of orthology-based pairs that are recovered by DiffPALM.

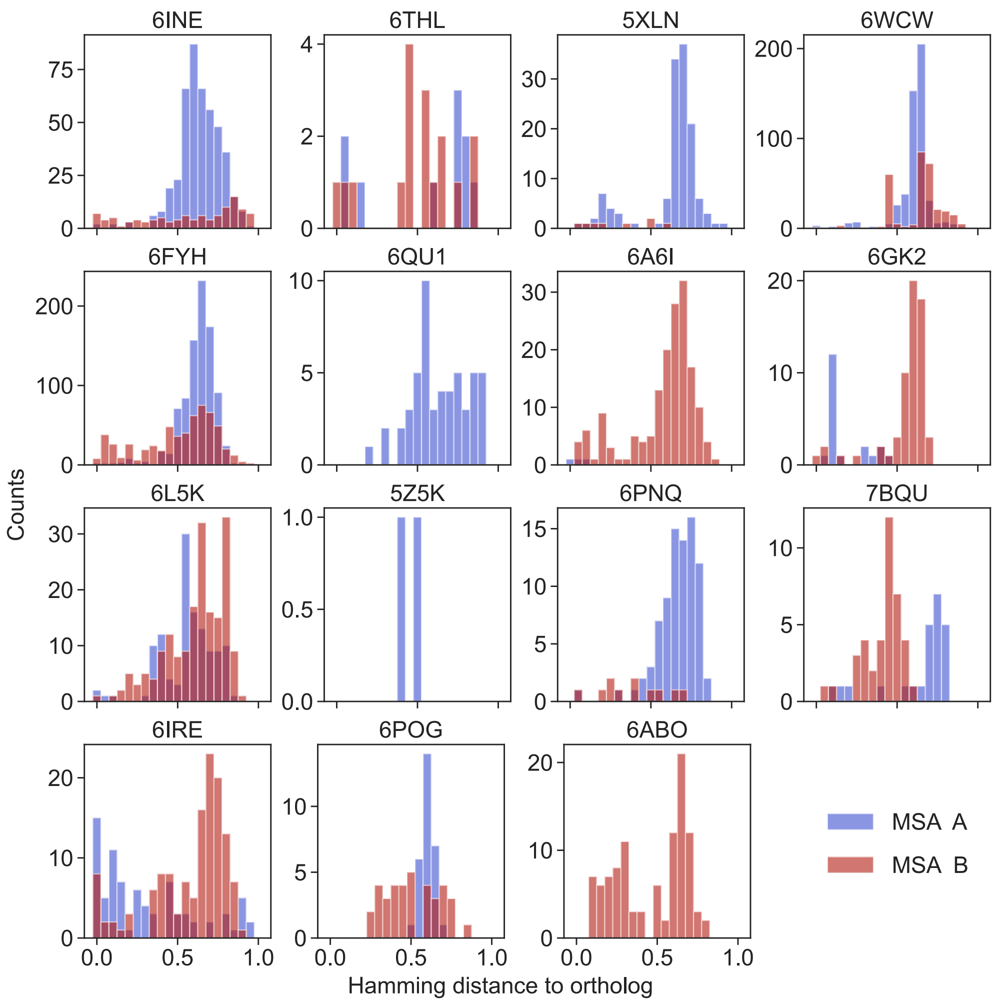

Figure 7: **Hamming distance to the orthologs of all the mismatched pairs predicted by DiffPALM.** We show, for the 15 complexes listed in Tab. 1, histograms of the Hamming distance between the partner predicted by DiffPALM and the one predicted by matching orthologs to the query sequences, whenever they differ. In practice, for each sequence A_1 in family A which is paired with a partner B_1 from family B using orthology, but with a different partner B_2 using DiffPALM, we measure the Hamming distance between B_1 and B_2. A similar protocol is conducted for each sequence B_1 in family B. These distances allow us to compare the pairs predicted by DiffPALM to the orthology-based pairs. Note that the total counts of the distributions regarding MSA A and MSA B generally differ. This happens because DiffPALM might pair orthologs to padding sequences of gaps: in this case, we do not report Hamming distances.

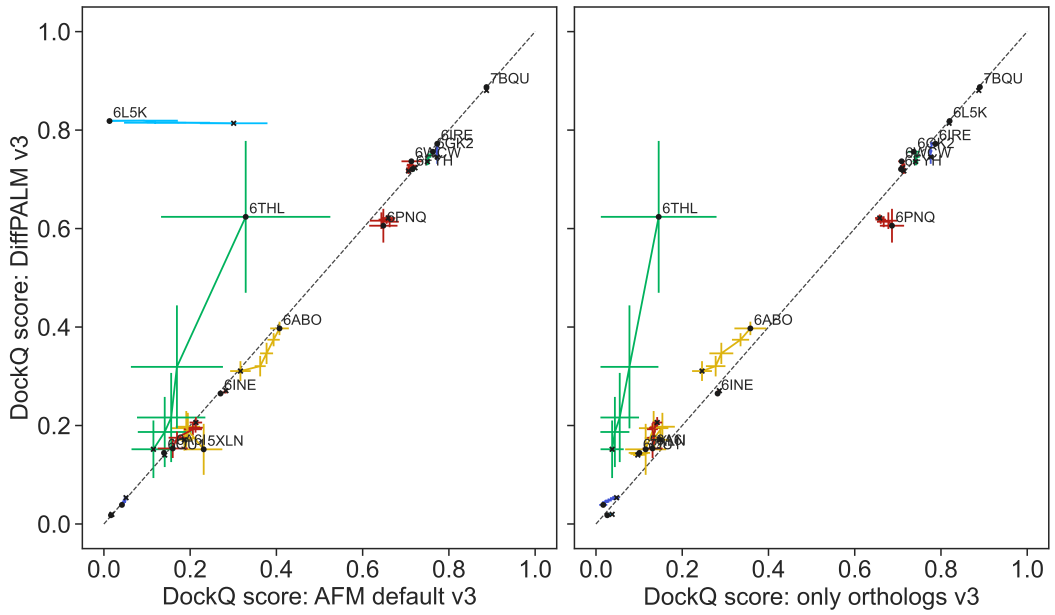

Figure 8: **Performance of structure prediction by AFM-v3 using different MSA pairing methods.** We report the performance of AFM-v3, in terms of DockQ scores, for the 15 complexes listed in Tab. 1, using three different pairing methods on the same initial unpaired MSAs. Left panel: DiffPALM versus default AFM pairing. Right panel: DiffPALM versus only pairing orthologs to the two query sequences. As in Fig. 2, for each complex, AFM is run five times, and the five top predicted structures by AFM confidence are considered each time, yielding 25 predicted structures total. For each complex, we show "trajectories" of performance starting from the top-confidence predicted structure (black circular marker) and ending with all predicted structures up to and including the fifth one (black cross marker). Results are averaged over the 5 runs and standard errors are shown as error bars. Points with DockQ below 0.1 are not labelled with their PDB ID for graphical reasons. Note that Fig. 2 restricts to those complexes where any two of these three pairing methods yield a significant difference ($> 10\%$) in average DockQ scores, among those shown here and listed in Tab. 1.

B METHODS

B.1 THE PARALOG MATCHING PROBLEM

Goal and notations. Paralog matching amounts to pairing a pair of MSAs, each one corresponding to one of the two protein families considered. We assume that interactions are one-to-one. Let $\mathcal{M}^{(\mathrm{A})}$ and $\mathcal{M}^{(\mathrm{B})}$ be the (single-chain) MSAs of two interacting protein families A and B, and let K denote the number of species represented in both MSAs and comprising more than one unmatched sequence in at least one MSA. Species represented in only one MSA are discarded since no within-species matching is possible for them. Species with only one unmatched sequence in each MSA are not considered further since pairing is trivial. There may also be N_{pos} known interacting pairs: they are treated separately, as positive examples (see below). Here we focus on the unmatched sequences. For $k = 1, \dots, K$, let $N_k^{(\mathrm{A})}$ and $N_k^{(\mathrm{B})}$ denote the number of unmatched sequences belonging to species k in $\mathcal{M}^{(\mathrm{A})}$ and $\mathcal{M}^{(\mathrm{B})}$ (respectively).

Dealing with asymmetric cases. The two protein families considered may have different numbers of paralogs within the same species. Assume, without loss of generality, that $N_k^{(\mathrm{A})} < N_k^{(\mathrm{B})}$ for a given k. To solve the matching problem with one-to-one interactions, we would like to pick, for each of the $N_k^{(\mathrm{A})}$ sequences in $\mathcal{M}^{(\mathrm{A})}$, a single and exclusive interaction partner out of the $N_k^{(\mathrm{B})}$ available sequences in $\mathcal{M}^{(\mathrm{B})}$. The remaining sequences of the species in $\mathcal{M}^{(\mathrm{B})}$ are left unpaired. In practice, we achieve this by augmenting the original set of species-k sequences from $\mathcal{M}^{(\mathrm{A})}$ with $N_k^{(\mathrm{B})} - N_k^{(\mathrm{A})}$ “padding sequences” made entirely of gap symbols. By doing so (and analogously when $N_k^{(\mathrm{A})} > N_k^{(\mathrm{B})}$), the thus-augmented interacting MSAs have the same number $N_k := \max(N_k^{(\mathrm{A})}, N_k^{(\mathrm{B})})$ of sequences from each species k. In practice, this method is used for the AFM complex structure prediction, while the curated benchmark prokaryotic MSAs do not have asymmetries (see B.7).

Formalization. The paralog matching problem corresponds to finding, within each species k, a mapping that associates one sequence of $\mathcal{M}^{(\mathrm{A})}$ to one sequence of $\mathcal{M}^{(\mathrm{B})}$ (and reciprocally). Thus, within each species k, one-to-one matchings can be encoded as permutation matrices of size $N_k \times N_k$. A brute-force search through all possible within-species one-to-one matchings would scale factorially with the size N_k of each species, making it prohibitive. Note that the Iterative Pairing Algorithm (IPA) (Bitbol et al., 2016; Bitbol, 2018) is an approximate method to solve this problem when optimizing coevolution scores. Here, we introduce another one, which allows to leverage the power of deep learning.

Exploiting known interacting partners. Our use of a language model allows for contextual conditioning, a common technique in natural language processing. Indeed, if any correctly paired sequences are already known, they can be included as part of the joint MSA input to MSA Transformer. In this case, we exclude their pairing from the optimization process – in particular, by not masking any of their amino acids, see below. We call these known paired sequences “positive examples”. In Fig. 5, we randomly sampled species and included all their pairs as positive examples, until we reached the desired depth $N_{\mathrm{pos}} \pm 10\%$. For eukaryotic complex structure prediction, we treated the query sequence pair as a positive example.

B.2 DIFFPALM: PARALOG MATCHING BASED ON MLM

Here, we explain our paralog matching method based on MLM, which we call DiffPALM. Background information on MSA Transformer and its MLM loss is collected in B.5. DiffPALM exploits our differentiable framework for optimizing matchings, see B.6. The key steps are summarized in Fig. 9.

Construction of an appropriate MLM loss. Using the tools just described, we consider two interacting MSAs (possibly augmented with padding sequences), still denoted by $\mathcal{M}^{(\mathrm{A})}$ and $\mathcal{M}^{(\mathrm{B})}$. Given species indexed by $k = 1, \dots, K$, we initialize a set $\{X_k\}_{k=1,\dots,K}$ of square matrices of size $N_k \times N_k$ (the case $K = 1$ corresponds to X in B.6). We call these “parameterization matrices”. By applying to them the matching operator M (see 2), we obtain the permutation matrices

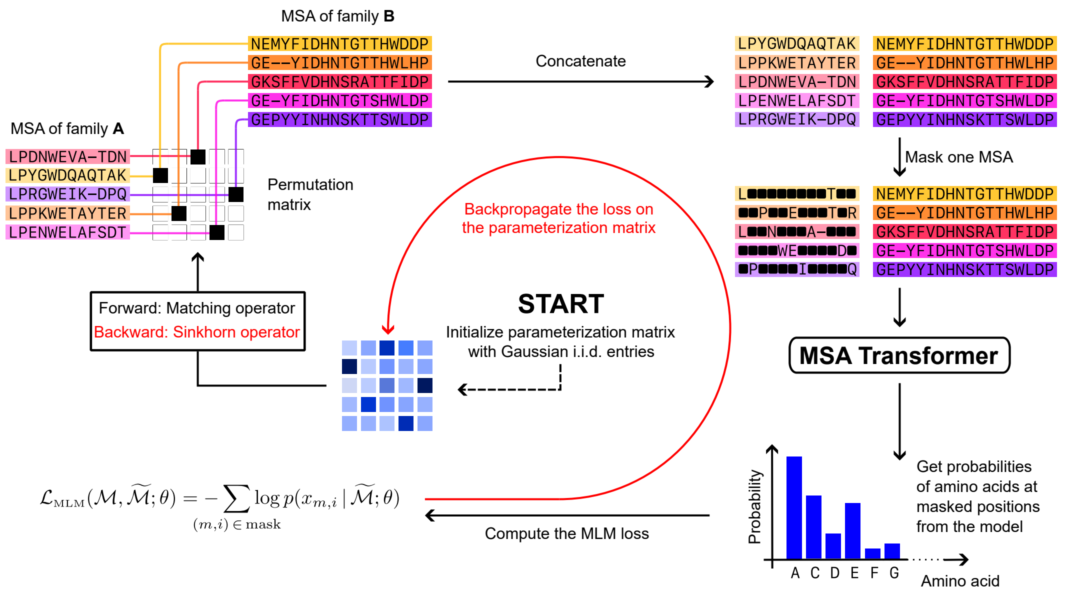
MSA of family B
NEMYFIDHNTGTTHWDDP
GE--YIDHNTGTTHWLHP
GKSFFVDHNSRATTFIDP
GE-YFIDHNTGTSHWLDP
GEPYYINHNSKTTSWLDP
MSA of family A
LPDNWEVA-TDN
LPYGWDQAQTAK
LPRGWEIK-DPQ
LPPKWETAYTER
LPENWELAFSDT
Permutation matrix
Forward: Matching operator
Backward: Sinkhorn operator
Concatenate
Backpropagate the loss on the parameterization matrix
START
Initialize parameterization matrix with Gaussian i.i.d. entries
Mask one MSA
MSA Transformer
Get probabilities of amino acids at masked positions from the model
Probability
Amino acid
A C D E F G
Compute the MLM loss
$\mathcal{L}_{\text{MLM}}(\mathcal{M}, \widetilde{\mathcal{M}}; \theta) = -\sum_{(m,i)\in\text{mask}} \log p(x_{m,i} \,|\, \widetilde{\mathcal{M}}; \theta)$

Figure 9: **Schematic of the DiffPALM method.** First, the parameterization matrices X_k are initialized, and then the following steps are repeated until the loss converges: (1) Compute the permutation matrix $M(X_k)$ and use it to shuffle $\mathcal{M}^{(\mathrm{A})}$ relative to $\mathcal{M}^{(\mathrm{B})}$. Then pair the two MSAs. (2) Randomly mask some tokens of one of the two sides of the paired MSA and compute the MLM loss 1. (3) Backpropagate the loss and update the parameterization matrices X_k, using the Sinkhorn operator $\hat{S}$ for the backward step instead of the matching operator M (see B.6).

$\{M(X_k)\}_{k=1,\dots,K}$, encoding matchings within each species in the paired MSA. Using gradient methods, we optimize the parameterization matrices so that the corresponding permutation matrices yield a paired MSA with low MLM loss. More precisely, paired MSAs are represented as concatenated MSAs with interacting partners placed on the same row,[1] and our MLM loss for this optimization is computed as follows:

1. Perform a shuffle of $\mathcal{M}^{(\mathrm{A})}$ relative to $\mathcal{M}^{(\mathrm{B})}$ using the permutation matrix $M(X_k)$ in each species k (plus an optional noise term, see below), to obtain a paired MSA $\mathcal{M}$;
2. Generate a mask for $\mathcal{M}$ (excluding any positive example tokens from the masking);
3. Compute MSA Transformer's MLM loss for that mask, see 1.

Importantly, we only mask tokens from one of the two MSAs, chosen uniformly at random within that MSA with a high masking probability $p \geq 0.7$.[2] Our rationale for using large masking probabilities is that it forces the model to predict masked residues in one of the two MSAs by using information coming mostly from the other MSA – see Fig. 4. We stress that, if padding sequences consisting entirely of gaps are present (see above), we mask these symbols with the same probability as those coming from ordinary sequences. Of the two MSAs to pair, we mask the one with shorter length if no padding sequences exist (i.e. here for our prokaryotic benchmark datasets). Else, if lengths are comparable but one MSA contains considerably more padding sequences than the other, we preferentially mask that MSA. Otherwise, we randomly choose which of the two MSAs to mask.

We fine-tuned all the hyperparameters involved in our algorithm using two joint MSAs of depth $\sim$50, constructed by selecting all the sequences of randomly sampled species from the HK-RR dataset (see B.7).

Noise and regularization. Following (Mena et al., 2018), after updating (or initializing) each X_k, we add to it a noise term given by a matrix of standard i.i.d. Gumbel noise multiplied by a scale

[1]We employ no special tokens to demarcate the boundary between one sequence and its partner.

[2]In contrast, uniformly random masking with $p = 0.15$ was used during MSA Transformer's pre-training (Rao et al., 2021b).

factor. The addition of noise ensures that the X_k do not get stuck at degenerate values for the right-hand side of 2, and more generally may encourage the algorithm to explore larger regions in the space of permutations. As scale factor for this noise we choose 0.1 times the sample standard deviation of the current entries of X_k, times a global factor tied to the optimizer scheduler (see next paragraph). Finally, since the matching operator is scale-invariant, we can regularize the matrices X_k to have small Frobenius norm. We find this to be beneficial and implement it through weight decay, set to be $w = 0.1$.

Optimization. We backpropagate the MLM loss on the parameterization matrices X_k. After each gradient step, we mean-center the parameterization matrices. We use the AdaDelta optimizer (Zeiler, 2012) with an initial learning rate $\gamma = 9$ and a "reduce on loss plateau" learning rate scheduler which decreases the learning rate by a factor of 0.8 if the loss has not decreased for more than 20 gradient steps after the learning rate was last set. The learning rate scheduler also provides the global scale factor which, together with the standard deviation of the entries of X_k, dynamically determines the magnitude of the Gumbel noise (see previous paragraph).

Exploring the loss landscape through multiple initializations. We observe that the initial choice of the parameterization set $\{X_k\}_{k=1,\ldots,K}$ strongly impact results. Slightly different initial conditions for X_k lead to very different final permutation matrices. Furthermore, we observe fast decrease in the loss when the X_k are initialized to be exactly zero (our use of Gumbel noise means that we break ties randomly when computing the permutation matrices $M(X_k)$; if noise is not used, similar results can be achieved by initializing X_k with entries very close to zero). Thus, we can cheaply probe different paths in the loss landscape by performing several short runs using zero-initialized parameterization matrices X_k. In practice, we use 20 different such short runs each consisting of 20 gradient steps. Then, we average all the final parameterizations together to warm-start a longer run made up of 400 gradient steps.

Result and confidence. We observe that, even though the loss generally converges to a minimum average value during our optimization runs, there are often several distinct hard permutations associated to the smallest loss values. This may indicate a flattening of the loss landscape relative to the inherent fluctuations in the MLM loss, and/or the existence of multiple local minima. To extract a single matching per species from one of our runs (or indeed from several runs, see B.2), we average the hard permutation matrices associated to the q lowest losses, and evaluate the matching operator [2] on the resulting averages. We find final precision-100 figures to be quite robust to the choice of q. On the other hand, for individual (warm-started) runs as described in B.2, precision-10 benefits from setting q to its maximum possible value of 400.

Furthermore, we propose using each entry in the averaged permutation matrices as an indicator of the model's confidence in the matching of the corresponding pair of sequences. Indeed, pairs that are present in most low-loss configurations are presumably essential for the optimization process and are captured most of the times, pushing their confidence value close to 1. Conversely, non-interacting pairs are in most of the cases associated to higher losses and therefore appear sporadically, obtaining confidences close to zero. Accordingly, we refer to the averaged hard permutations used to extract a single matching per species as "confidence matrices", and to the final in-species matchings as "consensus permutations".

Improving precision: MRA and IPA. We propose two methods for improving precision further. In the first method, which we call Multi-Run Aggregation (MRA), we perform N_{runs} independent optimization runs for each interacting MSA. Then, we collect together the hard permutations independently obtained from each run, and aggregate the $q = 400$ lowest-loss permutations from this larger collection to obtain more reliable confidence matrices and hard permutations.

The second method is an iterative procedure analogous to the Iterative Pairing Algorithm (IPA) of Refs. (Bitbol et al., 2016; Bitbol, 2018), and named after it. The idea is to gradually add pairs with high confidence as positive examples. Assuming a paired MSA containing a single species for notational simplicity, the n-th iteration (starting at $n = 1$) involves the following steps:

1. Perform an optimization run and extract from it a confidence matrix $C^{(n)}$ as described in B.2, using the currently available positive examples;

2. Compute the moving average $\tilde{C}^{(n)} = \text{mean}(C^{(n)}, \tilde{C}^{(n-1)}, \ldots, \tilde{C}^{(1)})$ (where $\tilde{C}^{(1)} \equiv C^{(1)}$);
3. Define candidate matchings via the consensus permutation $M^{(n)} = M(\tilde{C}^{(n)})$;
4. Repeat Steps 1-3 a maximum of 3 times, until the average MLM loss estimated using $M^{(n)}$, and 200 random masks, is lower or statistically insignificantly higher (based on a two-sample T-test: we say "statistically insignificantly" when $p \geq 0.95$, and "statistically significantly" when $p < 0.05$) than what could have been obtained using $M^{(n-1)}$ and the same positive examples as in Step 1;
5. If Step 4 fails, set $\tilde{C}^{(n)} = \tilde{C}^{(n-1)}$ and $M^{(n)} = M(\tilde{C}^{(n-1)})$ (but removing rows and columns corresponding to the positive examples added at iteration $n-1$);
6. Check that the average MLM loss estimated using $M^{(n)}$ and 200 random masks, but only regarding as positive examples those available at the beginning of iteration $n-1$, is not statistically significantly higher than the average MLM loss estimated using $M^{(n-1)}$ and those same positive examples;
7. If Step 6 fails, terminate the IPA. Otherwise, pick the top 5 pairs according to $\tilde{C}^{(n)}$, promote them to positive examples in all subsequent iterations, and remove them from the portion of the paired MSA to be optimized.

If several species are present, they are optimized together (see B.2) and confidence values from all species are used to select the top 5 pairs.

B.3 Pairing based on a single-sequence language model

To assess whether a single-sequence model is able to solve the paralog matching problem, we consider the 650M-parameter version of the model ESM-2 (Lin et al., 2023). We score candidate paired sequences using the MLM loss in 1. In contrast with MSA Transformer, the input of the model is not paired MSAs but single paired sequences. Therefore, it is sufficient to individually score each possible pair within each species, without needing to consider all permutations. Denoting by N_k the number of sequences from each family in species k, the number of possible pairs is N_k^2 while the number of permutations is $N_k!$. This complexity reduction allows us to evaluate the scores of all possible pairs. This removes the need of backpropagating the loss on the permutation matrix. Accordingly, this method is much faster, since we only need to use the model in evaluation mode, without gradient backpropagation.

For each candidate paired sequence, we evaluate the average of the MLM losses computed over multiple random masks (with masking probability p). Once the average MLM losses are computed for all the N_k^2 pairs, we compute the optimal one-to-one matching by using standard algorithms for linear assignment problems (Kuhn, 1955) on the $N_k \times N_k$ matrix containing all the losses.

B.4 Assessing the impact of pairing on AFM structure prediction

Pairing methods employed in AFM and ColabFold. When presented with a set of query chains, AFM retrieves homologs of each of the chains by running JackHMMER (Johnson et al., 2010) on UniProt, and further homology searches on other databases (Evans et al., 2021). UniProt hits are partitioned into species (while the official AFM implementation in `https://github.com/deepmind/alphafold` uses UniProt "entry names" to define species, when possible we instead use NCBI TaxIDs (via UniProt mappings to NCBI tax IDs, retrieved on 17 Dec 2022, corresponding to UniProt Knowledgebase release 2022_04), which are more accurate) and ranked within each species by decreasing Hamming distance to the relevant query sequence. A paired MSA is obtained by matching equal-rank hits. Sequences left unpaired are discarded. In addition, AFM produces "block MSAs" constructed by "pairing" hits from the remaining databases with padding sequences of gaps. The input for AFM comprises the paired MSA and the block MSAs.

While sharing the same architecture and weights as AFM, ColabFold retrieves homologs using MMseqs2 (Steinegger & Söding, 2017) on ColabFoldDB (Mirdita et al., 2022). In each species, hits are sorted by increasing E-value, and the best hits are paired (Zeng et al., 2018; Cong et al., 2019; Green et al., 2021; Pozzati et al., 2021; Bryant et al., 2022; Mirdita et al., 2022). Thus, contrary to the default AFM pipeline, the paired MSA in ColabFold contains at most one sequence pair per species for a heterodimer. Because the databases and homology search methods used by ColabFold differ from those used by AFM, a direct comparison does not allow one to isolate the effect of their different pairing schemes. Therefore, we employed the ColabFold pairing method starting from the sequences that are paired in the default AFM pipeline.

Pairing using DiffPALM. To assess the impact of DiffPALM on complex structure prediction by AFM, we started from the sequences that are paired in the default AFM pipeline. We left out species with large unbalances between the number of sequences in the two families considered. Specifically, if the ratio of the larger to the smaller of these two numbers exceeds an ad-hoc "maximum size ratio" MSR (see Tab. 1), if there is only one sequence in both families, or if there are more than 50 sequences in at least one family, then we do not attempt pairing via DiffPALM, and revert to default AFM pairing. When the full MSA to be paired with DiffPALM is too deep and/or long, optimizing it as a whole is not possible due to GPU memory limitations. Instead, we partition it into several small enough sub-MSAs, which we optimize independently. Note that this may reduce performance, since pairing quality increases with MSA depth. We always use the query sequences as positive examples. For all these optimization runs, we use a masking probability $p = 0.7$.

B.5 MSA Transformer and masked language modeling for MSAs

We use the MSA Transformer model (Rao et al., 2021b), which takes MSAs as inputs and was trained with a variant of the masked language modeling (MLM) objective (Devlin et al., 2019) on a training set of 26 million MSAs constructed from UniRef50 clusters. The model's training objective was to correctly predict the identity of randomly masked residue positions in the MSAs in its training set. Specifically, it was trained to minimize an MLM loss, which reads, for an MSA $\mathcal{M}$, and its masked version $\widetilde{\mathcal{M}}$:

$$\mathcal{L}_{\text{MLM}}(\mathcal{M}, \widetilde{\mathcal{M}}; \theta) = -\sum_{(m,i)\,\in\,\text{mask}} \log p(x_{m,i} \,|\, \widetilde{\mathcal{M}}; \theta)\,. \tag{1}$$

Here, $x_{m,i}$ denotes the amino acid at the i-th residue position (column) in the m-th sequence (row) of $\mathcal{M}$, while θ stands for all the model parameters. At each residue position in the input MSA, MSA Transformer outputs a probability for each of the 21 possible amino-acid and gap symbols, and $p(x_{m,i} \,|\, \widetilde{\mathcal{M}}; \theta)$ in 1 is the probability associated with the correct residue $x_{m,i}$ at MSA position (m, i). MSA Transformer's architecture interleaves multi-headed (tied) row attention blocks and (untied) column attention blocks, over several layers. Therefore, the accessible context for a masked residue consists not only of amino acids at different positions along the same sequence, but also of amino acids from other sequences (Rao et al., 2021b; Lupo et al., 2022). This allows the model to capture coevolution information. After pre-training, each term in the right-hand side of 1 can be interpreted as the model's estimate of the (negative) log-likelihood of the amino acid $x_{m,i}$ at a masked position (m, i) (Wang & Cho, 2019; Goyal et al., 2021; Rao et al., 2021a).

B.6 A differentiable formulation of paralog matching

We formulate a differentiable optimization problem that can be more efficiently solved than the brute-force search, using gradient methods. The goal is to obtain sets of within-species matchings (and thus permutations) that minimize our MLM loss.

The set $\mathcal{P}_N$ of permutation matrices of N objects can be parameterized exactly by square matrices X via the *matching operator*

$$M(X) = \underset{P\in\mathcal{P}_N}{\arg\max}[\text{trace}(P^{\text{T}}X)], \tag{2}$$

which can be computed using standard non-differentiable algorithms for linear assignment problems (Kuhn, 1955).[3]

We exploit the fact, shown in (Mena et al., 2018), that permutation matrices can be approximated arbitrarily well by using the *Sinkhorn operator* S, which is defined on square matrices X as follows:

$$S(X) = \lim_{l\to\infty} S^l(X), \quad \text{where} \quad S^l(X) = (\mathcal{T}_{\text{c}} \circ \mathcal{T}_{\text{r}})^l(\exp(X)), \tag{3}$$

[3]More precisely, the right-hand side of 2 has a unique solution for almost all X (Mena et al., 2018).

$\mathcal{T}_c$ and $\mathcal{T}_r$ are the row- and column-wise normalization operators, and $\exp$ denotes the component-wise matrix exponential.[4] More precisely, $M(X) = \lim_{\tau \to 0^+} S(X/\tau)$ for almost all X (Mena et al., 2018, Theorem 1). Hence, by choosing a suitably small value of τ, and using S^l [3] instead of S for a suitably large l, we can define a smooth mapping $\hat{S}(X) = S^l(X/\tau)$ which sends arbitrary square matrices to "soft permutations" approximating *bona fide* ("hard") permutations. In practice, we use $\tau = 1$ and $l = 10$.

Applying general soft permutations directly on an MSA (after one-hot encoding its residues) yields a dataset consisting of "amino acid mixtures" at each MSA position. Such datasets are out of distribution relative to MSA Transformer's pre-training since it was trained on single amino acid embeddings, not mixtures of them. Besides, we wish to optimize for an MLM loss defined on realistic MSAs. Therefore, in order to be able to backpropagate through $\hat{S}$, while also evaluating MLM losses only on MSAs shuffled by hard permutations, we compute the full matching operator M [2] in the forward pass, but propagate gradients backwards through $\hat{S}$ alone.[5]

B.7 Datasets

Benchmark prokaryotic datasets. We developed and tested DiffPALM using joint MSAs extracted from two datasets. The first dataset is composed of 23,632 cognate pairs of histidine kinases (HK) and response regulators (RR) from the P2CS database (Barakat et al., 2009; 2011), paired using genome proximity, and previously described in (Bitbol et al., 2016; Bitbol, 2018). The average number of pairs per species in this dataset is 10.23 (standard deviation: 7.85).

The second dataset consists of 17,950 ABC transporter protein pairs, homologous to the *Escherichia coli* MALG-MALK pair of maltose and maltodextrin transporters, also paired using genome proximity (Ovchinnikov et al., 2014; Bitbol et al., 2016). The average number of pairs per species in this dataset is 5.68 (standard deviation: 5.60). We also considered a similarly constructed dataset of 220 pairs homologous to the *Escherichia coli* NUOA-NUOJ pair of NADH-quinone oxidoreductase subunits, with an average number of pairs per species of 2.04.

Throughout, our focus is on pairing interaction partners among paralogs within each species. In all these benchmark datasets, species comprising only one pair of sequences were discarded. Indeed, pairing is trivial in these cases.

Out of each of these benchmark datasets of known interacting pairs, we consider paired MSAs of depth $\sim$50 (resp. $\sim$100 or $\sim$200), constructed by selecting all the sequences of randomly sampled species from the full dataset. Specifically, for a target MSA depth $\overline{D} = 50$, 100 or 200, we add randomly sampled complete species one by one; if the first m species (but no fewer) give an MSA depth $D \geq 0.9\overline{D}$, and the first $n \geq m$ species (but no more) give $D \leq 1.1\overline{D}$, then we select the first k species in our final MSA, with k picked uniformly at random between m and n. For such shallow MSAs, existing coevolution-based methods do not perform well. Note also that MSA Transformer's large memory footprint constrains the depth and length of input MSAs. Concretely, in our GPU (NVIDIA RTX A6000 with 48 GB of memory) it is possible to backpropagate the gradients for an input that has up to $\sim$40,000 tokens, i.e. $\sim$200 sequences of length $\sim$200 amino acids.

Eukaryotic complexes. We considered targets whose structures are not in the training set of AFM with v2 weights, and where default AFM predictions are poor. Specifically, we started from those eukaryotic targets from Table A1 of (Chen et al., 2023) and from the "Benchmark 2" dataset in (Evans et al., 2021) whose PDB structures were released after the training cutoff for the AFM v2 weights (April 30, 2018). Among those, we focused on multimers with no more than 2 different types of monomers, where both monomers come from the same species, and with paired sequences not longer than 500 amino acids, due to GPU memory constraints. Finally, we further restricted to the 15 targets with default AFM predictions yielding the poorest reported DockQ score. They are

[4]That is, S^l consists of applying $\exp$ and then iteratively normalizing rows and columns l times.

[5]See (Norn et al., 2021) for a similar use of "gradient bypassing" in the context of protein design. We write the hard permutation as $[M(X) - \hat{S}(X)] + \hat{S}(X)$, and halt gradient backpropagation through the term in square brackets. Schematically, let L denote the operator which takes an MSA as input, randomly masks it, passes the masked MSA to MSA Transformer, and finally computes the MLM loss. Then, if A is the MSA whose rows we wish to permute, we use $L'(M(X)A)\,S'(X)A$ instead of $L'(M(X)A)\,M'(X)A$ as our "gradient".

listed in Tab. 1. All of them are heterodimers, except 6ABO which is a heterotetramer complex made of two IFFO1 and two XRCC4 molecules. Note that we also show results for AFM with v3 weights (latest release, training cutoff September 30, 2021) in Fig. 8.

B.8 General points on AFM

For all structure prediction tasks, we use the five pre-trained AFM models with v2 weights (Evans et al., 2021), except in Fig. 8 where v3 weights are used. We use full genomic databases and code from release v2.3.1 of the official implementation in `https://github.com/deepmind/alphafold`. We use no structural templates, and perform 3 recycles for each structure, without early stopping. We relax all models using AMBER.

When using all pairing methods (default AFM, DiffPALM, orthology-based), we also retained the block MSAs retrieved by the default AFM pipeline (Bryant et al., 2022).

The AFM confidence score is defined as $0.8 \cdot \mathrm{iptm} + 0.2 \cdot \mathrm{ptm}$, where iptm is the predicted TM-score in the interface, and ptm the predicted TM-score of the entire complex (Evans et al., 2021).