SFCalculator: connecting deep generative models and crystallography

Minhuan Li[1], Kevin Dalton[2,3], Doeke Hekstra[1,2*]

[1]John A. Paulson School of Engineering & Applied Sciences, Harvard University.
[2]Department of Molecular & Cellular Biology, Harvard University.
[3]LCLS Data Systems, SLAC National Accelerator Laboratory .

*Corresponding author(s). E-mail(s): doeke_hekstra@harvard.edu;
Contributing authors: minhuanli@g.harvard.edu;
kmdalton@slac.stanford.edu;

Abstract

Proteins drive biochemical transformations by transitioning through distinct conformational states. Understanding these states is essential for modulating protein function. Although X-ray crystallography has enabled revolutionary advances in protein structure prediction by machine learning, this connection was made at the level of atomic models, not the underlying data. This lack of connection to crystallographic data limits the potential for further advances in both the accuracy of protein structure prediction and the application of machine learning to experimental structure determination. Here, we present SFCalculator, a differentiable pipeline that generates crystallographic observables from atomistic molecular structures with bulk solvent correction, bridging crystallographic data and neural network-based molecular modeling. We validate SFCalculator against conventional methods and demonstrate its utility by establishing three important proof-of-concept applications. First, SFCalculator enables accurate placement of molecular models relative to crystal lattices (known as phasing). Second, SFCalculator enables the search of the latent space of generative models for conformations that fit crystallographic data and are, therefore, also implicitly constrained by the information encoded by the model. Finally, SFCalculator enables the use of crystallographic data during training of generative models, enabling these models to generate an ensemble of conformations consistent with crystallographic data. SFCalculator, therefore, enables a new generation of analytical paradigms integrating crystallographic data and machine learning.

Keywords: Protein Structure, Protein Dynamics, Generative Models, X-ray crystallography, Structure Factors

1 Introduction

Machine learning (ML) is making rapid strides in the prediction of biomolecular structures and their conformational ensembles. Key capabilities include the ability to predict structure from sequence with near-experimental accuracy[4, 30, 36], to design practically viable structures from geometric or functional constraints[14, 19, 66], and to predict ensembles of possible conformations given an initial structure[29, 35]. The capability to generate credible conformations as a function of a sequence, sequence alignment, or a latent variable overcomes some sampling limitations inherent in stepwise Molecular Dynamics and Monte Carlo algorithms.

X-ray crystallography, on the other hand, has enabled experimental structure determination for many biomolecules, including assemblies such as ribosomes, nucleosomes, and proteasomes, and a wealth of smaller systems such as enzymes, ion channels, GPCRs, and kinases. Spurred on by powerful new X-ray Free Electron Lasers[49, 61] and synchrotron beamlines[26, 58], X-ray crystallography has also begun to enable visualization of the dynamics of proteins on timescales from femtoseconds to seconds—for example visualizing enzyme catalysis[13, 60], photosynthesis[6], K^+ ion channel permeation[34], and the first events in vision[24]—and across physical and chemical conditions—for example enabling crystallographic screening of interactions of drug targets with thousands of drug fragments[62].

Generative ML models and X-ray crystallographic data could be productively combined to improve the accuracy of generative models, and the accuracy and throughput of crystallographic structure determination. Recent work has taken important steps in this direction. McCoy et al.[40] showed that AlphaFold models are often sufficiently accurate to provide an initial solution to the crystallographic phase problem. Terwilliger et al.[64] showed that by iteratively feeding intermediate structure refinement models as templates into the AlphaFold module one can achieve higher-quality models.

This iterative procedure is necessary, however, because there was no direct interface between predictive models and crystallographic data.

Differentiable algorithms that integrate cryo-electron microscopy (cryo-EM) and nuclear magnetic resonance (NMR) data with machine learning have significantly accelerated structure determination, boosting both efficiency and throughput[33, 69]. Here, we introduce SFCalculator, a fast, differentiable likelihood-calculation tool that connects structural models—complete with solvent corrections—to crystallographic observables. SFCalculator is designed as a bridge between the rapidly evolving machine learning ecosystem and crystallographic datasets, as illustrated in Fig. 1.

We validate SFCalculator by benchmarking it against conventional calculations and demonstrate its utility through three examples. First, we show how SFCalculator can be combined with a hierarchical grid search algorithm[68, 69] and gradient-based optimization [32] to place a search model (here, an AlphaFold prediction, Figure 1a) relative to a protein crystal lattice, a procedure known as molecular replacement (Figure 1b). We then asked if the latent space of a pre-trained conformational generative model, the Boltzmann Generator [47], could serve as a search space to find a protein conformation maximally consistent with experimental data (Figure 1c). By doing so, we find that AlphaFold models can be refined to yield conformations with excellent stereochemistry and fit to the data.

Finally, we show how SFCalculator can be used to train generative models on crystallographic data, offering, at once, a perspective on the further improvement of generative models, and a principled strategy for tackling the crystallography inverse problem: inference of conformational ensembles that exhibit both strong data fit and favorable physical energies. These approaches naturally incorporate constraints encoded in the generative model design, potentially obviating the need for explicit physical restraints[25, 44] during refinement.

2 Results

SFCalculator provides a fast and differentiable interface

SFCalculator relates atomic models of protein structure to the primary crystallographic observables. Specifically, X-ray scattering methods measure the amplitudes of the Fourier components (structure factors) of the electron density of a sample. SFCalculator, likewise, provides access to the Fourier components of the electrostatic potential map and thus, in principle, supports electron scattering methods like cryo-electron microscopy and electron diffraction—we will address this further in future work.

The electron density of a protein crystal includes contributions from both macromolecular atoms and the surrounding solvent. The disordered solvent component is particularly significant in low-resolution data. A widely implemented approach for modeling this solvent contribution is the bulk-solvent model, which presumes a uniform solvent density in all regions not occupied by the atomic structure. A more refined variant, known as the probe-shrink model[28], permits variations in solvent density within a boundary region at the interface between the solvent and macromolecule to account for the ordered hydration layer. This model has become the default in contemporary crystallographic refinement programs[3, 45].

Despite its utility, the discretization and rounding operations inherent in this approach hinder differentiability, thereby precluding the simultaneous optimization of the macromolecular and bulk-solvent components during refinement. To address this, Fenn et al.[18] introduced a differentiable algorithm that smooths the solvent boundary using Gaussian or polynomial functions. However, this method is computationally intensitve and relies on Babinet's principle[43], neglecting the influence of the hydration layer. In SFCalculator, we have designed algorithms to ensure that both contributions are incorporated in a fast and differentiable manner, as illustrated in

Fig.2 (a). We adopt the direct summation method for the macromolecular contribution and a cutoff-sigmoid approximation of the solvent mask (see Methods). To assess the accuracy and performance of SFCalculator, we first performed a rigorous comparison with the widely used software PHENIX [3]. For a comprehensive evaluation, the comparison was conducted across a large sample of PDB entries (N=868), covering data resolutions from 0.8 to 3.0 Å, various spacegroups, and a range of molecular sizes (see Fig. A1). The results demonstrate strong agreement with PHENIX across key metrics, while showcasing the computational efficiency and scalability of SFCalculator.

A comparative analysis of solvent masks was performed between the probe-shrink method[28], the default in PHENIX, and the differentiable solvent mask generated by SFCalculator. Fig. 2(b) presents slice views of the probe-shrink mask, as comparison to our differentiable mask slice in Fig. 2(a). While the overall shapes of the two masks closely agree, notable differences are evident in the boundary regions. The probe-shrink method provides more nuanced details in these areas, primarily due to its "shrink" step, which excludes the hydration layer surrounding the macromolecule. This adjustment is crucial, as solvent molecules in the hydration layer are often ordered and are not adequately represented by a flat solvent model.

To assess the agreement between structure factors generated by SFCalculator and PHENIX, we calculated their correlation. For the protein component, SFCalculator shows perfect correlation with PHENIX (Fig. 2c), demonstrating the accuracy of its calculations. For the solvent component (Fig. 2d), high correlations are observed at low resolutions (high Å values) with less correlation at high resolutions due to differences in handling of the protein-solvent boundary.

We further compared the quality of structure factors calculated for the same deposited models by SFCalculator and PHENIX, using the cross-validation metric R_{free} to relate structure factors calculated by both to the data (see Methods). Overall, SFCalculator achieves high consistency, with discrepancies primarily for dataset/-model combinations with high R_{free} (Fig. 2e). These discrepancies are primarily due to the larger contribution of solvent at low resolution, where differences between the solvent models of SFCalculator and PHENIX become more pronounced. Despite these differences, SFCalculator maintains robust performance across a wide range of resolutions.

SFCalculator is computationally efficient. In performance tests comparing forward and backward computations, SFCalculator achieved speeds 50–200 times faster than PHENIX's phenix.fmodel (Fig. 2f), which operates exclusively on CPUs. This large performance gain derives from the ability of GPUs to leverage parallel processing and the differentiable backends integrated into SFCalculator.

We further evaluated the memory efficiency of SFCalculator (Fig. 2g). We find that SFCalculator can handle proteins with approximately 1000 residues at a resolution of 2 Å (PDB id: 4PKF) on a single Nvidia A100 80GB GPU. Handling even larger systems can be achieved by further partitioning certain bottleneck operations that involve large matrices with $N_{HKL} \times N_{atoms}$ elements. This scalability underscores the potential for large-scale applications in structural biology, making SFCalculator a valuable tool for analyzing complex macromolecular systems.

Taken together, these results establish the accuracy, efficiency, and scalability of SFCalculator, positioning it as a promising complement to established tools like PHENIX, for workflows requiring high throughput and differentiability.

SFCalculator Enables Efficient Molecular Replacement

As a first test of SFCalculator's potential to facilitate crystallographic data processing, we used it to implement a simple molecular replacement (MR) algorithm. MR is used in crystallography to find an initial, approximate solution to the phase problem and is practically useful when the target molecule shares structural similarity with a previously solved model. With the advent of highly successful protein structure prediction algorithms, many targets that were previously challenging or intractable––particularly those involving novel folds––can now be readily solved using MR [5, 31, 40, 42]. The immediate goal of MR is to establish the position and orientation of a protein relative to the crystal lattice—inherently, a 6-dimensional search. MR has been extensively studied, resulting in the development of several efficient programs[23, 39, 41] based on a decomposition of the 6D search into subsequent rotational and translational searches, comparing observed and calculated Patterson functions as proposed by Rossmann and Blow [56]. Our implementation follows the same principles but incorporates a hierarchical rotational grid constructed using Hopf Fibration [68, 69] (see Methods).

We illustrate molecular replacement using SFCalculator using a crystallographic dataset of the inverting cellulase PcCel45A, a fungal endoglucanase (PDB id: 3X2I) with a "Newton's cradle proton relay mechanism in its active site[46]. This enzyme crystallizes in an orthorhombic space group, $P2_12_12_1$, with symmetry-related hydrolase molecules within the unit cell. Each molecule represents a distinct ground-truth solutions for the pose search (Fig. 3 a).

The molecular replacement procedure begins with an initial model generated by AlphaFold2. Due to the random pose assignment of the predicted model, the starting R_{free} is extremely poor (0.659), as shown in Fig. 3 b, as is the Root-Mean-Square Deviation (R.M.S.D.) between the initial pose and any of the four ground-truth poses, ranging from 17.0 to 24.9 Å (Fig. 3c).

To find an approximate pose, we then employed a hierarchical grid search in decoupled rotation and translation spaces. This search identified a poses similar to one of the ground-truth poses (specifically, the pink copy), resulting in an R.M.S.D. of 1.12 Å and R_{free} of 0.509 (Fig. 3d). Subsequent gradient descent, relying on autodifferentiation enabled by SFCalculator, strongly improves R.M.S.D., to 0.51Å, and R_{free}, to

0.416, as shown in Fig. 3e. It is remarkable that, even after multiple rounds of hierarchical grid search——with the final round featuring a 0.95 degree step for rotation and 0.75 Å step for translation, gradient descent can still achieve such large improvements just by improving pose. This observation highlights a key limitation of hierarchical grid search: sensitivity to the initial candidates. If the initial base grid is not sufficiently fine, the search is prone to converging on local minima. As the computational cost of increased grid resolution scales cubically in step size, finer grids are impractical, underscoring the value of the final gradient descent step.

SFCalculator Enables Structure Refinement Using Generative Models

Initial structural models of a protein for an experimental condition of interest are often fraught with inaccuracies——errors that can obscure critical details of molecular mechanisms. Structure refinement addresses this by optimizing structural parameters such as atomic positions and atomic displacement factors (e.g., B factors). Structure refinement using experimental data generally faces two fundamental obstacles. First, experimental data rarely suffice to yield physically accurate models without the use of additional stereochemical information on, e.g. bond lengths, angles, and preferred side chain orientations, during model building. Second, the energy landscapes of proteins are extremely high-dimensional and 'rugged', with many local minima separated by barriers.

Current approaches to structure refinement typically address the need to include stereochemical information by combining a data term with explicit restraint terms in the overall objective function to be minimized. These restraints can be derived from high-resolution protein and small-molecule crystal structures[25], quantum calculations[38], or molecular mechanics forcefields [27][9][44]. Efficient navigation of conformational landscapes is typically achieved through a heuristic combination of different sampling and gradient-descent steps using several model representations [3].

We reasoned that conformational generative models could help address both major obstacles by enabling alternative approaches to protein structure refinement. Generative models are often trained to transform coordinates from internal ”latent-space” representations with relatively simple properties (e.g., following a Gaussian distribution) to complex real-world representations (e.g., protein atomic structures). As such, their latent spaces might be more readily traversed during structure refinement than traditional coordinate representation spaces, while their coordinate transforms may have learned to account for protein physics, enabling optimization of model fit to experimental data (a likelihood function) without the use of explicit constraints or restraints on conformations.

SFCalculator makes it possible to test this idea by providing a fully (auto)differentiable link between models and experimental likelihood evaluations. To do so, we adopted the normalizing flow architecture[15, 50] as the foundational framework for our model, building on the Boltzmann Generator approach pioneered by Noé and colleagues[47]. In this approach, we learn a transformation $x = T_\theta(z)$, from latent variable z to protein coordinates x. z follows a smooth, easily sampled distribution, such as a Gaussian ($z \sim \mu(z) = \mathcal{N}(0, I)$). Under the change of variables formula, the induced distribution in x-space can then be expressed as

$$q_\theta(x) = \mu_z(z) \left| R_{xz}(z, \theta) \right|^{-1} \tag{1}$$

where R_{zx} represents the determinant of the Jacobian matrix of the transformation $T_\theta(z)$. The objective is to learn model parameters θ such that the variational distribution $q_\theta(x)$ approximates the true Boltzmann distribution of protein conformations. A key advantage of the normalizing flow framework is its inherently invertible transformation design, which ensures compatibility with any initial structure model, allowing seamless integration without introducing deviations. The schematic architecture is composed of several key blocks, which we describe with detail in Appendix B.4.

We reimplemented the Boltzmann Generator of Noé et al., as depicted in Fig. 4a, employing two complementary training objectives based on correspondence with molecular dynamics (MD) snapshots and physical energy calculations for self-generated snapshots, as described in Methods. We illustrate the results with the same PcCel45A dataset used to illustrate Molecular Replacement. As intended, the generated samples displayed reasonable energies (Fig. 4c). We then froze the parameters θ of the Boltzmann Generator.

Next, we examined whether the latent space of this generative model would allow for efficient refinement of protein structure by unconstrained stochastic gradient optimization of the experimental data likelihood (see Methods) given the model, as illustrated in Fig. 4b. Importantly, the differentiability of SFCalculator enables gradient information to flow back, guiding the search in the latent space $\mathbf{z}$.

The optimization trajectory is shown in Fig. 4d. Our method achieved a data fit, measured by R_{free} , similar to the model refined by PHENIX, while the molecular mechanics energy from our refinement ($-4,105\ kT$) was significantly more reasonable than that of the PHENIX-refined model ($+6,584\ kT$). This difference primarily arises from PHENIX's use of a limited set of geometric restraints [25]. While effective, these restraints do not accurately account for features such as atomic clashes, and hydrogen bonding and electrostatic interactions.

In Fig 4e, we visualize the structural improvements achieved through our refinement process, on both the protein backbone and side chains. These results validate the utility of SFCalculator and demonstrate the potential of using deep generative models for structural refinement.

SFCalculator Enables Ensemble Refinement by Generative Modeling

In the above example, we first trained a generative model and then used its latent space as a search space to find a conformation that best accounted for the experimental data. SFCalculator also makes it possible to directly include crystallographic data during training of generative models. Notably, of course, many conformational generative models, including AlphaFold [30], have been trained on atomic models derived from crystallographic data, but none have been trained on the crystallographic data themselves. As a first example of doing so, we include crystallographic data during the training of a Boltzmann Generator in order to obtain conformational ensembles consistent with crystallographic data. Training a generative model this way serves as a form of ensemble refinement[10], yielding a variational approximation [55] of the true posterior distribution of conformations of a protein consistent with physical prior information and the crystallographic data.

As illustrated in Fig. 5a, the target distribution follows the unnormalized posterior distribution:

$$p^*(x) \propto p(x) \cdot p\left(F_o \mid x, \sigma_F\right)$$

where $p(x)$ represents the physical prior Boltzmann distribution, and $p\left(F_{o,} \mid x, \sigma_F\right)$ corresponds to the experimental likelihood of the observed structure factor amplitudes F_o given the protein coordinates x, and measurement uncertainty σ_F. Once the normalizing flow is trained, the model itself serves as an approximation of the posterior distribution. Consequently, samples generated from the model can be interpreted as effective results of ensemble refinement.

We evaluated this ensemble refinement approach on two systems: the hydrolase PcCel45A used above, and the proline isomerase Cyclophilin A (CypA, PDB ID:

3K0N). The results demonstrate the advantages of approximating the posterior distribution with flow-based generative models to achieve physically meaningful and experimentally consistent conformational ensembles.

For the hydrolase, as shown in Fig. 5c, the flow ensemble models strikes a balance between consistency with the data ($R_{\text{work}}/R_{\text{free}}$ =0.278 / 0.301) and a more reasonable MM energy E = -7,020 ± 177 kT. Further analysis of key residues highlights the structural improvements achieved by flow-based ensemble refinement, as illustrated by Fig. 5d for residues ALA29, GLY91, GLU117, and LYS124. When overlaid with electron density maps, the models refined by the flow ensemble show corrected backbone and side chain conformations, providing good agreement with the experimental data.

We further tested the refinement method on Cyclophilin A (CypA, PDB ID: 3K0N), a system known for exhibiting an important network of alternative conformations of residues along its central beta sheet [20], including PHE113. The flow ensemble successfully captures both conformational states of PHE113, as shown by the overlay with the electron density map in Fig. 5e. Additionally, the torsion angle distribution plot for χ_1 and χ_2 of PHE113 reveals the broader conformational diversity achieved by the flow ensemble compared to other models. This demonstrates the ability of the flow ensemble to capture diverse and physically meaningful conformational states.

3 Discussion

A defining strength of SFCalculator is its differentiability, which seamlessly connects crystallographic data with machine learning models for protein structure generation. This capability enables gradient-based optimization, allowing crystallographic observables to be integrated into end-to-end machine learning pipelines for joint refinement, structural validation, and iterative model improvement. Its scalability ensures compatibility with large datasets and high-throughput workflows, positioning SFCalculator as a critical bridge between traditional crystallographic methods and emerging computational approaches. This integration accelerates advancements in protein structure prediction and generation, which are addressing long-standing challenges in structural biology and opening new avenues for understanding molecular function and interactions.

Despite advances in modern refinement algorithms, significant challenges remain, particularly in automation. The rugged, non-convex optimization landscape of macromolecular refinement often leads algorithms to local minima. Overcoming this requires careful initialization and iterative strategies, such as simulated annealing, to navigate toward globally optimal solutions. However, automated methods still struggle to replicate the nuanced, context-dependent decisions made during expert manual inspection. Tasks like identifying alternative conformations or resolving ambiguities in poorly defined regions often rely on visual intuition and expertise[11, 17]. This reliance on manual intervention creates bottlenecks, limiting scalability for high-throughput datasets and underscoring the need for more adaptive and automated algorithms.

The complexity of the optimization landscape is closely tied to its representation. Transformations between coordinate systems, such as Cartesian and dihedral angles, can affect landscape navigability without eliminating inherent barriers. Generative modeling approaches in machine learning are designed to identify and exploit such transformations[7, 15, 47], facilitating the principled incorporation of diverse prior constraints. This makes the integration of generative models into refinement workflows particularly promising.

Leveraging the differentiability of SFCalculator, we demonstrate proof-of-concept applications that enhance the interpretation of crystallographic data within generative models of protein structure. By incorporating physical priors from molecular mechanics force fields——rather than relying solely on geometric constraints derived from small-molecule statistics——we establish a more physically grounded refinement framework. The combination of SFCalculator with deep generative models also enables rigorous variational inference, producing conformational ensembles that not only fit the experimental data but also maintain favorable physical energies. This synergy underscores the potential of integrating machine learning with crystallographic refinement to drive progress in structural biology.

Refinement of atomic displacement parameters

While the present work focuses on conformational refinement, two critical aspects of structural refinement remain unaddressed: the refinement of atomic displacement parameters (ADPs) and the placement of ordered solvent molecules, both of which are known to significantly impact model quality[3].

For ADP refinement in the guided search approach, we adopted a straightforward strategy: the ADPs from the PHENIX-refined model were copied and kept unchanged throughout the search. This ensures a fair comparison of R_{free} metrics between the methods. In the ensemble refinement approach, all ADPs were uniformly set to a small value of 5.0 Å^2, encouraging conformational exploration without biasing the refinement toward any particular set of displacement parameters. While gradient-based optimization of B-factors is straightforward to implement using SFCalculator, preliminary trials showed that refining ADPs led to negligible improvements in the final metrics. This suggests that further investigation and more comprehensive testing are required to better understand and optimize ADP refinement within this framework.

A better differentiable solvent model

The performed benchmarks revealed differences between the differentiable solvent mask implemented in SFCalculator and the Probe-Shrink method, particularly in the boundary regions. The Probe-Shrink method employs a non-differentiable "shrink" step which appears to better account for the hydration layer surrounding the macromolecule. This adjustment is essential, as solvent molecules within the hydration layer are often ordered and cannot be adequately modeled using a flat solvent approximation. As a result, SFCalculator yielded slightly higher R_{free} values compared to PHENIX in scenarios where solvent contributions were prominent, especially at lower resolution. Importantly, as well, our comparisons with PHENIX were based on models without explicitly modeled solvent molecules because the generative models used here do not generate ordered solvent molecules.

These observations emphasize the necessity for a more advanced differentiable solvent model. Potential avenues for improvement include the development of differentiable morphological operations or the integration of a data-driven pre-trained masking approach, both of which could leverage the inherent differentiability of SFCalculator framework to enhance accuracy and performance.

The placement of ordered solvent molecules presents a further challenge, particularly when MD-based generative models, due to the stringent requirements of molecular topology. However, the differentiable nature of SFCalculator offers a promising avenue for addressing this issue as well. It could enable the development of a solvent placement model conditioned on both the structural model and experimental observables. Such a predictive approach would represent a desirable solution to the problem, leveraging gradient information to integrate solvent modeling seamlessly into the refinement process.

Sensitivity to training scheme and mode collapse

The training process for current normalizing flow models can be particularly challenging, especially when incorporating objectives which involve the reverse Kullback-Leibler (KL) divergence. While forward KL-divergence encourages the learned distribution to be mass-covering, reverse KL-divergence drives the model to be mode-seeking. This tendency towards mode-seeking often results in mode collapse, where the learned distribution focuses on a narrow subset of the target distribution, neglecting other regions. To mitigate this issue, we used careful scheduling of the weights between these two objectives to balance mass-covering and mode-seeking behaviors, complicating the training process and limiting scalability. We emphasize that SFCalculator can be combined with any differentiable generative model and anticipate that these limitations will be overcome by the next generation of generative models.

4 Methods

Structure factor from protein molecules, $\mathbf{F}_{\text{protein}}$

We adopt the direct summation method for the macromolecular part contribution. As the Fourier transform is linear, the overall contribution from the whole molecule equals to the summation of all atoms[57]:

$$\mathbf{F}_{\text{protein}}\left(\vec{h}\right) = \sum_{G}\sum_{j} O_j \cdot f_{\vec{h},j} \cdot \text{DWF}(\vec{h}) \cdot \exp\left[2\pi i\vec{h} \cdot \left(\mathbf{R}_G\vec{x}_j + \vec{T}_G\right)\right] \tag{2}$$

where G is the index of symmetry operations (appearing as rotation matrix $\vec{R}_G$ and translation vector $\vec{T}_G$ given the space group; j is the atom index, O_j is occupancy and $\vec{x}_j$ is the fractional coordinates of atom j; $\vec{h}$ is the Miller index, $f_{\vec{h},j}$ is the atomic scattering factor for the atom type of atom j, and $\text{DWF}(\vec{h})$ is the Debye-Waller factor.

The atomic scattering functions are approximated with the nine-parameter Gaussian summation[12], which is used in nearly all protein crystallography programs to compute the wavelength-independent atomic scattering factor (or atomic form factor) $f_{\mathbf{S}}$ as a function of the scattering angle θ :

$$f_{\mathbf{S}} = \sum_{i=1}^{4} a_i \exp\left(-b_i|\mathbf{S}|^2/4\right) + c = \sum_{i=1}^{4} a_i \exp\left(-b_i(\sin\theta/\lambda)^2\right) + c$$

Here, a_i, b_i, and c are the Cromer-Mann coefficients. These coefficients are tabulated in the International Tables for Crystallography[53] In SFCalculator, atomic structure factor coefficients are accessed through GEMMI[67]. Additionally, to handle cryo-EM datasets, SFCalculator supports a "cryoem" mode in which the atomic structure factors are calculated using electron scattering factors parameterized as five Gaussians[53].

The Debye-Waller factor accounts for atomic displacement caused by thermal vibrations or structural disorder which results in slight positional variations of atoms within each unit cell, leading to additional phase differences. In crystallography, this effect is commonly parameterized as the B-factor. In isotropic case, the DWF becomes a standard Gaussian B-factor exponential:

$$\mathrm{DWF}_{\mathrm{iso}}\left(\vec{h}\right)=\exp\left[-B_{\mathrm{iso}}\left(\sin\theta/\lambda\right)^2\right]$$

In the anisotropic case, the displacement is usually parameterized with a symmetric matrix U_w, transforming the DWF into:

$$\mathrm{DWF}_{\mathrm{aniso}}\left(\vec{h}\right)=\exp\left(-2\pi^2\vec{h}^T U^*\vec{h}\right),\quad U^*=\mathbf{O}^{-1}U_w\left(\mathbf{O}^{-1}\right)^{\mathrm{T}}$$

where $\mathbf{O}^{-1}$ is the deorthogoniazation matrix of the unit cell. SFCalculator is able to handle both isotropic and anisotropic parametrizations.

Differentiable bulk solvent contribution $\mathbf{F}_{\mathbf{solvent}}$

Calculating the solvent mask in a differentiable manner is a non-trivial task. The widely used probe-shrink method generates the solvent mask in a semi-localized way[28], utilizing van der Waals atomic radii r along with two parameters, r_{probe} and r_{shrink} . While this approach achieves good agreement with experimental data, it involves non-differentiable operations such as rounding for discretization. Here, we propose an algorithm to approximate the solvent mask in a differentiable manner, as illustrated in Fig. 2a.

Algorithm 1 Differentiable solvent mask approximation

Require: $\mathbf{F}_{\mathrm{protein}}$

$\mathbf{F}_{P1} \leftarrow \mathrm{expand\_to\_P1}(\mathbf{F}_{\mathrm{protein}}, d_{min})$ ▷ Apply symmetry operations and low pass filter

$G \leftarrow \mathrm{reciprocal\_grid}(\mathbf{F}_{P1}, \mathrm{grid\_size})$ ▷ Assign value to reciprocal grid

$g \leftarrow \mathrm{real}(\mathrm{fft3d}(G))$ ▷ Fourier transform for real space grid with density

$g^* \leftarrow (g - \mathrm{mean}(g))/\mathrm{std}(g)$ ▷ Normalize the density

$\delta \leftarrow \mathrm{quantile}(g^*, \mathrm{solvent\_percentage})$ ▷ Get the solvent density cutoff

$g_s \leftarrow \mathrm{sigmoid}((\delta - g^*) * \mathrm{scale})$ ▷ Get solvent mask density grid

$G_s \leftarrow \mathrm{ifft3d}(g_s)$ ▷ Get reciprocal grid for solvent mask

$\mathbf{F}_{\mathrm{solvent}} \leftarrow \mathrm{assign\_value}(G_s, d'_{min})$ ▷ Apply low pass filter

The complex protein structure $\mathbf{F}_{\text{protein}}$, calculated as described earlier, is first subjected to a Fourier transform to generate a protein density map. Subsequently, a sigmoid operation is applied after subtracting the density cutoff to produce a binary-like solvent mask map. This map is then processed through an inverse Fourier transform to calculate the structure factors contributed by the bulk solvent. It is important to note that two low-pass filters are applied in conjunction with the Fourier and inverse Fourier transforms to suppress high-frequency noise in the solvent mask map. Resolution cutoff d'_{min} in the inverse Fourier transform is set as 3Å to be consistent with PHENIX default. This operation is conceptually aligned with the solvent flatness constraints previously proposed[59].

The algorithm incorporates three key hyperparameters: the low-pass filter cutoff d_{min}, the solvent percentage, and the sigmoid scale. The methodology for determining these hyperparameters and the impact of their adjustments on the performance of SFCalculator are included in Appendix B.2.

Scaling

After obtaining the two contributing components, $\mathbf{F}_{\text{protein}}$ and $\mathbf{F}_{\text{solvent}}$, additional parameters are introduced to scale the bulk-solvent model and align it with the experimental observables for accurate target function calculation. The scaling parameters in SFCalculator are initialized following the strategies outlined in [2], with enhancements achieved through gradient descent-based optimization.

The total model structure factor is defined as:

$$\mathbf{F}_c^{(s)} = k_{iso}^{(s)} \cdot \exp(-2\pi \mathbf{S}^T U_{aniso}^{(s)} \mathbf{S}) \cdot (\mathbf{F}_{\text{protein}} + k_{mask}^{(s)} \cdot \mathbf{F}_{\text{solvent}}) \tag{3}$$

where the $k_{iso}^{(s)}$, $U_{aniso}^{(s)}$ and $k_{mask}^{(s)}$ are scaling parameters in each resolution bin. The purpose of binning is to group data with common features, enabling each group to be characterized by a shared set of parameters. In this context, the primary parameter is the resolution d of reflections. SFCalculator employs a binning scheme that divides the resolution range uniformly on a logarithmic scale, $\ln(d)$ [65]. This approach ensures that higher-resolution bins contain more reflections than lower-resolution bins, while allowing for finer binning at low resolution without increasing the total number of bins. As highlighted in [2], the dependence of scale factors on resolution is approximately exponential. By using logarithmic binning, the variation of scale factors between bins is rendered more uniform, enhancing the algorithm's effectiveness.

The scaling parameters are ideally determined by minimizing the following least-square residues:

$$\mathrm{LS} = \sum \left(F_{\text{obs}} - |\mathbf{F}_c|\right)^2 , \tag{4}$$

To achieve fast and reliable convergence, we employ the root-finding and linear system-solving methods proposed in [2] to initialize the scaling parameters. Details of these algorithms are provided in the Appendix B.3. Following this initialization, we leverage the autodifferentiation back-end of our SFCalculator implementation to perform a few steps of gradient descent-based optimization, using either the ADAM optimizer[32] or L-BFGS[37]. This process minimizes the target function (Equation 4) to compute the final scaling parameters.

Molecular Replacement

The goal of molecular replacement is to determine the optimal rotation and translation of a molecule to best align with the experimental observables. Performing a full six-dimensional search, however, is computationally prohibitive. By decomposing the rotation and translation searches, the number of evaluations can be significantly reduced. Hierarchical grid search[68, 69] allows for additional efficiency gains.

Our molecular replacement method begins with a rapid packing score (see Appendix B.1) search performed on a coarse grid of center-of-mass (COM) positions within the asymmetric unit, using a random initial pose. This step identifies a candidate COM with the lowest clash score. The selected COM, in combination with the initial random rotation, is then used to construct an initial model and determine the scale factors. The identified COM is subsequently carried forward to the rotation search phase. It is important to note that the following hierarchical grid search remains indispensable despite the availability of a fully differentiable backend capable of gradient-based optimization. This necessity arises from the highly rugged landscape of the target function, which limits the radius of convergence for gradient descent. Therefore, identifying a suboptimal candidate through the hierarchical search is a critical prerequisite. The next phase involves a rotation search on a hierarchical SO(3) grid, leveraging Patterson functions to determine a suboptimal rotation matrix. This matrix is then utilized in a hierarchical translation search, which identifies a suboptimal translation vector. The pipeline culminates in a gradient descent optimization step, refining the pose to achieve the final solution.

Hierarchical rotation search

The decomposition of the rotation and translation search is achieved using Patterson functions [56]. The Patterson function is defined as the autocorrelation of the electron density, effectively representing a map of interatomic distance vectors:

$$P(\mathbf{u}) = \int_R \rho(\mathbf{r})\rho(\mathbf{r}+\mathbf{u})d\mathbf{r}$$

Leveraging the convolution theorem, the Patterson map can be expressed as the Fourier transform of the squared magnitudes of the structure factors, which are derived directly from experimental data:

$$P(u,v,w) = \frac{2}{V}\sum_{h=0}^{+\infty}\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} F_h^2 \cos 2\pi(hu+kv+lw) \tag{5}$$

Algorithm 2 Hierarchical Rotation Search

```
procedure OPTΦ(P_u^o, X)                ▷ Find the suboptimal rotation given a model
    N_round ← 5, N_candi ← 40
    Φ ← SO(3) × ℝ^2 grid at base resolution
    for iter = 1 ... N_round do
        for φ_i ∈ Φ do
            F_c(φ_i) ← SFCalculator(X, φ_i)
            S_rot(φ_i) ← PearsonR(P_u^o, P_u(F_c(φ_i)))
        end for
        Sort Φ by S_rot(φ_i) in descending order
        Φ_top ← {φ_1, φ_2, ..., φ_{N_candi}}      ▷ Select top N candidates based on S_rot(φ_i)
        φ* ← arg max(S_rot(φ_i))
        Φ_new ← {}
        for φ_i ∈ Φ_top do
            Φ_new ← Φ_new ∪ SUBDIVIDE(φ_i)
        end for
        Φ ← Φ_new
    end for
    return φ*
end procedure
```

As the autocorrelation of the electron density, the Patterson map inherently captures the interatomic features of the system. Specifically, by focusing on an appropriate range of Patterson vectors that primarily represent intramolecular pairwise distance features, the map becomes relatively insensitive to translational changes. Consequently, using the Patterson function as the target allows for the decomposition of the rotational and translational searches, significantly simplifying the molecular replacement process.

In practice, we use the following correlation function as the scoring function for hierarchical rotation search:

$$S_{\text{rot}} = \text{PearsonR}(\text{P}^{\text{o}}_{\mathbf{u}}, \text{P}^{\text{c}}_{\mathbf{u}}) \tag{6}$$

where $P^o_{\mathbf{u}}$ is the Patterson function calculated using experimental structure factor amplitudes [57], while $P^c_{\mathbf{u}}$ is the Patterson function calculated using the structure factor amplitudes of candidate models.

For the hierarchical grid search, we adopted the uniform multiresolution grids on $SO(3)$ as described in[68], utilizing the Healpix [21] grid for the sphere and the Hopf fibration to uniformly lift the grid to $SO(3)$. This approach is consistent with the method used in cryoDRGN[69]. The base grid on $SO(3)$ consists of 576 orientations, with a spacing of 30°. In each iteration, the 40 grids with the highest S_{rot}-scores are selected and further subdivided into higher-resolution neighboring grids, with each grid generating 8 neighbors. Typically, five rounds of refinement are performed, culminating in a final orientation resolution of 0.92 degrees. This hierarchical approach requires a total of 1,856 evaluations, significantly fewer than the brute-force search involving 36,864 rotations at a spacing of 7.5°, while achieving a much finer rotation spacing.

Hierarchical translation search

The concept of the hierarchical translation search mirrors that of the hierarchical rotation search, with the primary difference being that the subdivision of grids is performed in fractional coordinates using interpolation. At each level, each grid point generates $3^3 - 1 = 26$ neighboring grids in the next level. Additionally, since a suboptimal rotation matrix has already been determined from the preceding search, there is no need to rely on a Patterson function-based scoring method. Instead, we utilize the correlation of structure factor magnitudes, which reduces the computational cost by avoiding the calculation of the Patterson function. The correlation is quantified using the Pearson correlation coefficient between observed structure factor amplitudes, F_{obs}, and structure factor amplitudes calculated from the model, F_c:

$$S_{\mathrm{trans}} = \mathrm{PearsonR}\left(F_{\mathrm{obs}}\ , F_c\right)$$

The translation search can still be computationally expensive, even with a hierarchical strategy, due to the size of the unit cell. However, one advantageous property of crystallographic symmetry can significantly reduce the computational effort: the polar axis. The polar axis, present in specific space groups, is an axis along which translation operations do not affect the structure factor magnitudes. This means that any point along the polar axis can be arbitrarily chosen as the origin, eliminating the need to perform a search along that axis. SFCalculator implementation fully supports this functionality, allowing for more efficient translation searches.

Gradient-Descent Rigid Body Refinement

Once the suboptimal rotation ϕ_0 and translation v_0 have been determined, we assume that they lie within the radius of convergence for gradient descent. Gradient-based optimization is then performed using the following target function:

$$\phi', v' = \arg\min_{\phi', v'} \frac{\left(F_{\text{obs}} - F_{\text{c}}\left(X, \phi_0 + \phi', v_0 + v'\right)\right)^2}{\sigma^2_{F_{\text{obs}}}}$$

The optimal pose is subsequently defined as:

$$\phi^* = \phi_0 + \phi', \quad v^* = v_0 + v'$$

For this optimization, rotations are parameterized as quaternions to facilitate unconstrained optimization.

Crystallographic R factors

R factors are statistical metrics used in crystallography to assess the quality of a structural model, calculated by comparing the model-predicted structure factors to crystallographic data, with lower values indicating better agreement.

$$R = \frac{\sum_h |F_{\text{obs}}(h) - F_{\text{c}}(h)|}{\sum_h |F_{\text{obs}}(h)|}$$

Where h indicates triplets of so-called Miller indices that indicate the spatial frequencies of the corresponding Fourier components. Typically, two separate R factors are calculated during structure refinement: R_{work} is calculated over Miller indices included in refinement, while R_{free} is a cross-validation metric calculated using a reserved subset of experimental data not used during refinement. A smaller gap between R_{work} and R_{free} indicates less overfitting.

Training for physical prior embedded model

The physical prior is represented by the Boltzmann distribution, $p(x) = e^{-E(x)/RT}/Z$ at temperature T, with R the universal gas constant; Z is the partition function. To approximate the target distribution using our normalizing flow model, we utilize two training objectives:

$$\begin{aligned}
\mathcal{L}_{\text{forward}} &= \mathrm{KL}\left[\mu_X \| q_X\right] = -\mathbb{E}_{\mathbf{x}\sim\mu_X}\left[\log\mu_Z\left(T_{xz}(\mathbf{x},\theta)\right) + \log R_{xz}(\mathbf{x},\theta)\right] \\
&\qquad\qquad\qquad \approx -\mathbb{E}_{\mathbf{x}\sim\rho_X}\left[\log\mu_Z\left(T_{xz}(\mathbf{x};\theta)\right) + \log R_{xz}(\mathbf{x},\theta)\right] \\
\mathcal{L}_{\text{reverse}} &= \mathrm{KL}\left[\mu_Z \| q_Z\right] = -\mathbb{E}_{\mathbf{z}\sim\mu_Z}\left[\log\mu_X\left(T_{zx}(\mathbf{z},\theta)\right) + \log R_{zx}(\mathbf{z},\theta)\right] \\
&\qquad\qquad\qquad = \mathbb{E}_{\mathbf{z}\sim\mu_Z}\left[E_{mm}\left(T_{zx}(\mathbf{z},\theta)\right)/k_\beta T - \log R_{zx}(\mathbf{z},\theta)\right]
\end{aligned} \tag{7}$$

Here:

- $\mathcal{L}_{\text{forward}}$ corresponds to the negative log-likelihood of the training dataset, as in most generative models. Training samples are typically prepared by running molecular dynamics simulations for 100 ns using an implicit solvent model. μ_Z represents the prior density distribution in the latent space, which is Gaussian in this case; μ_X represents the ground truth distribution of protein conformations, and we approximate it with ρ_X, the observed density of MD samples.
- $\mathcal{L}_{\text{reverse}}$ is the energy-based term specific to cases where the (normalized) density function of the target distribution is known. It ensures that sampled conformations from the latent space exhibit physically reasonable energies. E_{mm} represents the molecular mechanics energy. It is important to note that most molecular mechanics

force fields can be easily wrapped as differentiable energy functions, since the (negative) gradient of the energy with respect to the coordinates corresponds directly to the force. In the work, we employed OpenMM[16].

The training process begins by optimizing $\mathcal{L}_{\text{forward}}$ alone to initialize the network. At this stage, generated samples resemble those in the molecular dynamics trajectories but often contain steric clashes or other nonphysical details, leading to high energies. Subsequently, the weight of $\mathcal{L}_{\text{reverse}}$ is gradually increased to regularize the latent space, incorporating physical energy constraints and ensuring sampled conformations align with the Boltzmann-distributed target.

Experimental data likelihood

The differentiability of SFCalculator enables a differentiable calculation of model structure factors $\mathbf{F}_c$, thus the likelihood target. We employ the following two objective likelihood functions:

1. Least Squares Loss:

$$L_{lse} = \sum \frac{\left(|E_C| - E_O\right)^2}{2\sigma_{E_O}^2} \tag{8}$$

2. Negative Log-Likelihood from Rice distribution:

$$L_{rice} = -\left[\sum_{h\in a} \log p_a\left(E_O; E_C\right) + \sum_{h\in c} \log p_c\left(E_O; E_C\right)\right] \tag{9}$$

where, according to [54], the likelihoods $p_a\left(E_O; E_C\right)$ for acentric reflections and $p_c\left(E_O; E_C\right)$ for centric reflections are defined as:

$$
\begin{aligned}
p_{\mathrm{a}}\left(E_{\mathrm{O}}; E_{\mathrm{C}}\right) = & \frac{2 E_{\mathrm{O}}}{1-\sigma_{\mathrm{A}}^2+2 \sigma_{E_{\mathrm{O}}}^2} \exp \left[-\frac{E_{\mathrm{O}}^2+\left(\sigma_{\mathrm{A}} E_{\mathrm{C}}\right)^2}{1-\sigma_{\mathrm{A}}^2+2 \sigma_{E_{\mathrm{O}}}^2}\right] \times I_0\left(\frac{2 \sigma_{\mathrm{A}} E_{\mathrm{O}} E_{\mathrm{C}}}{1-\sigma_{\mathrm{A}}^2+2 \sigma_{E_{\mathrm{O}}}^2}\right), \\
p_{\mathrm{c}}\left(E_{\mathrm{O}}; E_{\mathrm{C}}\right) = & \left[\frac{2}{\pi\left(1-\sigma_{\mathrm{A}}^2+\sigma_{E_{\mathrm{O}}}^2\right)}\right]^{1/2} \exp \left[-\frac{E_{\mathrm{O}}^2+\left(\sigma_{\mathrm{A}} E_{\mathrm{C}}\right)^2}{2\left(1-\sigma_{\mathrm{A}}^2+\sigma_{E_{\mathrm{O}}}^2\right)}\right] \\
& \times \cosh \left(\frac{\sigma_{\mathrm{A}} E_{\mathrm{O}} E_{\mathrm{C}}}{1-\sigma_{\mathrm{A}}^2+\sigma_{E_{\mathrm{O}}}^2}\right).
\end{aligned} \tag{10}
$$

Here, E_O, E_C, and σ_{E_O} represent the normalized F_o, F_c (such that $\langle E^2 \rangle = 1$), and σ_F, respectively. σ_A describes the correlation between model and data, as defined in [54, 63].

During the guided search approach, gradient descent is performed with respect to a linear combination of L_{lse} and L_{rice}. The weights of this combination are chosen empirically and depend on the specific system under consideration.

Training for posterior approximation

The training objective for the normalizing flow model approximating the posterior is to minimize the following reverse Kullback–Leibler divergence:

$$
\begin{aligned}
\mathcal{L}_{\text{posterior}} & = D_{KL}[q(x;\theta) \| p\left(x \mid F_o, \sigma_F\right)] \\
& = \underset{z \sim \mu_z(z)}{\mathbb{E}}[\log \mu_z(z) - \log R_{zx}(z,\theta)] - \underset{z \sim \mu_z(z)}{\mathbb{E}}[\log p\left(x \mid F_o, \sigma_F\right)] \\
& = -\underset{z \sim \mu_z(z)}{\mathbb{E}}[\log R_{zx}(z,\theta)] - \underset{z \sim \mu_z(z)}{\mathbb{E}}[\log p\left(F_o, \sigma_F \mid T_{zx}(z,\theta)\right) + \log p(T_{zx}(z,\theta)) - \log C]
\end{aligned} \tag{11}
$$

which is equivalent to maximizing the Evidence Lower Bound (ELBO) as defined in [55]. $\log C$ is the normalization constant.

By using the Boltzmann distribution as the physical prior, the energy term is incorporated, with the normalizing constant (partition function) absorbed into the constant term:

$$\log p\left(T_{zx}(z, \theta)\right) = -E_{mm}\left(T_{zx}(z, \theta)\right)/k_B T$$

Thus, the objective $\mathcal{L}_{\text{posterior}}$ is equivalent to $\mathcal{L}_{\text{reverse}}$ as defined in Equation 7, with the addition of the negative experimental log-likelihood term $-\log p\left(F_o, \sigma_F \mid T_{zx}(z, \theta)\right)$.

The training process begins by optimizing $\mathcal{L}_{\text{forward}}$ alone to initialize the network. Once the initial training is complete, the weight of $\mathcal{L}_{\text{posterior}}$ is gradually increased to regularize the latent space. This step incorporates both physical energy and experimental likelihood constraints, ensuring that the sampled conformations align with the posterior target.

In greater detail, during the training phase involving $\mathcal{L}_{\text{posterior}}$, the negative log-likelihood term initially uses the least squares term L_{lse} as defined in Equation 8. Over time, this is gradually transitioned to L_{rice}, as defined in Equation 9. This approach is adopted because, during the early stages of $\mathcal{L}_{\text{posterior}}$ training, the energy term is often very large due to the presence of non-physical details in the conformations. Under these conditions, the least squares term L_{lse} is more suitable for matching the scale of the other terms in the loss function. Transitioning to L_{rice} later in the process ensures a more accurate incorporation of the experimental likelihood once the system has been sufficiently regularized.

Code availability

SFCalculator has been implemented with PyTorch[52], JAX[8], and TensorFlow[1] backends to accommodate a broader range of users. The codebase is open-sourced and available at: https://github.com/Hekstra-Lab/SFcalculator.

Code related to molecular replacement, normalizing flow construction, training and refinement is maintained in a separate repository, `deeprefine`, which can be accessed at: https://github.com/minhuanli/deeprefine.

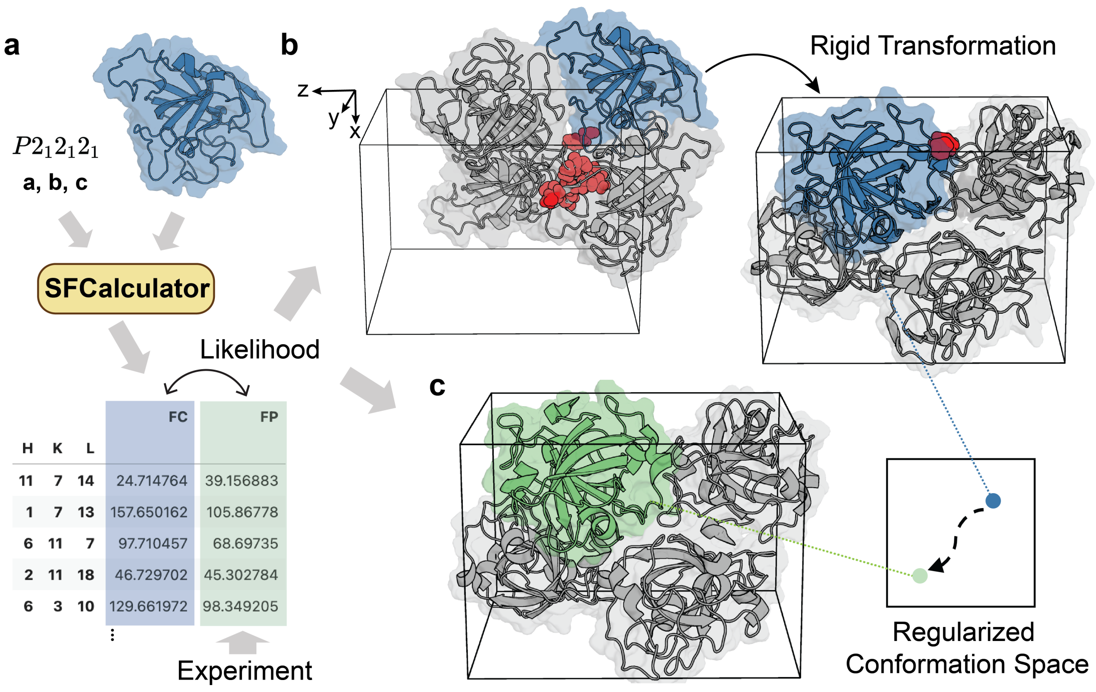

Fig. 1 SFCalculator as an interface. SFCalculator provides a differentiable forward function that calculates from molecular models, along with space group and unit cell information, expected values of experimental observables. This enables the computation of a differentiable likelihood or scoring function (**a**), facilitating downstream tasks such as molecular replacement (**b**), and structure refinement (**c**) within a unified framework. The red spheres in panel (**b**) represent atoms that clash with neighboring atoms.

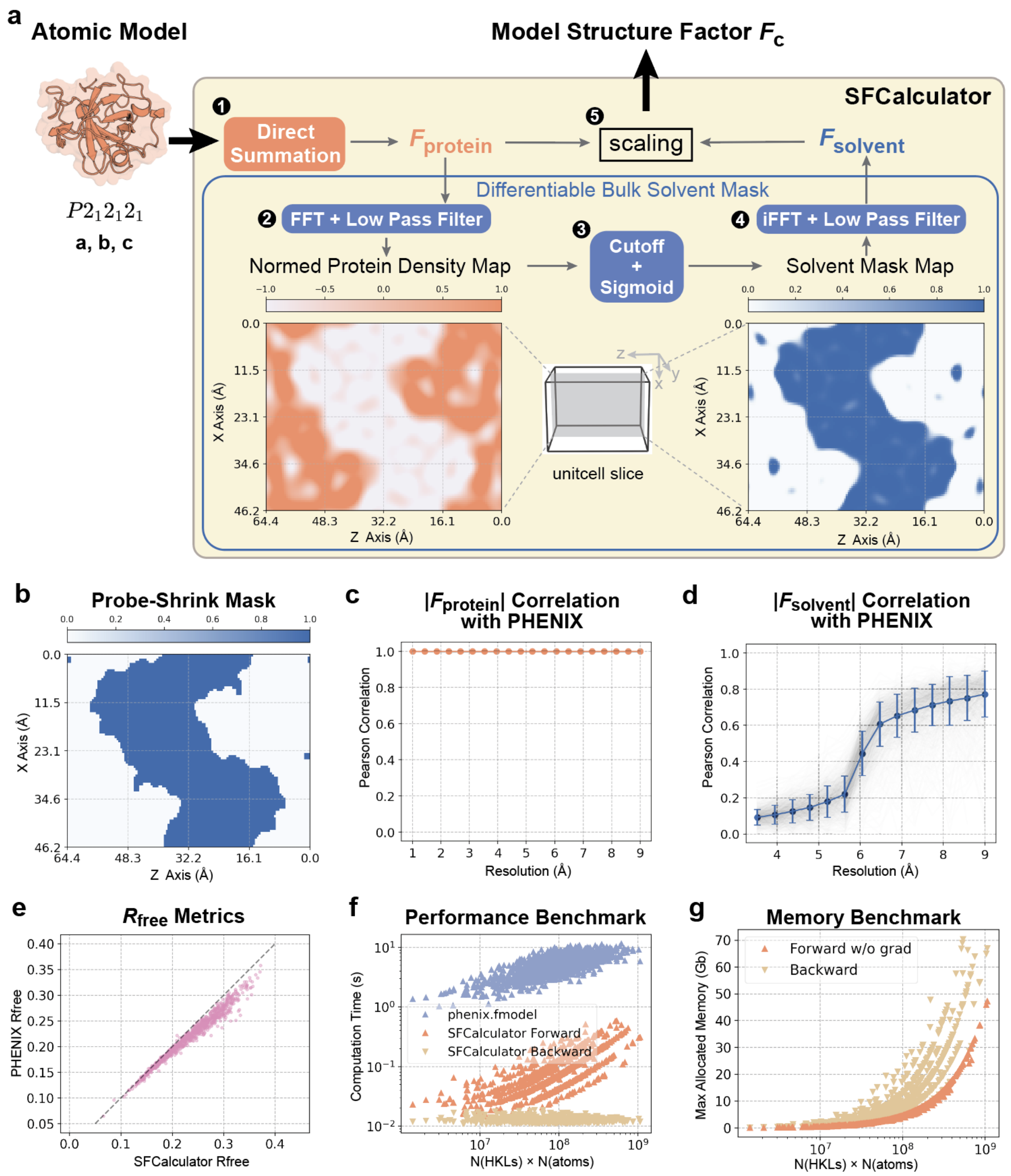

Fig. 2 Algorithms and benchmark of SFCalculator. (**a**) Algorithms of SFCalculator: The protein contribution is computed through direct summation, generating $\mathbf{F}_{\text{protein}}$. The differentiable bulk solvent mask algorithm is then applied to compute the solvent contribution, $\mathbf{F}_{\text{solvent}}$. These contributions are combined by determining parameterized scale factors to produce the final model structure factor F_c. (**b-g**) Accuracy and performance benchmarking of SFCalculator compared to the commonly used software PHENIX. (**b**) Slice view of the solvent mask generated using the Probe-Shrink method (default in PHENIX) compared to our differentiable solvent mask in (**a**). (**c**) Correlation statistics of protein structure factors between SFCalculator and PHENIX, showing perfect agreement. (**d**) Correlation statistics of solvent structure factors between SFCalculator and PHENIX. (**e**) Comparison of R_{free} calculations between SFCalculator and PHENIX. SFCalculator achieves high consistency overall, with discrepancies appearing for high-R_{free} datasets. (**f**) Performance benchmarking of SFCalculator's forward and backward computations compared to PHENIX's `phenix.fmodel`. SFCalculator is 50-200 times faster due to GPU utilization, while PHENIX runs solely on CPUs. Hardware details: PHENIX was tested on an Intel(R) Xeon(R) CPU @ 2.20 GHz , and SFCalculator was tested on an Nvidia A100 GPU. (**g**) Memory benchmarking of SFCalculator, demonstrating the ability to handle proteins with up to ∼1,000 residues at -2 Å resolution (PDB id: 4PKF) on a single Nvidia A100 80GB GPU.

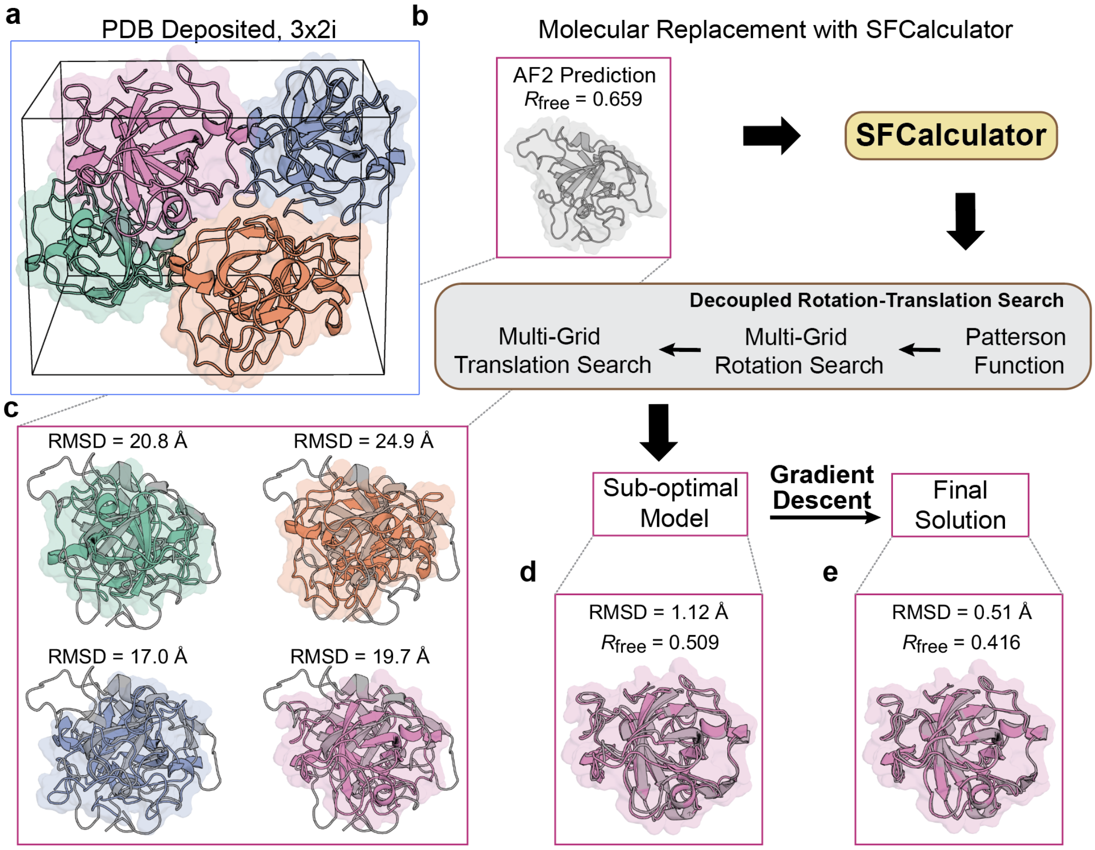

Fig. 3 Molecular Replacement using SFCalculator. (a) Visualization of the PcCel45A unit cell (PDB id: 3X2I), which contains four copies of the asymmetric unit, providing four ground-truth poses for the pose search. (b) Visualization of the AlphaFold-predicted structure for the enzyme system. The random initial pose assigned by the predictive model yields a poor R_{free} value (0.659), indicating a significant mismatch. (c) The initial pose exhibits a large R.M.S.D. (Root-Mean-Square Deviation) relative to each of the four ground-truth poses, with values ranging from 17.0 Å to 24.9 Å. (d) The hierarchical grid search identifies an approximate pose, closely resembling the pink ground-truth pose, with an improved R.M.S.D. of 1.12 Å and an R_{free} of 0.509. (**e**) Subsequent gradient descent refinement further improves the R.M.S.D. to 0.51 Å and reduces R_{free} to 0.416, underscoring the importance of the gradient descent step in achieving accurate pose refinement.

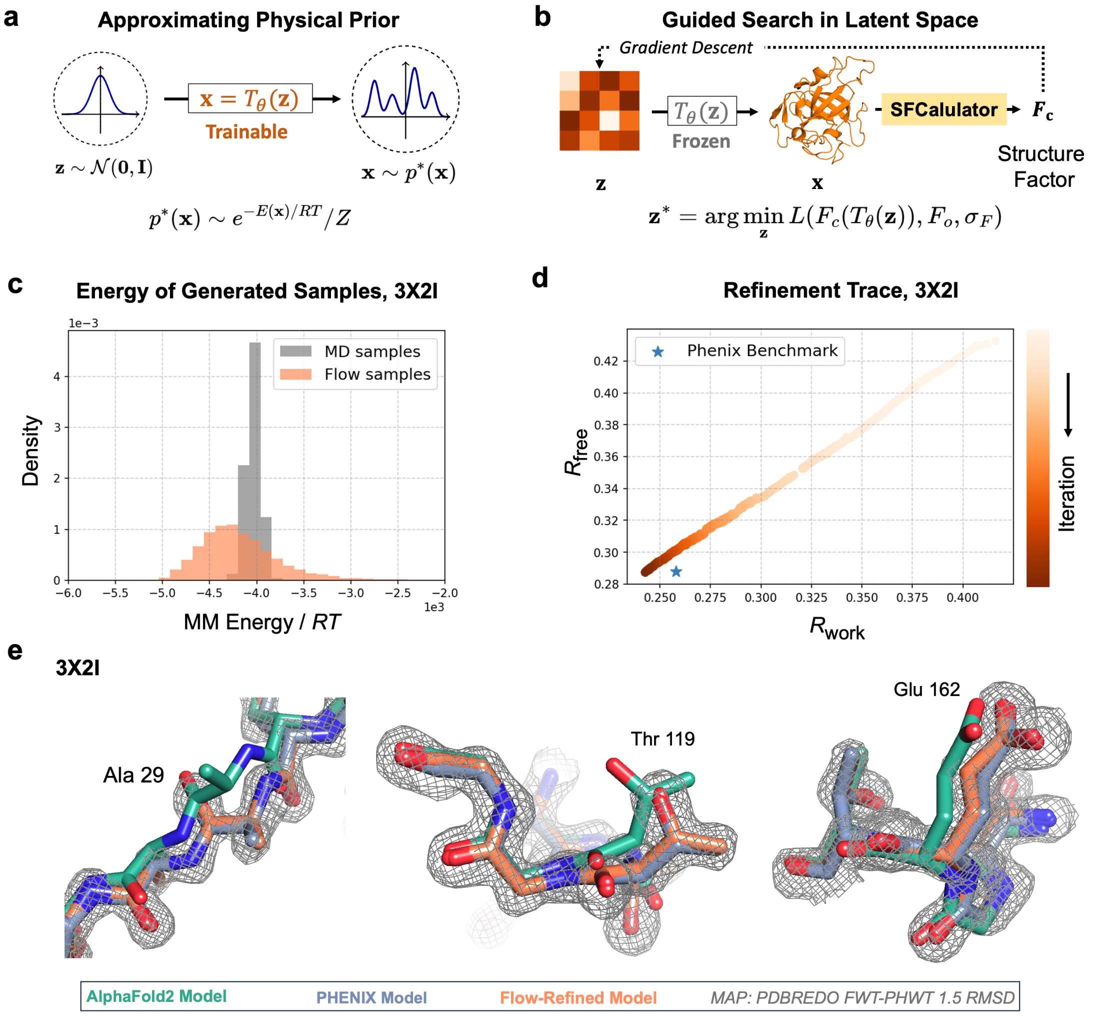

Fig. 4 Guided search in the latent space of a generative model encoding a physical prior. (**a**) The physical prior, represented by the Boltzmann distribution $\left(p^*(x) = e^{-E(x)/RT}/Z\right)$ at temperature T for conformations x with energy $E(x)$, is approximated using a normalizing flow model. This framework embeds the latent space z with the physical prior, where $z \sim \mathcal{N}(0, I)$, and the transformation $x = T_\theta(z)$ maps the latent space to the target space x to approximate the target distribution $p^*(x)$. (**b**) After training, the refinement process is performed through guided search in the latent space. Gradient descent is applied to minimize the data likelihood loss, $L\left(F_c\left(T_\theta(z)\right), F_o, \sigma_F\right)$, using SFCalculator to compute structure factors (F_c). (**c-e**) Refinement of the PcCel45A structure (PDB id: 3X2I) in a Latent Space with Embedded Physical Priors. (**c**) Energy distribution of samples generated by the trained normalizing flow, compared to samples from molecular dynamics (MD). The generated samples exhibit energies within a reasonable scale, consistent with physical expectations. (**d**) Refinement trace in the latent space. (**e**) Visualization of the improvements achieved through refinement in the latent space. The method effectively corrects both backbone and side chain conformations, showcasing structural improvements over the initial models.

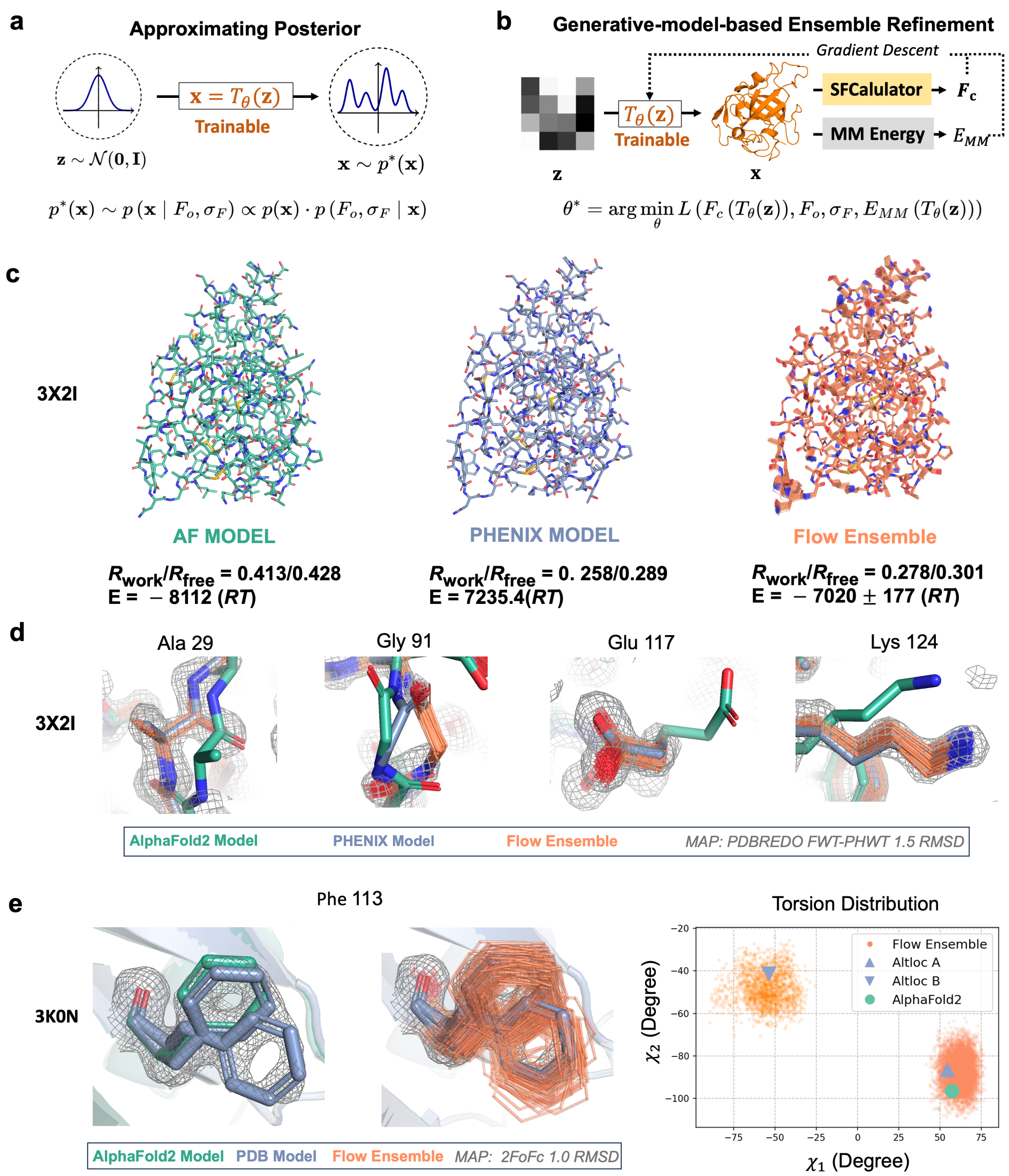

Fig. 5 Ensemble refinement using a generative model approximating the posterior distribution. (**a**) The target distribution of the normalizing flow is the posterior combining the physical prior $p(x)$ and the experimental likelihood. The transformation $x = T_\theta(z)$ is trained to approximate this posterior. (**b**) By including both the experimental likelihood and the molecular mechanics (MM) energy in the training objective, the parameters θ are optimized via gradient descent. This ensures that the samples from the normalizing flow align with the posterior distribution. SFCalculator computes structure factors (F_c), while the MM energy term (E_{MM}) incorporates physical constraints, refining the ensemble to balance data fit and physical plausibility. (**c**) Comparison of models for PcCel45A (PDB ID: 3X2I). The AlphaFold2 model after molecular replacement exhibits a high $R_{\text{work}}/R_{\text{free}}$ = (0.413 / 0.428) but a good molecular mechanics (MM) energy -8112 RT. The Phenix-refined model achieves better $R_{\text{work}}/R_{\text{free}}$ =(0.258 / 0.289) but has an unphysically high MM energy 7235.4 RT. The flow ensemble model strikes a balance with $R_{\text{work}}/R_{\text{free}}$ =(0.278 / 0.301) and a more reasonable MM energy E=-7020 ± 177 RT. (**d**) Visualization of key residues (ALA29, GLY91, GLU117, and LYS124) of PcCel45A. The AlphaFold2, Phenix, and flow ensemble models are overlaid with electron density maps, showcasing the structural improvements achieved by the flow ensemble refinement. Both backbone and side chain conformations are corrected, providing better agreement with the experimental data. (**e**) Refinement results for residue PHE113 in the isomerase Cyclophilin A (CypA, PDB ID: 3K0N). The $3k0n$ dataset, collected at room temperature, is known to exhibit alternative conformations for many residues, including PHE113. The flow ensemble successfully captured both conformational states. The torsion angle distribution plot for χ_1 and χ_2 highlights the broader conformational diversity achieved by the flow ensemble.

Acknowledgements

We thank Tom Terwilliger, Randy Read, Alisia Fadini, members of the Hekstra Lab and the Molecular Biophysics and Integrated Bioimaging at Lawrence Berkeley National Lab for fruitful discussions. This work was supported by a graduate fellowship of the Eric and Wendy Schmidt Center to M.H.L., a Career Award at the Scientific Interface from the Burroughs Wellcome Fund to K.M.D., and the NIH Director's New Innovator Award (DP2-GM141000) to D.R.H. Computations were run on the FASRC Cannon cluster supported by the FAS Division of Science Research Computing Group at Harvard University. We acknowledge the open source software projects used in this work, including reciprocalspaceship[22], GEMMI[67], and healpy[70].

Declarations

The authors declare no competing interests.

Author Contributions

M.H.L., K.M.D. and D.R.H. conceived the project. M.H.L. developed all code. M.H.L. and D.R.H. performed the analysis and drafted the manuscript. All authors edited the paper.

Appendix A Supplementary Figures

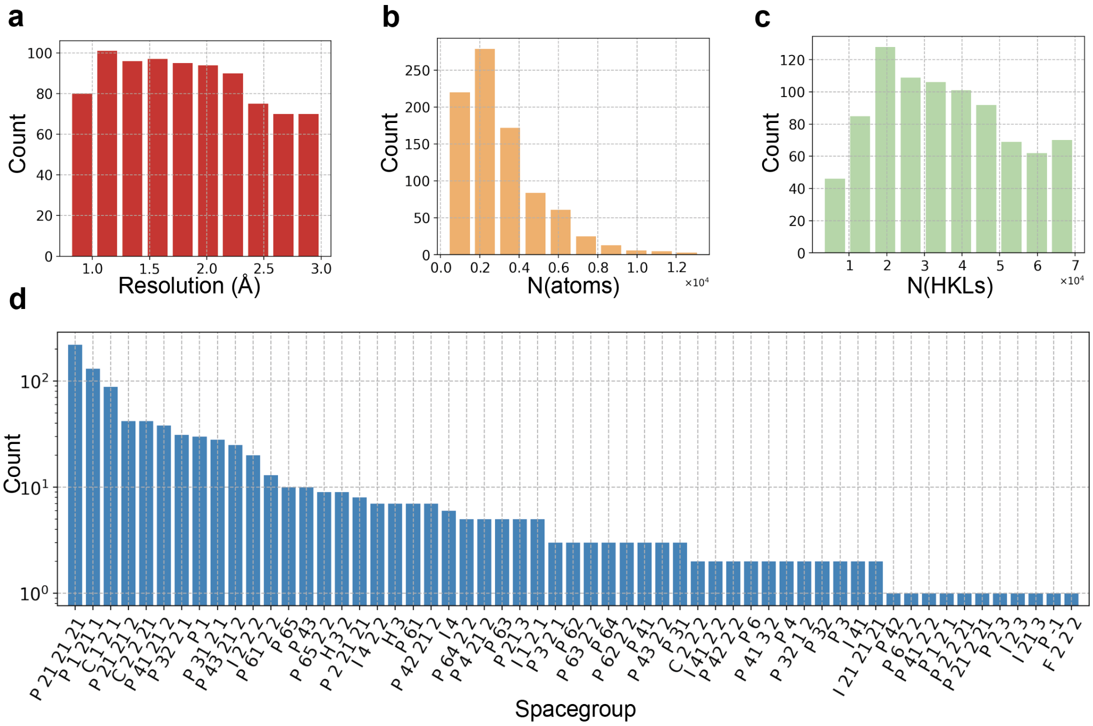

Fig. A1 Statistics of PDB entries used as benchmark datasets. Benchmarking was conducted across 868 PDB entries, encompassing data resolutions ranging from 0.8 Å to 3.0 Å (**a**), diverse molecular size (**b**), various number of reflections (**c**), and various space groups (**d**).

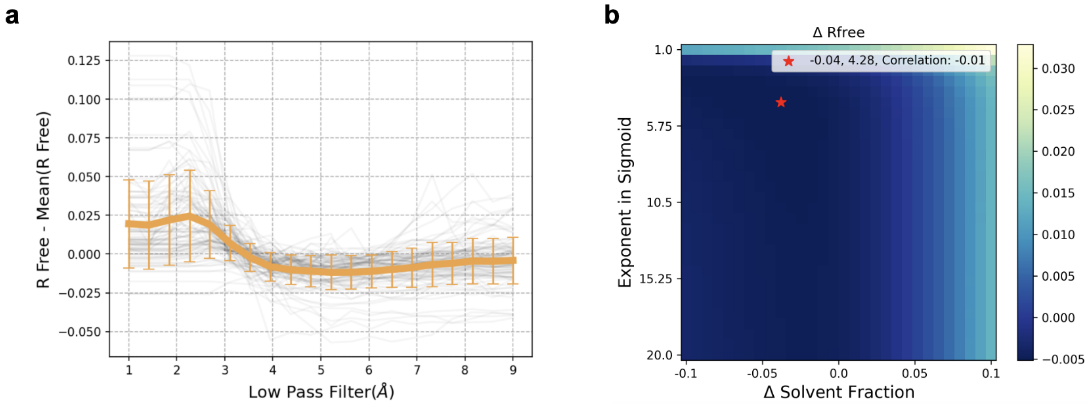

Fig. A2 Sweep of hyperparameters used in the differentiable solvent mask, with results based on statistics from 300 PDB entries. The final R_{free} value is used as the primary evaluation metric, with lower values indicating better performance. **a**. Effect of the low-pass filter cutoff. The optimal range is found to be 4-6 Å, with 5 Å selected as the default for its balance of accuracy and generality. **b**. The normalized voxel density distribution is passed through a sharp sigmoid function with inflection point dependent on the estimated solvent fraction (see Algorithm 1). Here we swept an offset, Δ between the sigmoid midpoint and estimated solvent fraction as well as the exponent of the sigmoid function. The results demonstrate that the solvent percentage algorithm is robust, with minimal sensitivity to changes as Δ within the range of -5% to $+2\%$. Similarly, the exponent of the sigmoid function is not important as long as it remains sufficiently large. A higher exponent ensures a more binary solvent mask. The default value of 10 was chosen for its consistency across tests.

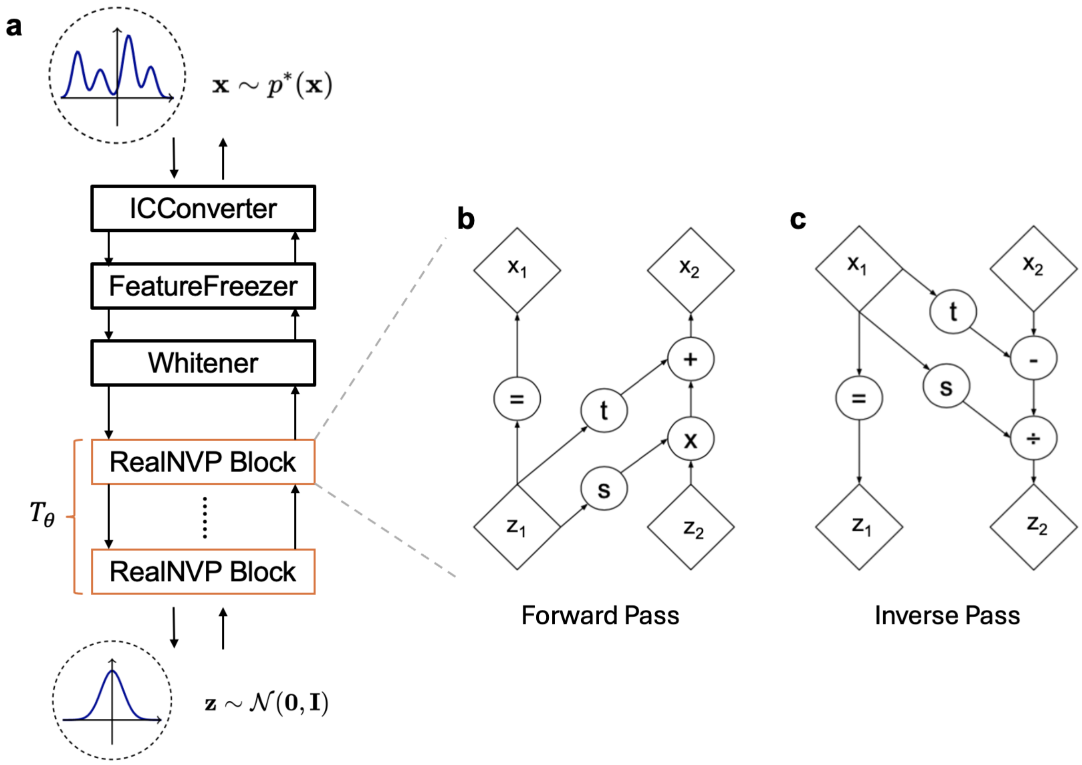

Fig. A3 Architecture of the normalizing flow model. **a**. Schematic representation of the model pipeline (see section B.4 for implementation details). Cartesian coordinates are first converted into internal coordinates using the ICConverter, followed by the FeatureFreezer, which locks features with low variability, such as bond lengths. Next, the Whitener normalizes the features and removes the six degrees of freedom associated with global rigid-body transformations. Finally, the data is processed through a stack of trainable RealNVP blocks. **b**. Each RealNVP block consists of two coupling layers that apply invertible transformation with channels swapped, combining scaling and the addition of a constant offset to one subset of the input vector (z_2), conditioned on the remaining subset (z_1). **c**. The inverse transformation of the coupling layer, illustrating the reversibility of the model. Panel **b** and **c** are adapted from reference[15].

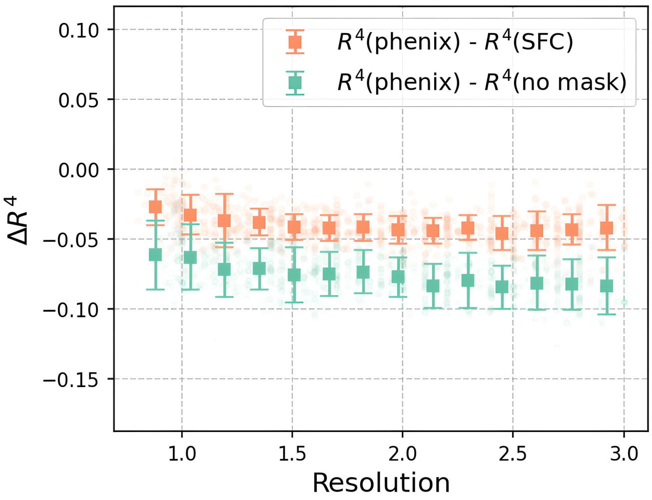

Fig. A4 Statistics of R^4 comparisons between SFCalculator, PHENIX, and the case with no solvent mask applied. R factors were calculated over reflections at resolutions ≥ 4 Å comparing observed structure factors against ones calculated from the 868 corresponding deposited models using SFCalculator or PHENIX with solvent mask, or using just the protein structure factors. Results were binned across datasets by their resolution, with error bars indicating standard deviations. The results clearly demonstrate that the differentiable solvent mask used in SFCalculator outperforms the absence of a mask across all ranges. However, due to its handling of the hydration layer, the Probe-Shrink method, the default in PHENIX, achieves better accuracy than SFCalculator.

Appendix B Detailed methodologies

B.1 Fast voxel rendering for solvent fraction calculations

The concept of fast solvent percentage calculation involves estimating the occupancy of protein molecules on a coarse unit cell grid, typically with a spacing of $z = 4.5$ Å. For this purpose, we adopt the occupancy function proposed in PyUUL[48]:

$$V_a^i(d, r_a) = \frac{1}{1 + e^{s(d - r_a)}}$$

Here, V_a^i represents the fraction of volume i occupied by an atom a with radius r_a at a distance d from its center. The parameter s controls the steepness of the decay. The total volume occupancy for each grid point is calculated as:

$$V^i = \sum_a V_a^i$$

If $V^i > c$, where c is the cutoff fraction, the volume is classified as "occupied." The solvent percentage is then computed as:

$$\text{Solvent Percentage} = 1 - \frac{\text{Number of occupied volumes}}{\text{Total number of unit cell volumes}}$$

For the steepness parameter s, we adopt an empirical formula:

$$s = \ln\left(\frac{1}{c} - 1\right) / \left(\frac{z}{2} - r\right)$$

This ensures that any atom center within $z/2$ of the volume center will occupy the volume. By default, we set $c = 10^{-3}$, resulting in a large s value to ensure rapid density decay. This ensures that a volume is considered occupied only when it is significantly filled by an atom, rather than being the result of a cumulative effect from multiple atoms contributing low-density values.

The same algorithm is employed for calculating packing scores and clash scores in the molecular replacement pipeline. The packing score is defined as the fraction of occupied volumes, while the clash score is defined as the fraction of volumes occupied by two or more atoms.

B.2 Hyperparameters in solvent mask calculation

To evaluate the impact of hyperparameters in the solvent model, we conducted a systematic sweep using 300 PDB entries, with the final R_{free} value serving as the primary evaluation metric (lower values indicate better performance), as shown in Fig. A2. The analysis of the low-pass filter cutoff revealed that the optimal range is between 4 and 6 Å, with 5 Å selected as the default for its balance between accuracy and generality. Additionally, we examined the effects of (1) applying an offset to the solvent fraction estimate calculated above for use in the solvent mask calculation (see Algorithm 1) and (2) varying the exponent in the sigmoid function. The solvent fraction algorithm proved robust, exhibiting minimal sensitivity to deviations in the solvent percentage offset within the range of -5% to $+2\%$. Similarly, the exponent in the sigmoid function was found to be effective as long as it remained sufficiently large to ensure a more binary solvent mask. A default exponent value of 10 was chosen based on its consistency across the tests.

B.3 Robust initialization of scaling parameters

We initialize the k_{iso}, k_{mask} and U_{aniso} in a fast and robust way following the reference[2], with necessary modifications.

First, we initialize k_{mask} and k_{iso} by solving the following overdetermined minimization problem for each resolution bin s:

$$\mathrm{LS}_s\left(K, k_{\text{mask}}\right)=\sum_h\left(\left|\mathbf{F}_{\text{protein}}+k_{\text{mask}}\, \mathbf{F}_{\text{solvent}}\right|^2-K F_{obs}^2\right)^2$$

where $K = k_{iso}^{-2}$, and in each resolution bin k_{iso} and k_{mask} are fixed for each reflection h. Minimizing the above equation lead to initializing k_{mask} by root finding of the following equation:

$$k_{\text{mask}}^3+a k_{\text{mask}}^2+b k_{\text{mask}}+c=0$$

where:

$$\begin{aligned}
a &= \left(C_3 Y_2-C_2 B_2-C_2 Y_3\right) / \left(D_3 Y_2-C_2^2\right) \\
b &= \left(B_3 Y_2-C_2 A_2-Y_3 B_2\right) / \left(D_3 Y_2-C_2^2\right) \\
c &= \left(A_3 Y_2-Y_3 A_2\right) / \left(D_3 Y_2-C_2^2\right)
\end{aligned}$$

and:

$$\begin{aligned}
A_2 &= \sum_s u_s I_s \\
B_2 &= 2 \sum_s v_s I_s \\
C_2 &= \sum_s w_s I_s \\
Y_2 &= \sum_s I_s^2
\end{aligned}$$

$$A_3 = \sum_s u_s v_s$$

$$B_3 = \sum_s (2v_s^2 + u_s w_s)$$

$$C_3 = 3\sum_s w_s v_s$$

$$D_3 = \sum_s w_s^2$$

$$Y_3 = \sum_s I_s v_s$$

where $w = |\mathbf{F}_{\text{solvent}}|^2$, $v = (\mathbf{F}_{\text{protein}}, \mathbf{F}_{\text{solvent}})$, $u = |\mathbf{F}_{\text{protein}}|^2$, and $I = F_{obs}^2$

If no positive root exists, k_{mask} is assigned a zero value, which implies the absence of a bulk-solvent contribution. If several roots with $k_{\text{mask}} \geq 0$ exist then the one that gives the smallest value of $\text{LS}_s(K, k_{\text{mask}})$ is selected. Once k_{mask} is determined, k_{iso} is calculated with:

$$K = \left(k_{\text{mask}}^2 C_2 + k_{\text{mask}} B_2 + A_2\right)/Y_2, \quad k_{iso} = 1/\sqrt{K}$$

Once we have initialized k_{mask} and k_{iso}, the remaining U_{aniso} is determined by solving a linear system:

$$\mathbf{U}_{aniso} = \mathbf{M}^{-1}\mathbf{b}$$

where $\mathbf{M} = \sum_{\mathbf{s}} \mathbf{V} \otimes \mathbf{V}, \mathbf{V} = \left(h^2, k^2, l^2, 2hk, 2hl, 2kl\right)^t$, $\mathbf{b} = -\sum_{\mathrm{s}} Z\mathbf{V}$, and $Z = \left[1/\left(2\pi^2\right)\right] \ln\left[F_{\text{obs}}\left(k_{iso}|\mathbf{F}_{\text{protein}} + k_{\text{solvent}}\mathbf{F}_{\text{mask}}|\right)^{-1}\right]$

B.4 Architecture of the Normalizing Flow model

The schematic architecture, illustrated in Fig. A3a, is composed of several key blocks, which we describe in detail below:

ICConverter

We adopt the mixed coordinate transformation layer described in [47]. In this approach, the Cartesian coordinates of macromolecules are partitioned into backbone atoms (N , C_α, C for each residue) and auxiliary atoms. The backbone atoms are retained as Cartesian coordinates (x_C), while the auxiliary atoms are converted into internal coordinates (x_I) as follows: for each auxiliary atom i, three "parent" particles j, k, and l are defined. The Cartesian coordinates of particles i, j, k, and l are then transformed into distance, angle, and dihedral coordinates ($d_{ij}, \alpha_{ijk}, \phi_{ijkl}$). Consequently, the Cartesian coordinates x are transformed into a combination of Cartesian and internal coordinates, expressed as $x \to [x_C, x_I]$.

The inverse transformation is straightforward. Auxiliary atoms are positioned sequentially: first, those whose parent particles are entirely within the Cartesian set, followed by particles whose parents have just been placed, and so on. The conversion from internal coordinates to Cartesian coordinates is performed using the NeRF algorithm [51].

FeatureFreezer

The FeatureFreezer module is newly introduced to lock features with low variability, thereby reducing the dimensionality of the problem. This is achieved through the following steps:

1. Identify the features to be frozen based on statistical analysis of the training set.
2. Exclude these features during the forward pass.
3. Reintroduce them in the inverse pass using their statistical mean values.

The selection of frozen features is governed by the following rules:

1. Cartesian signals are never frozen.
2. Distance signals (covalent bonds' lengths) are frozen if the ratio of standard deviation to the mean (std/mean) is less than 0.05.
3. Angle signals, represented as $\mathbf{v}_i = [\sin(\theta), \cos(\theta)]$, are frozen if the circular concentration coefficient $c = |\sum \mathbf{v}_i| / \sum |\mathbf{v}_i|$ exceeds 0.996 , corresponding to a standard deviation of approximately 5 degrees.

Whitener

The Whitener layer employs Principal Component Analysis (PCA) to normalize feature scales and remove their means. Two separate PCA operations are applied:

1. For Cartesian signals (x_C), PCA performs whitening and removes the six degrees of freedom associated with global rigid-body transformations.

2. For internal coordinates, PCA is applied independently to normalize these features.

The whitening and inverse whitening matrices are stored, enabling efficient application during both the forward and inverse passes.

RealNVP block

The RealNVP block incorporates trainable parameters to define and shape the transformation. Each RealNVP block consists of two stacked coupling layers, with channels swapped between the layers to ensure that both channels are transformed [15, 47]. A schematic representation of a single coupling layer is shown in Fig. A3b and c. The forward transformation is defined as:

$$f_{xx}(\mathbf{x}_1, \mathbf{x}_2) : \begin{cases} \mathbf{z}_1 = \mathbf{x}_1 \\ \mathbf{z}_2 = \mathbf{x}_2 \odot \exp(S(\mathbf{x}_1, \theta)) + T(\mathbf{x}_1, \theta) \end{cases}$$

where $\odot$ is element-wise multiplication, and the log-determinant of the Jacobian given by:

$$\log R_{xz} = \sum S_i\left(\mathbf{x}_1, \theta\right)$$

The inverse transformation operates as follows:

$$f_{x=}\left(\mathbf{z}_1, \mathbf{z}_2\right) : \begin{cases} \mathbf{x}_1 = \mathbf{z}_1 \\ \mathbf{x}_2 = \left(\mathbf{z}_2 - T\left(\mathbf{x}_1, \theta\right)\right) \odot \exp\left(-S\left(\mathbf{z}_1, \theta\right)\right) \end{cases}$$

with the corresponding log-determinant:

$$\log R_{xx} = -\sum_i S_i\left(\mathbf{z}_1, \theta\right)$$

This setup ensures that the transformation remains invertible while allowing fast computation of the log-determinant for both forward and inverse passes.

References

[1] Abadi M, Barham P, Chen J, et al (2016) {TensorFlow}: a system for {Large-Scale} machine learning. In: 12th USENIX symposium on operating systems design and implementation (OSDI 16), pp 265–283

[2] Afonine P, Grosse-Kunstleve R, Adams P, et al (2013) Bulk-solvent and overall scaling revisited: faster calculations, improved results. Acta Crystallographica Section D: Biological Crystallography 69(4):625–634

[3] Afonine PV, Grosse-Kunstleve RW, Echols N, et al (2012) Towards automated crystallographic structure refinement with phenix. refine. Acta Crystallographica Section D: Biological Crystallography 68(4):352–367

[4] Baek M, DiMaio F, Anishchenko I, et al (2021) Accurate prediction of protein structures and interactions using a three-track neural network. Science 373(6557):871–876

[5] Barbarin-Bocahu I, Graille M (2022) The x-ray crystallography phase problem solved thanks to alphafold and rosettafold models: a case-study report. Acta Crystallographica Section D: Structural Biology 78(4):517–531

[6] Bhowmick A, Hussein R, Bogacz I, et al (2023) Structural evidence for intermediates during o2 formation in photosystem ii. Nature 617(7961):629–636

[7] Bond-Taylor S, Leach A, Long Y, et al (2021) Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. IEEE transactions on pattern analysis and machine intelligence 44(11):7327–7347

[8] Bradbury J, Frostig R, Hawkins P, et al (2018) JAX: composable transformations of Python+NumPy programs. URL http://github.com/jax-ml/jax

[9] Bruenger AT, Karplus M (1991) Molecular dynamics simulations with experimental restraints. Accounts of chemical research 24(2):54–61

[10] Burnley BT, Afonine PV, Adams PD, et al (2012) Modelling dynamics in protein crystal structures by ensemble refinement. elife 1:e00311

[11] Croll TI (2018) Isolde: a physically realistic environment for model building into low-resolution electron-density maps. Acta Crystallographica Section D: Structural Biology 74(6):519–530

[12] Cromer DT, Mann JB (1968) X-ray scattering factors computed from numerical hartree–fock wave functions. Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography 24(2):321–324

[13] Dasgupta M, Budday D, De Oliveira SH, et al (2019) Mix-and-inject xfel crystallography reveals gated conformational dynamics during enzyme catalysis. Proceedings of the National Academy of Sciences 116(51):25634–25640

[14] Dauparas J, Anishchenko I, Bennett N, et al (2022) Robust deep learning–based protein sequence design using proteinmpnn. Science 378(6615):49–56

[15] Dinh L, Sohl-Dickstein J, Bengio S (2016) Density estimation using real nvp. arXiv preprint arXiv:160508803

[16] Eastman P, Galvelis R, Peláez RP, et al (2023) Openmm 8: molecular dynamics simulation with machine learning potentials. The Journal of Physical Chemistry B 128(1):109–116

[17] Emsley P, Cowtan K (2004) Coot: model-building tools for molecular graphics. Acta crystallographica section D: biological crystallography 60(12):2126–2132

[18] Fenn T, Schnieders M, Brunger A (2010) A smooth and differentiable bulk-solvent model for macromolecular diffraction. Acta Crystallographica Section D: Biological Crystallography 66(9):1024–1031

[19] Frank C, Khoshouei A, Fuβ L, et al (2024) Scalable protein design using optimization in a relaxed sequence space. Science 386(6720):439–445

[20] Fraser JS, Clarkson MW, Degnan SC, et al (2009) Hidden alternative structures of proline isomerase essential for catalysis. Nature 462(7273):669–673

[21] Gorski KM, Hivon E, Banday AJ, et al (2005) Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. The Astrophysical Journal 622(2):759

[22] Greisman JB, Dalton KM, Hekstra DR (2021) reciprocalspaceship: a python library for crystallographic data analysis. Journal of Applied Crystallography 54(5):1521–1529

[23] Grosse-Kunstleve RW, Adams PD (2001) Patterson correlation methods: a review of molecular replacement with cns. Acta Crystallographica Section D: Biological Crystallography 57(10):1390–1396

[24] Gruhl T, Weinert T, Rodrigues MJ, et al (2023) Ultrafast structural changes direct the first molecular events of vision. Nature 615(7954):939–944

[25] Headd JJ, Echols N, Afonine PV, et al (2012) Use of knowledge-based restraints in phenix. refine to improve macromolecular refinement at low resolution. Acta Crystallographica Section D: Biological Crystallography 68(4):381–390

[26] Henning RW, Kosheleva I, Šrajer V, et al (2024) Biocars: Synchrotron facility for probing structural dynamics of biological macromolecules. Structural Dynamics 11(1)

[27] Jack A, Levitt M (1978) Refinement of large structures by simultaneous minimization of energy and r factor. Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography 34(6):931–935

[28] Jiang JS, Brünger AT (1994) Protein hydration observed by x-ray diffraction: solvation properties of penicillopepsin and neuraminidase crystal structures. Journal of molecular biology 243(1):100–115

[29] Jing B, Berger B, Jaakkola T (2024) Alphafold meets flow matching for generating protein ensembles. arXiv preprint arXiv:240204845

[30] Jumper J, Evans R, Pritzel A, et al (2021) Highly accurate protein structure prediction with alphafold. nature 596(7873):583–589

[31] Keegan RM, Simpkin AJ, Rigden DJ (2024) The success rate of processed predicted models in molecular replacement: implications for experimental phasing in the alphafold era. Biological Crystallography 80(11)

[32] Kingma DP (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980

[33] Klukowski P, Riek R, Güntert P (2022) Rapid protein assignments and structures from raw nmr spectra with the deep learning technique artina. Nature

Communications 13(1):6151

[34] Lee B, White KI, Socolich M, et al (2025) Direct visualization of electric-field-stimulated ion conduction in a potassium channel. Cell 188(1):77–88

[35] Lewis S, Hempel T, Jiménez Luna J, et al (2024) Scalable emulation of protein equilibrium ensembles with generative deep learning. bioRxiv pp 2024–12

[36] Lin Z, Akin H, Rao R, et al (2023) Evolutionary-scale prediction of atomic-level protein structure with a language model. Science 379(6637):1123–1130

[37] Liu DC, Nocedal J (1989) On the limited memory bfgs method for large scale optimization. Mathematical programming 45(1):503–528

[38] Lundgren KJ, Caldararu O, Oksanen E, et al (2024) Quantum refinement in real and reciprocal space using the phenix and orca software. IUCrJ 11(6)

[39] McCoy AJ, Grosse-Kunstleve RW, Adams PD, et al (2007) Phaser crystallographic software. Journal of applied crystallography 40(4):658–674

[40] McCoy AJ, Sammito MD, Read RJ (2022) Implications of alphafold2 for crystallographic phasing by molecular replacement. Acta Crystallographica Section D: Structural Biology 78(1):1–13

[41] Millán C, Jiménez E, Schuster A, et al (2020) Alixe: a phase-combination tool for fragment-based molecular replacement. Acta Crystallographica Section D: Structural Biology 76(3):209–220

[42] Millán C, Keegan RM, Pereira J, et al (2021) Assessing the utility of casp14 models for molecular replacement. Proteins: Structure, Function, and Bioinformatics 89(12):1752–1769

[43] Moews P, Kretsinger R (1975) Refinement of the structure of carp muscle calcium-binding parvalbumin by model building and difference fourier analysis. Journal of molecular biology 91(2):201–225

[44] Moriarty NW, Janowski PA, Swails JM, et al (2020) Improved chemistry restraints for crystallographic refinement by integrating the amber force field into phenix. Acta Crystallographica Section D: Structural Biology 76(1):51–62

[45] Murshudov GN, Skubák P, Lebedev AA, et al (2011) Refmac5 for the refinement of macromolecular crystal structures. Acta Crystallographica Section D: Biological Crystallography 67(4):355–367

[46] Nakamura A, Ishida T, Kusaka K, et al (2015) "newton's cradle" proton relay with amide–imidic acid tautomerization in inverting cellulase visualized by neutron crystallography. Science advances 1(7):e1500263

[47] Noé F, Olsson S, Köhler J, et al (2019) Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. Science 365(6457):eaaw1147

[48] Orlando G, Raimondi D, Duran-Romaña R, et al (2022) Pyuul provides an interface between biological structures and deep learning algorithms. Nature communications 13(1):961

[49] Pandey S, Bean R, Sato T, et al (2020) Time-resolved serial femtosecond crystallography at the european xfel. Nature methods 17(1):73–78

[50] Papamakarios G, Nalisnick E, Rezende DJ, et al (2021) Normalizing flows for probabilistic modeling and inference. Journal of Machine Learning Research 22(57):1–64

[51] Parsons J, Holmes JB, Rojas JM, et al (2005) Practical conversion from torsion space to cartesian space for in silico protein synthesis. Journal of computational chemistry 26(10):1063–1068

[52] Paszke A, Gross S, Massa F, et al (2019) Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems 32

[53] Prince E (2004) International Tables for Crystallography, Volume C: Mathematical, physical and chemical tables. Springer Science & Business Media

[54] Read RJ, McCoy AJ (2016) A log-likelihood-gain intensity target for crystallographic phasing that accounts for experimental error. Acta Crystallographica Section D: Structural Biology 72(3):375–387

[55] Rezende D, Mohamed S (2015) Variational inference with normalizing flows. In: International conference on machine learning, PMLR, pp 1530–1538

[56] Rossmann M, Blow DM (1962) The detection of sub-units within the crystallographic asymmetric unit. Acta crystallographica 15(1):24–31

[57] Rupp B (2009) Biomolecular Crystallography, 0th edn. Garland Science, https://doi.org/10.1201/9780429258756, URL https://www.taylorfrancis.com/books/9781134064199

[58] Sanchez-Weatherby J, Sandy J, Mikolajek H, et al (2019) Vmxi: a fully automated, fully remote, high-flux in situ macromolecular crystallography beamline. Journal of synchrotron radiation 26(1):291–301

[59] Schevitz RW, Podjarny AD, Zwick M, et al (1981) Improving and extending the phases of medium-and low-resolution macromolecular structure factors by density

modification. Foundations of Crystallography 37(5):669–677

[60] Schmidt M (2013) Mix and inject: reaction initiation by diffusion for time-resolved macromolecular crystallography. Advances in Condensed Matter Physics 2013(1):167276

[61] Schmidt M (2019) Time-resolved macromolecular crystallography at pulsed x-ray sources. International journal of molecular sciences 20(6):1401

[62] Schuller M, Correy GJ, Gahbauer S, et al (2021) Fragment binding to the nsp3 macrodomain of sars-cov-2 identified through crystallographic screening and computational docking. Science advances 7(16):eabf8711

[63] Srinivasan R, Ramachandran G (1965) Probability distribution connected with structure amplitudes of two related crystals. v. the effect of errors in the atomic coordinates on the distribution of observed and calculated structure factors. Acta Crystallographica 19(6):1008–1014

[64] Terwilliger TC, Poon BK, Afonine PV, et al (2022) Improved alphafold modeling with implicit experimental information. Nature methods 19(11):1376–1382

[65] Urzhumtsev A, Afonine PV, Adams PD (2009) On the use of logarithmic scales for analysis of diffraction data. Acta Crystallographica Section D: Biological Crystallography 65(12):1283–1291

[66] Watson JL, Juergens D, Bennett NR, et al (2023) De novo design of protein structure and function with rfdiffusion. Nature 620(7976):1089–1100

[67] Wojdyr M (2022) Gemmi: A library for structural biology. Journal of Open Source Software 7(73):4200

[68] Yershova A, LaValle SM, Mitchell JC (2010) Generating uniform incremental grids on so (3) using the hopf fibration. In: Algorithmic Foundation of Robotics VIII: Selected Contributions of the Eight International Workshop on the Algorithmic Foundations of Robotics, Springer, pp 385–399

[69] Zhong ED, Bepler T, Berger B, et al (2021) Cryodrgn: reconstruction of heterogeneous cryo-em structures using neural networks. Nature methods 18(2):176–185

[70] Zonca A, Singer L, Lenz D, et al (2019) healpy: equal area pixelization and spherical harmonics transforms for data on the sphere in python. Journal of Open Source Software 4(35):1298