Diffusion Models II

SDE formulation and AlphaFold 3

October 8th, 2024

Recap

- Denoising Diffusion Probabilistic Models (DDPMs) are a popular formulation of diffusion.
  - Forward process: $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$
  - Backward process: $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$
  - Reformulation resulted in new loss function with improved sample quality

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2$$

Generalizing the design of diffusion models

- Other frameworks for diffusion models exist (e.g. Noise Conditioned Score Networks) where the data distribution is perturbed with Gaussian noise
- How can we unite these frameworks? How can we easily change the design of diffusion models?
- Idea: previous diffusion models added noise in discrete steps, but we can imagine this as a discretization of a continuous process.
- “Score-based Generative Modeling through Stochastic Differential Equations” gives a continuous formulation of diffusion models [1].

What is a Stochastic Differential Equation?

- Differential Equation

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t)$$

- Stochastic Differential Equation (SDE)

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + g(t)\frac{\mathrm{d}\mathbf{W}}{\mathrm{d}t}$$

White Noise = “Derivative of Gaussian Random variable”

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{W}$$

Forward Process

- If $\mathbf{x}(0) \sim p_0$ is the data distribution, then the SDE $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{W}$ will perturb this distribution with white noise.

- Typically, for a long period of time T the distribution of $\mathbf{x}(T) \sim p_T$ will have almost no information about the initial distribution (most often a normal distribution).

- The SDE is thus describing the *forward process* of the diffusion model. It is describing a continuous way of adding noise.

- The backward process will also be given by an SDE with initial condition given by $\mathbf{x}(T) \sim p_T$.

Backward Process

- To sample, we solve a reverse-time SDE

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\overline{\mathbf{W}}$$

which is guaranteed to have the same distributions as the forward SDE.

- Want to learn the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ for $0 \leq t \leq T$. For fixed t the objective becomes

$$\mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(t)|\mathbf{x}(0)} \| \mathbf{s}_\theta(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0)) \|^2$$

where $p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0))$ is the density function of the solution given the initial value is fixed at $\mathbf{x}(0)$.

- Most models choose the SDE so that p_{0t} has an exact formula and can be sampled.

Benefits of SDE formulation

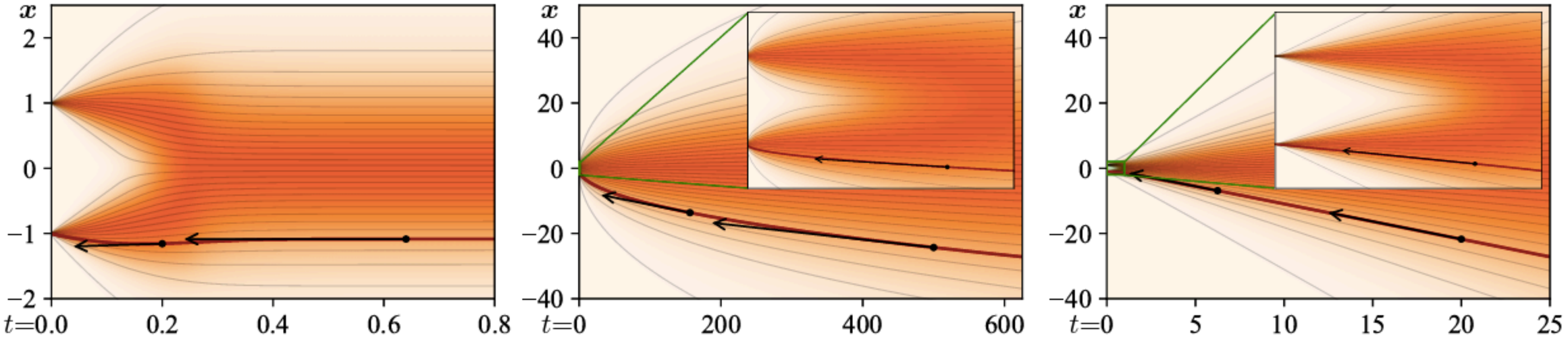
- SDE formulation generalizes previous models
  - Taking increasingly smaller steps in the DDPM and treating β_i as discretization of continuous function $\beta(t)$ gives a forward process that converges to
  - Variance Preserving SDE: $\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{W}$
  - Simplified loss becomes slightly rescaled version of denoising score matching objective
- Deterministic sampling method:

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t$$

Improving sampling

- Problem: it usually takes 100s-1000s of model evaluations to solve reverse SDE/ODE
- SDE formulation allows a flexible framework, so we can adjust parameters to improve sampling.
- “Elucidating the Design Space of Diffusion-Based Generative Models” discusses choices to improve sampling time [2].
  - Variance exploding SDE: $\mathrm{dx} = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}\mathbf{W}$
  - Choosing $\sigma(t) = t$ and parameterizing $\nabla p_t(x) = (D(x,t) - x)/t^2$ leads to ODE

$$\frac{dx}{dt} = \frac{x - D(x,t)}{t}$$



Left: VP SDE, Center: VE SDE, Right: Karras et al. [2]

Algorithm 2 Our stochastic sampler with $\sigma(t) = t$ and $s(t) = 1$.

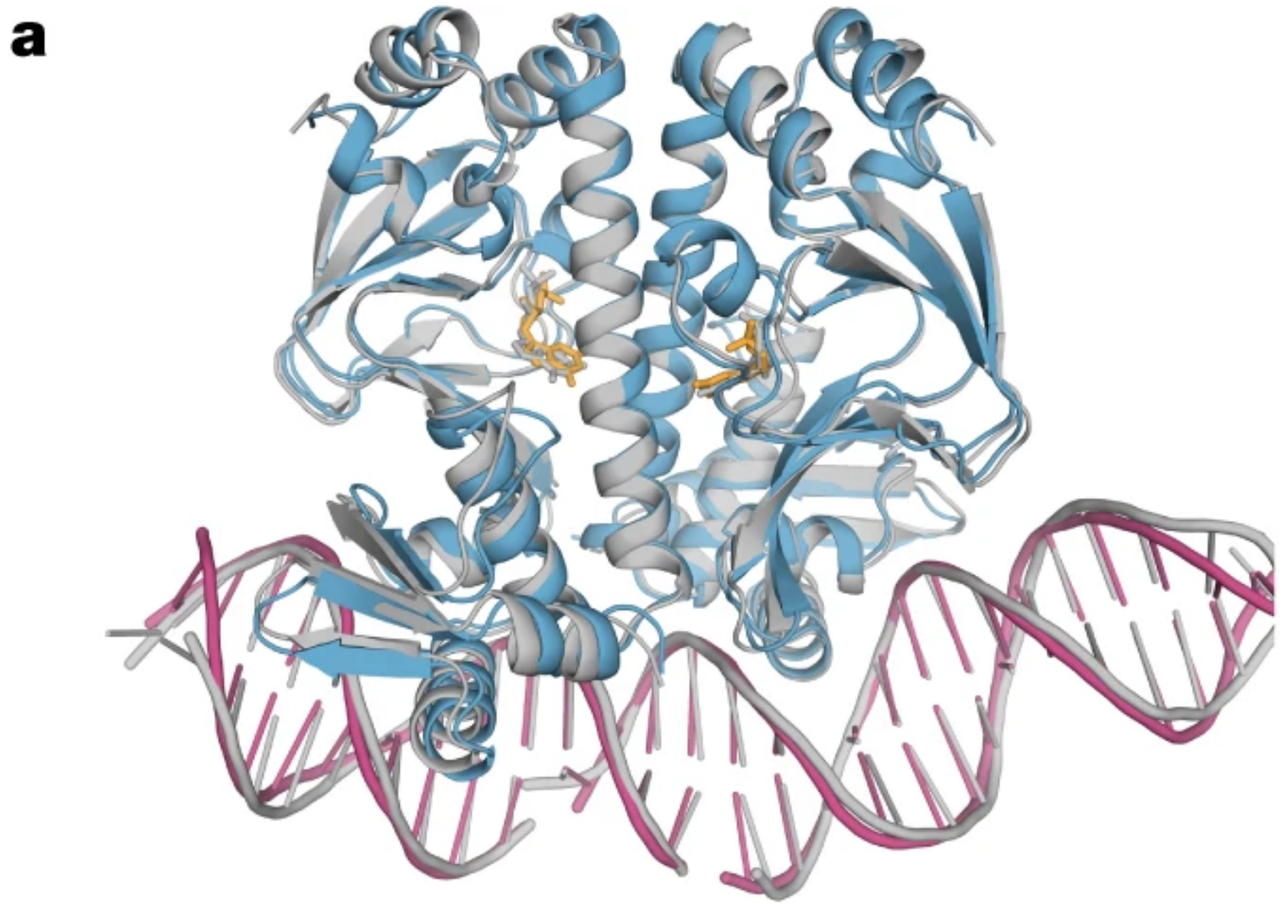
procedure StochasticSampler($D_\theta(\boldsymbol{x}; \sigma)$, $t_{i\in\{0,\dots,N\}}$, $\gamma_{i\in\{0,\dots,N-1\}}$, S_{noise})
  sample $\boldsymbol{x}_0 \sim \mathcal{N}\big(\mathbf{0},\ t_0^2\ \mathbf{I}\big)$
  for $i \in \{0, \dots, N-1\}$ **do**
    sample $\boldsymbol{\epsilon}_i \sim \mathcal{N}\big(\mathbf{0},\ S_{\text{noise}}^2\ \mathbf{I}\big)$
    $\hat{t}_i \leftarrow t_i + \gamma_i t_i$ ▷ Select temporarily increased noise level $\hat{t}_i$
    $\hat{\boldsymbol{x}}_i \leftarrow \boldsymbol{x}_i + \sqrt{\hat{t}_i^2 - t_i^2}\ \boldsymbol{\epsilon}_i$ ▷ Add new noise to move from t_i to $\hat{t}_i$
    $\boldsymbol{d}_i \leftarrow \big(\hat{\boldsymbol{x}}_i - D_\theta(\hat{\boldsymbol{x}}_i; \hat{t}_i)\big)/\hat{t}_i$ ▷ Evaluate $\mathrm{d}\boldsymbol{x}/\mathrm{d}t$ at $\hat{t}_i$
    $\boldsymbol{x}_{i+1} \leftarrow \hat{\boldsymbol{x}}_i + (t_{i+1} - \hat{t}_i)\boldsymbol{d}_i$ ▷ Take Euler step from $\hat{t}_i$ to t_{i+1}
    if $t_{i+1} \neq 0$ **then**
      $\boldsymbol{d}'_i \leftarrow \big(\boldsymbol{x}_{i+1} - D_\theta(\boldsymbol{x}_{i+1}; t_{i+1})\big)/t_{i+1}$ ▷ Apply 2nd order correction
      $\boldsymbol{x}_{i+1} \leftarrow \hat{\boldsymbol{x}}_i + (t_{i+1} - \hat{t}_i)\big(\frac{1}{2}\boldsymbol{d}_i + \frac{1}{2}\boldsymbol{d}'_i\big)$
  return $\boldsymbol{x}_N$

$$\triangleright\ \gamma_i = \begin{cases} \min\left(\frac{S_{\text{churn}}}{N}, \sqrt{2}-1\right) & \text{if } t_i \in [S_{\text{tmin}}, S_{\text{tmax}}] \\ 0 & \text{otherwise} \end{cases}$$

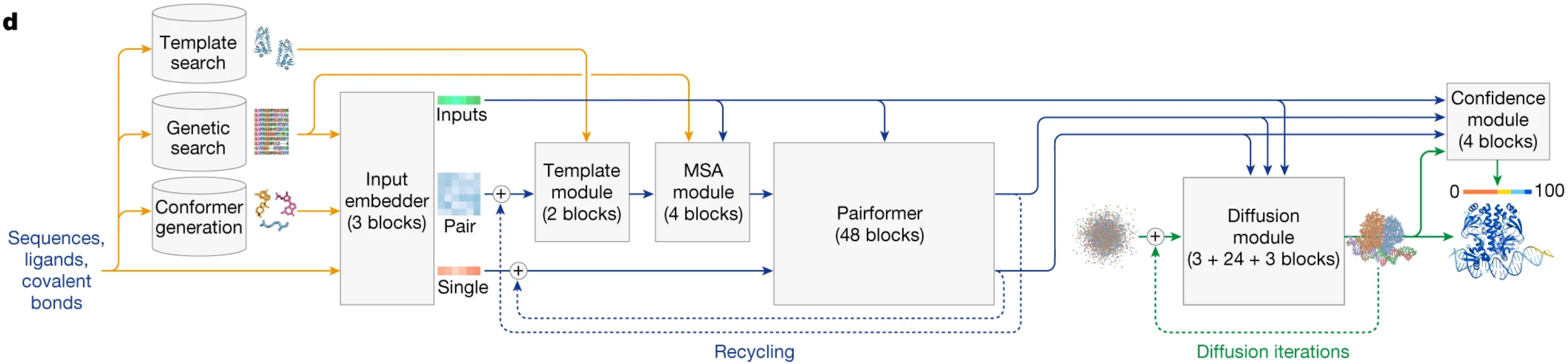
Stochastic sampler from [2]

AlphaFold 3

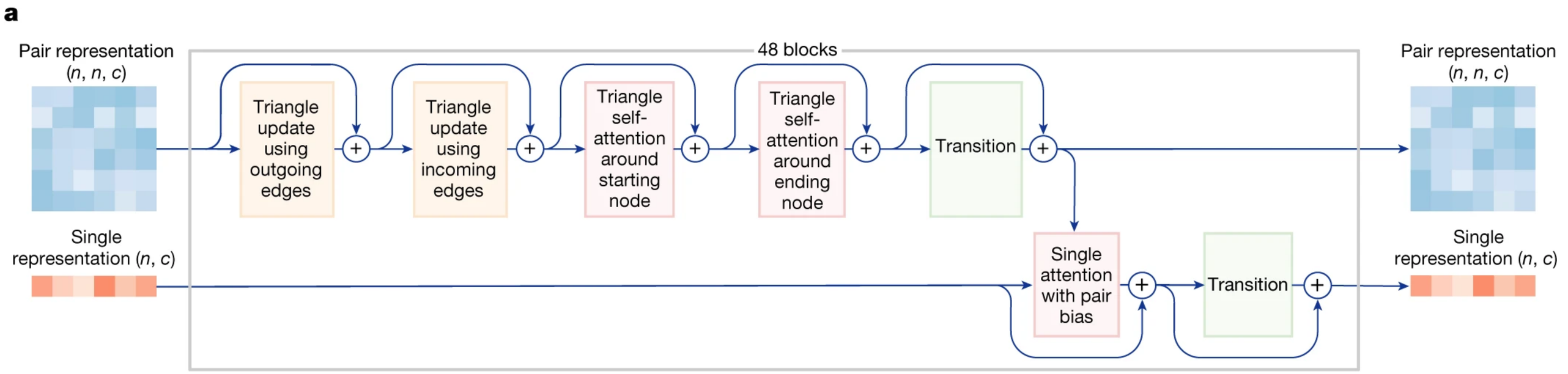
- The most recent iteration of AlphaFold 3 expands its modeling capabilities from just protein structure to the joint structure of complexes including proteins, nucleic acids, ligands, etc.
- Much of the architecture remains similar to AF2, but there are some notable change:
  - The main trunk acts on representations of tokens (polymer residues or atoms from small molecules) instead of just protein residues
  - The evoformer is replaced with a pairformer. The use of MSA is simplified from AF2, and the representation is not retained as the pair representation is updated
  - The structure module is replaced with an all-atom diffusion model
- AF3 is essentially a *conditional* diffusion model, where most of the compute is done on the conditioning information.



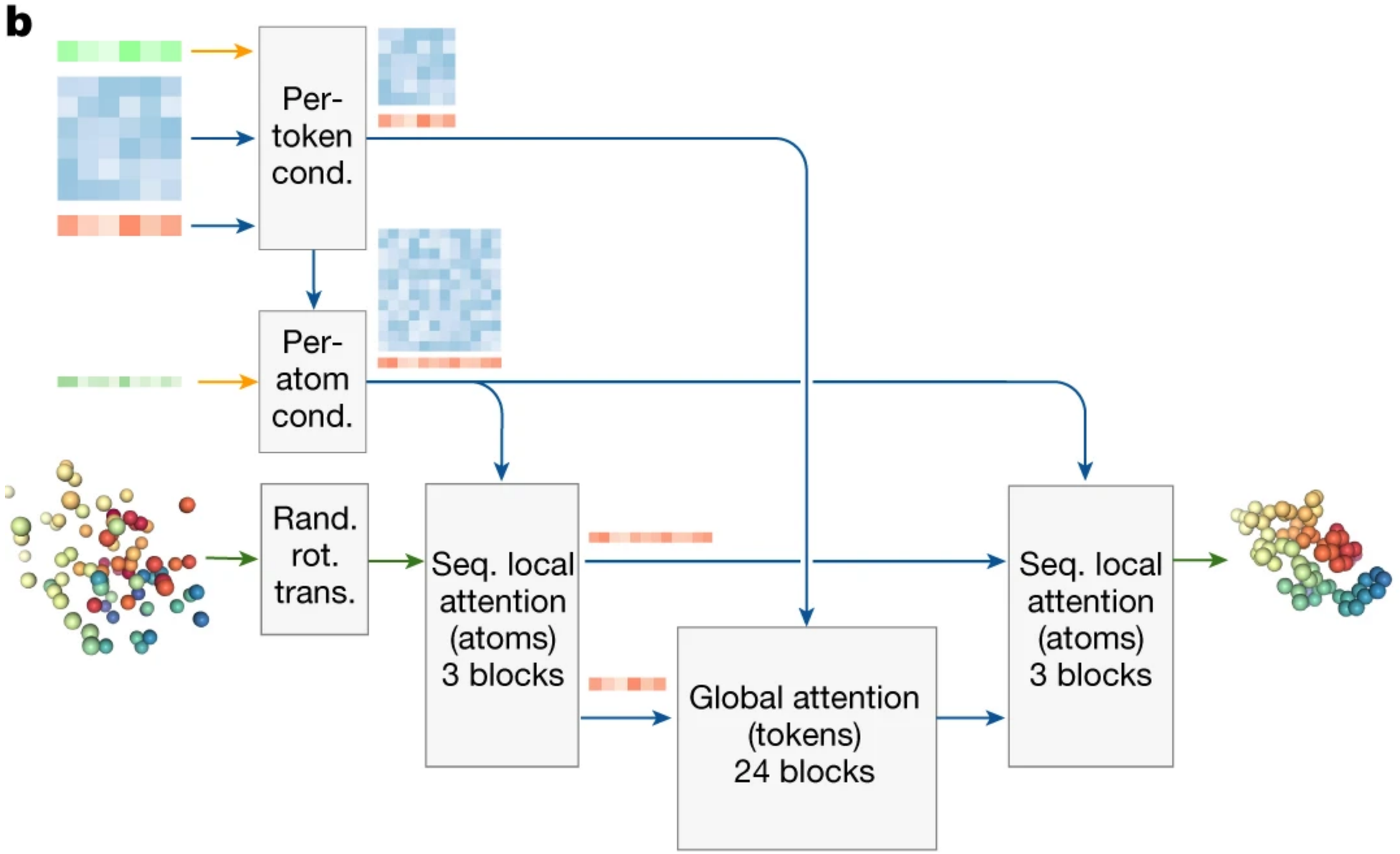
Example structure from AF3 [3]



AF3 pipeline [3]



Pairformer architecture [2]



Diffusion module [3]

Algorithm 18 Sample Diffusion

def SampleDiffusion($\{\mathbf{f}^*\}$, $\{\mathbf{s}_i^{\text{inputs}}\}$, $\{\mathbf{s}_i^{\text{trunk}}\}$, $\{\mathbf{z}_{ij}^{\text{trunk}}\}$, Noise Schedule $[c_0, c_1, \ldots, c_T]$, $\gamma_0 = 0.8$, $\gamma_{\text{min}} = 1.0$, noise scale $\lambda = 1.003$, step scale $\eta = 1.5$):

$\vec{\mathbf{x}}_l \sim c_0 \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$ $\quad \vec{\mathbf{x}}_l \in \mathbb{R}^3$

for all $c_\tau \in [c_1, \ldots, c_T]$ **do**

  $\{\vec{\mathbf{x}}_l\} \leftarrow \text{CentreRandomAugmentation}(\{\vec{\mathbf{x}}_l\})$

  $\gamma = \gamma_0$ if $c_\tau > \gamma_{\text{min}}$ else 0

  $\hat{t} = c_{\tau-1}(\gamma + 1)$

  $\vec{\xi}_l = \lambda\sqrt{\hat{t}^2 - c_{\tau-1}^2} \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$ $\quad \vec{\xi}_l \in \mathbb{R}^3$

  $\vec{\mathbf{x}}_l^{\text{noisy}} = \vec{\mathbf{x}}_l + \vec{\xi}_l$

  $\{\vec{\mathbf{x}}_l^{\text{denoised}}\} = \text{DiffusionModule}(\{\vec{\mathbf{x}}_l^{\text{noisy}}\}, \hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\})$

  $\vec{\delta}_l = (\vec{\mathbf{x}}_l - \vec{\mathbf{x}}_l^{\text{denoised}})/\hat{t}$

  $dt = c_\tau - \hat{t}$

  $\vec{\mathbf{x}}_l \leftarrow \vec{\mathbf{x}}_l^{\text{noisy}} + \eta \cdot dt \cdot \vec{\delta}_l$

end for

return $\{\vec{\mathbf{x}}_l\}$

Algorithm 20 Diffusion Module

def DiffusionModule($\{\vec{\mathbf{x}}_l^{\text{noisy}}\}, \hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\}$, $\sigma_{\text{data}} = 16, c_{\text{atom}} = 128, c_{\text{atompair}} = 16, c_{\text{token}} = 768$) :

 # Conditioning

$\{\mathbf{s}_i\}, \{\mathbf{z}_{ij}\} = \text{DiffusionConditioning}(\hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\}, \sigma_{\text{data}})$

 # Scale positions to dimensionless vectors with approximately unit variance.

$\mathbf{r}_l^{\text{noisy}} = \vec{\mathbf{x}}_l^{\text{noisy}} / \sqrt{\hat{t}^2 + \sigma_{\text{data}}^2}$ $\qquad \mathbf{r}_l^{\text{noisy}} \in \mathbb{R}^3$

 # Sequence-local Atom Attention and aggregation to coarse-grained tokens

$\{\mathbf{a}_i\}, \{\mathbf{q}_l^{\text{skip}}\}, \{\mathbf{c}_l^{\text{skip}}\}, \{\mathbf{p}_{lm}^{\text{skip}}\} = \text{AtomAttentionEncoder}(\{\mathbf{f}^*\}, \{\mathbf{r}_l^{\text{noisy}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}\}, c_{\text{atom}}, c_{\text{atompair}}, c_{\text{token}})$ $\qquad \mathbf{a}_i \in \mathbb{R}^{c_{\text{token}}}$

 # Full self-attention on token level.

$\mathbf{a}_i \mathrel{+}= \text{LinearNoBias}(\text{LayerNorm}(\mathbf{s}_i))$

$\{\mathbf{a}_i\} \leftarrow \text{DiffusionTransformer}(\{\mathbf{a}_i\}, \{\mathbf{s}_i\}, \{\mathbf{z}_{ij}\}, \beta_{ij} = 0, N_{\text{block}} = 24, N_{\text{head}} = 16)$

$\mathbf{a}_i \leftarrow \text{LayerNorm}(\mathbf{a}_i)$

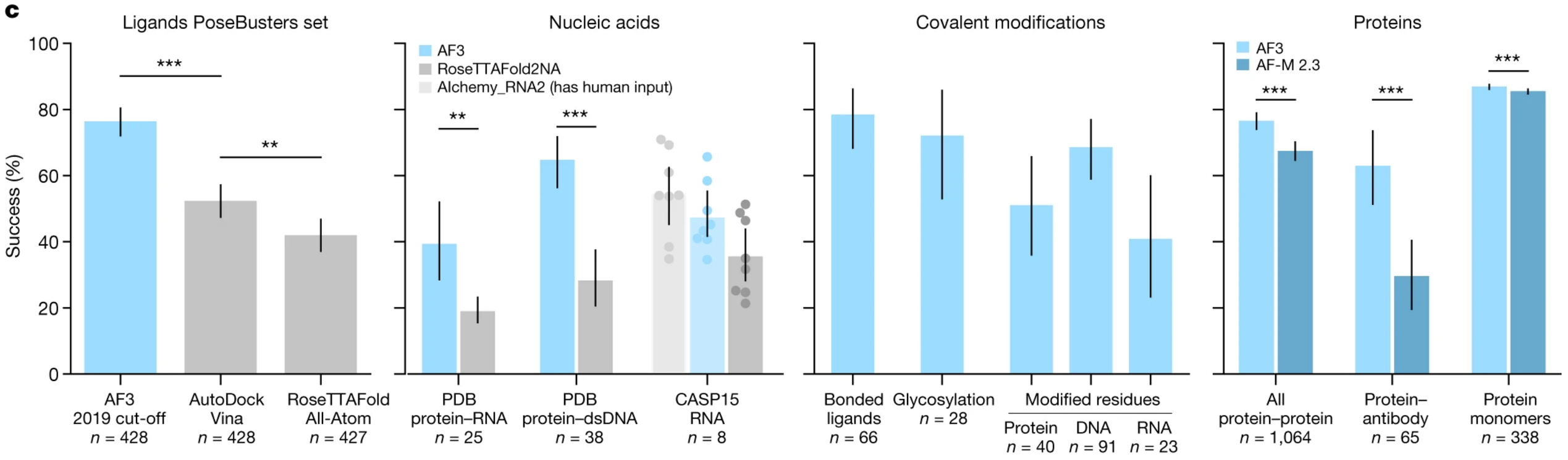
 # Broadcast token activations to atoms and run Sequence-local Atom Attention

$\{\mathbf{r}_l^{\text{update}}\} = \text{AtomAttentionDecoder}(\{\mathbf{a}_i\}, \{\mathbf{q}_l^{\text{skip}}\}, \{\mathbf{c}_l^{\text{skip}}\}, \{\mathbf{p}_{lm}^{\text{skip}}\})$

 # Rescale updates to positions and combine with input positions

$\vec{\mathbf{x}}_l^{\text{out}} = \sigma_{\text{data}}^2 / (\sigma_{\text{data}}^2 + \hat{t}^2) \cdot \vec{\mathbf{x}}_l^{\text{noisy}} + \sigma_{\text{data}} \cdot \hat{t} / \sqrt{\sigma_{\text{data}}^2 + \hat{t}^2} \cdot \mathbf{r}_l^{\text{update}}$

return $\{\vec{\mathbf{x}}_l^{\text{out}}\}$


c
Ligands PoseBusters set
Nucleic acids
Covalent modifications
Proteins
Success (%)
100
80
60
40
20
0
***
**
AF3
RoseTTAFold2NA
AIchemy_RNA2 (has human input)
AF3
AF-M 2.3
AF3 2019 cut-off n = 428
AutoDock Vina n = 428
RoseTTAFold All-Atom n = 427
PDB protein–RNA n = 25
PDB protein–dsDNA n = 38
CASP15 RNA n = 8
Bonded ligands n = 66
Glycosylation n = 28
Modified residues
Protein n = 40
DNA n = 91
RNA n = 23
All protein–protein n = 1,064
Protein–antibody n = 65
Protein monomers n = 338

Task	Dataset	Metric	Notes	Method	N	Mean	95% CI
Ligands	PoseBusters V1	% RMSD < 2 Å	–	RoseTTAFold All-Atom	427	42.0	37.2 – 46.8
				AF3 (2019 cutoff)	428	**76.4**	72.1 – 80.3
			Holo protein struct. given	EquiBind	428	2.6	1.3 – 4.6
				TankBind	428	15.0	11.7 – 18.7
				DiffDock	428	37.9	33.2 – 42.6
			Pocket residues specified	Vina on AF-M 2.3	428	13.1	10.0 – 16.7
				DeepDock	428	17.8	14.3 – 21.7
				Uni-Mol	428	22.9	19.0 – 27.2
				UMol	428	45.0	40.3 – 49.9
				Gold	428	51.2	46.3 – 56.0
				Vina	428	52.3	47.5 – 57.2
				Uni-Mol Docking V2	428	77.6	73.3 – 81.4
				AF3 (2019 cutoff) pocket specified	428	**90.2**	87.0 – 92.8
	PoseBusters V2	% RMSD < 2 Å	–	AF3 (2019 cutoff)	308	**80.5**	75.6 – 84.8
			Holo protein struct. given	EquiBind	308	1.9	0.7 – 4.2
				TankBind	308	15.9	12.0 – 20.5
				DiffDock	308	38.0	32.5 – 43.7
			Pocket residues specified	Vina on AF-M 2.3	308	15.3	11.4 – 19.8
				DeepDock	308	19.5	15.2 – 24.4
				Uni-Mol	308	21.8	17.3 – 26.8
				Gold	308	58.1	52.4 – 63.7
				Vina	308	59.7	54.0 – 65.3
				AF3 (2019 cutoff) pocket specified	308	**93.2**	89.8 – 95.7
Nucleic Acids	Protein-RNA	iLDDT		RoseTTAFold2NA	25	19.0	15.6 – 23.2
				AF3	25	**39.4**	28.5 – 51.9
	Protein-dsDNA	iLDDT		RoseTTAFold2NA	38	28.3	20.7 – 37.5
				AF3	38	**64.8**	56.4 – 71.7
	CASP 15 RNA	RNA LDDT		RoseTTAFold2NA	8	35.5	28.3 – 43.8
				AF3	8	47.3	41.7 – 55.2
				Alchemy_RNA2 (has human input)	8	**54.5**	45.3 – 62.4
				RNApolis (has human input)	8	50.5	45.2 – 55.8
				Chen (has human input)	8	49.8	40.7 – 58.5
				Kiharalab	8	40.9	35.1 – 54.3
				UltraFold	8	37.8	32.5 – 45.0
Covalent Mod.	Bonded ligands	% RMSD < 2 Å		AF3	66	**78.5**	68.3 – 86.2
	Glycosylation	% RMSD < 2 Å	high-quality, single-residue	AF3	28	**72.1**	53.1 – 85.7
			all-quality, single-residue	AF3	167	**46.0**	40.0 – 52.1
			all-quality, multi-residue	AF3	131	**42.4**	35.4 – 49.3
	Modified residues	% RMSD < 2 Å		AF3	154	**59.9**	52.4 – 67.0
	Modified protein residues	% RMSD < 2 Å		AF3	40	**51.0**	36.0 – 65.6
	Modified DNA residues	% RMSD < 2 Å		AF3	91	**68.6**	59.0 – 76.9
	Modified RNA residues	% RMSD < 2 Å		AF3	23	**40.9**	23.4 – 59.9
Proteins	All Protein-Protein	% dockq > 0.23		AF-M 2.3	1064	67.5	64.7 – 70.1
				AF3	1064	**76.6**	74.0 – 78.9
	Protein-Antibody	% dockq > 0.23		AF-M 2.3	65	29.6	19.6 – 40.4
				AF3	65	**62.9**	51.4 – 73.5
	Monomers	LDDT		AF-M 2.3	338	85.5	84.7 – 86.1
				AF3	338	**86.9**	86.2 – 87.6

References

[1] Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv preprint arXiv:2011.13456* (2020).

[2] Karras, Tero, et al. "Elucidating the design space of diffusion-based generative models." *Advances in neural information processing systems* 35 (2022): 26565-26577.

[3] Abramson, Josh, et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3." *Nature* (2024): 1-3.