EQUIVARIANT SCALAR FIELDS FOR MOLECULAR DOCKING WITH FAST FOURIER TRANSFORMS

Bowen Jing[1], **Tommi Jaakkola**[1], **Bonnie Berger**[1 2]
[1]CSAIL, Massachusetts Institute of Technology
[2]Dept. of Mathematics, Massachusetts Institute of Technology
bjing@mit.edu, {tommi, bab}@csail.mit.edu

ABSTRACT

Molecular docking is critical to structure-based virtual screening, yet the throughput of such workflows is limited by the expensive optimization of scoring functions involved in most docking algorithms. We explore how machine learning can accelerate this process by learning a scoring function with a functional form that allows for more rapid optimization. Specifically, we define the scoring function to be the cross-correlation of multi-channel ligand and protein scalar fields parameterized by equivariant graph neural networks, enabling rapid optimization over rigid-body degrees of freedom with fast Fourier transforms. The runtime of our approach can be amortized at several levels of abstraction, and is particularly favorable for virtual screening settings with a common binding pocket. We benchmark our scoring functions on two simplified docking-related tasks: decoy pose scoring and rigid conformer docking. Our method attains similar but faster performance on crystal structures compared to the widely-used Vina and Gnina scoring functions, and is more robust on computationally predicted structures. Code is available at https://github.com/bjing2016/scalar-fields.

1 INTRODUCTION

Proteins are the macromolecular machines that drive almost all biological processes, and much of early-stage drug discovery focuses on finding molecules which bind to and modulate their activity. *Molecular docking*—the computational task of predicting the binding pose of a small molecule to a protein target—is an important step in this pipeline. Traditionally, molecular docking has been formulated as an optimization problem over a *scoring function* designed to be a computational proxy for the free energy (Torres et al., 2019; Fan et al., 2019). Such scoring functions are typically a sum of pairwise interaction terms between atoms with physically-inspired functional forms and empirically tuned weights (Quiroga & Villarreal, 2016). While these terms are simple and hence fast to evaluate, exhaustive sampling or optimization over the space of ligand poses is difficult and leads to the significant runtime of docking software.

ML-based scoring functions for docking have been an active area of research, ranging in sophistication from random forests to deep neural networks (Yang et al., 2022; Crampon et al., 2022). These efforts have largely sought to more accurately model the free energy based on a docked pose, which is important for downstream identification of binders versus non-binders (*virtual screening*). However, they have not addressed nor reduced the computational cost required to produce these poses in the first place. Hence, independently of the accuracy of these workflows, molecular docking for large-scale structure-based virtual screening remains computationally challenging, especially with the growing availability of large billion-compound databases such as ZINC (Tingle et al., 2023).

In this work, we explore a different paradigm and motivation for machine learning scoring functions, with the specific aim of *accelerating scoring and optimization* of ligand poses for high-throughput molecular docking. To do so, we forego the physics-inspired functional form of commonly used scoring functions, and instead frame the problem as that of learning *scalar fields* independently associated with the 3D structure of the protein and ligand, respectively. We then define the score to be the cross-correlation between the overlapping scalar fields when oriented according to the ligand pose. While seemingly more complex than existing scoring functions, these cross-correlations can

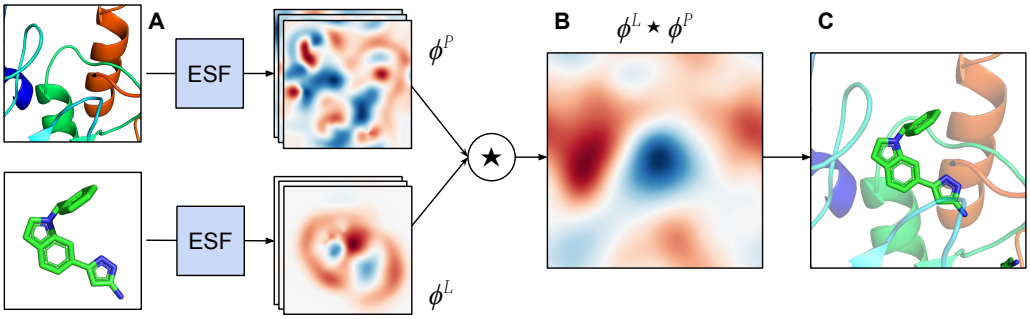

Figure 1: **Overview** of the scalar field-based scoring function and docking procedure. The translational FFT procedure is shown here; the rotational FFT is similar, albeit harder to visualize. (A) The protein pocket and ligand conformer are independently passed through equivariant scalar field networks (ESFs) to produce scalar fields. (B) The fields are cross-correlated to produce heatmaps over ligand translations. (C) The ligand coordinates are translated to the argmax of the heatmap. Additional scalar field visualizations are in Appendix C.

be rapidly evaluated over a large number of ligand poses simultaneously using Fast Fourier Transforms (FFT) over both the translational space $\mathbb{R}^3$ and the rotational space $SO(3)$. This property allows for significant speedups in the optimization over these degrees of freedom.

Scalar fields representing molecules in 3D space have been previously parameterized as *neural fields* (Zhong et al., 2019), i.e., neural networks taking coordinates of a query point as input and producing field values as output. However, our scalar fields must be defined relative to the ligand (or protein) structure—represented as a graph embedded in 3D space—and be $SE(3)$ *equivariant* in order to yield an invariant scoring function. To satisfy these requirements, we introduce *equivariant scalar field* networks (ESFs), which parameterize the scalar fields with message passing E3NNs (Thomas et al., 2018; Geiger & Smidt, 2022). Unlike neural fields, these networks yield a compact representation of the scalar field in a single forward pass and are automatically $SE(3)$-equivariant.

Contrasting with existing ML scoring functions, the computational cost of our method can be *amortized* at several levels of abstraction, significantly accelerating runtimes for optimized workflows. For example, unlike methods that require one neural network forward pass per pose, our network is evaluated once per protein structure or ligand conformer *independently*. Post-amortization, we attain translational and rotational optimization runtimes as fast as 160 μs and 650 μs, respectively, with FFTs. Such throughputs, when combined with effective sampling and optimization, could make docking of very large compound libraries feasible with only modest resources.

Empirically, we evaluate our method on two simplified docking-related tasks: (1) decoy pose scoring and (2) rigid conformer docking. On both tasks, our scoring function is competitive with—but faster than—the scoring functions of Gnina (Ragoza et al., 2017; McNutt et al., 2021) and Vina (Trott & Olson, 2010) on PDBBind crystal structures and is significantly better on ESMFold structures. We then demonstrate the further advantages of runtime amortization on the virtual screening-like setup of the PDE10A test set (Tosstorff et al., 2022), where—since there is only one unique protein structure—our method obtains a 50x speedup in total inference time at no loss of accuracy.

To summarize, our contributions are:

- We are the first to propose *learning* protein-ligand scoring functions based on cross-correlations of scalar fields in order to accelerate pose optimization in molecular docking.
- We formulate a neural network parameterization and training procedure for learning *equivariant scalar fields* for molecules.
- We demonstrate that the performance and runtime of our scoring function holds promise when evaluated on docking-related tasks.

2 BACKGROUND

Molecular docking. The two key components of a molecular docking algorithm are (1) one or more scoring functions for ligand poses, and (2) a search, sampling, or optimization procedure. There is considerable variation in the design of these components and how they interact with each other, ranging from exhaustive enumeration and filtering (Shoichet et al., 1992; Meng et al., 1992) to genetic, gradient-based, or MCMC optimization algorithms (Trott & Olson, 2010; Morris et al., 1998; McNutt et al., 2021). We refer to reviews elsewhere (Ferreira et al., 2015; Torres et al., 2019; Fan et al., 2019) for comprehensive details. These algorithms have undergone decades of development and have been given rise to well-established software packages in academia and industry, such as AutoDock (Morris & Lim-Wilby, 2008), Vina (Trott & Olson, 2010) and Glide (Halgren et al., 2004). In many of these, the scoring function is designed not only to identify the binding pose, but also to predict the binding affinity or activity of the ligand (Su et al., 2018). In this work, however, we focus on learning and evaluating scoring functions for the rapid prediction of binding poses.

ML methods in docking. For over a decade, ML methods have been extensively explored to improve scoring functions for already-docked ligand poses, i.e., for prediction of activity and affinity in structural-based virtual screens (Li et al., 2021; Yang et al., 2022; Crampon et al., 2022). On the other hand, developing ML scoring functions as the direct optimization objective has required more care due the enormous number of function evaluations involved. MedusaNet (Jiang et al., 2020) and Gnina (Ragoza et al., 2017; McNutt et al., 2021) proposed to sparsely use CNNs for guidance and re-ranking (respectively) in combination with a traditional scoring function. DeepDock (Méndez-Lucio et al., 2021) used a hypernetwork to predict complex-specific parameters of a simple statistical potential. Recently, geometric deep learning models have explored entirely different paradigms for docking via direct prediction of the binding pose (Stärk et al., 2022; Zhang et al., 2022; Lu et al., 2022) or via a generative model over ligand poses (Corso et al., 2023).

FFT methods in docking. Methods based on fast Fourier transforms have been widely applied for the related problem of *protein-protein docking*. Katchalski-Katzir et al. (1992) first proposed using FFTs over the translational space $\mathbb{R}^3$ to rapidly evaluate poses using scalar fields that encode the shape complementarity of the two proteins. Later works extended this method to rotational degrees of freedom (Ritchie & Kemp, 2000; Ritchie et al., 2008; Padhorny et al., 2016), electrostatic potentials and solvent accessibility (Gabb et al., 1997; Mandell et al., 2001; Chen & Weng, 2002), and data-driven potentials (Neveu et al., 2016). Today, FFT methods are a routine step in established protein-protein docking programs such as PIPER (Kozakov et al., 2006), ClusPro (Kozakov et al., 2017), and HDOCK (Yan et al., 2020), where they enable the evaluation of billions of poses, typically as an initial screening step before further evaluation with a more accurate scoring function.

In contrast, FFT methods have been significantly less studied for protein-ligand docking. While a few works have explored this direction (Padhorny et al., 2018; Ding et al., 2020; Nguyen et al., 2018), these algorithms have not been widely adopted nor been incorporated into established docking software. A key limitation is that protein-ligand scoring functions are typically more complicated than protein-protein scoring functions and cannot be easily expressed as a cross-correlation between scalar fields (Ding et al., 2020). To our knowledge, no prior works have explored the possibility of overcoming this limitation by *learning* cross-correlation based scoring functions.

3 METHOD

3.1 EQUIVARIANT SCALAR FIELDS

We consider the inputs to a molecular docking problem to be a pair of protein structure and ligand molecule, encoded as a featurized graphs G^P, G^L, and with the protein structure associated with alpha carbon coordinates $\mathbf{X}^P = [\mathbf{x}_1^P, \dots \mathbf{x}_{N_P}^P] \in \mathbb{R}^{3\times N_P}$. The molecular docking problem is to find the ligand atomic coordinates $\mathbf{X}^L = [\mathbf{x}_1^L, \dots \mathbf{x}_{N_L}^L] \in \mathbb{R}^{3\times N_L}$ of the true binding pose. To this end, our aim is to parameterize and learn (multi-channel) scalar fields $\phi^P := \phi(\mathbf{x}; G^P, \mathbf{X}^P)$ and $\phi^L := \phi^L(\mathbf{x}; G^L, \mathbf{X}^L)$ associated with the protein and ligand structures, respectively, such that the scoring function evaluated on any pose $\mathbf{X}^L \in \mathbb{R}^{3N_L}$ is given by

$$E(\mathbf{X}^P, \mathbf{X}^L) = \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}; G^P, \mathbf{X}^P)\phi_c^L(\mathbf{x}; G^L, \mathbf{X}^L)\, d^3\mathbf{x} \tag{1}$$

where ϕ_c refers to the cth channel of the scalar field. While neural fields that directly learn functions $\mathbb{R}^3 \to \mathbb{R}$ have been previously developed as encodings of molecular structures (Zhong et al., 2019), such a formulation is unsuitable here as the field must be defined relative to the variable-sized structure graphs G^P, G^L and transform appropriately with rigid-body motions of their coordinates.

Instead, we propose to parameterize the scalar field as a sum of contributions from each ligand atom or protein alpha-carbon, where each contribution is defined by its coefficients in a *spherical harmonic expansion* centered at that atom (or alpha-carbon) coordinate in 3D space. To do so, we choose a set $R_j : \mathbb{R}^+ \to \mathbb{R}$ of radial basis functions (e.g., Gaussian RBFs) in 1D and let Y_m^ℓ be the real spherical harmonics. Then we define

$$\phi_c(\mathbf{x}; G, \mathbf{X}) = \sum_{n,j,\ell,m} A_{cnj\ell m}(G, \mathbf{X}) R_j(\|\mathbf{x} - \mathbf{x}_n\|) Y_\ell^m \left(\frac{\mathbf{x} - \mathbf{x}_n}{\|\mathbf{x} - \mathbf{x}_n\|} \right) \tag{2}$$

where here (and elsewhere) we drop the superscripts L, P for common definitions. Given some constraints on how the vector of coefficients $A_{cnj\ell m}$ transforms under $SE(3)$, this parameterization of the scalar field satisfies the following important properties:

Proposition 1. *Suppose the scoring function is parameterized as in Equation 2 and for any $R \in SO(3), \mathbf{t} \in \mathbb{R}^3$ we have $A_{cnj\ell m}(G, R.\mathbf{X} + \mathbf{t}) = \sum_{m'} D^\ell_{mm'}(R) A_{cnj\ell m'}(G, \mathbf{X})$ where $D^\ell(R)$ are the (real) Wigner D-matrices, i.e., irreducible representations of $SO(3)$. Then for any $g \in SE(3)$,*

1. *The scalar field transforms equivariantly: $\phi_c(\mathbf{x}; G, g.\mathbf{X}) = \phi_c(g^{-1}.\mathbf{x}; G, \mathbf{X})$.*
2. *The scoring function is invariant: $E(g.\mathbf{X}^P, g.\mathbf{X}^L) = E(\mathbf{X}^P, \mathbf{X}^L)$.*

See Appendix A for the proof. We choose to parameterize $A_{cnj\ell m}(G, R.\mathbf{X})$ with E3NN graph neural networks (Thomas et al., 2018; Geiger & Smidt, 2022), which are specifically designed to satisfy these equivariance properties and produce all coefficients in a single forward pass. The core of our method consists of the training of two such equivariant scalar field networks (ESFs), one for the ligand and one for the protein, which then parameterize their respective scalar fields. While the second property (invariance of the scoring function) is technically the only one required by the problem symmetries, the first property ensures that different ligand poses related by rigid-body transformations can be evaluated via transformations of the scalar field itself (without re-evaluating the neural network) and is thus essential to our method.

Next, we show how this parameterization enables ligand poses related by rigid body motions to some reference pose to be rapidly evaluated with fast Fourier transforms (all derivations in Appendix A). There are actually two ways to do so: we can evaluate the score of all poses generated by *translations* of the reference pose, or via *rotations* around some fixed point (which we always choose to be the center of mass of ligand). These correspond to FFTs over $\mathbb{R}^3$ and $SO(3)$, respectively.

3.2 FFT Over Translations

We first consider the space of poses generated by translations. Given some reference pose $\mathbf{X}^L$, the score as a function of the translation is just the cross-correlation of the fields ϕ^L and ϕ^P:

$$E(\mathbf{X}^P, \mathbf{X}^L + \mathbf{t}) = \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}) \phi_c^L(\mathbf{x} - \mathbf{t}) \, d^3\mathbf{x} = \sum_c (\phi_c^L \star \phi_c^P)(\mathbf{t}) \tag{3}$$

where we have dropped the dependence on $G, \mathbf{X}$ for cleaner notation and applied Proposition 1. By the convolution theorem, these cross-correlations may be evaluated using Fourier transforms:

$$\phi_c^L \star \phi_c^P = \frac{1}{(2\pi)^{3/2}} \mathcal{F}^{-1} \left\{ \overline{\mathcal{F}\left[\phi_c^L\right]} \cdot \mathcal{F}\left[\phi_c^P\right] \right\} \tag{4}$$

Hence, in order to simultaneously evaluate all possible translations of the reference pose, we need to compute the Fourier transforms of the protein and ligand scalar fields. One naive way of doing so would be to explicitly evaluate Equation 2 at an evenly-spaced grid of points spanning the structure and then apply a fast Fourier transform. However, this would be too costly, especially during training time. Instead, we observe that the functional form allows us to immediately obtain the Fourier transform via the expansion coefficients $A_{cnj\ell m}$:

$$\mathcal{F}\left[\phi_c\right](\mathbf{k}) = \sum_n e^{-i\mathbf{k}\cdot\mathbf{x}_n} \sum_\ell (-i)^\ell \sum_{m,n} A_{cnj\ell m} \mathcal{H}_\ell[R_j](\|\mathbf{k}\|) Y_\ell^m\left(\mathbf{k}/\|\mathbf{k}\|\right) \tag{5}$$

where now Y_ℓ^m must refer to the complex spherical harmonics and the coefficients must be transformed correspondingly, and

$$\mathcal{H}_\ell[R_j](k) = \sqrt{\frac{2}{\pi}} \int_0^\infty j_\ell(kr) R_j(r) r^2 \, dr \tag{6}$$

is the ℓth order spherical Bessel transform of the radial basis functions. Conceptually, this expression corresponds to first evaluating the Fourier transform of each atom's scalar field in a coordinate system centered at its location, and then translating it via multiplication with $e^{-i\mathbf{k}\cdot\mathbf{x}_n}$ in Fourier space. Importantly, $\mathcal{H}_\ell[R_j]$ and Y_ℓ^m can be precomputed and cached at a grid of points *independently* of any specific structure, such that only the translation terms and expansion coefficients need to be computed for every new example.

3.3 FFT Over Rotations

We next consider the space of poses generated by rotations. Suppose that given some reference pose $\mathbf{X}^L$, the protein and ligand scalar fields are both expanded around some common coordinate system origin using the complex spherical harmonics and a set of *global radial basis functions* $S_j(r)$:

$$\phi_c(\mathbf{x}) = \sum_{j,\ell,m} B_{cj\ell m} S_j(\|\mathbf{x}\|) Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \tag{7}$$

We seek to simultaneously evaluate the score of poses generated via rigid rotations of the ligand, which (thanks again to Proposition 1) is given by the rotational cross-correlation

$$E(\mathbf{X}^P, R.\mathbf{X}^L) = \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}) \phi_c^L(R^{-1}\mathbf{x}) \, d^3\mathbf{x} \tag{8}$$

Cross-correlations of this form have been previously studied for rapid alignment of crystallographic densities (Kovacs & Wriggers, 2002) and of signals on the sphere in astrophysics (Wandelt & Górski, 2001). It turns out that they can also be evaluated in terms of Fourier sums:

$$\int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}) \phi_c^L(R^{-1}\mathbf{x}) \, d^3\mathbf{x} = \sum_{\ell,m,h,n} d_{mh}^\ell d_{hn}^\ell I_{mn}^\ell e^{i(m\xi + h\eta + n\omega)} \tag{9}$$

where ξ, η, ω are related to the the Euler angles of the rotation R, d^ℓ is the (constant) Wigner D-matrix for a rotation of $\pi/2$ around the y-axis, and

$$I_{mn}^\ell = \sum_{j,k} B_{cj\ell m}^P \overline{B_{ck\ell n}^L} G_{jk} \quad \text{where} \quad G_{jk} = \int_0^\infty S_j(r) S_k(r) r^2 \, dr \tag{10}$$

Thus the main task is to compute the complex coefficients $B_{cj\ell m}$ of the ligand and protein scalar fields, respectively. This is not immediate as the fields are defined using expansions in "local" radial and spherical harmonic bases, i.e., with respect to the individual atom positions as opposed to the coordinate system origin. Furthermore, since we cannot (in practice) use a complete set of radial or angular basis functions, it is generally not possible to express the ligand or protein scalar field as defined in Equation 2 using the form in Equation 7. Instead, we propose to find the coefficients $B_{cj\ell m}$ that give the best approximation to the true scalar fields, in the sense of least squared error.

Specifically, suppose that $\mathbf{R} \in \mathbb{R}^{N_{\text{grid}} \times N_{\text{local}}}$ are the values of N_{local} real local basis functions (i.e., different origins, RBFs, and spherical harmonics) evaluated at N_{grid} grid points and $\mathbf{A} \in \mathbb{R}^{N_{\text{local}}}$ is the vector of coefficients defining the scalar field ϕ_c. Similarly define $\mathbf{S} \in \mathbb{R}^{N_{\text{grid}} \times N_{\text{global}}}$ using the real versions of the global basis functions. We seek to find the least-squares solution $\mathbf{B} \in \mathbb{R}^{N_{\text{global}}}$ to the overdetermined system of equations $\mathbf{RA} = \mathbf{SB}$, which is given by

$$\mathbf{B} = (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T\mathbf{RA} \tag{11}$$

Notably, this is simply a linear transformation of the local coefficients $A_{cnj\ell m}$. Thus, if we can precompute the inverse Gram matrix of the global bases $(\mathbf{S}^T\mathbf{S})^{-1}$ and the inner product of the global and local bases $\mathbf{S}^T\mathbf{R}$, then for any new scalar field ϕ_c the real global coefficients are immediately available via a linear transformation. The desired complex coefficients can then be easily obtained via a change of bases. At first glance, this still appears challenging due to the continuous space of possible atomic or alpha-carbon positions, but an appropriate discretization makes the precomputation relatively inexpensive without a significant loss of fidelity.

3.4 TRAINING AND INFERENCE

We now study how the rapid cross-correlation procedures presented thus far are used in training and inference. For a given training example with protein structure $\mathbf{X}^P$, the scoring function $E(\mathbf{X}^P, \mathbf{X}^L)$ should ideally attain a maximum at the true ligand pose $\mathbf{X}^L = \mathbf{X}^{L\star}$. We equate this task to that of learning an *energy based model* to maximize the log-likelihood of the true pose under the model likelihood $p(\mathbf{X}^L) \propto \exp\left[E(\mathbf{X}^P, \mathbf{X}^L)\right]$. However, as is typically the case for energy-based models, directly optimizing this objective is difficult due to the intractable partition function.

Instead, following Corso et al. (2023), we conceptually decompose the ligand pose $\mathbf{X}^L$ to be a tuple $\mathbf{X}^L = (\mathbf{X}^C, R, \mathbf{t})$ consisting of a zero-mean conformer $\mathbf{X}^C$, a rotation R, and a translation $\mathbf{t}$, from which the pose coordinates are obtained: $\mathbf{X}^L = R.\mathbf{X}^C + \mathbf{t}$. Then consider the following *conditional* log-likelihoods:

$$\log p(\mathbf{t} \mid \mathbf{X}^C, R) = E(\mathbf{X}^P, \mathbf{X}^L) - \log \int_{\mathbb{R}^3} \exp\left[E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}')\right] \, d^3\mathbf{t}' \tag{12a}$$

$$\log p(R \mid \mathbf{X}^C, \mathbf{t}) = E(\mathbf{X}^P, \mathbf{X}^L) - \log \int_{SO(3)} \exp\left[E\left(\mathbf{X}^P - \mathbf{t}, R'.\mathbf{X}^C\right)\right] \, dR' \tag{12b}$$

We observe that these integrands are precisely the cross-correlations in Equations 3 and 8, respectively, and can be quickly evaluated and summed for all values of $\mathbf{t}'$ and R' using fast Fourier transforms. Thus, the integrals—which are the marginal likelihoods $p(\mathbf{X}^C, R)$ and $p(\mathbf{X}^C, \mathbf{t})$—are tractable and the conditional log-likelihoods can be directly optimized in order to train the neural network. Although neither technically corresponds to the joint log-likelihood of the pose, we find that these training objectives work well in practice and optimize their sum in our training procedure.

At inference time, a rigid protein structure $\mathbf{X}^P$ is given and the high-level task is to score or optimize candidate ligand poses $\mathbf{X}^L$. A large variety of possible workflows can be imagined; however, for proof of concept and for our experiments in Section 4 we describe and focus on the following relatively simple inference workflows (presented in greater detail in Appendix B):

- **Translational FFT (TF)**. Given a conformer $\mathbf{X}^C$, we conduct a grid-based search over R and use FFT to optimize $\mathbf{t}$ in order to find the best pose $(\mathbf{X}^C, R, \mathbf{t})$. To do so, we compute the Fourier coefficients (Equation 5) of the protein $\mathbf{X}^P$ *once* and for *each* possible ligand orientation $R.\mathbf{X}^C$. We then use translational cross-correlations (Equation 3) to find the best translation $\mathbf{t}$ for each R and return the highest scoring combination.
- **Rotational FFT (RF)**. Given a conformer $\mathbf{X}^C$, we conduct a grid-based search over $\mathbf{t}$ and use FFT to optimize R. To do so, we compute the global expansion coefficients $B^P_{cj\ell m}$ of the protein $\mathbf{X}^L - \mathbf{t}$ relative to *each* possible ligand position $\mathbf{t}$ and *once* for the ligand $\mathbf{X}^C$ relative to its (zero) center of mass (Equation 11). We then use rotational cross-correlations (Equation 8) to find the best orientation R for each $\mathbf{t}$ and return the highest scoring combination.
- **Translational scoring (TS)**. Here we instead are given a list of poses $(\mathbf{X}^C, R, \mathbf{t})$ and wish to score them. Because the values of R nor $\mathbf{t}$ may not satisfy a grid structure, we cannot use the FFT methods. Nevertheless, we can compute the (translational) Fourier coefficients of the protein $\mathbf{X}^P$ and for each unique oriented conformer $R.\mathbf{X}^C$ of the ligand using Equation 5. We then evaluate

  $$E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}) = \sum_c \int_{\mathbb{R}^3} \overline{\mathcal{F}[\phi^P_c](\mathbf{k})} \cdot \mathcal{F}[\phi^L_c(\,\cdot\,; R.\mathbf{X}^C)](\mathbf{k}) \cdot e^{-i\mathbf{k}\cdot\mathbf{t}} \, d^3\mathbf{k} \tag{13}$$

  Since the Fourier transform is an orthogonal operator on functional space, this is equal to the real-space cross-correlation.
- **Rotational scoring (RS)**. Analogously, we can score a list of poses $(\mathbf{X}^C, R, \mathbf{t})$ using the global spherical expansions $B_{cj\ell m}$. We obtain the real expansion coefficients of the protein relative to each $\mathbf{t}$ and for each ligand conformer $\mathbf{X}^C$ using Equation 11. The score for $(\mathbf{X}^C, R, \mathbf{t})$ is then given by the rotational cross-correlation

  $$E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}) = \sum_{c,j,k,\ell,m,n} B^P_{cj\ell m}(\mathbf{X}^P - \mathbf{t}) B^L_{ck\ell n}(\mathbf{X}^C) D^\ell_{mn}(R) G_{jk} \tag{14}$$

  where G_{jk} is as defined in Equation 10 and D^ℓ_{mn} are the real Wigner D-matrices.

Table 1: **Typical runtimes** of the computations involved in inference-time scoring and optimization procedures, measured on PDBBind with one V100 GPU. The three sets of rows delineate computations that are protein-specific, ligand-specific, or involve both protein and ligand, respectively.

Frequency	Computation	TF	RF	TS	RS	Runtime
Per protein structure	Coefficients $A_{cnj\ell m}$	✓	✓	✓	✓	65 ms
	FFT coefficients	✓		✓		7.0 ms
↪ Per translation	Global expansion $B_{cj\ell m}$		✓		✓	80 ms
Per ligand conformer	Coefficients $A_{cnj\ell m}$	✓	✓	✓	✓	4.3 ms
	Global expansion $B_{cj\ell m}$		✓		✓	17 ms
↪ Per rotation	FFT coefficients	✓		✓		1.6 ms
Per conformer × rotation	Translational FFT	✓				160 μs
Per conformer × translation	Rotational FFT		✓			650 μs
Per pose	Translational scoring			✓		1.0 μs
	Rotational scoring				✓	8.2 μs

The runtime of these workflows can vary significantly depending on the parameters, i.e., number of proteins, ligands, conformers, rotations, and translations, with amortizations possible at several levels. Table 1 provides a summary of the computations in each workflow, their frequencies, and typical runtimes. We highlight that the **RF** workflow is well-suited for virtual screening since the precomputations for the protein and ligand translations within a pocket can be amortized across all ligands. Furthermore, if the ligands are drawn from a shared library, their coefficients can also be precomputed independent of any protein, leaving only the rotational FFT as the cost per ligand-protein pair. See Appendix F for further discussion on runtime amortization.

4 EXPERIMENTS

We train and test our model on the PDBBind dataset (Liu et al., 2017) with splits as defined by Stärk et al. (2022). We train two variants of our model: ESF and ESF-N, where the latter is trained with rotational and translational noise injected into the examples to increase model robustness. All other model and inference-time hyperparameters are discussed in Appendix E. For the test set, we consider both the co-crystal structures in the PDBBind test split and their counterpart ESMFold complexes as prepared by Corso et al. (2023). We also collect a test set of 77 crystal structures (none of which are in PDBBind) of phosphodiesterase 10A (PDE10A) with different ligands bound to the same pocket (Tosstorff et al., 2022). This industrially-sourced dataset is representative of a real-world use case for molecular docking and benchmarks the benefits of runtime amortization with our approach.

To evaluate our method against baselines, we note that a scoring function by itself is not directly comparable to complete docking programs, which also include tightly integrated conformer search, pose clustering, and local refinement algorithms. Here, however, we focus on the development of the *scoring function* itself independently of these other components. Thus, we consider two simplified settings for evaluating our model: (1) **scoring decoy poses** with the aim of identifying the best pose among them, and (2) **docking rigid conformers** to a given pocket, similar to the re-docking setup in Stärk et al. (2022). The first setting focuses on evaluating only the quality of the scoring function itself, whereas the second is a simplified version of a typical docking setting that circumvents some of the confounding factors while still allowing the benchmarking of FFT-accelerated optimization.

For the baselines, we select Gnina (McNutt et al., 2021) as the representative traditional docking software, which runs parallel MCMC chains to collect pose candidates that are then refined and re-ranked to produce the final prediction. Among the scoring functions supported by Gnina, we evaluate its namesake CNN (Ragoza et al., 2017) scoring function and the traditional scoring function of Vina (Trott & Olson, 2010). For recent deep learning baselines, we adapt TANKBind (Lu et al., 2022) and DiffDock (Corso et al., 2023) for decoy-pose scoring. Specifically, with DiffDock we use the provided confidence model—which is natively trained as an <2 Å RMSD classifier—to score all poses. With TANKBind, we use the predicted distance map and ground truth conformer interatomic distances to calculate $\mathcal{L}_{\text{generation}}$ as the pose score. We do not evaluate these methods on pocket-level conformer docking as they cannot be easily adapted for this task.

Table 2: **Decoy scoring results** (median over test complexes). All RMSDs are heavy-atom symmetry aware. The best results from our method (ESF) are underlined if not bolded. †These methods have been adapted for the pose scoring task; see main text.

Method	Crystal structures				ESMFold structures				Time per	
	<2 Å AUROC	Top RMSD	Top Rank	% <2 Å	<2 Å AUROC	Top RMSD	Top Rank	% <2 Å	Pose	Complex
Vina	0.93	**0.54**	**2**	**91**	0.86	2.43	419	43	3.4 ms	110 s
Gnina	0.90	0.59	3	83	0.84	2.19	1110	46	13.0 ms	426 s
TANKBind†	0.69	4.01	6811	10	0.64	4.22	8538	9	1.1 μs	62 ms
DiffDock†	**0.96**	0.66	3	87	**0.89**	2.01	143	50	62 ms	2041 s
ESF-TS	0.87	<u>0.59</u>	<u>3</u>	<u>87</u>	0.82	**1.38**	24	**57**	1.0 μs	3.2 s
ESF-RS	0.87	0.63	<u>3</u>	85	0.82	1.75	**22**	53	8.2 μs	5.7 s
ESF-N-TS	<u>0.92</u>	0.69	4	81	<u>0.87</u>	1.64	**22**	54	1.0 μs	3.2 s
ESF-N-RS	<u>0.92</u>	0.75	5	80	<u>0.87</u>	1.74	26	53	8.2 μs	5.7 s

4.1 Scoring Decoys

For each PDBBind test complex, we generate $32^3 - 1 = 32767$ decoy poses by sampling 31 translational, rotational, and torsional perturbations to the ground truth pose and considering all their possible combinations. On median, the RMSD of the closest decoy is 0.4 Å, and 1.6% of all poses ($n = 526.5$) are below 2 Å RMSD (Appendix D.1). We then score all poses using the baselines and with our method in both **TS** (Equation 13) and **RS** (Equation 14) modes. The quality of each scoring function is evaluated with the AUROC when used as a <2Å RMSD classifier, the RMSD of the top-ranked pose (Top RMSD), the rank of the lowest-RMSD pose (Top Rank), and the fraction of complexes for which the top-ranked pose is under 2 Å RMSD.

As shown in Table 2, our method is competitive with the Gnina and Vina baselines on crystal structures and better on ESMFold structures. This improved robustness is expected since the interaction terms in traditional scoring functions are primarily mediated by sidechain atoms, which are imperfectly predicted by ESMFold, whereas our scalar fields only indirectly depend on the sidechains via residue-level coefficients. Our method matches and outperforms the DiffDock confidence model on crystal and ESMFold structures, respectively, in terms of identifying the best poses, although DiffDock obtains better <2 Å RMSD AUROC. Finally, our method substantially ourperforms TANKBind in all metrics. Overall, the performance of our method is superior in the **TS** mode relative to **RS**, likely due to the spatially coarser representation of the scalar fields in the global spherical harmonic expansion (i.e., Equation 7) relative to the grid-based Cartesian expansion.

In terms of runtime per pose, our method is faster than Vina by several orders of magnitude, with even greater acceleration compared to the CNN-based Gnina and the E3NN-based DiffDock, both of which require one neural network evaluation for every pose. The runtime improvement per complex is more tempered since the different proteins and ligand in every complex limit the opportunity for amortization. In fact, of the total runtime per complex in Table 2, only 1% (**TS**) to 5% (**RS**) is due to the pose scoring itself, with the rest due to preprocessing that must be done for every new protein and ligand independently. Similar to our method, TANKBind requires only a single neural network evaluation per complex and thus has comparable per-pose runtime, but enjoys faster per-complex runtime due to the lack of such preprocessing.

4.2 Docking Conformers

We consider the task of pocket-level docking where all methods are given as input the ground-truth conformer in a random orientation. Following common practice (McNutt et al., 2021), we aim to provide 4 Å of translational uncertainty around the true ligand pose in order to define the binding pocket. To do so, we provide Gnina with a bounding box with 4 Å of padding around the true pose, and provide our method with a cube of side length 8 Å as the search space for $\mathbf{t}$ (with a random grid offset). For PDE10A, we define the pocket using the pose of the first listed complex (PDB 5SFS) and cross-dock to that protein structure. In all docking runs, we deactivate all torsion angles so that Gnina docks the provided conformer to the pocket.

Table 3: **Rigid conformer docking results** (median over test complexes). All RMSDs are heavy-atom symmetry aware. The median RMSD of our method (ESF) is lower-bounded at 0.5–0.6 Å by the resolution of the search grid (Appendix G.2). The runtime is shown as an average per complex, excluding / including pre-computations that can be amortized.

	PDBBind test					PDE10A		
	Crystal		ESMFold					
Method	% <2 Å	Med. RMSD	% <2 Å	Med. RMSD	Runtime	% <2 Å	Med. RMSD	Runtime
Vina	**79**	**0.32**	24	6.1	20 s	**74**	**0.75**	6.1 s
Gnina	77	0.33	28	5.9	23 s	73	0.77	6.0 s
ESF-TF	70	1.13	31	4.6	0.8 s / 8.3 s	67	1.20	1.0 s / 7.1 s
ESF-RF	71	<u>0.97</u>	32	4.4	0.5 s / 67 s	<u>73</u>	<u>0.82</u>	0.5 s / 1.5 s
ESF-N-TF	72	1.10	46	**2.9**	0.7 s / 8.2 s	64	1.11	1.0 s / 7.2 s
ESF-N-RF	<u>73</u>	1.00	**47**	3.0	0.5 s / 68 s	70	1.00	0.5 s / 1.5 s

As shown in Table 3, the baseline scoring functions obtain excellent success rates on the PDBBind crystal structures (79%). Our method is slightly weaker but also obtains high success rates (73%). The performance decrease in terms of Median RMSD is somewhat larger, likely due to the coarse search grid over non-FFT degrees of freedom (Appendix G.2) and the lack of any refinement steps (which are an integral part of Gnina) in our pipeline. On ESMFold structures, however, our method obtains nearly twice the success rate (47% vs 28%) of the baseline scoring functions. Unlike in decoy scoring, noisy training noticeably contributes to the performance on ESMFold structures, and the **RF** procedure generally outperforms **TF**, likely due to the relatively finer effective search grid in rotational cross-correlations (Appendix G.2).

Because of the nature of the PDBBind workflow, the total runtime is comparable to or slower than the baselines when precomputations are taken into account. However, in terms of the pose optimization itself, our method is significantly faster than the Gnina baselines, despite performing a brute force search over the non-FFT degrees of freedom. While it is also possible to trade-off performance and runtime by changing various Gnina settings from their default values, our method expands the Pareto front currently available with the Gnina pipeline (Appendix G.3; Figure 8). This favorable tradeoff affirms the practical value-add of our method in the context of existing approaches.

To more concretely demonstrate the runtime improvements of our method with amortization, we then dock the conformers in the PDE10A dataset. Our method again has similar accuracy to the baselines (Table 3); however, because of the common pocket, all protein-level quantities are computed *only once* and the total runtime is significantly accelerated. For the **RF** procedure in particular, the computation of protein global coefficients on the translational grid is by far the most expensive step (Table 9), and the remaining ligand precomputations are very cheap. The amortization of these coefficients leads to a 45x speedup in the overall runtime (67 s $\rightarrow$ 1.5 s). (The runtime for Gnina is also accelerated, although to a lesser extent, due to the smaller ligand size.) As the number of ligands increases further, the total runtime per complex of our method would further decrease.

5 Conclusion

We have proposed a machine-learned based scoring function for accelerating pose optimization in molecular docking. Different from existing scoring functions, the score is defined as a cross-correlation between scalar fields, which enables the use of FFTs for rapid search and optimization. We have formulated a novel parameterization for such scalar fields with equivariant neural networks, as well as training and inference procedures with opportunities for significant runtime amortization. Our scoring function shows comparable performance but improved runtime on two simplified docking-related tasks relative to standard optimization procedures and scoring functions. Thus, our methodology holds promise when integrated with other components into a full docking pipeline. These integrations may include multi-resolution search, refinement with traditional scoring functions, and architectural adaptations for conformational (i.e., torsional) degrees of freedom—all potential directions of future work.

ACKNOWLEDGMENTS

We thank Hannes Stärk, Samuel Sledzieski, Ruochi Zhang, Michael Brocidiacono, Gabriele Corso, Xiang Fu, Felix Faltings, Ameya Daigavane, and Mario Geiger for helpful feedback and discussions. This work was supported by the NIH NIGMS under grant #1R35GM141861 and a Department of Energy Computational Science Graduate Fellowship.

REFERENCES

Rong Chen and Zhiping Weng. Docking unbound proteins using shape complementarity, desolvation, and electrostatics. *Proteins: Structure, Function, and Bioinformatics*, 47(3):281–294, 2002.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi S Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *The Eleventh International Conference on Learning Representations*, 2023.

Kevin Crampon, Alexis Giorkallos, Myrtille Deldossi, Stéphanie Baud, and Luiz Angelo Steffenel. Machine-learning methods for ligand–protein molecular docking. *Drug discovery today*, 27(1): 151–164, 2022.

Xinqiang Ding, Yujin Wu, Yanming Wang, Jonah Z Vilseck, and Charles L Brooks III. Accelerated cdocker with gpus, parallel simulated annealing, and fast fourier transforms. *Journal of chemical theory and computation*, 16(6):3910–3919, 2020.

Jiyu Fan, Ailing Fu, and Le Zhang. Progress in molecular docking. *Quantitative Biology*, 7:83–89, 2019.

Leonardo G Ferreira, Ricardo N Dos Santos, Glaucius Oliva, and Adriano D Andricopulo. Molecular docking and structure-based drug design strategies. *Molecules*, 20(7):13384–13421, 2015.

Henry A Gabb, Richard M Jackson, and Michael JE Sternberg. Modelling protein docking using shape complementarity, electrostatics and biochemical information. *Journal of molecular biology*, 272(1):106–120, 1997.

Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks. *arXiv preprint arXiv:2207.09453*, 2022.

Thomas A Halgren, Robert B Murphy, Richard A Friesner, Hege S Beard, Leah L Frye, W Thomas Pollard, and Jay L Banks. Glide: a new approach for rapid, accurate docking and scoring. 2. enrichment factors in database screening. *Journal of medicinal chemistry*, 2004.

Huaipan Jiang, Mengran Fan, Jian Wang, Anup Sarma, Shruti Mohanty, Nikolay V Dokholyan, Mehrdad Mahdavi, and Mahmut T Kandemir. Guiding conventional protein–ligand docking software with convolutional neural networks. *Journal of Chemical Information and Modeling*, 60 (10):4594–4602, 2020.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *arXiv preprint arXiv:2206.01729*, 2022.

Ephraim Katchalski-Katzir, Isaac Shariv, Miriam Eisenstein, Asher A Friesem, Claude Aflalo, and Ilya A Vakser. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proceedings of the National Academy of Sciences*, 89 (6):2195–2199, 1992.

Julio A Kovacs and Willy Wriggers. Fast rotational matching. *Acta Crystallographica Section D: Biological Crystallography*, 58(8):1282–1286, 2002.

Dima Kozakov, Ryan Brenke, Stephen R Comeau, and Sandor Vajda. Piper: an fft-based protein docking program with pairwise potentials. *Proteins: Structure, Function, and Bioinformatics*, 65 (2):392–406, 2006.

Dima Kozakov, David R Hall, Bing Xia, Kathryn A Porter, Dzmitry Padhorny, Christine Yueh, Dmitri Beglov, and Sandor Vajda. The cluspro web server for protein–protein docking. *Nature protocols*, 12(2):255–278, 2017.

Hongjian Li, Kam-Heung Sze, Gang Lu, and Pedro J Ballester. Machine-learning scoring functions for structure-based virtual screening. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 11(1):e1478, 2021.

Zhihai Liu, Minyi Su, Li Han, Jie Liu, Qifan Yang, Yan Li, and Renxiao Wang. Forging the basis for developing protein–ligand interaction scoring functions. *Accounts of Chemical Research*, 50 (2):302–309, 2017.

Wei Lu, Qifeng Wu, Jixian Zhang, Jiahua Rao, Chengtao Li, and Shuangjia Zheng. Tankbind: Trigonometry-aware neural networks for drug-protein binding structure prediction. *Advances in neural information processing systems*, 2022.

Jeffrey G Mandell, Victoria A Roberts, Michael E Pique, Vladimir Kotlovyi, Julie C Mitchell, Erik Nelson, Igor Tsigelny, and Lynn F Ten Eyck. Protein docking using continuum electrostatics and geometric fit. *Protein engineering*, 14(2):105–113, 2001.

Andrew T McNutt, Paul Francoeur, Rishal Aggarwal, Tomohide Masuda, Rocco Meli, Matthew Ragoza, Jocelyn Sunseri, and David Ryan Koes. Gnina 1.0: molecular docking with deep learning. *Journal of cheminformatics*, 13(1):1–20, 2021.

Oscar Méndez-Lucio, Mazen Ahmad, Ehecatl Antonio del Rio-Chanona, and Jörg Kurt Wegner. A geometric deep learning approach to predict binding conformations of bioactive molecules. *Nature Machine Intelligence*, 3(12):1033–1039, 2021.

Elaine C Meng, Brian K Shoichet, and Irwin D Kuntz. Automated docking with grid-based energy evaluation. *Journal of computational chemistry*, 13(4):505–524, 1992.

Garrett M Morris and Marguerita Lim-Wilby. Molecular docking. In *Molecular modeling of proteins*, pp. 365–382. Springer, 2008.

Garrett M Morris, David S Goodsell, Robert S Halliday, Ruth Huey, William E Hart, Richard K Belew, and Arthur J Olson. Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of computational chemistry*, 19(14):1639–1662, 1998.

Emilie Neveu, David W Ritchie, Petr Popov, and Sergei Grudinin. Pepsi-dock: a detailed data-driven protein–protein interaction potential accelerated by polar fourier correlation. *Bioinformatics*, 32 (17):i693–i701, 2016.

Trung Hai Nguyen, Huan-Xiang Zhou, and David DL Minh. Using the fast fourier transform in binding free energy calculations. *Journal of computational chemistry*, 39(11):621–636, 2018.

Dzmitry Padhorny, Andrey Kazennov, Brandon S Zerbe, Kathryn A Porter, Bing Xia, Scott E Mottarella, Yaroslav Kholodov, David W Ritchie, Sandor Vajda, and Dima Kozakov. Protein–protein docking by fast generalized fourier transforms on 5d rotational manifolds. *Proceedings of the National Academy of Sciences*, 113(30):E4286–E4293, 2016.

Dzmitry Padhorny, David R Hall, Hanieh Mirzaei, Artem B Mamonov, Mohammad Moghadasi, Andrey Alekseenko, Dmitri Beglov, and Dima Kozakov. Protein–ligand docking using fft based sampling: D3r case study. *Journal of computer-aided molecular design*, 32:225–230, 2018.

Rodrigo Quiroga and Marcos A Villarreal. Vinardo: A scoring function based on autodock vina improves scoring, docking, and virtual screening. *PloS one*, 11(5):e0155183, 2016.

Matthew Ragoza, Joshua Hochuli, Elisa Idrobo, Jocelyn Sunseri, and David Ryan Koes. Protein–ligand scoring with convolutional neural networks. *Journal of chemical information and modeling*, 57(4):942–957, 2017.

David W Ritchie and Graham JL Kemp. Protein docking using spherical polar fourier correlations. *Proteins: Structure, Function, and Bioinformatics*, 39(2):178–194, 2000.

David W Ritchie, Dima Kozakov, and Sandor Vajda. Accelerating and focusing protein–protein docking correlations using multi-dimensional rotational fft generating functions. *Bioinformatics*, 24(17):1865–1873, 2008.

Brian K Shoichet, Irwin D Kuntz, and Dale L Bodian. Molecular docking using shape descriptors. *Journal of computational chemistry*, 13(3):380–397, 1992.

Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Regina Barzilay, and Tommi Jaakkola. Equibind: Geometric deep learning for drug binding structure prediction. In *International Conference on Machine Learning*, pp. 20503–20521. PMLR, 2022.

Minyi Su, Qifan Yang, Yu Du, Guoqin Feng, Zhihai Liu, Yan Li, and Renxiao Wang. Comparative assessment of scoring functions: the casf-2016 update. *Journal of chemical information and modeling*, 59(2):895–913, 2018.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint*, 2018.

Benjamin I Tingle, Khanh G Tang, Mar Castanon, John J Gutierrez, Munkhzul Khurelbaatar, Chinzorig Dandarchuluun, Yurii S Moroz, and John J Irwin. Zinc-22– a free multi-billion-scale database of tangible compounds for ligand discovery. *Journal of Chemical Information and Modeling*, 63(4):1166–1176, 2023.

Pedro HM Torres, Ana CR Sodero, Paula Jofily, and Floriano P Silva-Jr. Key topics in molecular docking for drug design. *International journal of molecular sciences*, 20(18):4574, 2019.

Andreas Tosstorff, Markus G Rudolph, Jason C Cole, Michael Reutlinger, Christian Kramer, Hervé Schaffhauser, Agnès Nilly, Alexander Flohr, and Bernd Kuhn. A high quality, industrial data set for binding affinity prediction: performance comparison in different early drug discovery scenarios. *Journal of Computer-Aided Molecular Design*, 36(10):753–765, 2022.

Oleg Trott and Arthur J Olson. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

Benjamin D Wandelt and Krzysztof M Górski. Fast convolution on the sphere. *Physical review D*, 63(12):123002, 2001.

Wikipedia. Hankel transform — Wikipedia, the free encyclopedia, 2023. URL `https://en.wikipedia.org/wiki/Hankel_transform#Fourier_transform_in_three_dimensions`.

Yumeng Yan, Huanyu Tao, Jiahua He, and Sheng-You Huang. The hdock server for integrated protein–protein docking. *Nature protocols*, 15(5):1829–1852, 2020.

Chao Yang, Eric Anthony Chen, and Yingkai Zhang. Protein–ligand docking in the machine-learning era. *Molecules*, 27(14):4568, 2022.

Anna Yershova, Swati Jain, Steven M Lavalle, and Julie C Mitchell. Generating uniform incremental grids on so (3) using the hopf fibration. *The International journal of robotics research*, 29(7):801–812, 2010.

Yangtian Zhang, Huiyu Cai, Chence Shi, Bozitao Zhong, and Jian Tang. E3bind: An end-to-end equivariant network for protein-ligand docking. *arXiv preprint arXiv:2210.06069*, 2022.

Ellen D Zhong, Tristan Bepler, Joseph H Davis, and Bonnie Berger. Reconstructing continuous distributions of 3d protein structure from cryo-em images. In *International Conference on Learning Representations*, 2019.

A MATHEMATICAL DETAILS

A.1 PROOF OF PROPOSITION 1

Proposition 1. *Suppose the scoring function is parameterized as in Equation 2 and for any $R \in SO(3), \mathbf{t} \in \mathbb{R}^3$ we have $A_{cnj\ell m}(G, R.\mathbf{X} + \mathbf{t}) = \sum_{m'} D^{\ell}_{mm'}(R) A_{cnj\ell m'}(G, \mathbf{X})$ where $D^{\ell}(R)$ are the (real) Wigner D-matrices, i.e., irreducible representations of $SO(3)$. Then for any $g \in SE(3)$,*

1. *The scalar field transforms equivariantly: $\phi_c(\mathbf{x}; G, g.\mathbf{X}) = \phi_c(g^{-1}.\mathbf{x}; G, \mathbf{X})$.*
2. *The scoring function is invariant: $E(g.\mathbf{X}^P, g.\mathbf{X}^L) = E(\mathbf{X}^P, \mathbf{X}^L)$.*

Proof. Let the action of $g = (R, \mathbf{t}) \in SE(3)$ be written as $g : \mathbf{x} \mapsto R\mathbf{x} + \mathbf{t}$ and hence $g^{-1} : \mathbf{x} \mapsto R^T(\mathbf{x} - \mathbf{t})$. We first note that $\|\mathbf{x} - g.\mathbf{x}_n\| = \|g^{-1}.\mathbf{x} - \mathbf{x}_n\|$ and $R^T(\mathbf{x} - g.\mathbf{x}_n) = g^{-1}.\mathbf{x} - \mathbf{x}_n$. Then

$$\begin{aligned}
\phi_c(\mathbf{x}; G, g.\mathbf{X}) &= \sum_{n,j,\ell,m} A_{cnj\ell m}(G, R.\mathbf{X} + \mathbf{t}) R_j(\|\mathbf{x} - g.\mathbf{x}_n\|) Y_\ell^m \left(\frac{\mathbf{x} - g.\mathbf{x}_n}{\|\mathbf{x} - g.\mathbf{x}_n\|} \right) \\
&= \sum_{n,j,\ell,m'} A_{cnj\ell m'}(G, \mathbf{X}) R_j(\|\mathbf{x} - g.\mathbf{x}_n\|) \sum_m D^{\ell}_{mm'}(R) Y_\ell^m \left(\frac{\mathbf{x} - g.\mathbf{x}_n}{\|\mathbf{x} - g.\mathbf{x}_n\|} \right) \\
&= \sum_{n,j,\ell,m'} A_{cnj\ell m'}(G, \mathbf{X}) R_j(\|\mathbf{x} - g.\mathbf{x}_n\|) Y_\ell^{m'} \left(\frac{R^T(\mathbf{x} - g.\mathbf{x}_n)}{\|\mathbf{x} - g.\mathbf{x}_n\|} \right) \\
&= \sum_{n,j,\ell,m'} A_{cnj\ell m'}(G, \mathbf{X}) R_j(\|g^{-1}.\mathbf{x} - \mathbf{x}_n\|) Y_\ell^{m'} \left(\frac{g^{-1}.\mathbf{x} - \mathbf{x}_n}{\|g^{-1}.\mathbf{x} - \mathbf{x}_n\|} \right) \\
&= \phi_c(g^{-1}.\mathbf{x}; G, \mathbf{X})
\end{aligned}$$

Next,

$$\begin{aligned}
E(g.\mathbf{x}^P, g.\mathbf{x}^L) &= \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}; G^P, g.\mathbf{X}^P) \phi_c^L(\mathbf{x}; G^L, g.\mathbf{X}^L)\, d^3\mathbf{x} \\
&= \sum_c \int_{\mathbb{R}^3} \phi_c^P(g^{-1}\mathbf{x}; G^P, \mathbf{X}^P) \phi_c^L(g^{-1}\mathbf{x}; G^L, \mathbf{X}^L)\, d^3\mathbf{x} \\
&= \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}'; G^P, \mathbf{X}^P) \phi_c^L(\mathbf{x}'; G^L, \mathbf{X}^L)\, d^3\mathbf{x}'
\end{aligned}$$

where the last line has substitution $\mathbf{x}' = g^{-1}\mathbf{x}$ with g volume preserving on $\mathbb{R}^3$. □

A.2 DERIVATIONS

In this section we describe the derivations for the various equations presented in the main text. We use the following convention for the (one-dimensional) Fourier transform and its inverse:

$$\mathcal{F}[f](k) = \frac{1}{\sqrt{2\pi}} \int e^{-ikx} f(x)\, dx \tag{15a}$$

$$\mathcal{F}^{-1}[f](x) = \frac{1}{\sqrt{2\pi}} \int e^{ikx} f(k)\, dk \tag{15b}$$

Equation 5 It is well known (Wikipedia, 2023) that given a function over $\mathbb{R}^3$ with complex spherical harmonic expansion

$$f(\mathbf{r}) = \sum_{\ell,m} f_{\ell,m}(\|\mathbf{r}\|) Y_\ell^m(\mathbf{r}/\|\mathbf{r}\|) \tag{16}$$

its Fourier transform is given by

$$f(\mathbf{k}) = \sum_{\ell,m} (-i)^\ell F_{\ell,m}(\|\mathbf{k}\|) Y_\ell^m(\mathbf{k}/\|\mathbf{k}\|) \tag{17}$$

where

$$F_{\ell,m}(k) = \frac{1}{\sqrt{k}} \int_0^\infty \sqrt{r} f_{\ell,m}(r) J_{\ell+1/2}(kr)\, r\, dr \tag{18}$$

with J_ℓ the ℓth-order Bessel function of the first kind. Relating these to the spherical Bessel functions j_ℓ via $J_{\ell+1/2}(x) = \sqrt{2x/\pi} j_\ell(x)$, we obtain

$$F_{\ell,m}(k) = \sqrt{\frac{2}{\pi}} \int_0^\infty f_{\ell,m}(r) j_\ell(kr)\, r^2\, dr \quad := \mathcal{H}_\ell[f_{\ell,m}](k) \tag{19}$$

which is the form of Equation 6. To apply this to our scalar fields, we define the *translation operator* $\mathcal{T}_{\mathbf{r}}[f](\mathbf{x}) = f(\mathbf{x} - \mathbf{r})$ and note its composition with the Fourier transform

$$(\mathcal{F} \circ \mathcal{T}_{\mathbf{r}})[f] = e^{-i\mathbf{k}\cdot\mathbf{r}} \mathcal{F}[f] \tag{20}$$

We then decompose the form of our scalar fields (Equation 2) into contributions from zero-origin spherical harmonic expansions

$$\phi_c(\mathbf{x}) = \sum_n \mathcal{T}_{\mathbf{x}_n}[\phi_{cn}](\mathbf{x}) \tag{21a}$$

$$\phi_{cn}(\mathbf{x}) = \sum_{\ell,m} \underbrace{\sum_j A_{cnj\ell m} R_j(\|\mathbf{x}\|)}_{\phi_{cn\ell m}(\|\mathbf{x}\|)} Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \tag{21b}$$

Hence, the Fourier transform of each contribution is

$$\mathcal{F}[\phi_{cn}](\mathbf{k}/\|\mathbf{k}\|) = \sum_{\ell,m} (-i)^\ell \mathcal{H}_\ell[\phi_{cn\ell m}](\|\mathbf{k}\|) Y_\ell^m(\mathbf{k}/\|\mathbf{k}\|) \tag{22}$$

Equation 5 is then obtained via Equation 20 and the linearity of the Fourier and spherical Bessel transforms.

Equation 9 We source (with some modifications) the derivation from Kovacs & Wriggers (2002). We consider the cross-correlation

$$c(R) = \int_{\mathbb{R}^3} \phi(\mathbf{x}) \overline{\psi(R^{-1}\mathbf{x})}\, d^3\mathbf{x} \tag{23}$$

which is the same as Equation 8 with $\phi = \phi_c^P$ and $\psi = \phi_c^L$ since ϕ_c^L is a real field. Expanding in complex spherical harmonics Y_ℓ^m and radial bases S_j:

$$\phi(\mathbf{x}) = \sum_{j,\ell,m} \Phi_{j\ell m} S_j(\|\mathbf{x}\|) Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \qquad \psi(\mathbf{x}) = \sum_{j,\ell,m} \Psi_{j\ell m} S_j(\|\mathbf{x}\|) Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \tag{24}$$

We then obtain

$$c(R) = \sum_{j,j',\ell,\ell',m,n,m'} \overline{D_{nm'}^\ell(R)} \Phi_{j\ell m} \overline{\Psi_{j'\ell'm'}} \int_{\mathbb{R}^3} [S_j \cdot S_{j'}](\|\mathbf{x}\|) [Y_\ell^m \cdot \overline{Y_{\ell'}^n}](\mathbf{x}/\|\mathbf{x}\|)\, d^3\mathbf{x} \tag{25a}$$

$$= \sum_{j,j',\ell,\ell',m,n,m'} \overline{D_{nm'}^\ell(R)} \Phi_{j\ell m} \overline{\Psi_{j'\ell'm'}} \underbrace{\int_0^\infty [S_j \cdot S_{j'}](r)\, r^2\, dr}_{G_{jj'}} \underbrace{\int_{S^2} [Y_\ell^m \cdot \overline{Y_{\ell'}^n}](\hat{\mathbf{r}})\, d\hat{\mathbf{r}}}_{\delta_{\ell\ell'}\delta_{mn}} \tag{25b}$$

$$= \sum_{\ell,m,m'} \overline{D_{mm'}^\ell(R)} \underbrace{\sum_{j,j'} \Phi_{j\ell m} \overline{\Psi_{j'\ell m'}} G_{jj'}}_{I_{mm'}^\ell} \tag{25c}$$

Now to evaluate the complex Wigner D-matrix, we adopt the extrinsic zyz convention for Euler angles (applied right-to-left) and note that any rotation (ϕ, θ, ψ) can be decomposed as

$$R(\phi,\theta,\psi) = R_z(\underbrace{\phi - \pi/2}_{\xi}) R_y(\pi/2) R_z(\underbrace{\pi - \theta}_{\eta}) R_y(\pi/2) R_z(\underbrace{\psi - \pi/2}_{\omega}) \tag{26}$$

Next, one can easily check (using the standard spherical harmonics) that the Wigner D-matrix for a rotation about the z-axis is diagonal and given by $D_{mn}^\ell(R_z(\omega)) = \delta_{mn} e^{-in\omega}$. Hence,

$$D_{mn}^\ell(R(\phi,\theta,\psi)) = e^{-im\xi} d_{mh}^\ell e^{-h\eta} d_{hn}^\ell e^{-i\omega n} \tag{27}$$

where $d^\ell = D^\ell(R_y(\pi/2))$ are constant and real. Complex conjugation then gives Equation 9.

Equation 12 The conditional likelihood is

$$\log p(\mathbf{t} \mid \mathbf{X}^C, R) = \log \frac{p(\mathbf{X}^C, R, \mathbf{t})}{p(\mathbf{X}^C, R)} \tag{28a}$$

$$= \log p(\mathbf{X}^C, R, \mathbf{t}) - \log \int_{\mathbb{R}^3} p(\mathbf{X}^C, R, \mathbf{t}')\, d^3\mathbf{t}' \tag{28b}$$

$$= \log E(\mathbf{X}^P, \mathbf{X}^L) - \log \int_{\mathbb{R}^3} \exp\left[E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}')\right]\, d^3\mathbf{t}' \tag{28c}$$

Similarly,

$$\log p(R \mid \mathbf{X}^C, \mathbf{t}) = \log \frac{p(\mathbf{X}^C, R, \mathbf{t})}{p(\mathbf{X}^C, \mathbf{t})} \tag{29a}$$

$$= \log p(\mathbf{X}^C, R, \mathbf{t}) - \log \int_{SO(3)} p(\mathbf{X}^C, R', \mathbf{t})\, dR' \tag{29b}$$

$$= \log E(\mathbf{X}^P, \mathbf{X}^L) - \log \int_{SO(3)} \exp\left[E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t})\right]\, dR' \tag{29c}$$

Finally, we move $\mathbf{t}$ to the protein coordinates (invoking the invariance of the score E) to obtain a form consistent with the rotational cross-correlations (Equation 8).

Equation 13 Given a pose $\mathbf{X}^L = R.\mathbf{X}^C + \mathbf{t}$, we evaluate

$$E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}) = \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}) \phi_c^L(\mathbf{x}; R.\mathbf{X}^C + \mathbf{t})\, d^3\mathbf{x} \tag{30}$$

The functional inner product is equivalent in Fourier space:

$$E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}) = \sum_c \int_{\mathbb{R}^3} \overline{\mathcal{F}[\phi_c^P](\mathbf{k})} \cdot \mathcal{F}[\phi_c^L(\,\cdot\,; R.\mathbf{X}^C + \mathbf{t})](\mathbf{k})\, d^3\mathbf{k} \tag{31}$$

Then with the translation operator $\mathcal{T}$ defined previously,

$$\phi_c^L(\mathbf{x}; R.\mathbf{X}^C + \mathbf{t}) = \mathcal{T}_{\mathbf{t}}[\phi(\,\cdot\,; R.\mathbf{X}^C)](\mathbf{x}) \tag{32a}$$

$$\mathcal{F}[\phi_c^L(\,\cdot\,; R.\mathbf{X}^C + \mathbf{t})](\mathbf{k}) = e^{-i\mathbf{k}\cdot\mathbf{t}} \mathcal{F}[\phi_c^L(\,\cdot\,; R.\mathbf{X}^C)](\mathbf{k}) \tag{32b}$$

We then substitute into Equation 31 to obtain Equation 13.

Equation 14 Given a pose $\mathbf{X}^L = R.\mathbf{X}^C + \mathbf{t}$, we assume that the field $\phi_c^P(\,\cdot\,; \mathbf{X}^P - \mathbf{t})$ and $\phi_c^L(\,\cdot\,; \mathbf{X}^C)$ are written in the real global spherical harmonic expansion:

$$\phi_c^P(\mathbf{x}; \mathbf{X}^P - \mathbf{t}) = \sum_{j,\ell,m} B_{cj\ell m}^P S_j(\|\mathbf{x}\|) Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \tag{33a}$$

$$\phi_c^L(\mathbf{x}; \mathbf{X}^C) = \sum_{j,\ell,m} B_{cj\ell m}^L S_j(\|\mathbf{x}\|) Y_\ell^m(\mathbf{x}/\|\mathbf{x}\|) \tag{33b}$$

Then, analogously to Equation 25,

$$E(\mathbf{X}^P, R.\mathbf{X}^C + \mathbf{t}) = E(\mathbf{X}^P - \mathbf{t}, R.\mathbf{X}^C) \tag{34a}$$

$$= \sum_c \int_{\mathbb{R}^3} \phi_c^P(\mathbf{x}; \mathbf{X}^P - \mathbf{t}) \phi_c^L(R^{-1}\mathbf{x}; \mathbf{X}^C)\, d^3\mathbf{x} \tag{34b}$$

$$= \sum_{c,\ell,m,m'} D_{mm'}^\ell(R) \sum_{j,j'} B_{cj\ell m}^P B_{cj'\ell m'}^L G_{jj'} \tag{34c}$$

Complex conjugation has been omitted because the coefficients and D-functions are now real.

B ALGORITHMIC DETAILS

Below, we present in detail the four inference procedures introduced in Section 3.4. The three blocks of computations are color-coded corresponding to protein preprocessing (green), ligand preprocessing (blue), and the core computation (red) and labelled with typical runtimes from Table 1 (unlabelled lines have negligible runtime). The various loop levels make clear that depending on the workflow, the protein and ligand processing precomputations can be amortized and approaches a negligible fraction of the total runtime. Note, however, that for readability we have presented the algorithms assuming that all possible combinations (i.e., of proteins, ligand conformers, rotations, and translations) are of interest; if this is not true (for example in PDBBind, or in any typical pose-scoring setting), then the full benefits of amortization may not be fully realized.

Algorithm 1: TRANSLATIONAL FFT

Input: Proteins $\{(G_i^P, \mathbf{X}_i^P)\}$, conformers $\{(G_h^L, \mathbf{X}_h^L)\}$
Output: Docked poses $(\mathbf{X}_i^P, \mathbf{X}_{ih}^L)\ \forall i, h$

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// protein preprocessing`
  Compute coefficients $\mathbf{A}_i^P = \{A_{cjn\ell m}^P(G_i^P, \mathbf{X}_i^P)\}$ with neural network ; `// 65 ms`
  Compute Fourier-space field values $\mathcal{F}[\phi^P]_i$ using $\mathbf{A}_i^P, \mathbf{x}_i^P$; `// 7.0 ms`

foreach $(G_h^L, \mathbf{X}_h^L)$ **do** `// ligand preprocessing`
  Compute coefficients $\mathbf{A}_h^L = \{A_{cjn\ell m}^L(G_h^L, \mathbf{X}_h^L)\}$ with neural network ; `// 4.3 ms`
  foreach $R_k \in \{R\}_{grid} \subset SO(3)$ **do**
    Compute rotated coefficients $\mathbf{A}_{h,k}^L$ using $D^\ell(R_k)$;
    Compute Fourier-space field values $\mathcal{F}[\phi^L]_{h,k}$ using $\mathbf{A}_{h,k}^L, R_k\mathbf{X}_h^L$; `// 1.6 ms`

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// pose optimization`
  foreach $(G_h^L, \mathbf{X}_h^L)$ **do**
    foreach $R_k \in \{R\}_{grid} \subset SO(3)$ **do**
      Compute $E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t}), \forall \mathbf{t}$ using FFT; `// 160 μs`
      $E_k^\star, \mathbf{t}_k^\star \leftarrow \{\max, \arg\max\}_{\mathbf{t}} E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t})$;
    $k^\star \leftarrow \arg\max_k E_k^\star$;
    $\mathbf{X}_{ih}^L \leftarrow R_{k^\star}\mathbf{X}_h^L + \mathbf{t}_{k^\star}^\star$;

Algorithm 2: ROTATIONAL FFT

Input: Proteins $\{(G_i^P, \mathbf{X}_i^P)\}$, conformers $\{(G_h^L, \mathbf{X}_h^L)\}$
Output: Docked poses $(\mathbf{X}_i^P, \mathbf{X}_{ih}^L)\ \forall i, h$

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// protein preprocessing`
  Compute coefficients $\mathbf{A}_i^P = \{A_{cjn\ell m}^P(G_i^P, \mathbf{X}_i^P)\}$ with neural network ; `// 65 ms`
  for $\mathbf{t}_k \in \{\mathbf{t}\}_{grid} \subset \mathbb{R}^3$ **do**
    Compute global expansion $\mathbf{B}_{i,k}^P = \{B_{cj\ell m}\}$ from $\mathbf{A}_i^P, \mathbf{X}_i^P - \mathbf{t}_k$; `// 80 ms`

foreach $(G_h^L, \mathbf{X}_h^L)$ **do** `// ligand preprocessing`
  Compute coefficients $\mathbf{A}_h^L = \{A_{cjn\ell m}^L(G_h^L, \mathbf{X}_h^L)\}$ with neural network ; `// 4.3 ms`
  Compute global expansion $\mathbf{B}_h^L = \{B_{cj\ell m}\}$ from $\mathbf{A}_h^L, \mathbf{X}_h^L$; `// 17 ms`

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// pose optimization`
  foreach $(G_h^L, \mathbf{X}_h^L)$ **do**
    foreach $\mathbf{t}_k \in \{\mathbf{t}\}_{grid} \subset \mathbb{R}^3$ **do**
      Compute $E(\mathbf{X}_i^P - \mathbf{t}_k, R.\mathbf{X}_h^L), \forall R$ using FFT ; `// 650 μs`
      $E_k^\star, R_k^\star \leftarrow \{\max, \arg\max\}_R E(\mathbf{X}_i^P - \mathbf{t}_k, R.\mathbf{X}_h^L + \mathbf{t})$;
    $k^\star \leftarrow \arg\max_k E_k^\star$;
    $\mathbf{X}_{ih}^L \leftarrow R_{k^\star}^\star\mathbf{X}_h^L + \mathbf{t}_{k^\star}$;

Algorithm 3: TRANSLATIONAL SCORING

Input: Proteins $\{(G_i^P, \mathbf{X}_i^P)\}$, conformers $\{(G_h^L, \mathbf{X}_h^L)\}$, rotations $\{R_k\}$, translations $\{\mathbf{t}_\ell\}$
Output: Scores $E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t}_\ell)\ \forall i, h, k, \ell$

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// protein preprocessing`
  Compute coefficients $\mathbf{A}_i^P = \{A_{cjn\ell m}^P(G_i^P, \mathbf{X}_i^P)\}$ with neural network ; `// 65 ms`
  Compute Fourier-space field values $\mathcal{F}[\phi^P]_i$ using $\mathbf{A}_i^P, \mathbf{x}_i^P$; `// 7.0 ms`

foreach $(G_h^L, \mathbf{X}_h^L)$ **do** `// ligand preprocessing`
  Compute coefficients $\mathbf{A}_h^L = \{A_{cjn\ell m}^L(G_h^L, \mathbf{X}_h^L)\}$ with neural network ; `// 4.3 ms`
  foreach R_k **do**
    Compute rotated coefficients $\mathbf{A}_{h,k}^L$ using $D^\ell(R_k)$;
    Compute Fourier-space field values $\mathcal{F}[\phi^L]_{h,k}$ using $\mathbf{A}_{h,k}^L, R_k\mathbf{X}_h^L$; `// 1.6 ms`

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// scoring`
  foreach $(G_h^L, \mathbf{X}_h^L)$ **do**
    foreach R_k **do**
      foreach $\mathbf{t}_\ell$ **do**
        Compute $E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t}_\ell)$ using Equation 13; `// 1.0` μs

Algorithm 4: ROTATIONAL SCORING

Input: Proteins $\{(G_i^P, \mathbf{X}_i^P)\}$, conformers $\{(G_h^L, \mathbf{X}_h^L)\}$, rotations $\{R_k\}$, translations $\{\mathbf{t}_\ell\}$
Output: Scores $E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t}_\ell)\ \forall i, h, k, \ell$

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// protein preprocessing`
  Compute coefficients $\mathbf{A}_i^P = \{A_{cjn\ell m}^P(G_i^P, \mathbf{X}_i^P)\}$ with neural network ; `// 65 ms`
  for $\mathbf{t}_k \in \{\mathbf{t}\}_{grid} \subset \mathbb{R}^3$ **do**
    Compute global expansion $\mathbf{B}_{i,k}^P = \{B_{cj\ell m}\}$ from $\mathbf{A}_i^P, \mathbf{X}_i^P - \mathbf{t}_k$; `// 80 ms`

foreach $(G_h^L, \mathbf{X}_h^L)$ **do** `// ligand preprocessing`
  Compute coefficients $\mathbf{A}_h^L = \{A_{cjn\ell m}^L(G_h^L, \mathbf{X}_h^L)\}$ with neural network ; `// 4.3 ms`
  Compute global expansion $\mathbf{B}_h^L = \{B_{cj\ell m}\}$ from $\mathbf{A}_h^L, \mathbf{X}_h^L$; `// 17 ms`

foreach $(G_i^P, \mathbf{X}_i^P)$ **do** `// scoring`
  foreach $(G_h^L, \mathbf{X}_h^L)$ **do**
    foreach R_k **do**
      foreach $\mathbf{t}_\ell$ **do**
        Compute $E(\mathbf{X}_i^P, R_k\mathbf{X}_h^L + \mathbf{t}_\ell)$ using Equation 14; `// 8.2` μs

C LEARNED SCALAR FIELDS

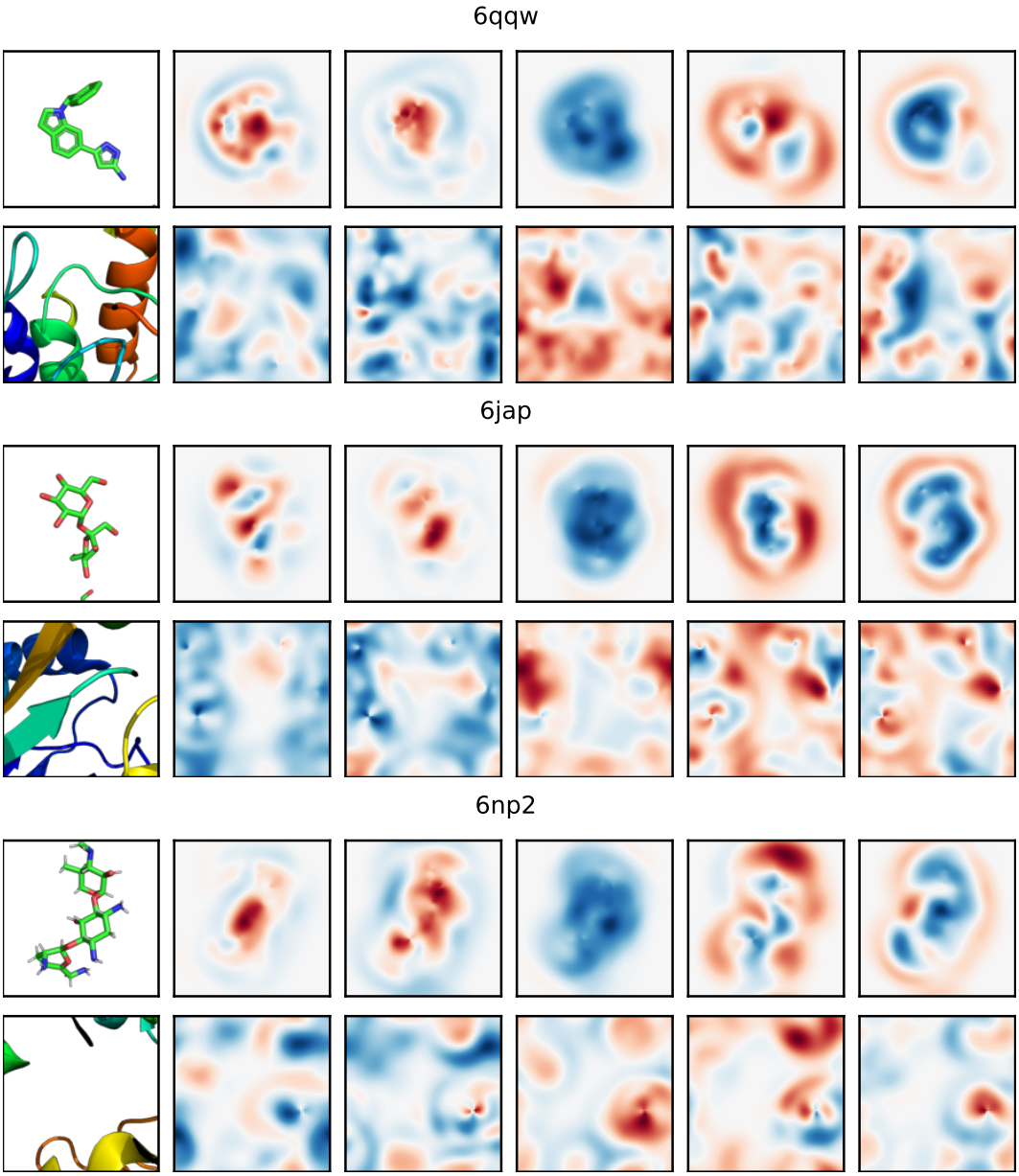

Figure 2: **Visualizations of learned scalar fields**. All five channels of the **ESF-N** learned scalar fields ϕ^L (top row) and ϕ^P (bottom row) are shown on the xy-plane passing through the center of mass of the ligand, with a box diameter of 20 Å. Positive values of the field are in blue and negative values in red. At left, the ligand and pocket structures are shown looking down the z-axis. Note that as the fields are only 2D slices, not all 3D features visible in the structures are visible in the fields.

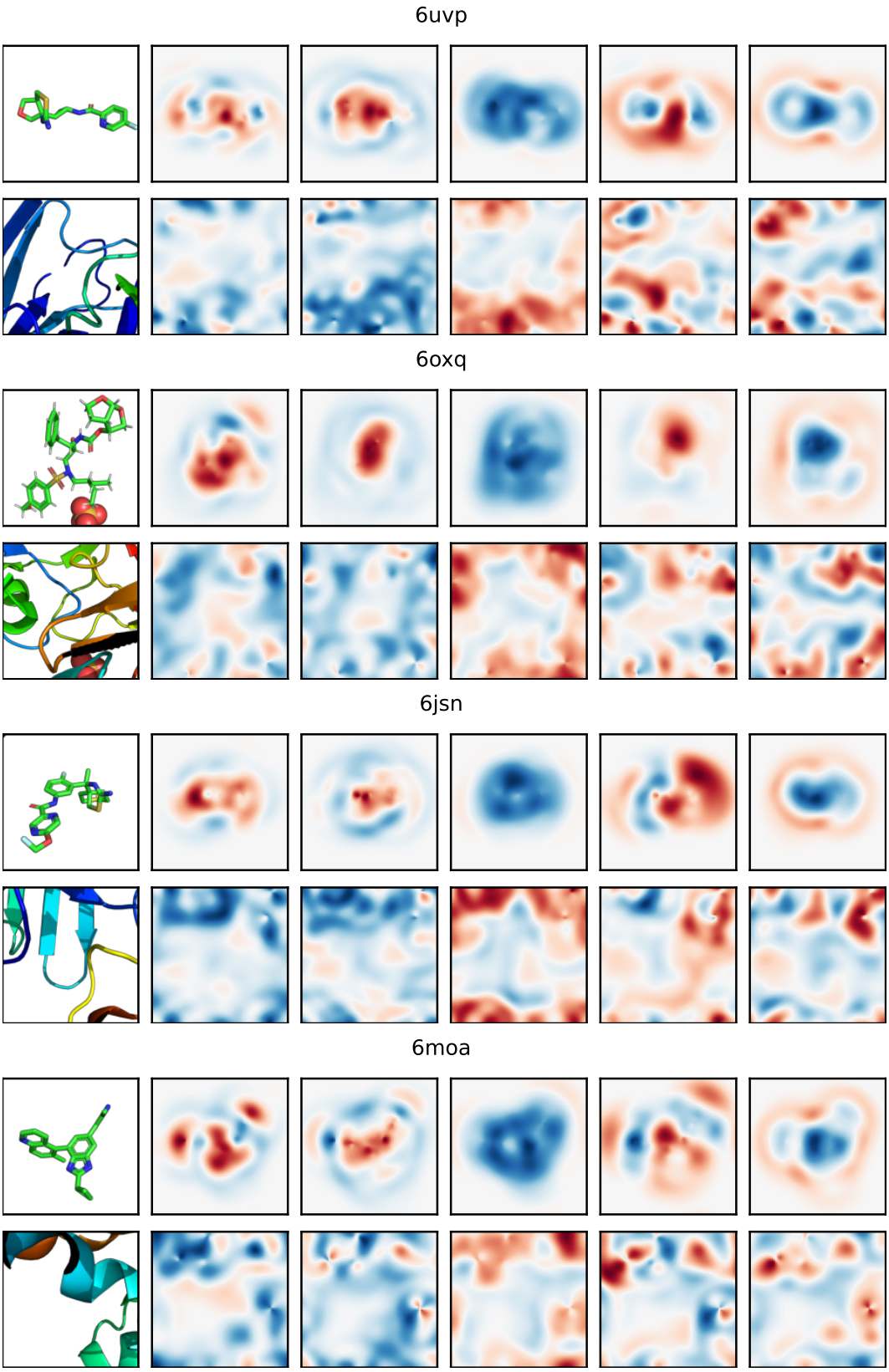

Figure 3: **Visualizations of learned scalar fields**, continued.

D EXPERIMENTAL DETAILS

D.1 DECOY SET

Given a zero-mean ground-truth ligand pose $\mathbf{X}^{L^\star}$, we generate $32^3 - 1 = 32767$ decoy poses via the following procedure.

- Sample 31 translational pertubations: $\mathbf{t}_i \sim \mathcal{N}(0, \mathbf{I}_3), i = 1 \ldots 31$ and set $\mathbf{t}_0 = \mathbf{0}$, with units in Å.
- Sample 31 rotational perturbations: $R_j = \texttt{FromRotvec}(\mathbf{r}_j), \mathbf{r}_j \sim \mathcal{N}(0, 0.5\mathbf{I}_3), j = 1 \ldots 31$ and set $R_0 = \mathbf{I}_3$.
- Sample 31 noisy conformers $\mathbf{X}_k^C, k = 1 \ldots 31$ by sampling torsional updates $\Delta\boldsymbol{\tau}_k \sim \mathcal{N}_{\mathbb{T}}(0, (\pi/2)\mathbf{I}_m)$ where $\mathcal{N}_{\mathbb{T}}$ is a wrapped normal distribution (Jing et al., 2022) and m is the number of torsion angles. The torsional updates are applied to the smaller side of the molecule. Set $\mathbf{X}_0^C = \mathbf{X}^{L^\star}$.
- Set $\mathbf{X}_{ijk}^L = R_j \mathbf{X}_k^C + \mathbf{t}_i, i, j, k = 0 \ldots 31$ and discard $\mathbf{X}_{000}^L = \mathbf{X}^{L^\star}$.

PDB ID 6A73 is excluded from the procedure due to the high level of graph symmetry and significant runtime for computing RMSDs for all decoys. Summary statistics for the decoy sets of the remaining 362 PDB IDs are presented in Figure 4.

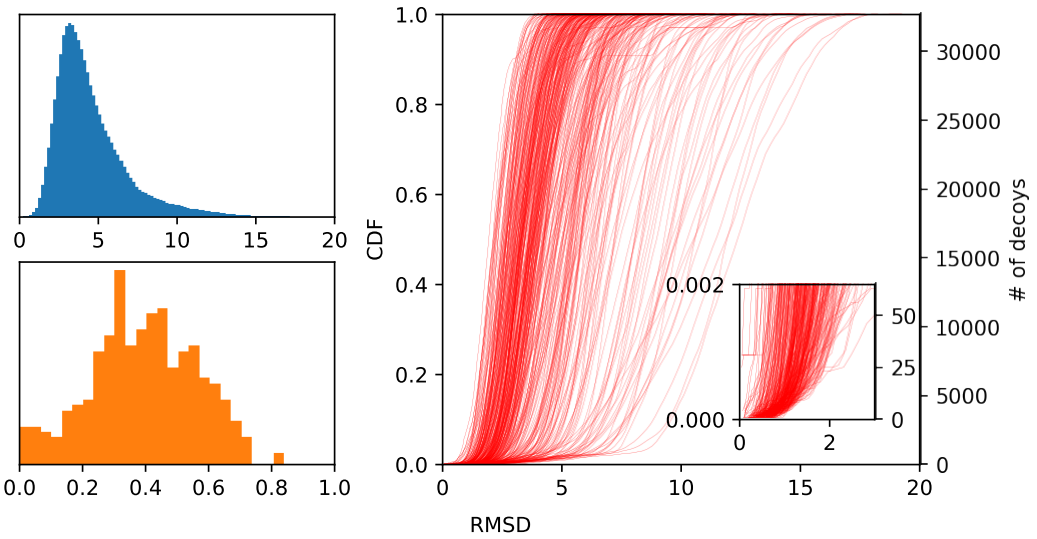

Figure 4: **Decoy set statistics.** *Top left*: histogram of RMSDs across all decoys sets (12 M total). *Bottom left*: histogram of minimum RMSDs among the decoy sets. All sets have a pose less than RMSD <1 Å from the true pose. *Right*: cumulative density function of RMSDs in each decoy set. *Bottom right inset*: all decoy sets have at least 23 poses with RMSD <2 Å.

D.2 RUNTIME MEASUREMENTS

All runtime measurements were performed on a machine with 64 Intel Xeon Gold 6130 CPUs and 8 Nvidia Tesla V100 GPUs. Gnina was run with default thread count settings. All of our processes were run on a single V100 GPU. For our method, we performed runtime analysis using CUDA events to remove the effects of asynchronous CUDA execution. Script loading, model loading, and algorithmic-level precomputations (which, if necessary, can be cached on disk) were excluded from the analysis. For Gnina, we attempted to remove similar overhead by timing single-pose scoring-only runs as representative of constant overhead costs. We report conformer docking runtimes in Table 3 using the PDBBind crystal structures; ESMFold runtimes are marginally shorter. Typical

runtimes reported in Table 1 and Appendix B are obtained from timing runs with our method across the entire PDBBind test set.

D.3 DATASETS

As noted previously, we use train, validation, and test splits from Stärk et al. (2022). However, due to RDKit parsing issues with Gnina-docked poses, the following 30 complexes are excluded (leaving 333 remaining) from all rigid conformer docking comparisons against Gnina: 6HZB, 6E4C, 6PKA, 6E3P, 6OXT, 6OY0, 6HZA, 6E6W, 6OXX, 6HZD, 6K05, 6NRH, 6OXW, 6RTN, 6D3Z, 6HLE, 6PY0, 6OXS, 6E3O, 6HZC, 6Q38, 6E7M, 6OIE, 6D3Y, 6D40, 6UHU, 6CJP, 6E3N, 6Q4Q, 6D3X. Scoring comparisons include all test complexes except 6A73, for which decoy poses could not be generated.

We download the 77 PDB IDs provided in Tosstorff et al. (2022) from the PDB to form the PDE10A dataset, keeping the A chain of each assymetric unit and the Ligand of Interest (LOI) interacting with it. We then align all ligands to the crystal structure of 5SFS using the procedure described in Corso et al. (2023) for aligning ESMFold structures, except transforming the ligand rather than the protein. This constitutes the construction of a *cross-docking* dataset due to the use of the same pocket for all ligands. Due to RDKit parsing errors with the Gnina-docked poses, the following 7 PDB IDs are excluded from all comparisons: 5SFA, 5SED, 5SFO, 5SEV, 5SF9, 5SDX, 5SFC. The remaining 70 ligands are shown superimposed on the 5SFS pocket in Figure 5.

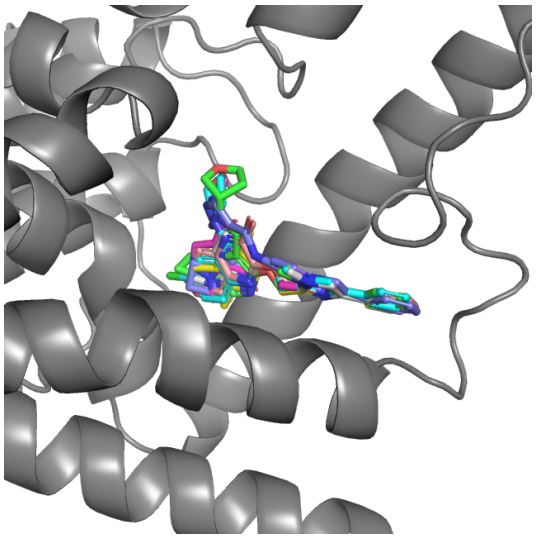

Figure 5: **PDE10A ligands** aligned on 5SFS

E HYPERPARAMETERS

Model hyperparameters The main model hyperparameters are the aspects of the scalar field parameterization (Equation 2) which dictate the number of coefficients $A_{cnj\ell m}$ predicted by the neural network. Intuitively, these determine the expressivity or "intrinsic resolution" of the scalar field which can be parameterized by the ESF neural network. These hyperparameters and their search spaces are detailed in Table 4, with full supporting experimental results in Appendix G.1.

Our data featurization and E3NN neural network architecture are largely adapted from the default settings of Corso et al. (2023). Both the protein and ligand neural networks operate on all heavy atom nodes; however, in the protein network only the alpha-carbons nodes emit $A_{cnj\ell m}$ coefficients that contribute to the scalar field. For simplicity, we also omit ESM features.

Table 4: Training-time model hyperparameters and their search spaces. The parameters corresponding to experiments in the main text are in bold.

Parameter	Search Space
# scalar field channels	3, **5**, 8
# radial basis functions	3, **5**, 8
RBF cutoff radius	3 Å, **5 Å**, 8 Å
Order of spherical harmonics	1, **2**, 3

Inference hyperparameters The primary inference-time hyperparameters control the "search resolution" over the FFT and grid-search degrees of freedom. For the grid-search, the resolution is controlled directly by the selection of grid points. For the FFT, the resolution is effectively controlled by frequency domain representation of the scalar field. Specifically,

- In the **translational** case, this resolution is expressed in terms of a sampling resolution which corresponds to the grid of frequency vectors at which the translational Fourier coefficients (Equation 5) are evaluated. To generate these frequency vectors, we use a 40 Å cubical domain with periodic boundary conditions.
- In the **rotational** case, this resolution is given by the maximum order of spherical harmonics in the global spherical basis (Equation 7) used for evaluating rotational cross-correlations.

The search space for these resolution hyperpameters is detailed in Table 5. Increasing the search resolution improves accuracy but at the cost of core optimization runtime. The default settings chosen for the main experiments are based on a qualitative assessment of the best tradeoff. Figure 6 visualizes this tradeoff, with full supporting experimental results in Appendix G.2.

Table 5: Inference-time model hyperparameters and their search spaces. The search grid points over $SO(3)$ are generated according to Zhong et al. (2019); Yershova et al. (2010). The translational search grid points are defined by evenly spacing the stated number of points along each dimension of a 8 Å cube. The parameters corresponding to experiments in the main text are in bold.

Procedure	Parameter	Search Space
TF	Translational signal resolution	0.5 Å, **1 Å**
	Rotataional search grid points	576, **4608**
RF	Rotational signal order	10, **25**, 50
	Translational search grid points	7^3, $\mathbf{9^3}$, 13^3

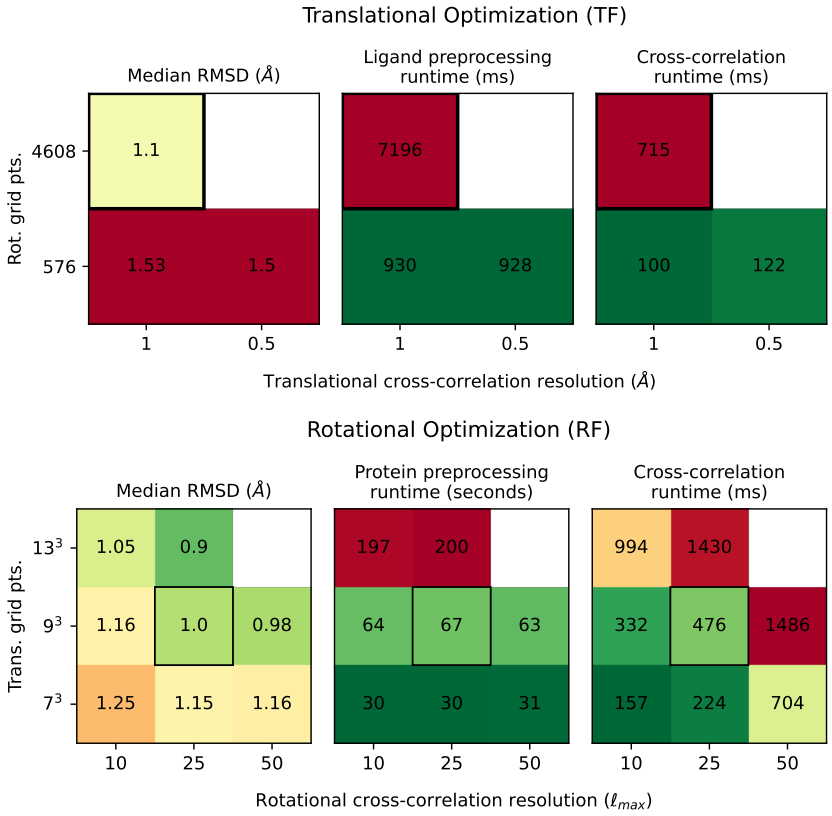

Figure 6: Inference-time hyperparameters and their effect on performance and average runtime. In the **TF** procedure (*top*), the protein preprocessing time is roughly constant and is not shown; similarly in the **RF** procedure (*bottom*) the ligand preprcessing time is constant. The selected default settings are boxed. All numbers are taken from full results on PDBBind in Appendix G.2

F RUNTIME AMORTIZATION

In this section we discuss in greater detail the practical implications of runtime amortization for various common workflows in molecular docking, focusing on rigid conformer docking as a proxy task. Conceptually, the total runtime of a docking workflow is given by

$$C_{\text{prot}} \cdot \begin{pmatrix} \text{\# unique} \\ \text{proteins} \end{pmatrix} + C_{\text{lig}} \begin{pmatrix} \text{\# unique ligand} \\ \text{conformers} \end{pmatrix} + C_{\text{pair}} \cdot (\text{\# complexes}) \tag{35}$$

where $C_{\text{prot}}, C_{\text{lig}}, C_{\text{pair}}$ refer to the protein precomputations, ligand precomputations, and cross-correlation (i.e., pose optimization) runtimes, respectively. If the number of complexes grows faster than either the number of unique proteins or ligand conformers individually, then the runtime contributions from the latter can be *amortized* across the complexes—meaning that the unit cost per complex *decreases* as their number increases.

To illustrate this amortization, suppose that we have N protein pockets (for simplicity we assume each is on a seperate protein) and M rigid conformers and wish to dock all MN complexes. Then, following the exposition and typical runtimes provided in Section 3.4, we can compute (Table 6) the total docking cost of this workflow for the **TF** procedure:

$$(72 \text{ ms}) \cdot M + (7400 \text{ ms}) \cdot N + (740 \text{ ms}) \cdot MN \tag{36}$$

Similarly, the total cost under the **RF** procedure is:

$$(58400 \text{ ms}) \cdot M + (21 \text{ ms}) \cdot N + (470 \text{ ms}) \cdot MN \tag{37}$$

Note that $C_{\text{prot}} \ll C_{\text{lig}}$ for the **TF** procedure, whereas $C_{\text{prot}} \gg C_{\text{lig}}$ for the **RF** procedure. This is because that the grid search over *rotations* in TF involves a new set of translational Fourier coefficients for each rotation of the ligand, while the protein is kept fixed; on the other hand, the grid search over *translations* in **RF** requires a new global spherical expansion $B_{nj\ell m}$ of the protein around each grid point, but only once for the ligand around its center of mass. This discrepancy has significant implications for the suitability of each procedure depending on the behavior of M, N:

- In the **virtual screening**-like setting, we keep a fixed protein ($M = 1$) and let the number of ligands increase ($N \to \infty$). Hence, the **TF** procedure is unsuitable, as the cost per ligand (and thus per complex) is a constant ≈8 seconds, approximately on the same order as traditional docking methods. On the other hand, the cost per ligand under the **RF** procedure begins at nearly ≈60 s with a single ligand, but asymptotically approaches ≈500 ms as the number of ligands increases (Figure 7, *left*).
- In the **inverse-screening**-like setting, we keep a fixed ligand ($N = 1$) and let the number of proteins increase ($M \to \infty$). Now the **RF** procedure is unsuitable, as the cost per protein is a constant ≈60 s, which is far too high to be useful. On the under hand, the cost per protein under the **TF** procedure begins at ≈8 s and asymptotically approaches ≈800 ms as the number of proteins increases (Figure 7, *right*).
- In the **general setting**, we divide both Equation 36 and 37 by MN to find that when $M \gg N$ (i.e., more proteins than ligands), the **TF** procedure is more efficient, and when $M \ll N$ (more ligands than proteins), the **RF** procedure is more efficient. However, both procedures converge to similar per-complex runtimes (740 ms vs 470 ms) in the limit of $M, N \to \infty$.

Table 6: Computing the runtime of the **TF** and **RF** procedures as a function of the number of proteins M and ligands N. T and R are the number of points used in the grid search over rotations and translations for **TF** and **RF**, respectively; they are fixed constants on the order of ≈1000 (we use $T = 729$, $R = 4608$ by default). The rows in red contribute the greatest preprocessing runtime.

Translational Optimization (TF)

Computation		Count	Runtime
Protein scalar field		$M\times$	65 ms
Protein FFT coeffs		$M\times$	7.0 ms
Ligand scalar field		$N\times$	4.3 ms
Ligand FFT coeffs		$RN\times$	1.6 ms
Cross-correlation	$+$	$RMN\times$	160 μs
	$=$		(Equation 36)

Rotational Optimization (RF)

Computation		Count	Runtime
Protein scalar field		$M\times$	65 ms
Protein global coeffs		$TM\times$	80 ms
Ligand scalar field		$N\times$	4.3 ms
Ligand global coeffs		$N\times$	17 ms
Cross-correlation	$+$	$TMN\times$	650 μs
	$=$		(Equation 37)

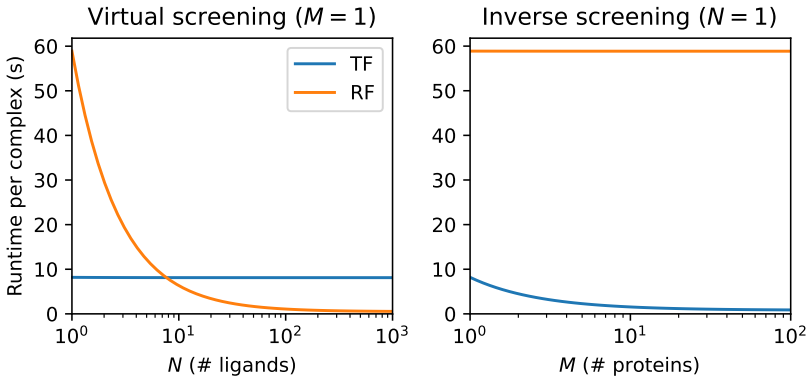

Figure 7: Amortized conformer docking runtimes per protein-ligand complex in the virtual screening (*left*) and inverse screening (*right*) settings.

G ADDITIONAL RESULTS

G.1 MODEL HYPERPARAMETERS

In Table 7, we explore the impact of varying model hyperparameters that determine the expressivity of the scalar field from their default values. We use the **ESF-N** model and evaluate on the rigid conformer docking task on PDBBind crystal structures. As the results illustrate, increasing the expressivity of the scalar field can bring additional improvements in performance. On the other hand, decreasing the expressivity degrades the performance of our method to the extent that it is no longer competitive with the baselines (Table 3).

Table 7: Rigid conformer docking results for varying model hyperparameters. Runtimes are shown averaged over complexes, excluding / including pre-computations that can be amortized.

	TF			**RF**		
Method	% <2 Å	Med. RMSD	Runtime	% <2 Å	Med. RMSD	Runtime
Baseline	72	1.10	0.7 s / 8.2 s	73	1.00	0.5 s / 68 s
# channels = 3	70	1.07	0.8 s / 6.5 s	71	0.98	0.4 s / 70 s
# channels = 8	71	1.15	0.9 s / 12 s	70	1.09	0.7 s / 71 s
# RBFs = 3	63	1.27	0.9 s / 8.0 s	64	1.13	0.5 s / 70 s
# RBFs = 8	77	1.07	0.8 s / 9.3 s	76	0.95	0.5 s / 74 s
RBF cutoff = 3 Å	72	1.16	0.8 s / 8.4 s	72	1.00	0.5 s / 68 s
RBF cutoff = 8 Å	76	1.16	0.8 s / 8.4 s	71	1.08	0.5 s / 69 s
Spherical harmonics $\ell_{\text{max}} = 1$	62	1.35	0.9 s / 7.6 s	62	1.23	0.5 s / 62 s
Spherical harmonics $\ell_{\text{max}} = 3$	80	0.99	0.9 s / 10 s	77	0.87	0.5 s / 81 s

G.2 INFERENCE HYPERPARAMETERS

In Tables 8–11 below, we explore the impact of inference-time hyperparameters on the performance and runtime of our method on the rigid conformer docking task. We use the **ESF-N** model variant and experiment with the PDBBind crystal test set and PDE10A test set. As discussed in Appendix E, for the **TF** procedure, we adjust (1) the number of grid points over $SO(3)$ (576 or 4608); and (2) the spatial resolution for translational cross-correlation (either 1 Å or 0.5 Å). For the **RF** procedure, we adjust (1) the number of translational grid points in the 8 Å cube (7^3, 9^3, or 13^3); and (2) the resolution for the rotational cross-correlation, expressed in terms of the maximal spherical harmonic order (10, 25, or 50). The rows corresponding to results in the main Table 3 are bolded.

In all rows, the effective number of poses searched over via both degrees of freedom is computed. To provide an idea of the impact of discretization, we compute the median RMSD of the *closest grid point* to the ground-truth pose (decomposed into rotational and translational contributions). This serves as a hard lower bound for the median RMSD of the output docked pose. In the **TF** procedure, increasing the resolution is memory-intensive; thus, the **RF** procedure is more effective at leveraging FFT to conduct fine-grained search over the accelerated degree of freedom. The default reported performance is attained with a translational offset of 0.4 Å and a rotational offset of 0.16 Å. While performance improves with smaller grid offsets, the returns are rapidly dimishing.

The runtime of the method (averaged over 333 PDBBind complexes and 70 PDE10A complexes) is reported and color-coded according to Appendix B: protein preprocessing (green), ligand preprocessing (blue), and the pose optimization (red). The effect of the non-FFT grid resolution is also color-coded, i.e., in **TF** the explicit enumeration over $SO(3)$ grid points directly scales the ligand preprocessing, whereas in **RF** the enumeration over $\mathbb{R}^3$ scales the protein preprocessing. As the tables show, the preprocessing of these explicit grid points contributes to the majority of the non-amortizeable runtime. In general, the $SO(3)$ grid / ligand preprocessing in **TF** is less expensive, however, it cannot be amortized when moving from PDBBind to PDE10A (where the ligands are still distinct). On the other hand, the $\mathbb{R}^3$ grid / protein preprocessing time in **RF** is significantly reduced (very roughly on the order of 70-fold, as expected) in PDE10A compared to PDBBind.

Table 8: **PDBBind TF**

Trans. resol.	$SO(3)$ grid pts.	Effective # poses	Grid offset Tr.	Grid offset Rot.	Grid offset All	Med. RMSD	% <2Å	Runtime (ms) Prot. prep.	Runtime (ms) Lig. prep.	Runtime (ms) Opt.
1 Å	576	420k	0.52	0.84	0.98	1.53	63	65	931	100
1 Å	**4608**	**3.4M**	**0.50**	**0.42**	**0.67**	**1.10**	**72**	**72**	**7196**	**715**
0.5 Å	576	2.8M	0.25	0.80	0.84	1.50	64	70	928	123

Table 9: **PDBBind RF**

Trans. grid pts.	$SO(3)$ ℓ_{max}	Effective # poses	Grid offset Tr.	Grid offset Rot.	Grid offset All	Med. RMSD	% <2Å	Runtime (ms) Prot. prep.	Runtime (ms) Lig. prep.	Runtime (ms) Opt.
7^3	10	3.2M	0.65	0.38	0.80	1.25	70	30k	85	158
7^3	25	45M	0.67	0.15	0.70	1.15	69	31k	87	225
7^3	50	353M	0.65	0.08	0.67	1.16	70	32k	85	704
9^3	10	6.8M	0.49	0.36	0.64	1.16	73	64k	85	333
$\mathbf{9^3}$	**25**	**97M**	**0.50**	**0.15**	**0.53**	**1.00**	**73**	**67k**	**87**	**476**
9^3	50	751M	0.51	0.08	0.52	0.98	74	63k	84	1487
13^3	10	20M	0.33	0.37	0.51	1.05	74	198k	85	995
13^3	25	291M	0.33	0.15	0.37	0.90	72	200k	86	1430

Table 10: **PDE10A TF**

Trans. resol.	$SO(3)$ grid pts.	Effective # poses	Grid offset Tr.	Grid offset Rot.	Grid offset All	Med. RMSD	% <2Å	Runtime (ms) Prot. prep.	Runtime (ms) Lig. prep.	Runtime (ms) Opt.
1 Å	576	420k	0.51	0.88	1.00	1.85	56	22	761	89
1 Å	**4608**	**3.4M**	**0.50**	**0.48**	**0.69**	**1.11**	**64**	**21**	**6159**	**736**
0.5 Å	576	2.8M	0.26	0.89	0.93	2.05	50	20	756	106
0.5 Å	4608	23M	0.26	0.44	0.51	1.00	73	20	6147	892

Table 11: **PDE10A RF**

Trans. grid pts.	$SO(3)$ ℓ_{max}	Effective # poses	Grid offset Tr.	Grid offset Rot.	Grid offset All	Med. RMSD	% <2Å	Runtime (ms) Prot. prep.	Runtime (ms) Lig. prep.	Runtime (ms) Opt.
7^3	10	3.2M	0.72	0.38	0.83	1.60	54	476	44	161
7^3	25	45M	0.57	0.16	0.59	1.21	63	549	42	227
7^3	50	353M	0.65	0.08	0.65	1.30	64	635	59	718
9^3	10	6.8M	0.46	0.39	0.63	1.05	64	1014	42	327
$\mathbf{9^3}$	**25**	**97M**	**0.48**	**0.16**	**0.51**	**1.00**	**70**	**946**	**43**	**465**
9^3	50	751M	0.49	0.09	0.50	0.99	64	943	42	1483
13^3	10	20M	0.34	0.41	0.55	1.17	64	2798	42	986
13^3	25	291M	0.33	0.16	0.36	0.96	69	2912	45	1469

G.3 PERFORMANCE-RUNTIME TRADEOFF

In Figure 8, we further investigate the tradeoff between speed and performance offered by our method compared to Gnina (with the Vina scoring function). While in the main results (Table 3) we run Gnina using all default settings, it is possible to reduce the runtime (and performance) by adjusting these settings. In particular, we explore setting `--max_mc_steps` and `--minimize_iters` to 5 independently and in combination. Together with the default runs and the `--score_only` runs, these trace out a *Pareto frontier* representing the tradeoff between runtime per complex and <2 Å RMSD success rate. With the default settings, Gnina outperforms all variants of our method on the PDBBind crystal and PDE10A test sets. However, Figure 8 shows that we can reach previously inaccessible regions in the accuracy v.s. runtime tradeoff landscape.

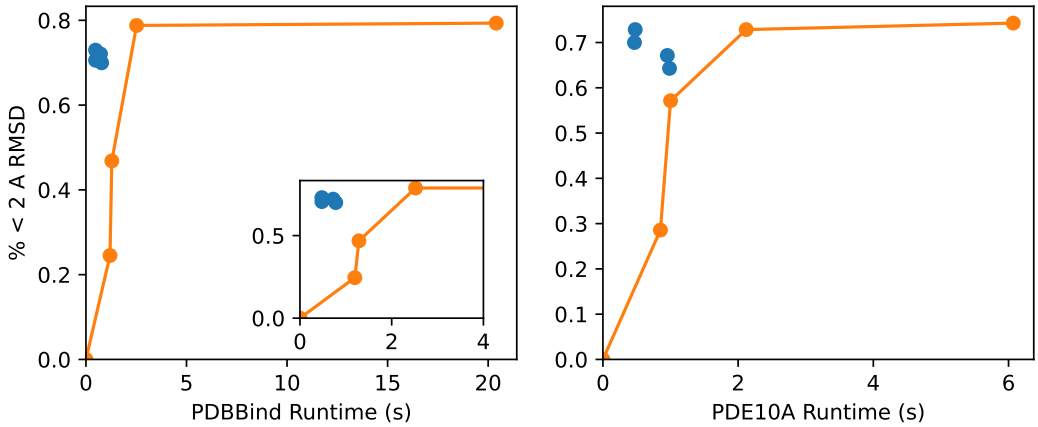

Figure 8: **Tradeoff between speed and accuracy** using our method compared to Gnina on PDBBind crystal structures (*left*) and PDE10A (*right*). In both cases, variants of our method (blue dots) enable possibilities not reachable with Gnina (orange curve).