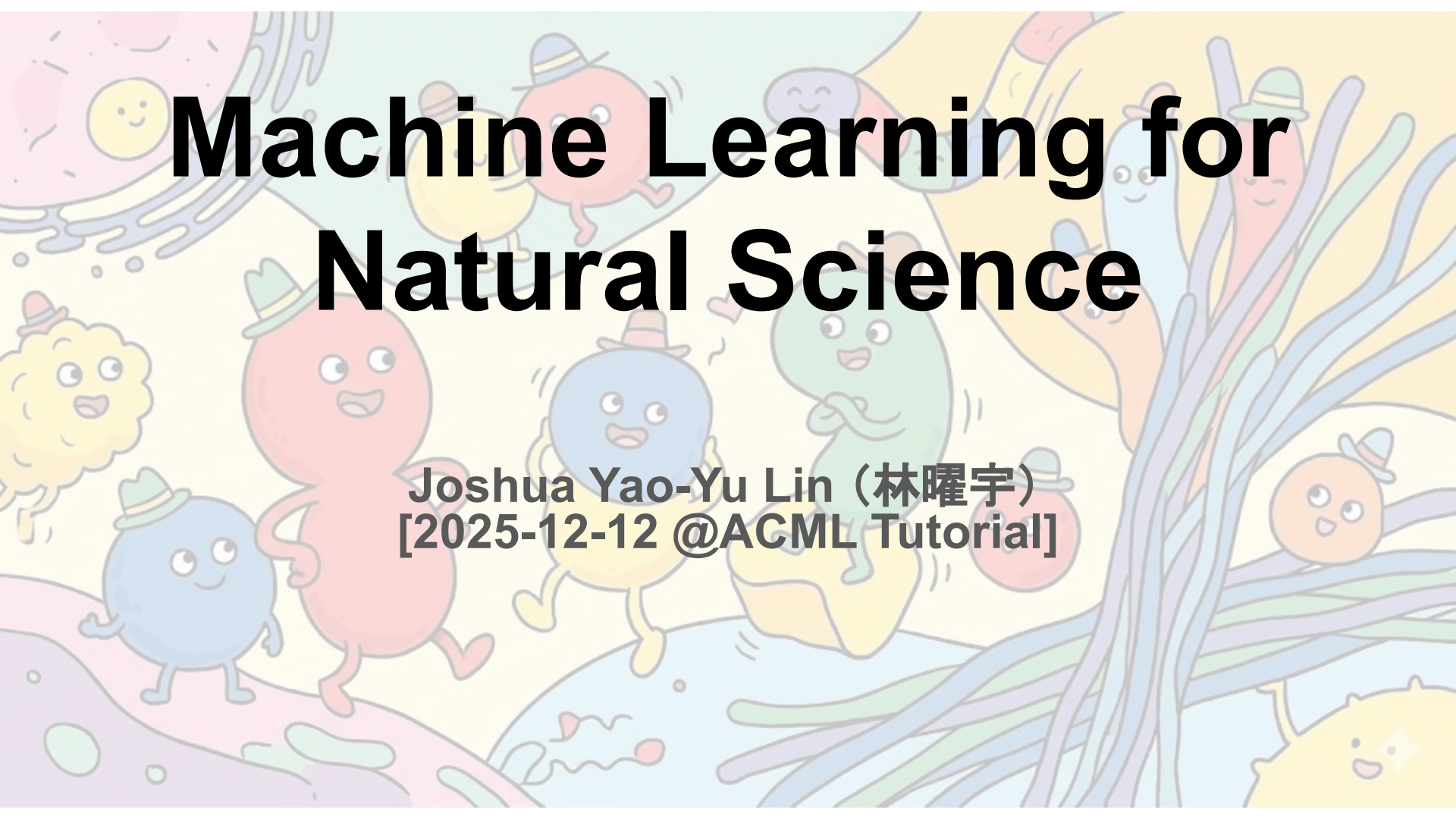
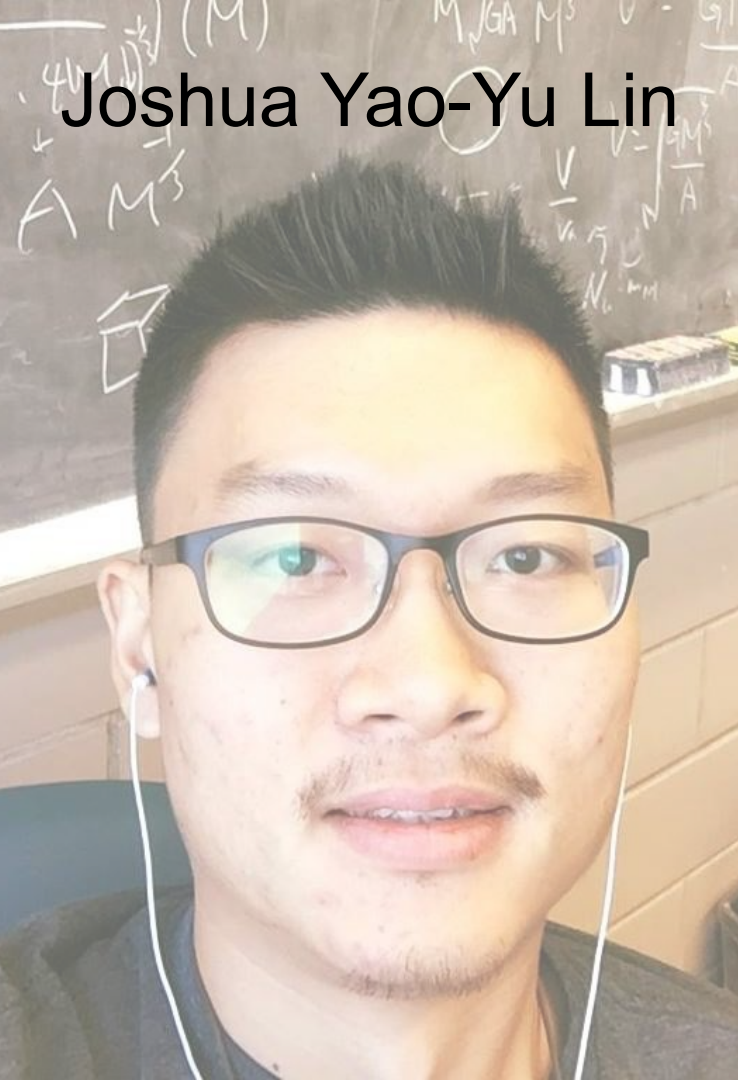
# Machine Learning for Natural Science



Joshua Yao-Yu Lin（林曜宇）
[2025-12-12 @ACML Tutorial]

# Joshua Yao-Yu Lin



- Currently a Machine Learning Scientist at Genentech (New York office)

- UIUC Physics Ph.D. (2016-2022), MS at NTU, and BS at NTHU in Taiwan.

- My past research spans a wide range of Machine Learning application for astrophysics, including black hole image and dark matter/strong lensing 

- ML Research Interest: **ML for Natural Science**, **ML for drug discovery & Protein Design**

- Previous ML experience: Genentech Postdoc with Prof. Kyunghyun Cho, Simons Foundation/Flatiron Institute (CCA), Google Research (2021)

- I like: Traveling, Jazz, Bouldering/Climbing, Brewing hard cider

# Prescient Design@Genentech/Roche

**Vladimir Gligorijevic**

Co-Founder and Senior Director, Prescient Design, Genentech

**Richard Bonneau**

Co-Founder and Executive Director, Prescient Design, Genentech

**Kyunghyun Cho**

Co-Founder and Senior Director, Prescient Design, Genentech

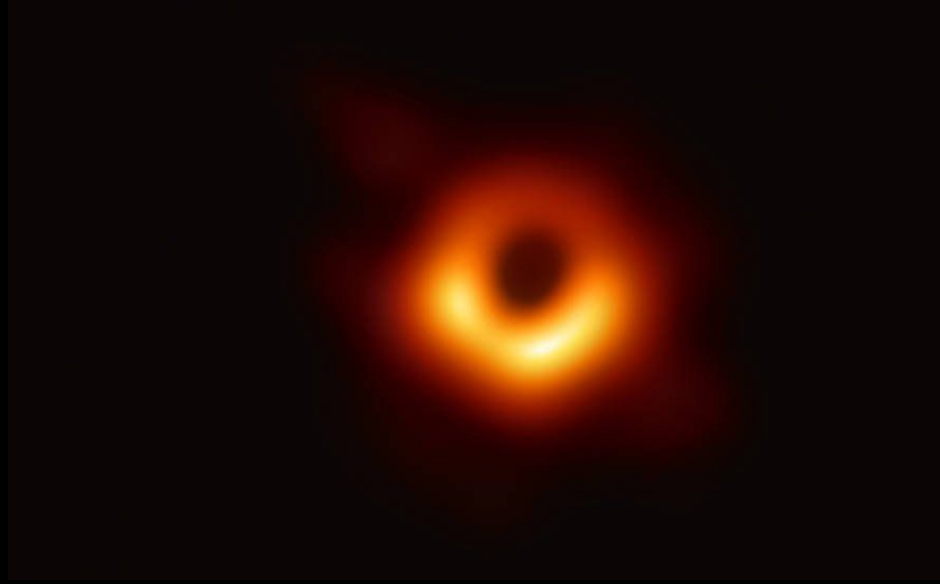
Prev. @Flatiron Institute/NYU

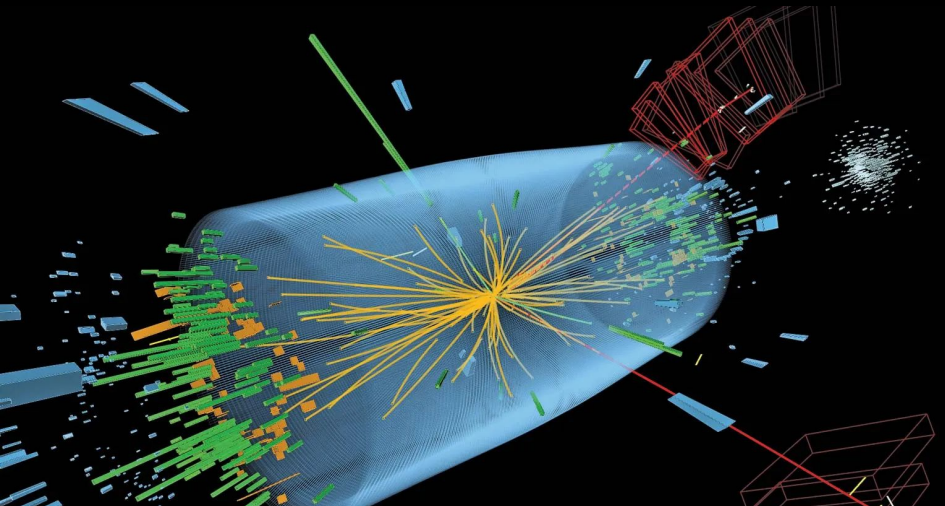
NYU CS/Data Science

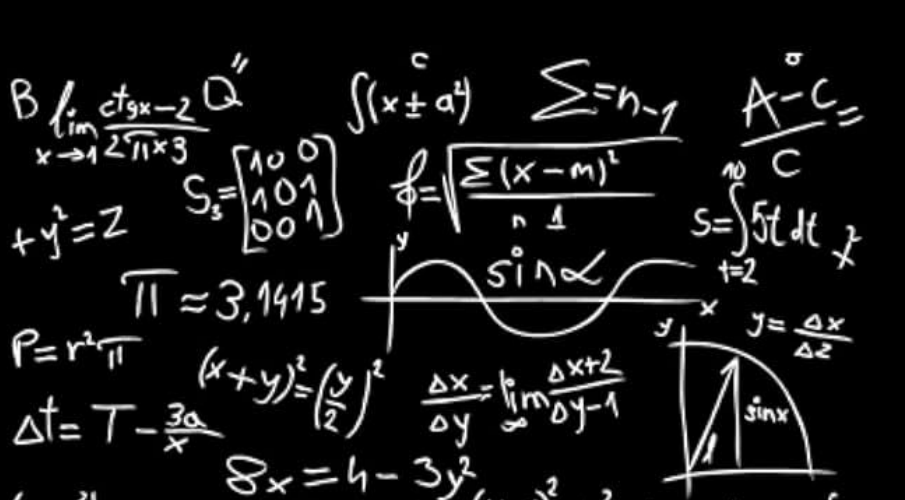
- Founded in Jan 2021, focusing on machine learning for **Protein Design/drug discovery**
- Acquired by **Genentech/Roche** ~ August 2021
- Around 70 people in the team (ML Scientist/ Engineer, Bio/Chem)
- We're hiring!

Prescient Design

A Genentech Accelerator

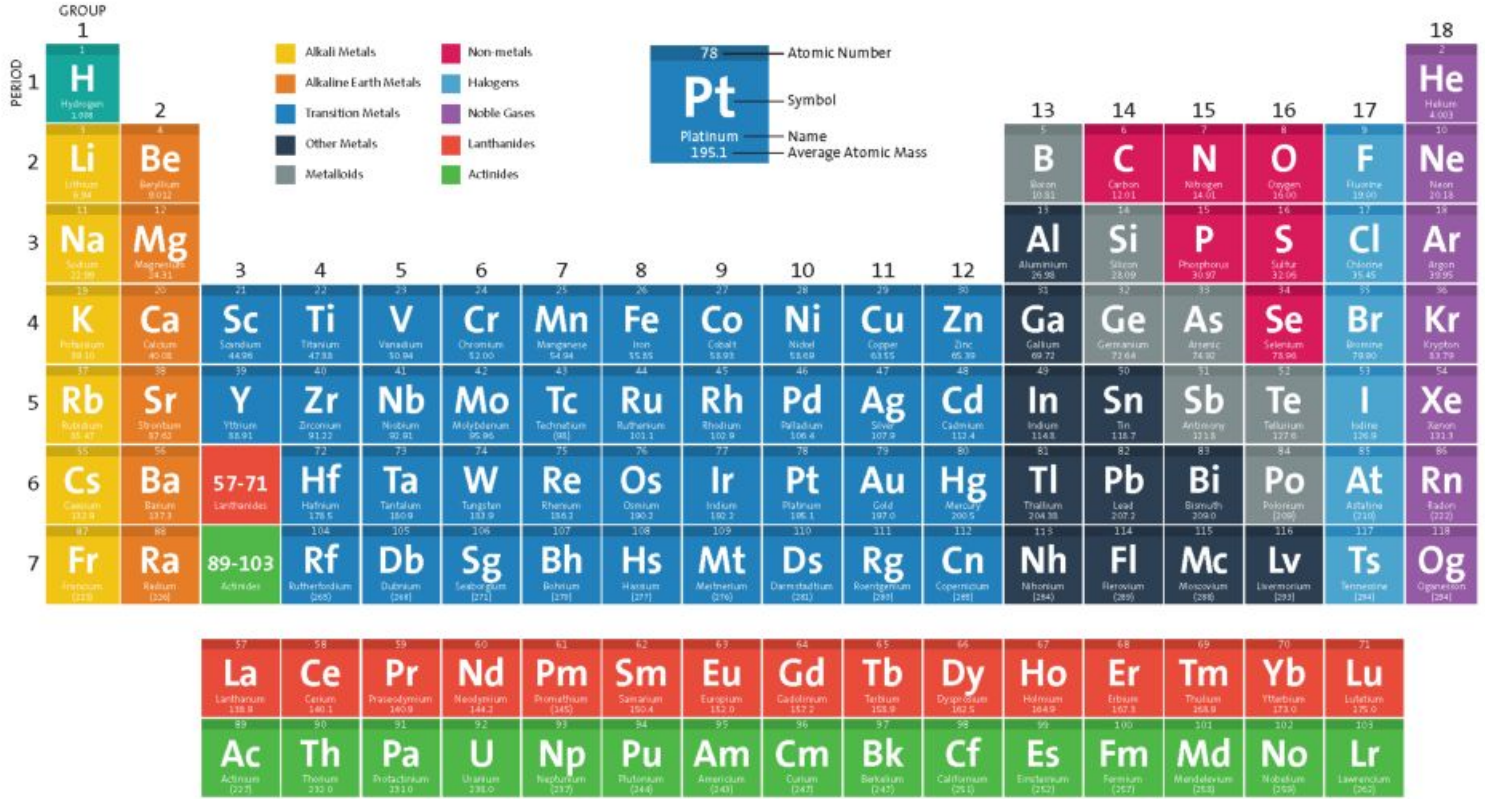





sinx
sinx

ACS Chemistry for Life®

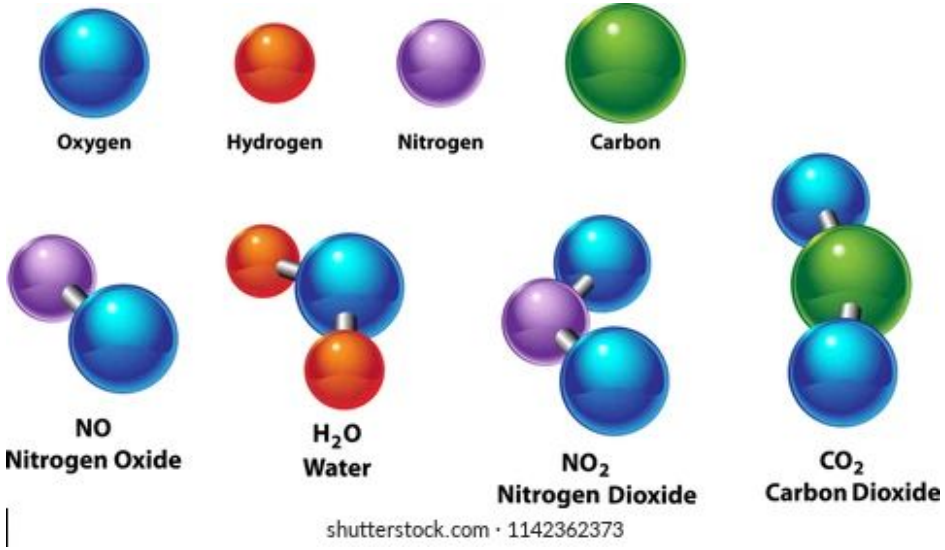
# PERIODIC TABLE OF ELEMENTS



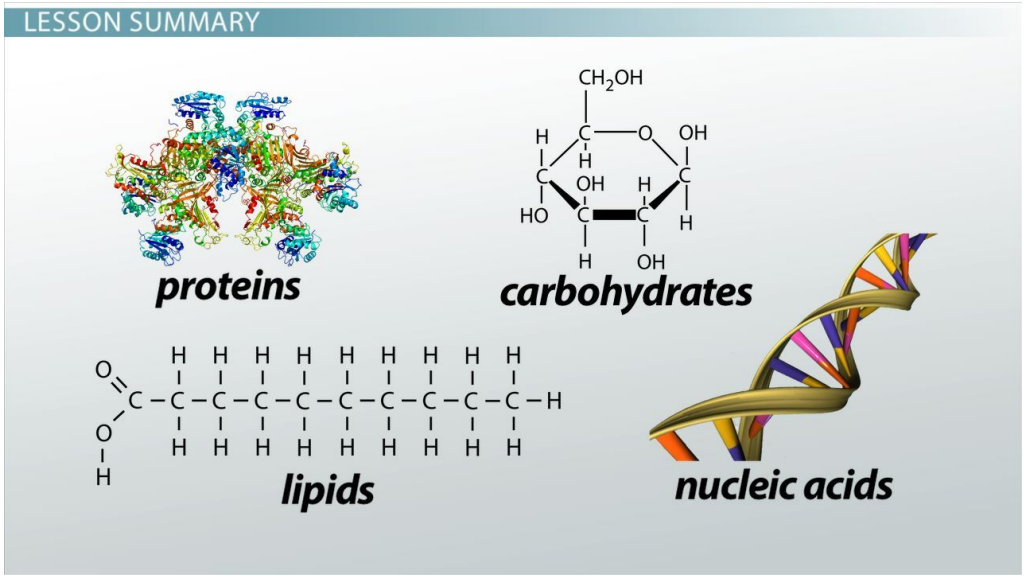

American Chemical Society

www.acs.org/outreach

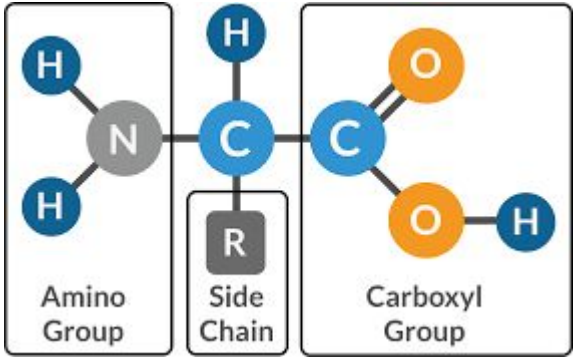
# Atom -> Molecules

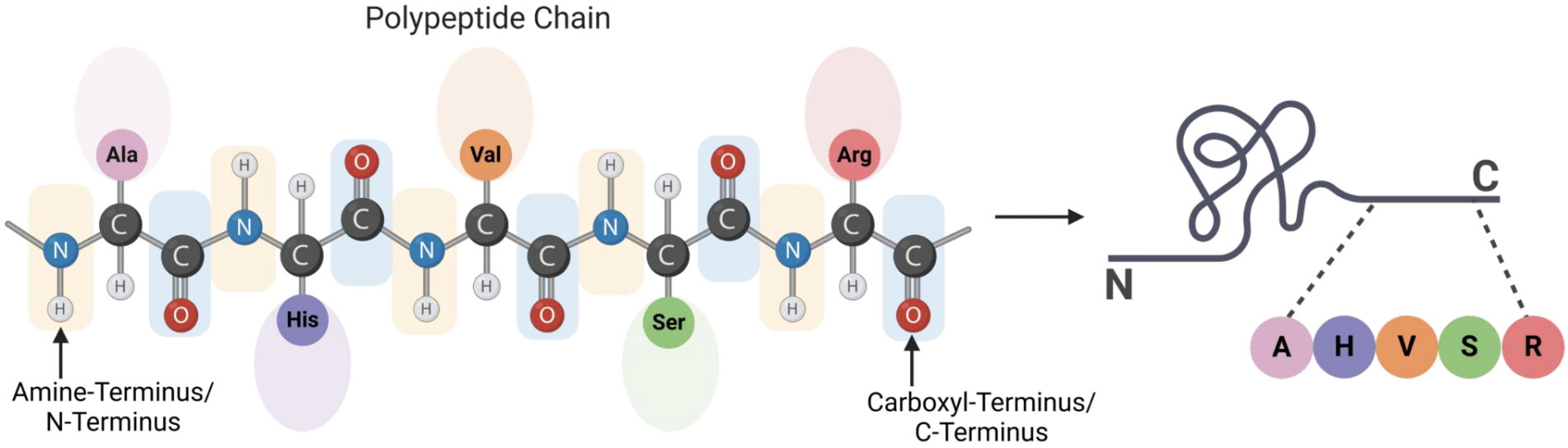


# Large Molecules



# Amino Acids

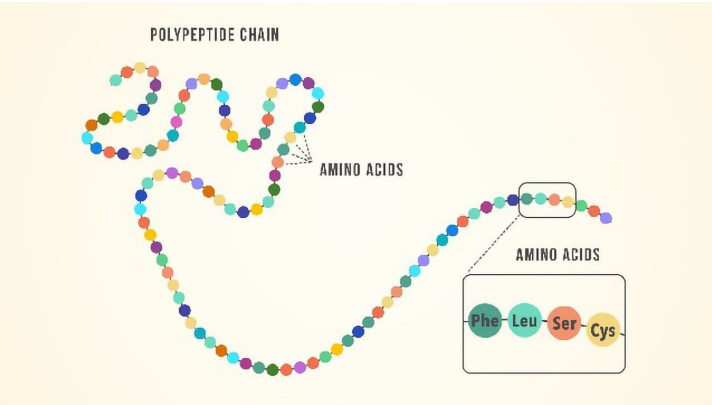

Amino Group
Side Chain
Carboxyl Group

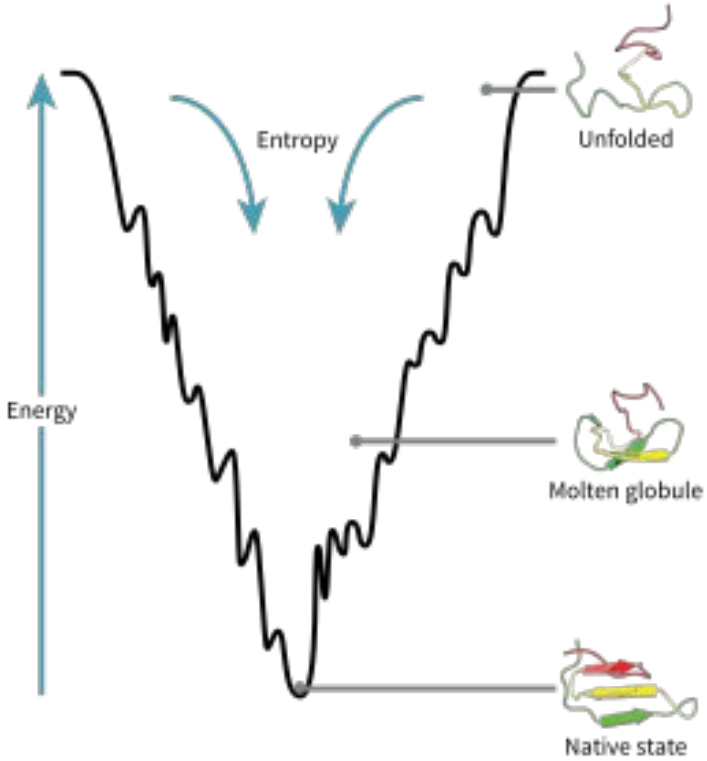

Polypeptide Chain
Ala
His
Val
Ser
Arg
Amine-Terminus/
N-Terminus
Carboxyl-Terminus/
C-Terminus
N
C
A
H
V
S
R

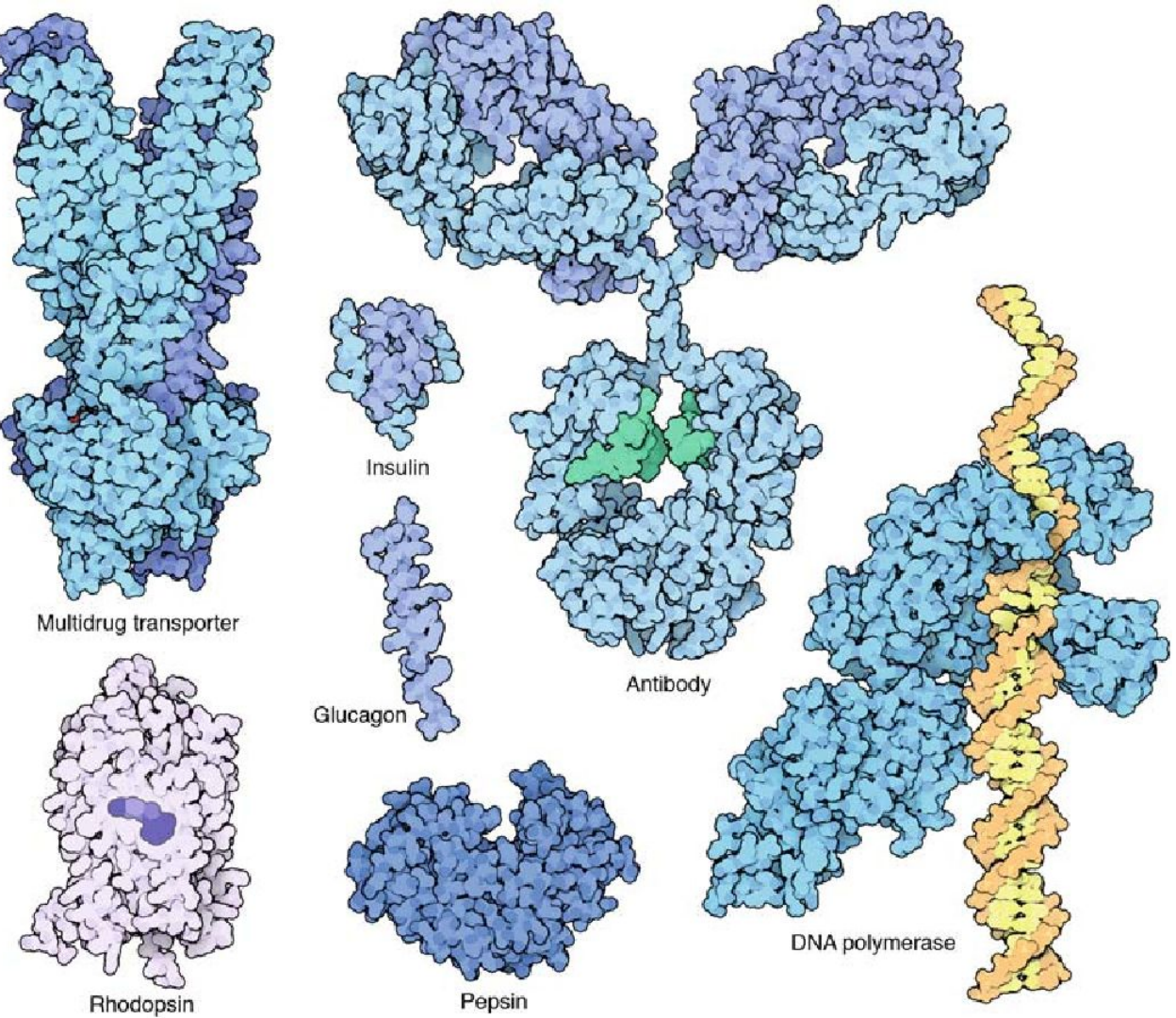
# Protein folding problem

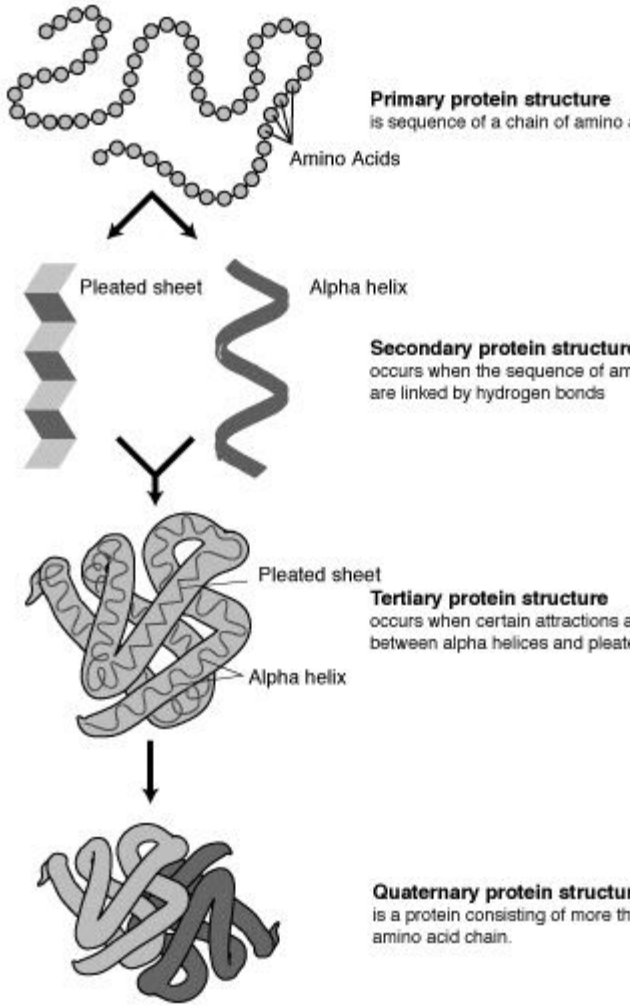
Protein’s amino acid sequence -> three-dimensional atomic structure prediction.

The notion of a folding “problem” first emerged around 1960, with the appearance of the first atomic-resolution protein structures




Multidrug transporter
Rhodopsin
Insulin
Glucagon
Antibody
Pepsin
DNA polymerase


Amino Acids
Primary protein structure
is sequence of a chain of amino
Pleated sheet
Alpha helix
Secondary protein structur
occurs when the sequence of am
are linked by hydrogen bonds
Pleated sheet
Alpha helix
Tertiary protein structure
occurs when certain attractions a
between alpha helices and pleat
Quaternary protein structur
is a protein consisting of more th
amino acid chain.

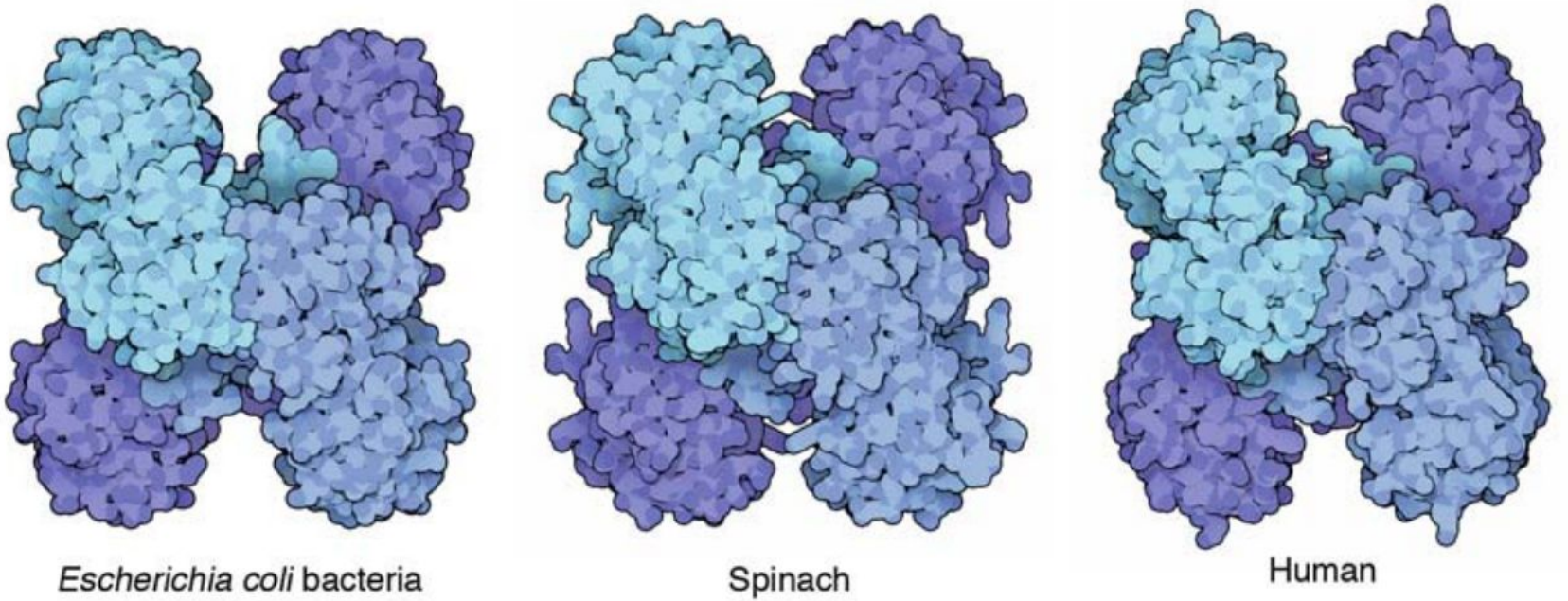
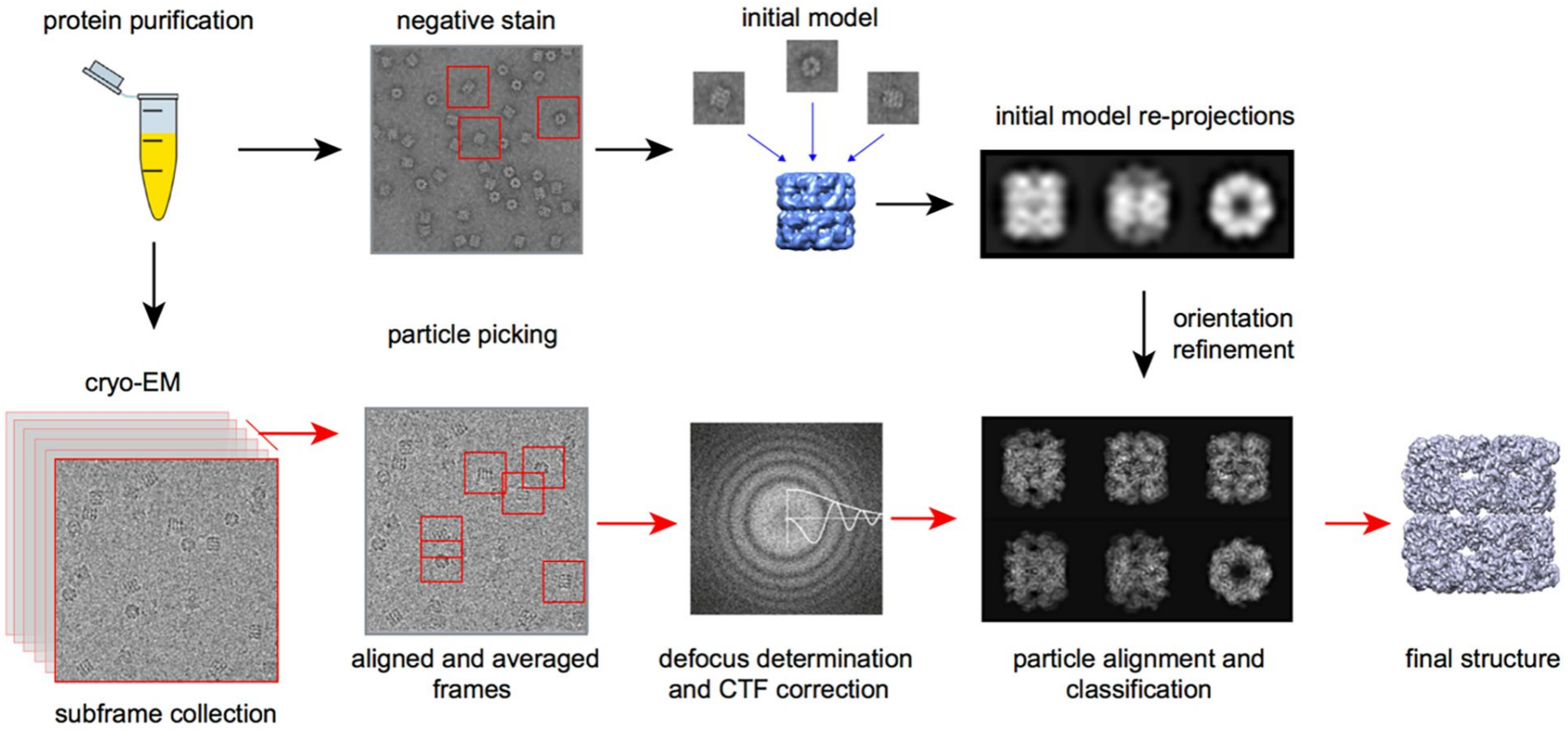
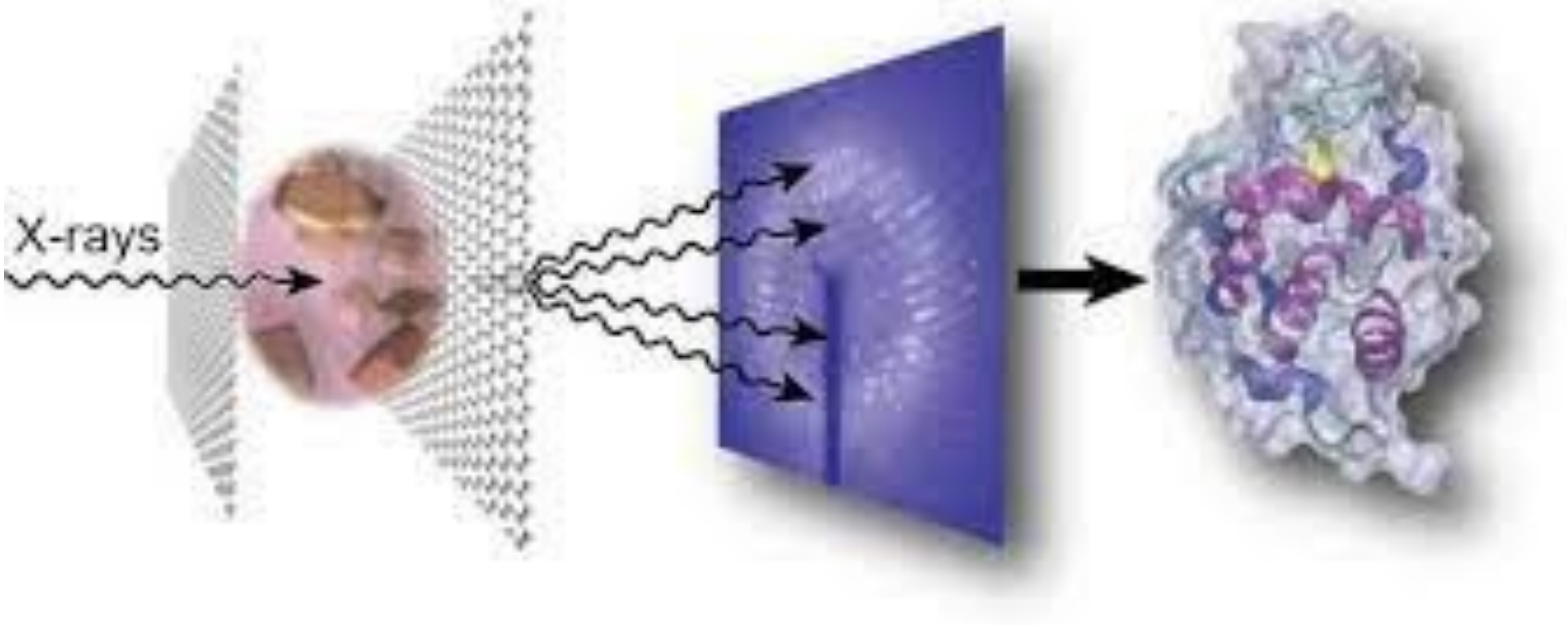



**Fig. 1.2 Molecular Machinery** Many molecular machines are virtually identical in all living cells. This is particularly true for molecules that play an essential role in the processes of life, such as the enzyme glyceraldehyde-3-phosphate dehydrogenase, which is vital for the metabolism of sugar in all three organisms. This illustration shows the similar form of the enzyme from a bacterial cell (left), a plant cell (center), and human cells (right) (5,000,000 X)

# How to get Protein Structures: Cryo-EM

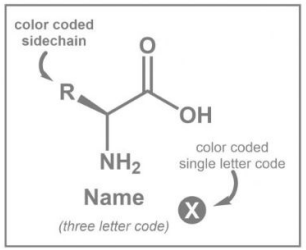


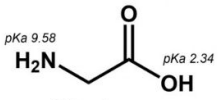
# How to get Protein Structures: Crystallography



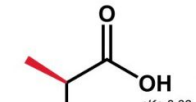
# THE 20 COMMON AMINO ACIDS

ALIPHATIC · AROMATIC · AMIDIC · HYDROXYLIC · ⊖ CHARGED · ⊕ CHARGED · SULFUR CONTAINING

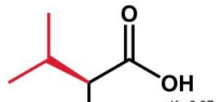


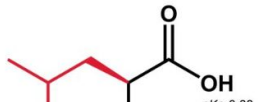

**Glycine** (Gly) G



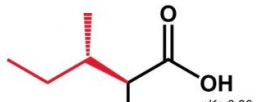

**Alanine** (Ala) A



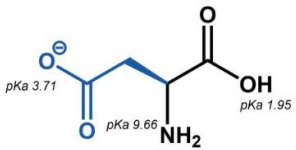

**Valine** (Val) V



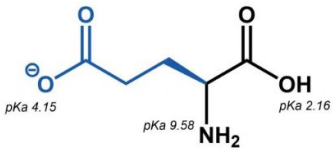

**Leucine** (Leu) L



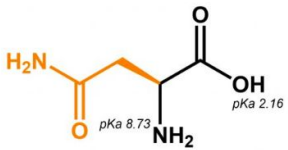

**Isoleucine** (Ile) I



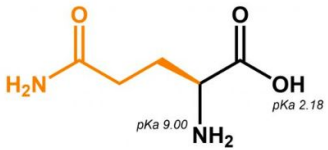

**Aspartic acid** (Asp) D



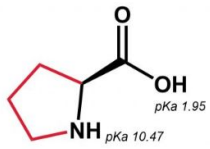

**Glutamic acid** (Glu) E



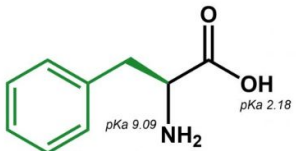

**Asparagine** (Asn) N



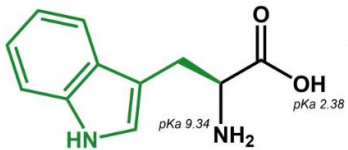

**Glutamine** (Gln) Q



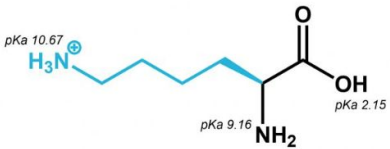

**Proline** (Pro) P



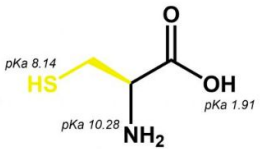

**Phenylalanine** (Phe) F



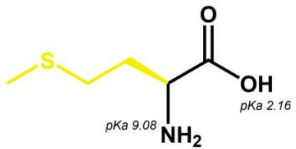

**Tryptophan** (Trp) W



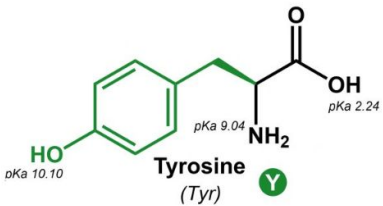

**Lysine** (Lys) K



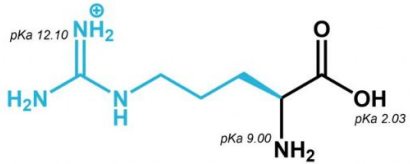

**Cysteine** (Cys) C



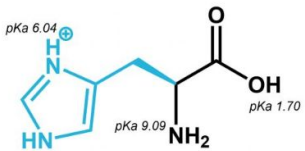

**Methionine** (Met) M



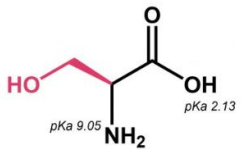

**Tyrosine** (Tyr) Y



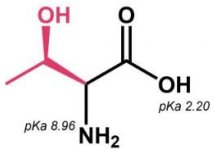

**Arginine** (Arg) R




**Histidine** (His) H




**Serine** (Ser) S

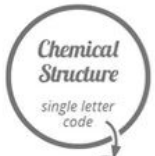


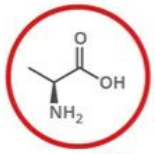

**Threonine** (Thr) T

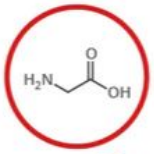
# A GUIDE TO THE TWENTY COMMON AMINO ACIDS

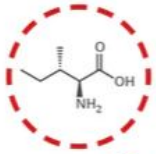
AMINO ACIDS ARE THE BUILDING BLOCKS OF PROTEINS IN LIVING ORGANISMS. THERE ARE OVER 500 AMINO ACIDS FOUND IN NATURE - HOWEVER, THE HUMAN GENETIC CODE ONLY DIRECTLY ENCODES 20. 'ESSENTIAL' AMINO ACIDS MUST BE OBTAINED FROM THE DIET, WHILST NON-ESSENTIAL AMINO ACIDS CAN BE SYNTHESISED IN THE BODY.

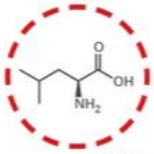
**Chart Key:** ALIPHATIC · AROMATIC · ACIDIC · BASIC · HYDROXYLIC · SULFUR-CONTAINING · AMIDIC · NON-ESSENTIAL · ESSENTIAL

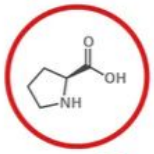


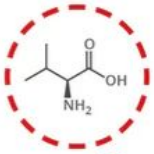



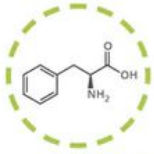



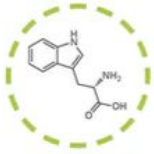



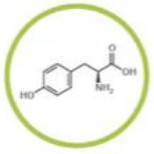



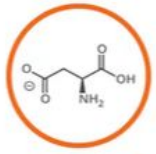



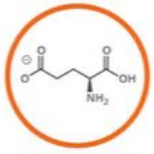



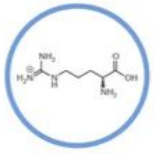



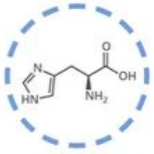



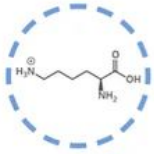



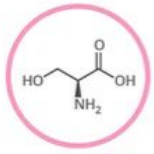



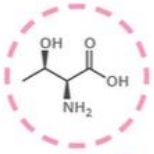



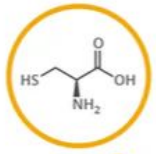



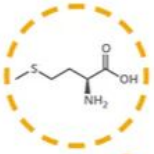



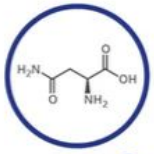



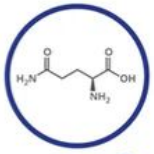



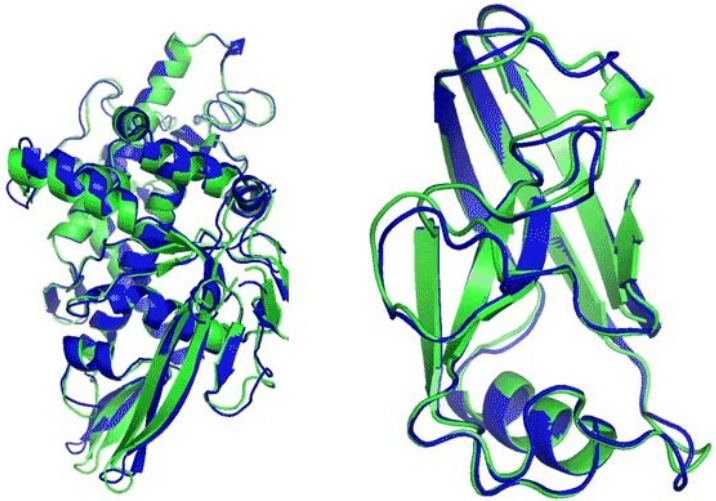




**Note:** This chart only shows those amino acids for which the human genetic code directly codes for. Selenocysteine is often referred to as the 21st amino acid, but is encoded in a special manner. In some cases, distinguishing between asparagine/aspartic acid and glutamine/glutamic acid is difficult. In these cases, the codes asx (B) and glx (Z) are respectively used.

# Critical Assessment of protein Structure Prediction (CASP)

- Biennial **global blind-prediction experiment** (since 1994)
- Fully blind evaluation → no data leakage → highest credibility



*Protein folding example, where predictions (in blue) are aligned to the known protein structure (in green) in the Protein DataBase (PDB). (image source: DeepMind blog)*

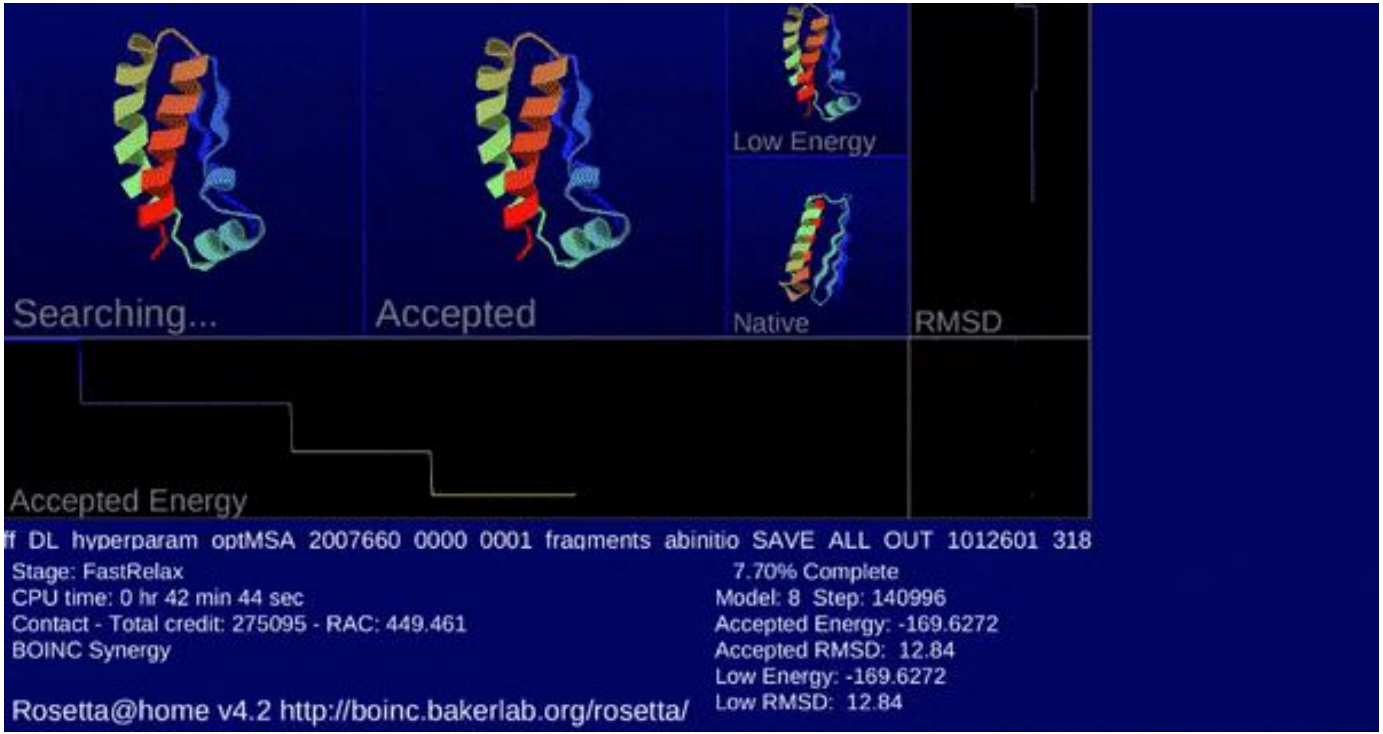
# David Baker's Lab

- Professor at Univ Washington
- Known for **computational Protein Design**
- **Rosetta** (physics-based)
- **RoseTTAFold** (**ML** for protein structure prediction)
- **Protein MPNN** (**ML** for protein inverse folding)
- **RF diffusion** (**generative model** for protein)

David Baker

David Baker's Lab

# Rosetta (Physics based structure prediction)




Andrew Leaver-Fay
& Rosetta Com

# Alphafold II

John Jumper (DeepMind)

Article

# Highly accurate protein structure prediction with AlphaFold

https://doi.org/10.1038/s41586-021-03819-2

Received: 11 May 2021

Accepted: 12 July 2021

Published online: 15 July 2021

Open access

Check for updates

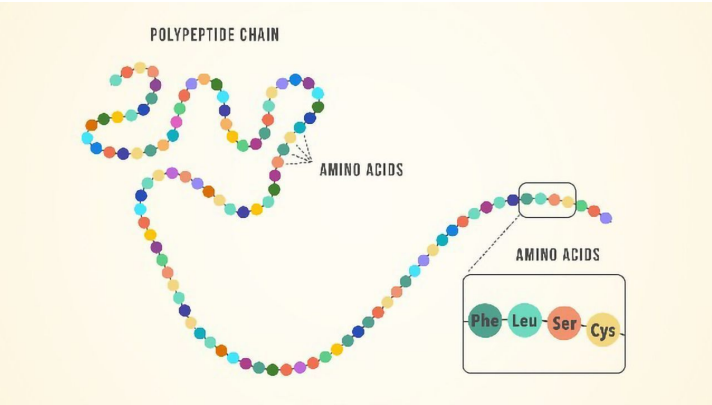
John Jumper[1,4]✉, Richard Evans[1,4], Alexander Pritzel[1,4], Tim Green[1,4], Michael Figurnov[1,4], Olaf Ronneberger[1,4], Kathryn Tunyasuvunakool[1,4], Russ Bates[1,4], Augustin Žídek[1,4], Anna Potapenko[1,4], Alex Bridgland[1,4], Clemens Meyer[1,4], Simon A. A. Kohl[1,4], Andrew J. Ballard[1,4], Andrew Cowie[1,4], Bernardino Romera-Paredes[1,4], Stanislav Nikolov[1,4], Rishub Jain[1,4], Jonas Adler[1], Trevor Back[1], Stig Petersen[1], David Reiman[1], Ellen Clancy[1], Michal Zielinski[1], Martin Steinegger[2,3], Michalina Pacholska[1], Tamas Berghammer[1], Sebastian Bodenstein[1], David Silver[1], Oriol Vinyals[1], Andrew W. Senior[1], Koray Kavukcuoglu[1], Pushmeet Kohli[1] & Demis Hassabis[1,4]✉

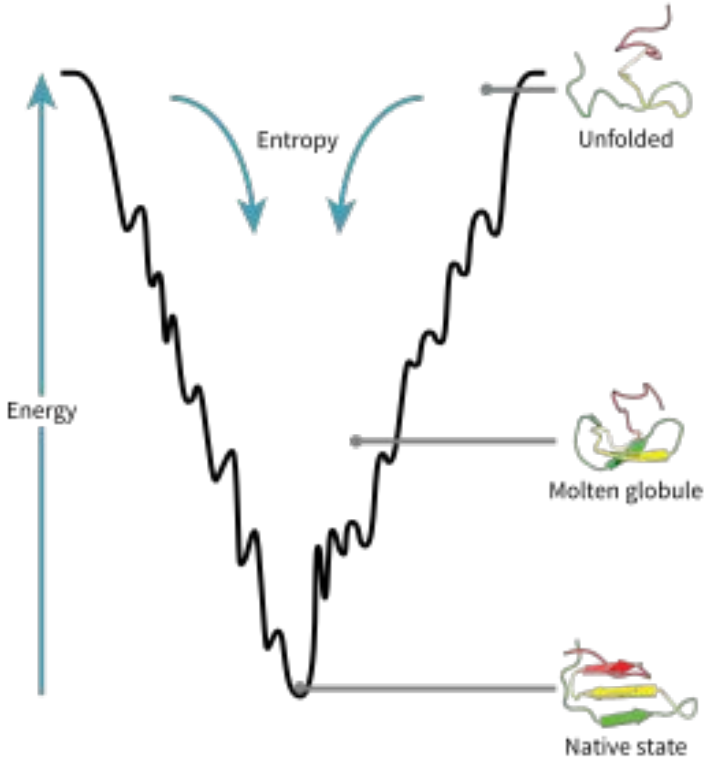
Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort[1–4], the structures of around 100,000 unique proteins have been determined[5], but this represents a small fraction of the billions of known protein sequences[6,7]. Structural coverage is bottlenecked by the months to years of painstaking effort required to determine a single protein structure. Accurate computational approaches are needed to address this gap and to enable large-scale structural bioinformatics. Predicting the three-dimensional structure that a protein will adopt based solely on its amino acid sequence—the structure prediction component of the 'protein folding problem'[8]—has been an important open research problem for more than 50 years[9]. Despite recent progress[10–14], existing methods fall far short of atomic accuracy, especially when no homologous structure is available. Here we provide the first computational method that can regularly predict protein structures with atomic accuracy even in cases in which no similar structure is known. We validated an entirely redesigned version of our neural network-based model, AlphaFold, in the challenging 14th Critical Assessment of protein Structure Prediction (CASP14)[15], demonstrating accuracy competitive with experimental structures in a majority of cases and greatly outperforming other methods. Underpinning the latest version of AlphaFold is a novel machine learning approach that incorporates physical and biological knowledge about protein structure, leveraging multi-sequence alignments, into the design of the deep learning algorithm.

# Protein folding problem

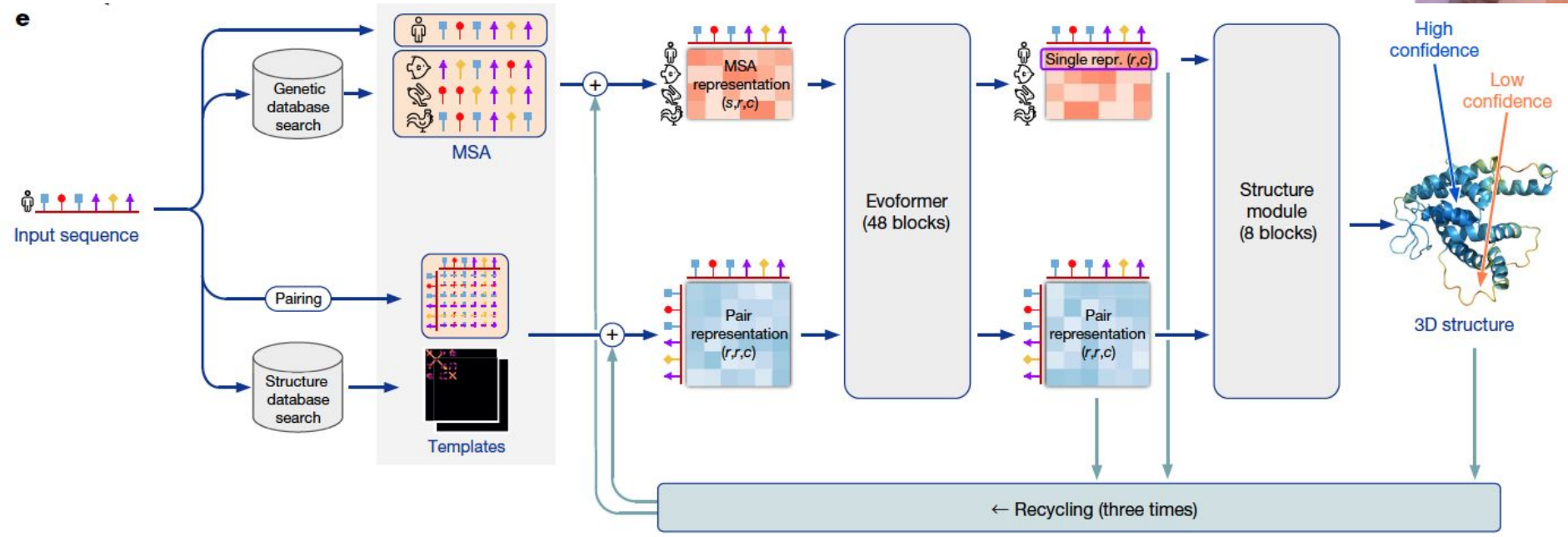
Protein’s amino acid sequence -> three-dimensional atomic structure prediction.

The notion of a folding “problem” first emerged around 1960, with the appearance of the first atomic-resolution protein structures



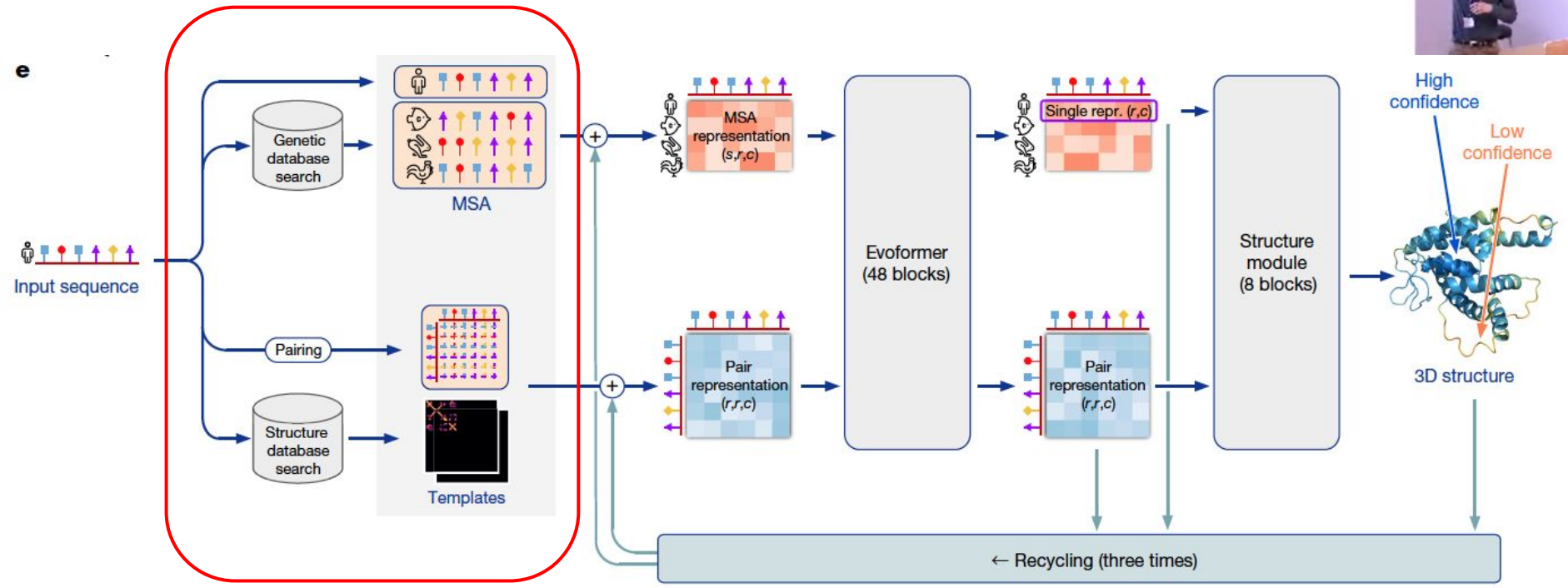



# Network Alphafold2



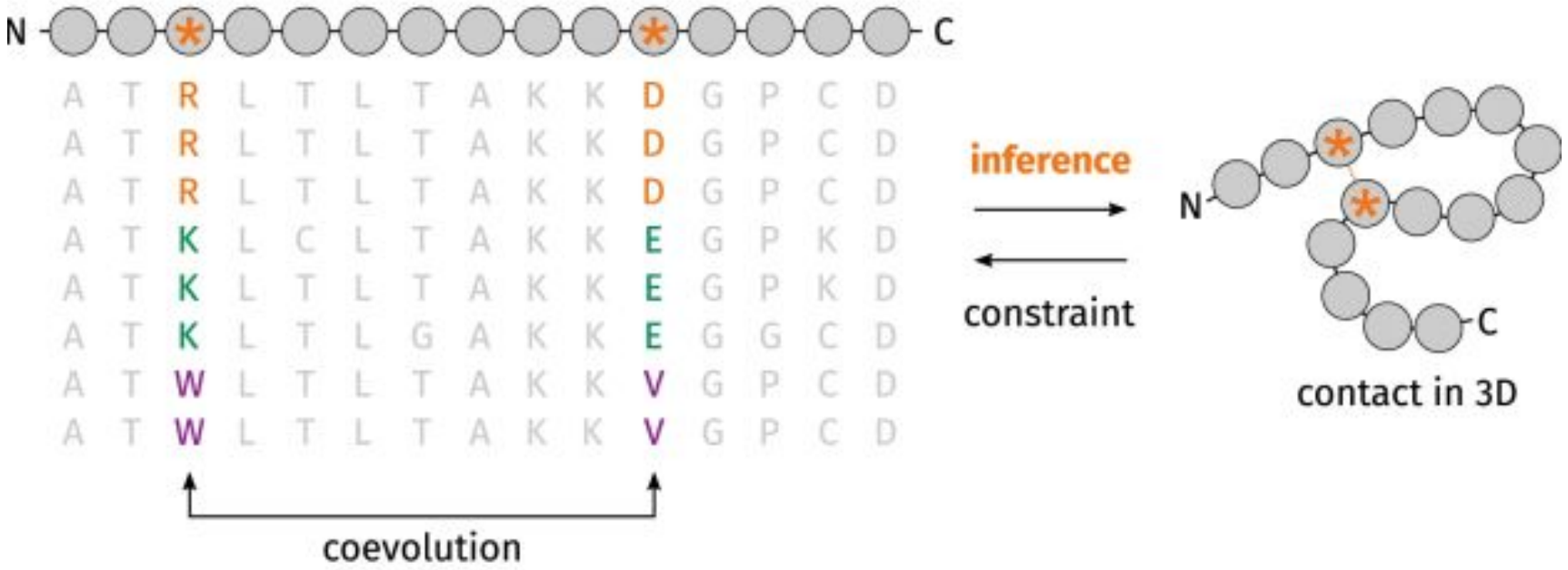

The Structure Module predicts a rotation + translation to place each residue.
A small network predicts side chain chi angles. The final structure is run through a relaxation process.

# Network Alphafold2

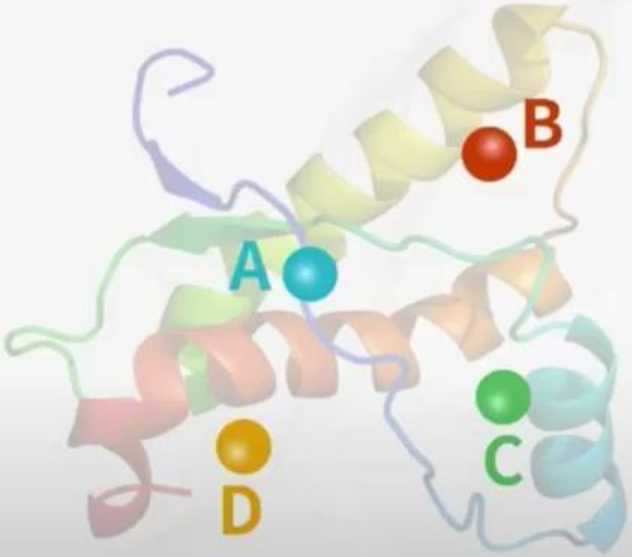



The Structure Module predicts a rotation + translation to place each residue.
A small network predicts side chain chi angles. The final structure is run through a relaxation process.

# Multiple Sequence Alignment (MSA)

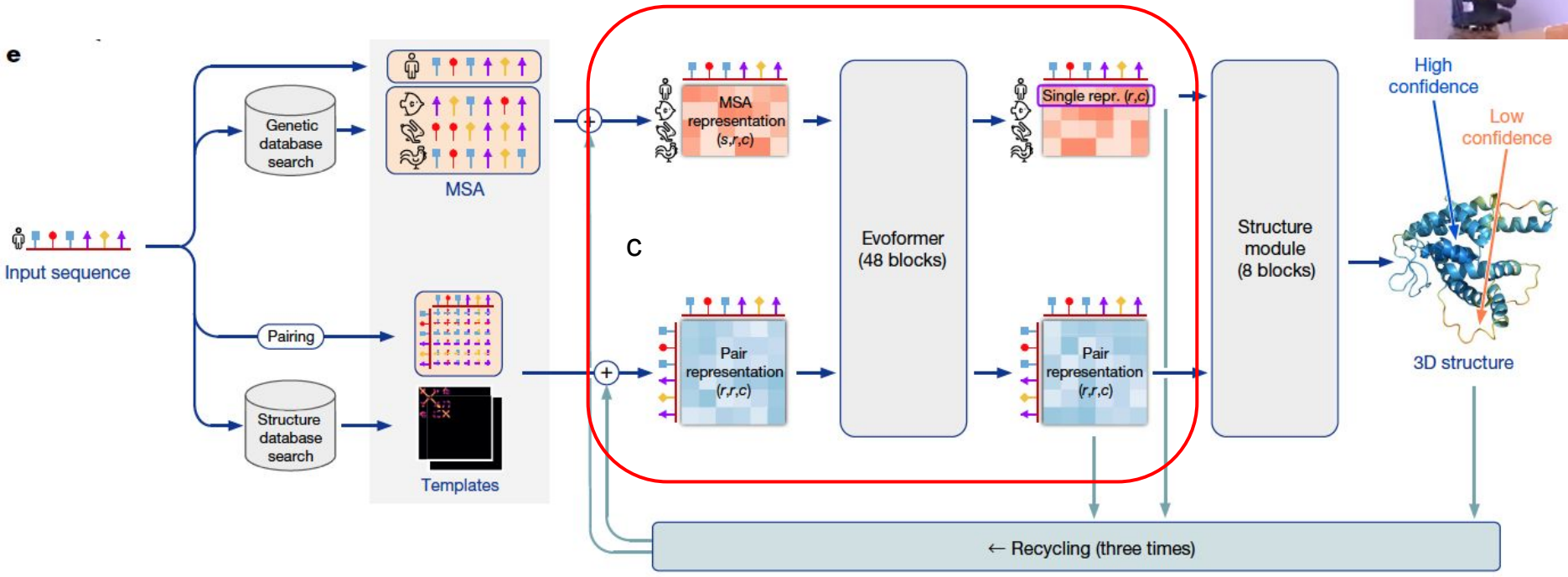


# Distance Matrix



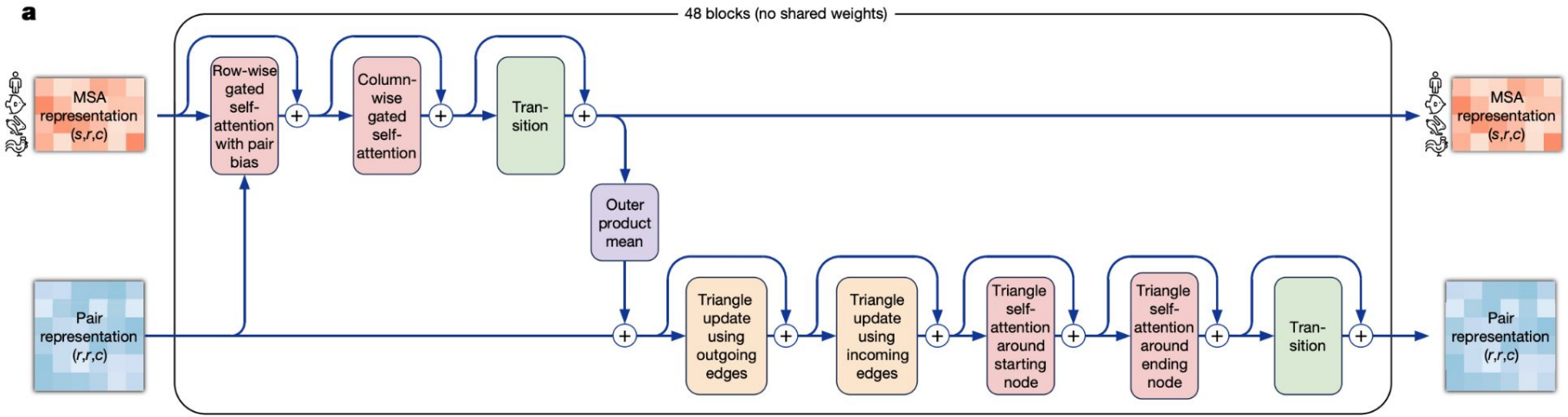

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | | | |
| B | | 0 | | |
| C | | | 0 | |
| D | | | | 0 |

# Network Alphafold2



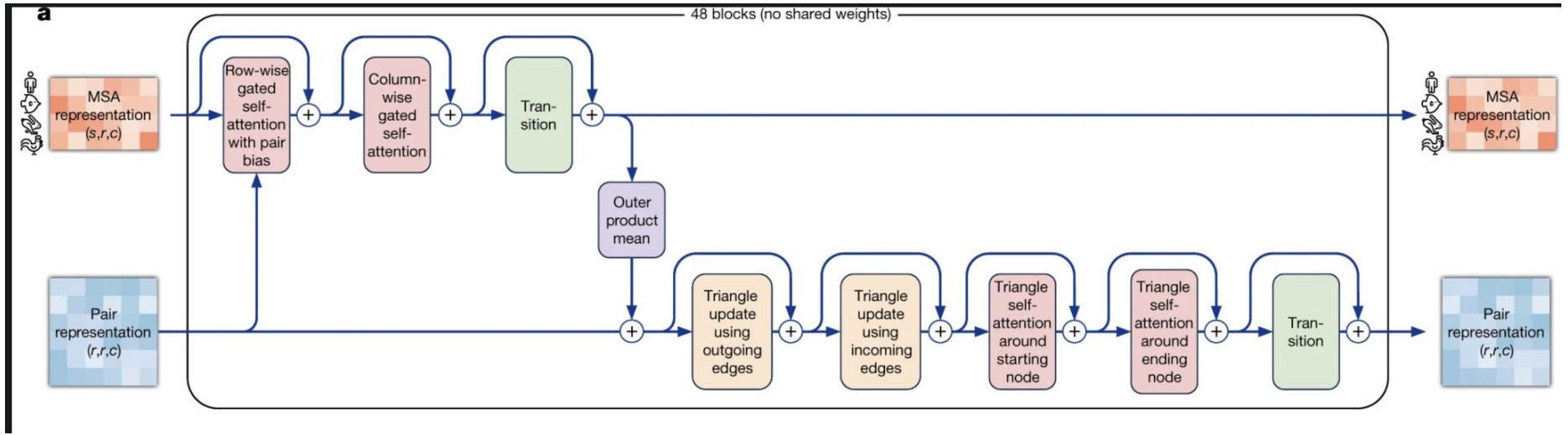

The Structure Module predicts a rotation + translation to place each residue.
A small network predicts side chain chi angles. The final structure is run through a relaxation process.

# Evoformer

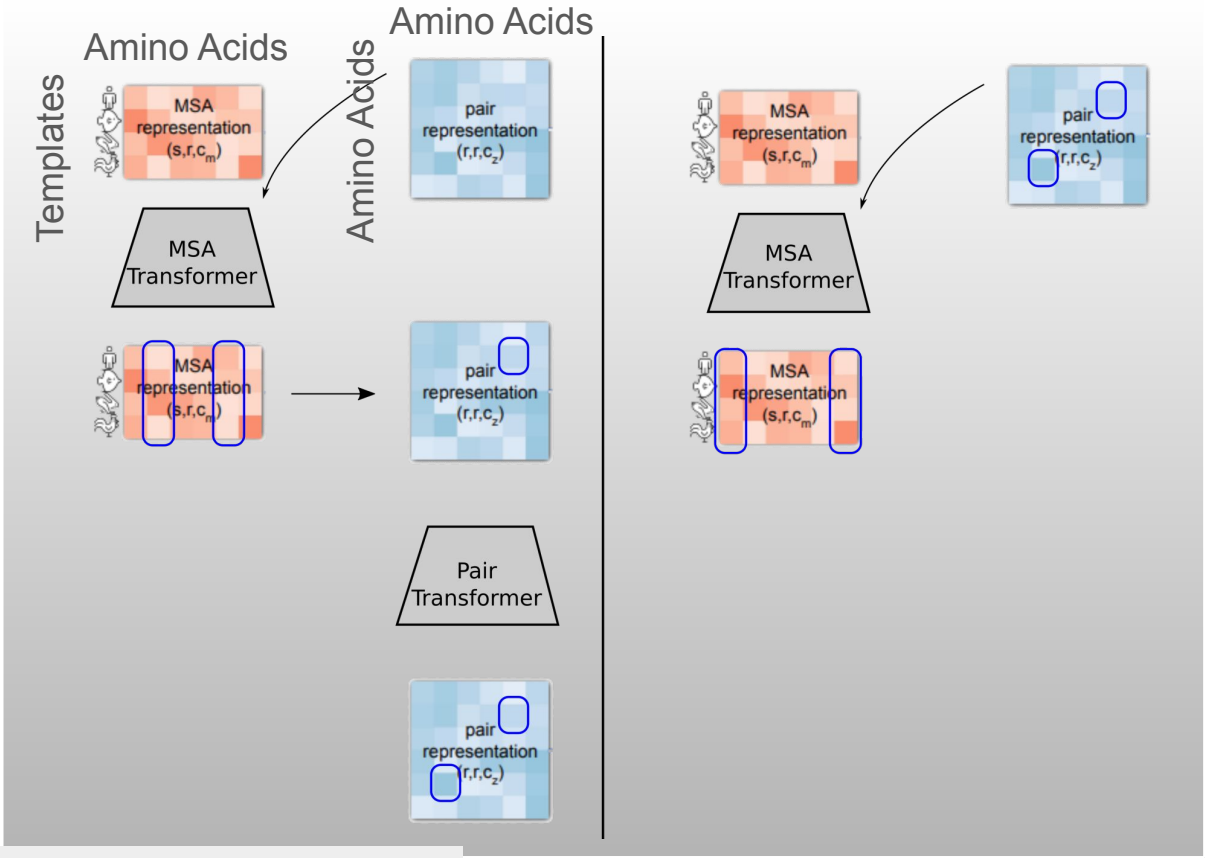

a
48 blocks (no shared weights)
MSA representation (s,r,c)
Row-wise gated self-attention with pair bias
Column-wise gated self-attention
Tran-sition
Outer product mean
Pair representation (r,r,c)
Triangle update using outgoing edges
Triangle update using incoming edges
Triangle self-attention around starting node
Triangle self-attention around ending node
Tran-sition
MSA representation (s,r,c)
Pair representation (r,r,c)

# What is going on in the swap?

1D of many proteins and 2D of one protein -> Update both

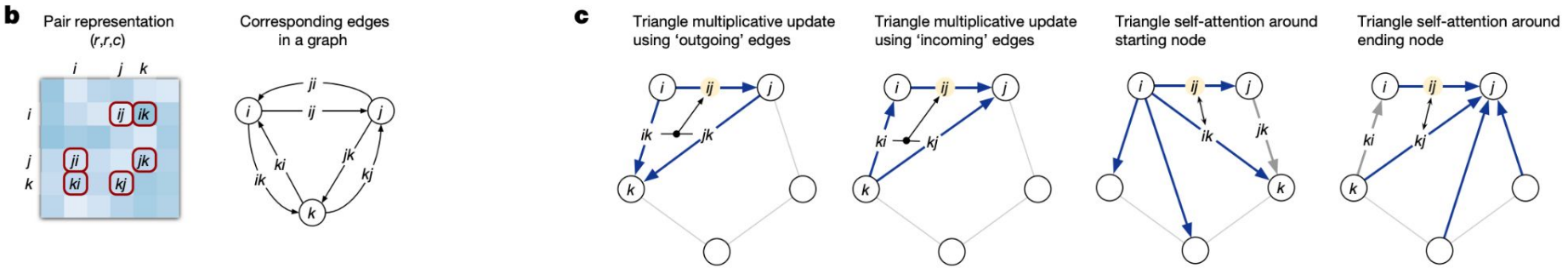


# Representations: Swap between 1D and 2D



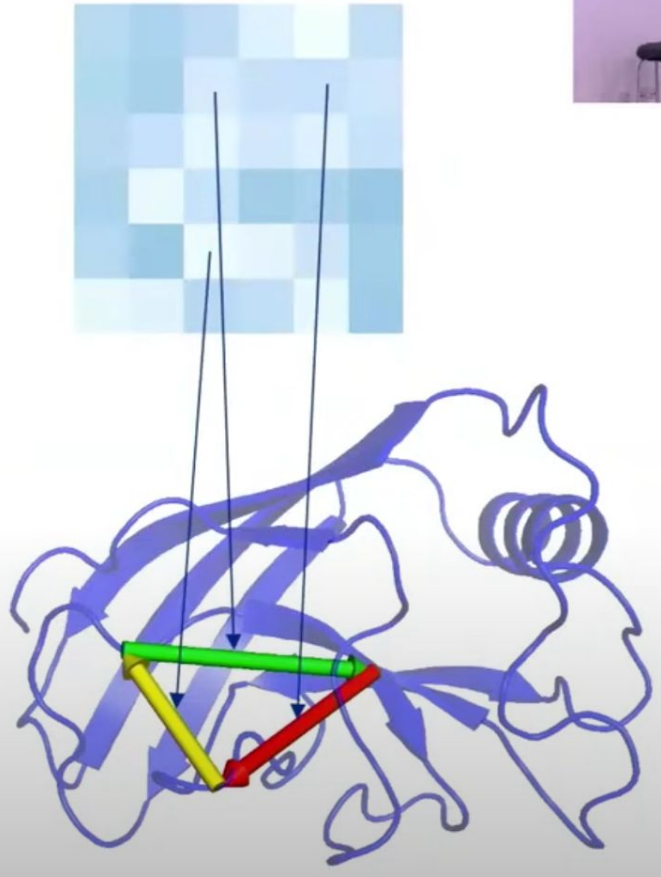

AA: Amino Acids in 1 Protein

# Triangular Attention



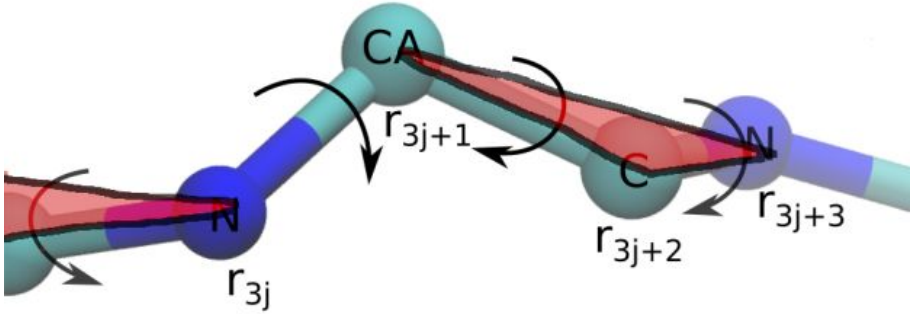
# Vignette 1: Triangular Attention

- **Take 3 points A, B, C**
  - If Distance AB and distance BC known strong constraint on AC (triangle inequality)
  - Evolution & Sequence gives information about relations between residues
- **Pair Embedding encodes relations**
  - Update for pair AC should depend on BC, AB
  - All about who communicates in the network, not what is computed



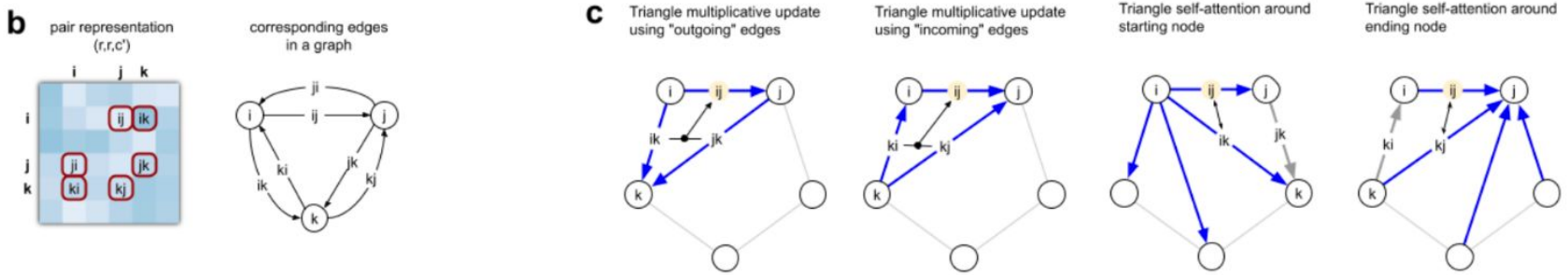
MPLT

# What is going on in the swap?

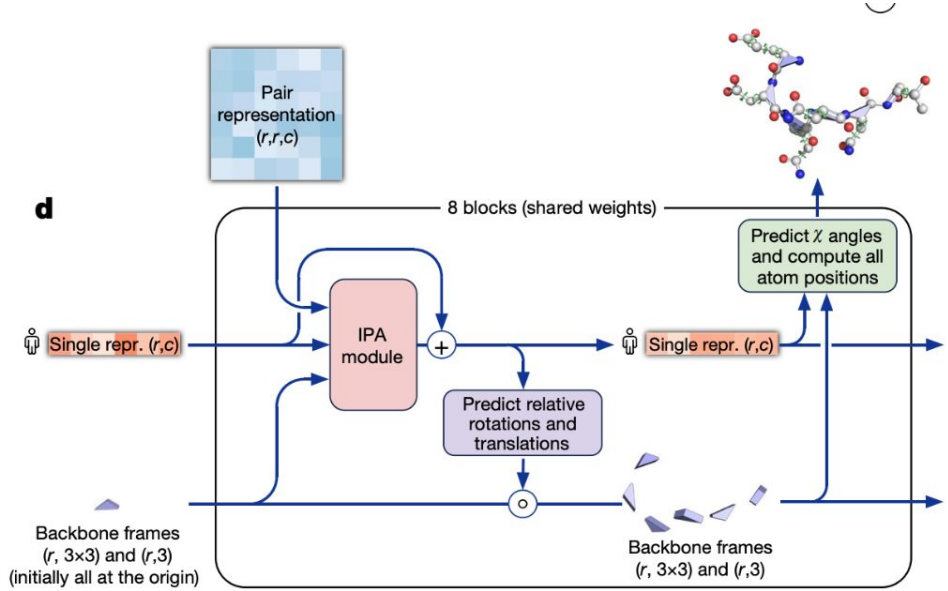


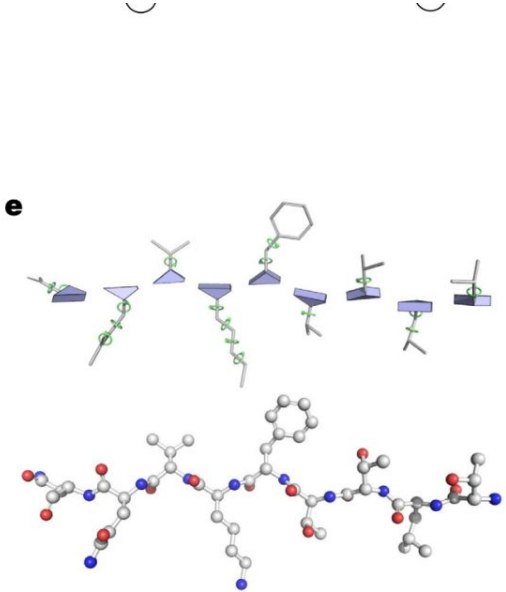

Concepts: Affine Transforms, Triangle Inequality

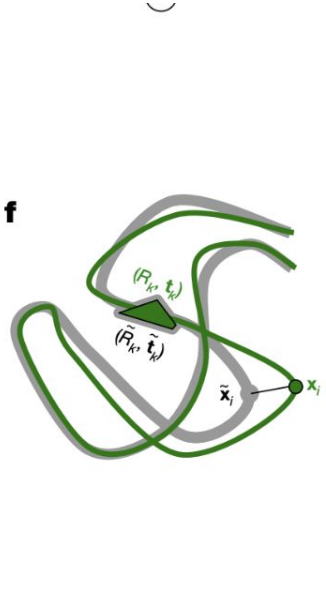
We preserve triangle inequality but *don't* care about overlaps till the very end.



# Structure Prediction



# Vignette 2: Structure module

- **End-to-end folding** instead of gradient descent
- Protein backbone = gas of 3-D rigid bodies (chain is learned!)

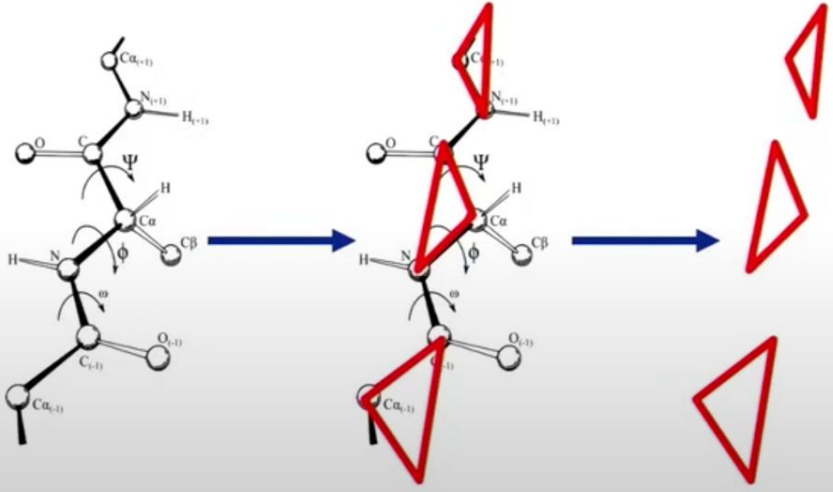
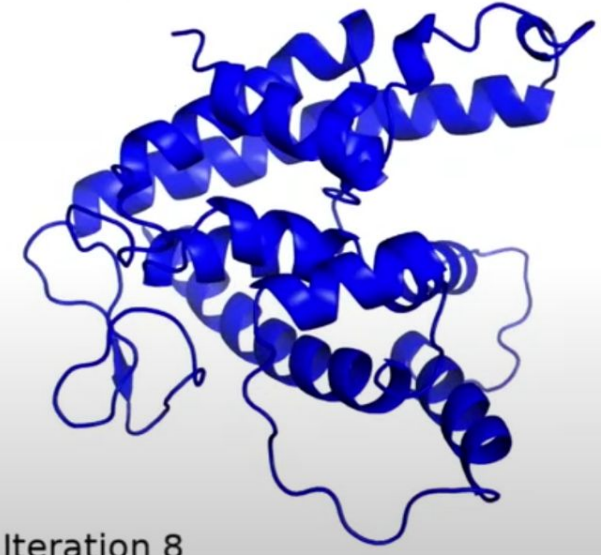


*Image: Dcrjsr, vectorised Adam Rędzikowski (CC BY 3.0, Wikipedia)*

- **3-D equivariant transformer architecture** updates the rigid bodies / backbone
  - Also builds the side chains from torsion angles

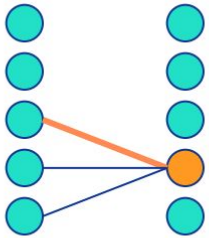


Iteration 8

Target: T1041

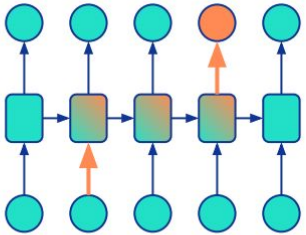
# Inductive Bias for Deep Learning Models





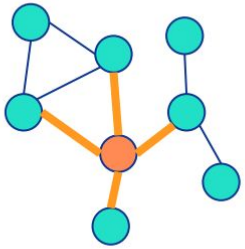
**Convolutional Networks (e.g. computer vision)**

- data in regular grid
- information flow to local neighbours



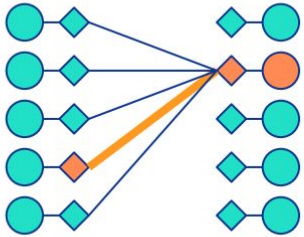
**Recurrent Networks (e.g. language)**

- data in ordered sequence
- information flow sequentially



**Graph Networks (e.g. recommender systems or molecules)**

- data in fixed graph structure
- information flow along fixed edges



**Attention Module (e.g. language)**

- data in unordered set
- information flow dynamically controlled by the network (via keys and queries)

Dzmitry Bahdanau Jacobs, **KyungHyun Cho**, Yoshua Bengio (ICLR 2015)

# Training loss functions

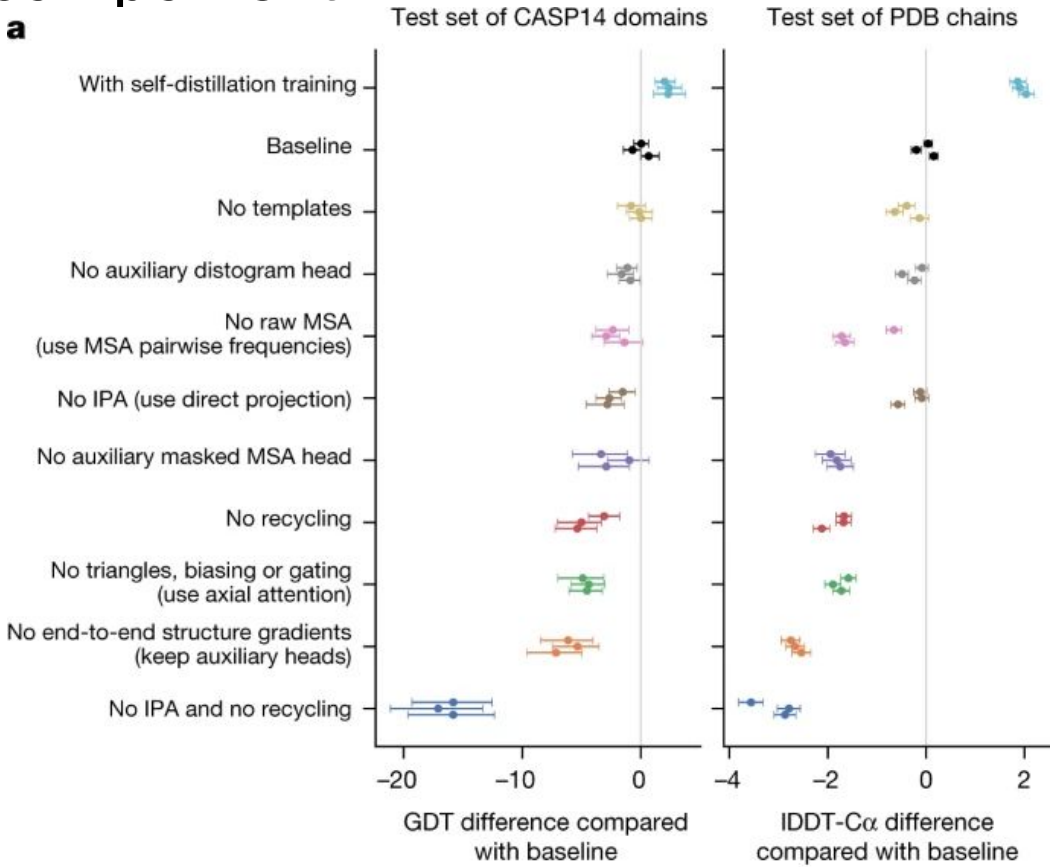
$$\mathcal{L} = \begin{cases} 0.5\mathcal{L}_{\text{FAPE}} + 0.5\mathcal{L}_{\text{aux}} + 0.3\mathcal{L}_{\text{dist}} + 2.0\mathcal{L}_{\text{msa}} + 0.01\mathcal{L}_{\text{conf}} & \text{training} \\ 0.5\mathcal{L}_{\text{FAPE}} + 0.5\mathcal{L}_{\text{aux}} + 0.3\mathcal{L}_{\text{dist}} + 2.0\mathcal{L}_{\text{msa}} + 0.01\mathcal{L}_{\text{conf}} + 0.01\mathcal{L}_{\text{exp resolved}} + 1.0\mathcal{L}_{\text{viol}} & \text{fine-tuning} \end{cases}$$

Training losses are designed to emphasize contributions from different architectural parts:

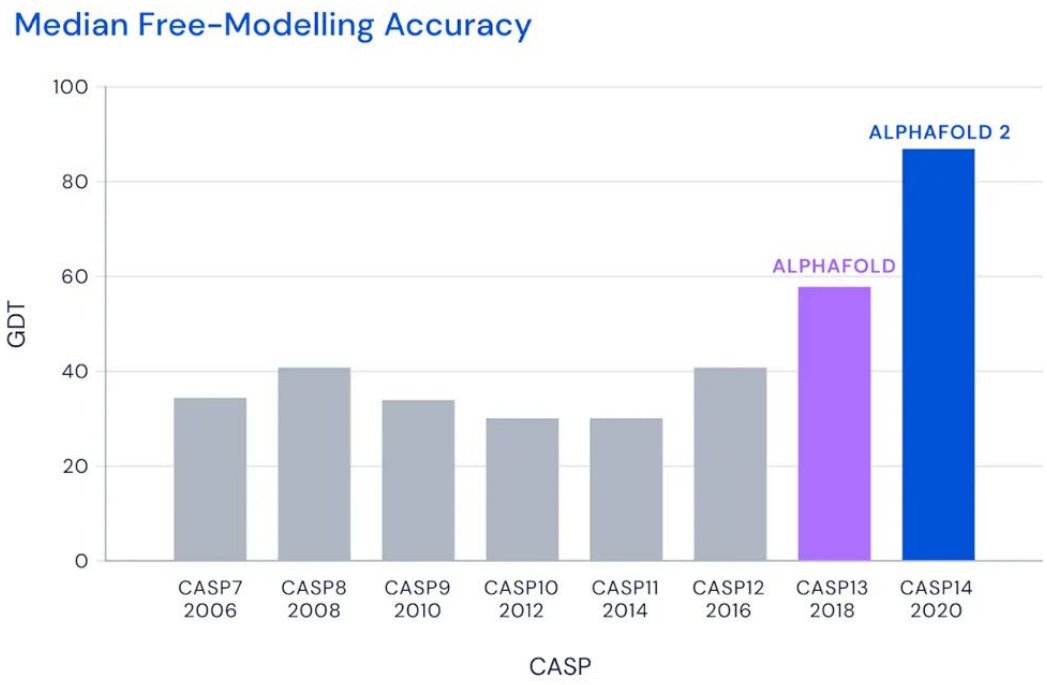
- $L_{aux}$: averaged FAPE and torsion angle losses from intermediate structure module layers
- $L_{dist}$: distogram prediction (cross-entropy)
- $L_{msa}$: masked-MSA prediction (cross-entropy)

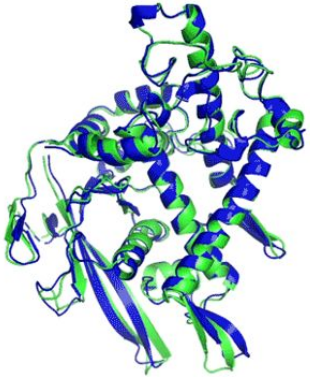
Credit: Amy Lu

# Ablation study of each compoment

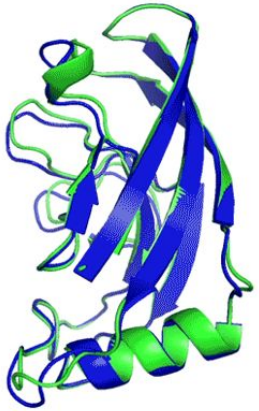


# AlphaFold 2 in CASP challenge

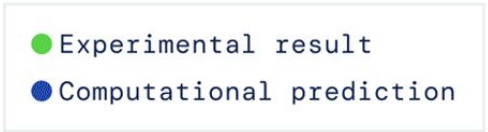


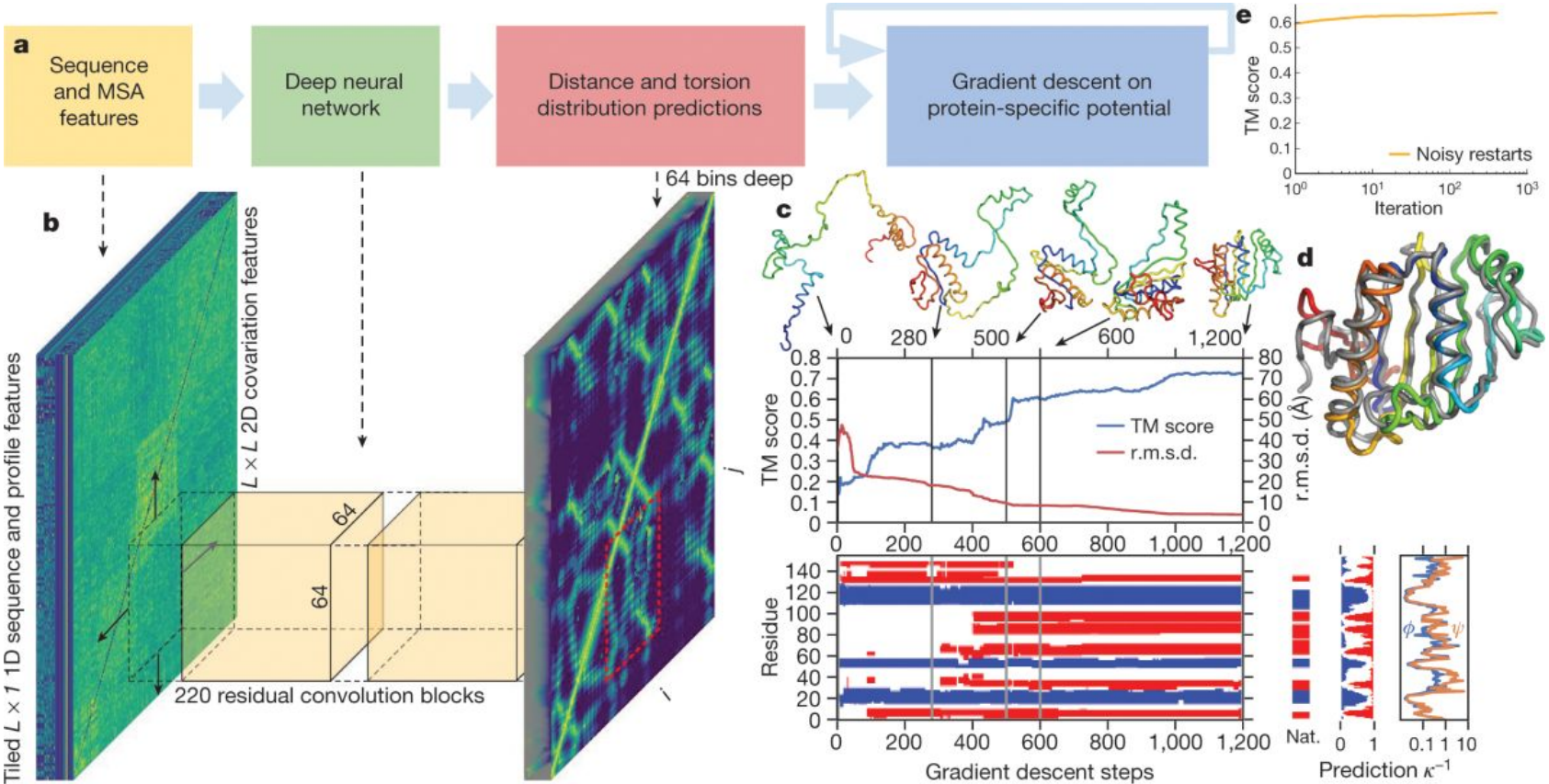
**T1037 / 6vr4**
90.7 GDT
(RNA polymerase domain)



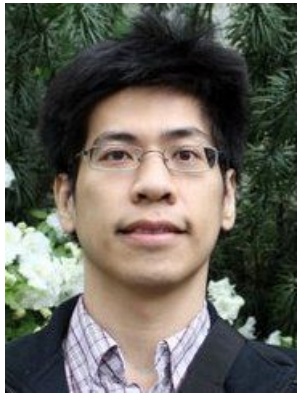
**T1049 / 6y4f**
93.3 GDT
(adhesin tip)



# Comparison to Alphafold 1



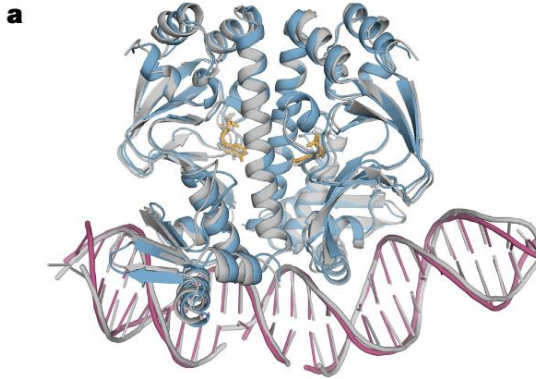
# Alphafold 2 vs Alphafold 3 (improvement)

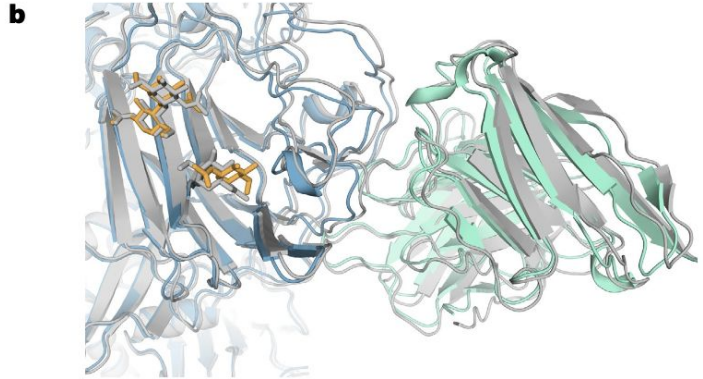


Chia-Chun (Alden) Hung
Isomorphic Lab

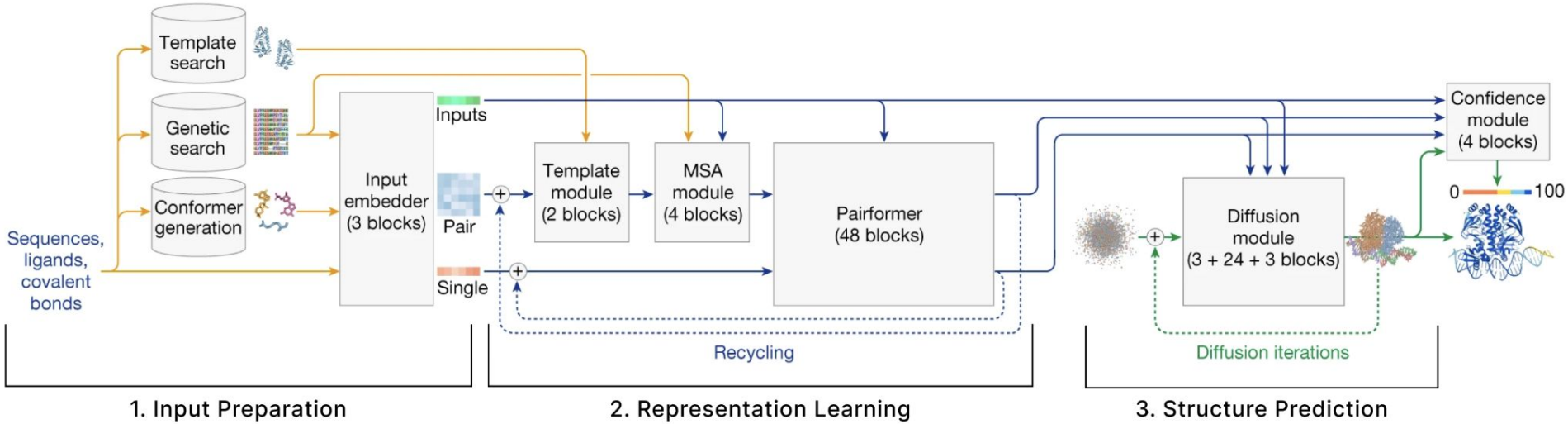
AF3 has these improvement compare with AF2

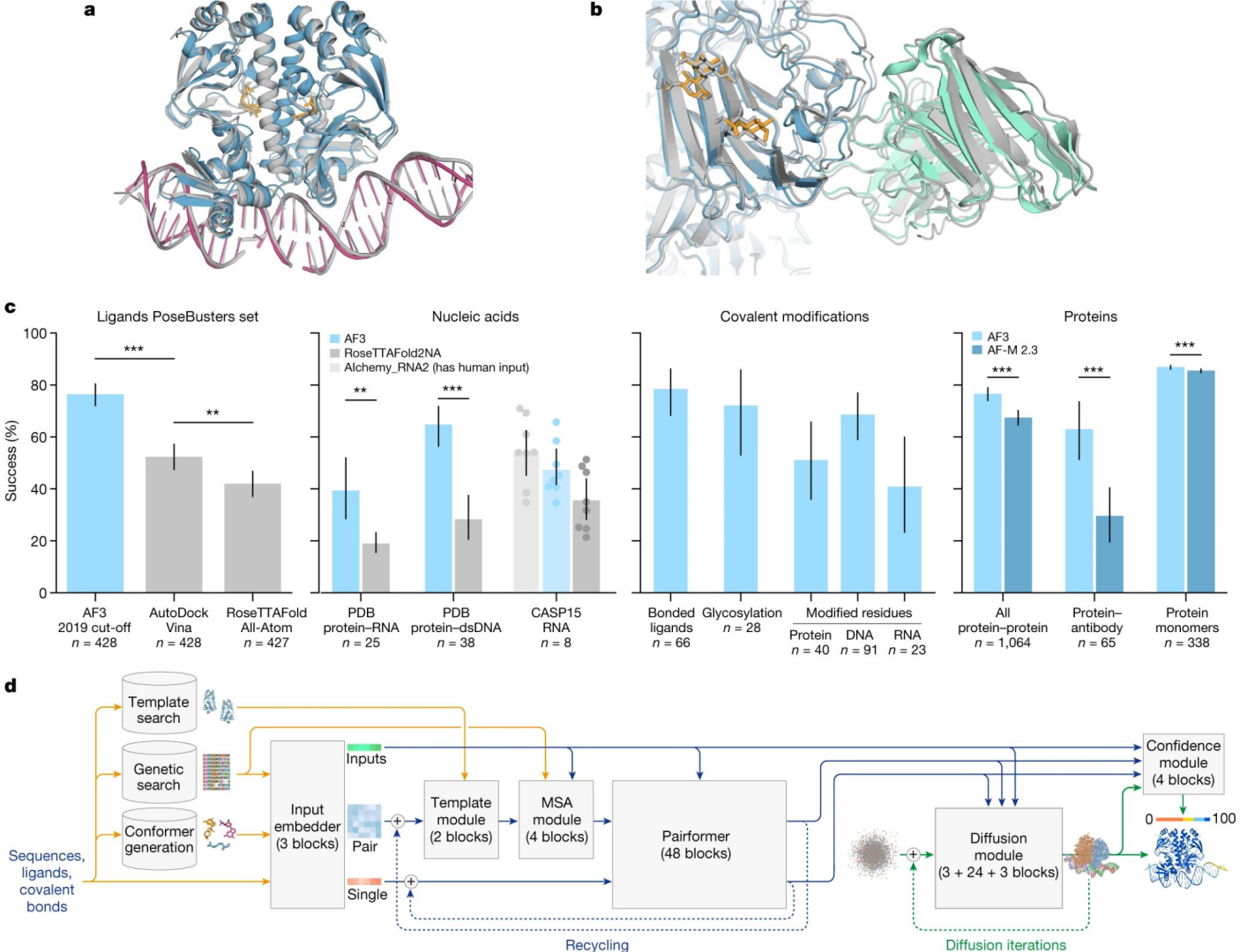
- multiple proteins
- DNA
- RNA
- small molecule ligands interaction with proteins
- Improved Antibody structures prediction
- Pseudo-opensource model
- Remove equivariant models, infuse diffusion model in protein structure prediction



# Alphafold 3 archecture



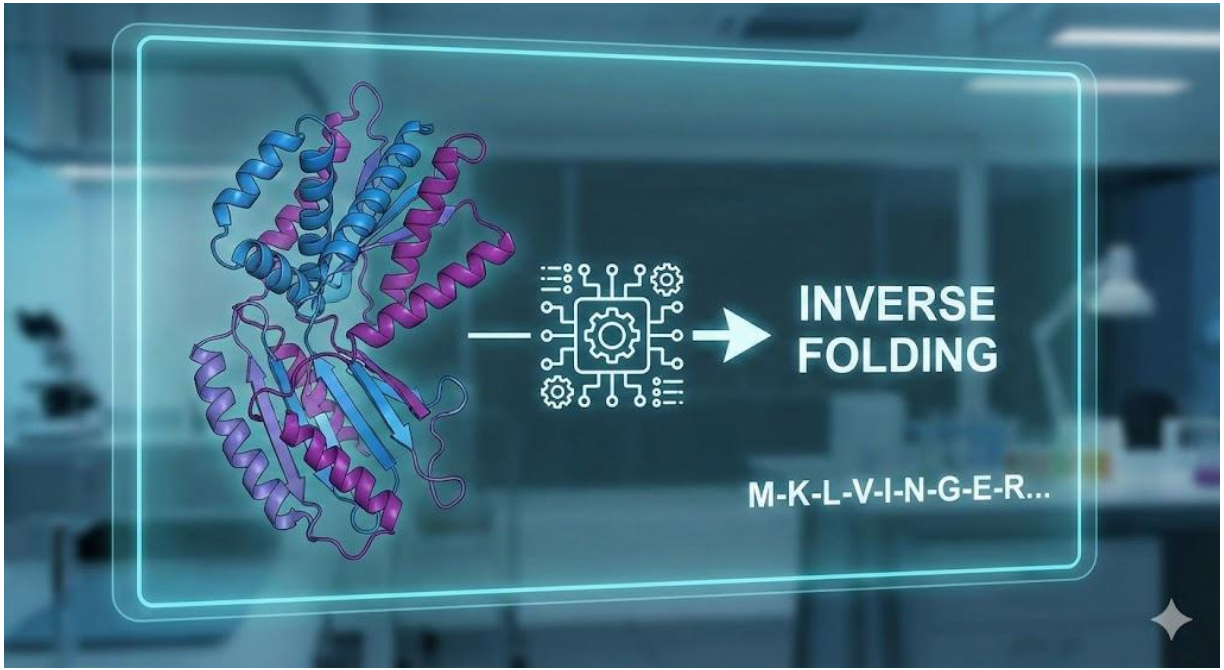

a
b
c
Ligands PoseBusters set
Nucleic acids
Covalent modifications
Proteins
Success (%)
AF3
RoseTTAFold2NA
Alchemy_RNA2 (has human input)
AF3
AF-M 2.3
AF3 2019 cut-off n = 428
AutoDock Vina n = 428
RoseTTAFold All-Atom n = 427
PDB protein–RNA n = 25
PDB protein–dsDNA n = 38
CASP15 RNA n = 8
Bonded ligands 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
d
Template search
Genetic search
Conformer generation
Sequences, ligands, covalent bonds
Input embedder (3 blocks)
Inputs
Pair
Single
Template module (2 blocks)
MSA module (4 blocks)
Pairformer (48 blocks)
Recycling
Diffusion module (3 + 24 + 3 blocks)
Diffusion iterations
Confidence module (4 blocks)
0
100

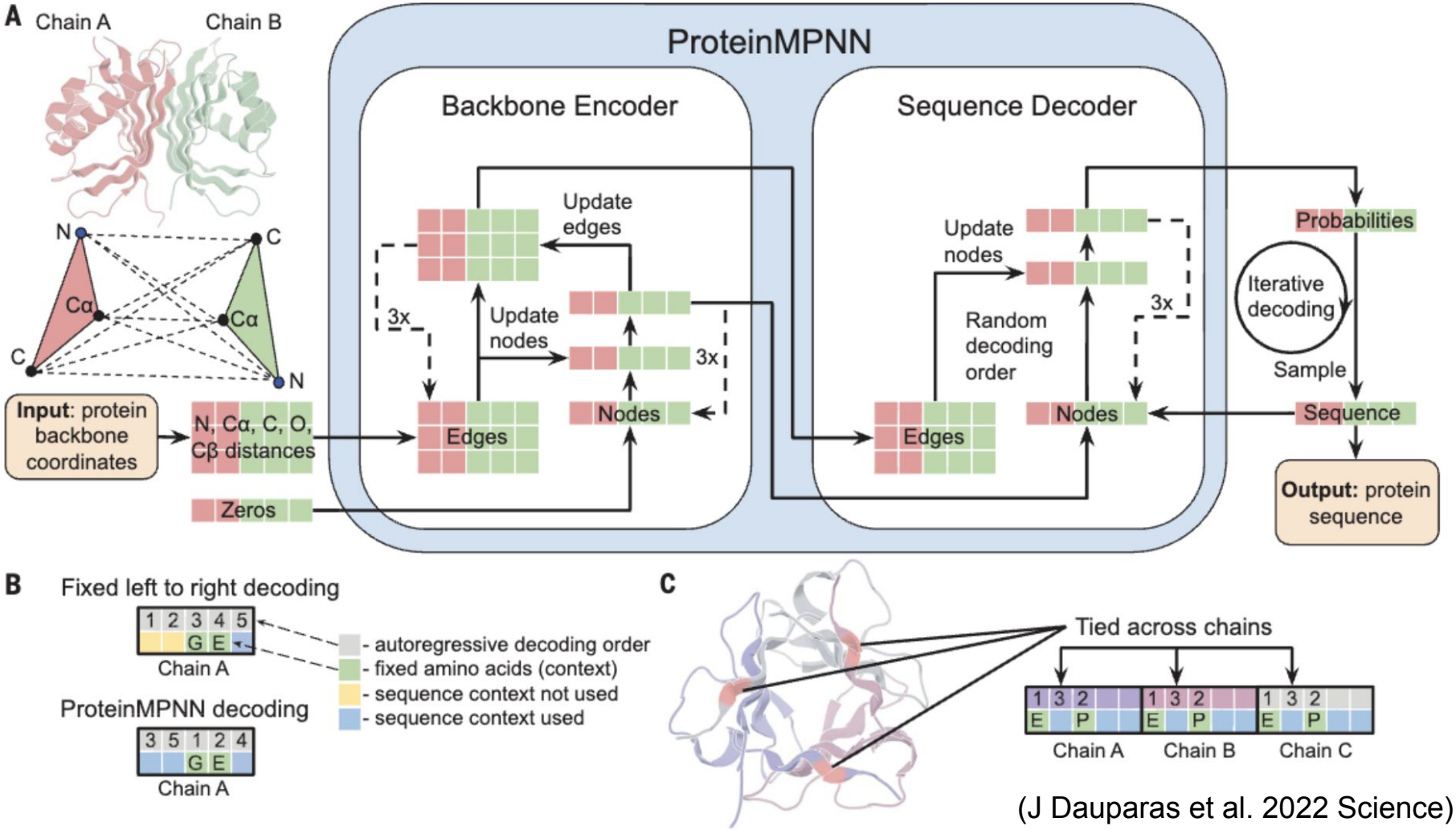

THE NOBEL PRIZE
IN CHEMISTRY 2024
Illustrations: Niklas Elmehed
David Baker
"for computational protein design"
Demis Hassabis
John M. Jumper
"for protein structure prediction"
THE ROYAL SWEDISH ACADEMY OF SCIENCES

# Live Demo

- PDB
- uniprot
- ColabFold
- ProteinMPNN
- RFdiffusion

# What about the inverse folding?






(J Dauparas et al. 2022 Science)

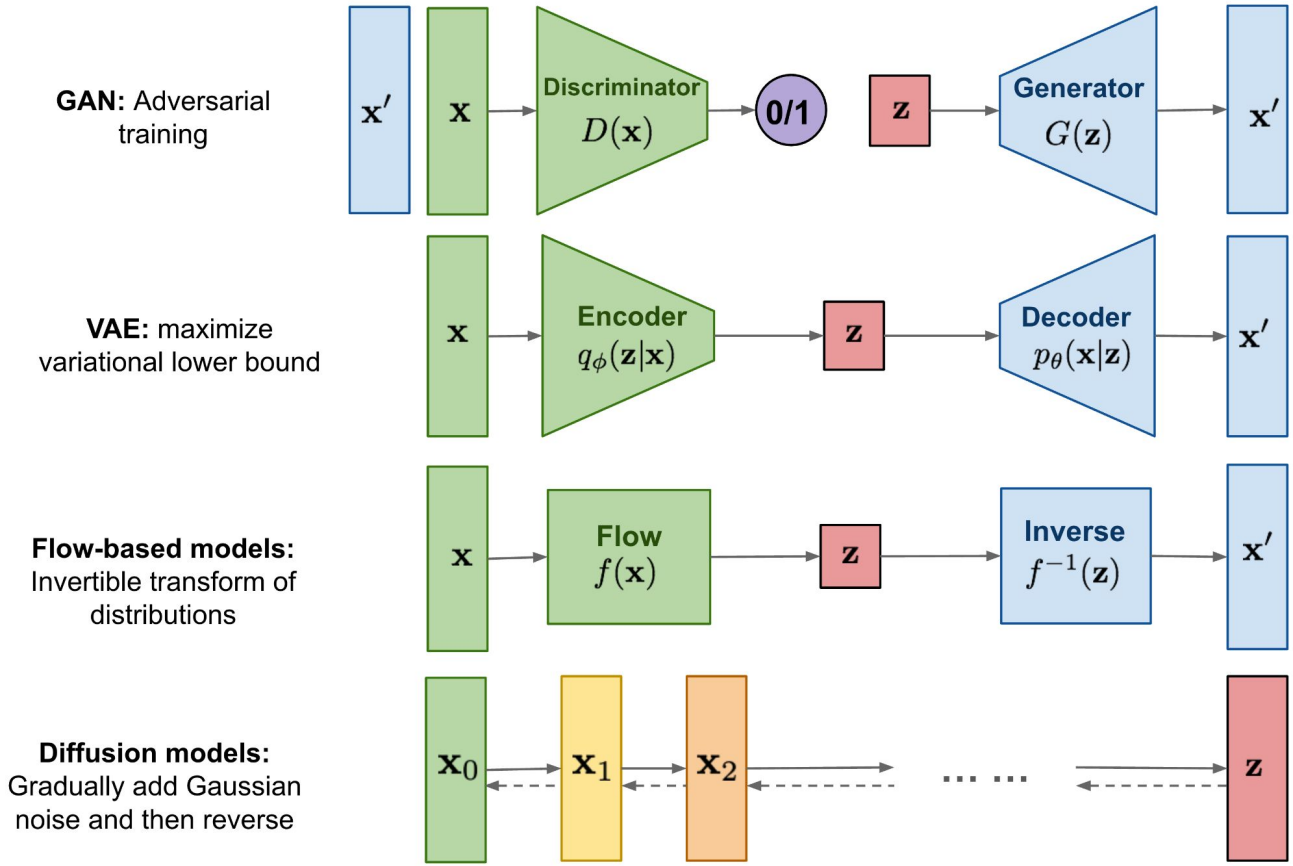
# Live Demo

- PDB
- uniprot
- ColabFold
- ProteinMPNN
- RFdiffusion

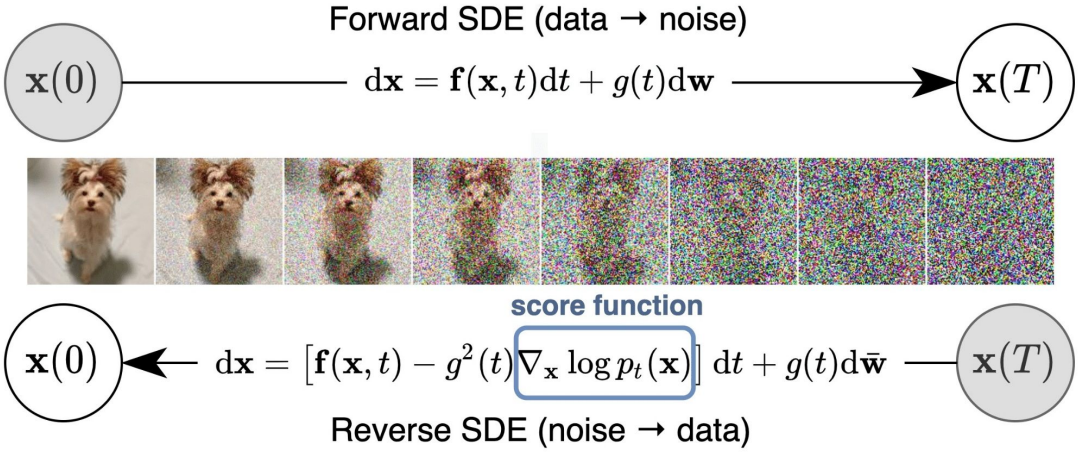
# Why do we need ML for protein design

- Even if we could actually predicture the structure of protein, the possible amino acid sequence is more than 20^300.
- **Generative model** for protein is needed

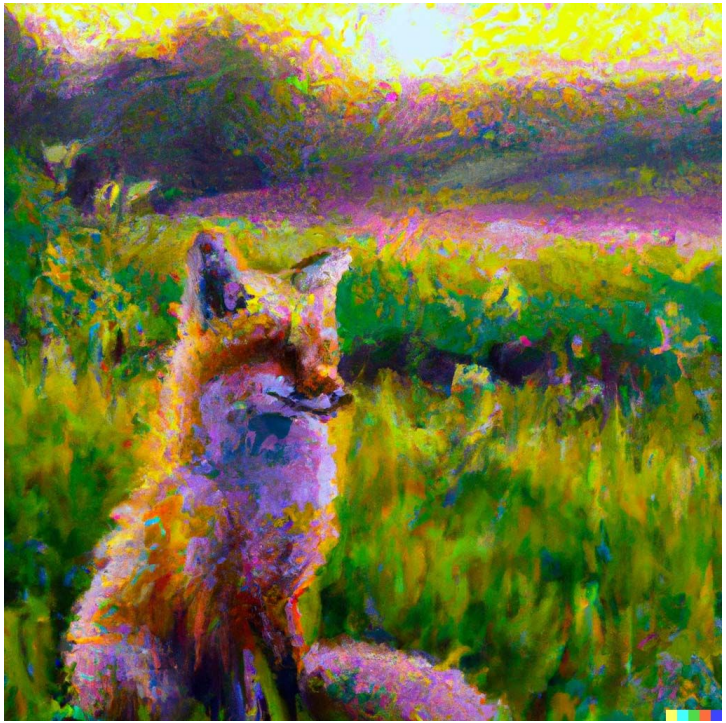
# Family of generative models



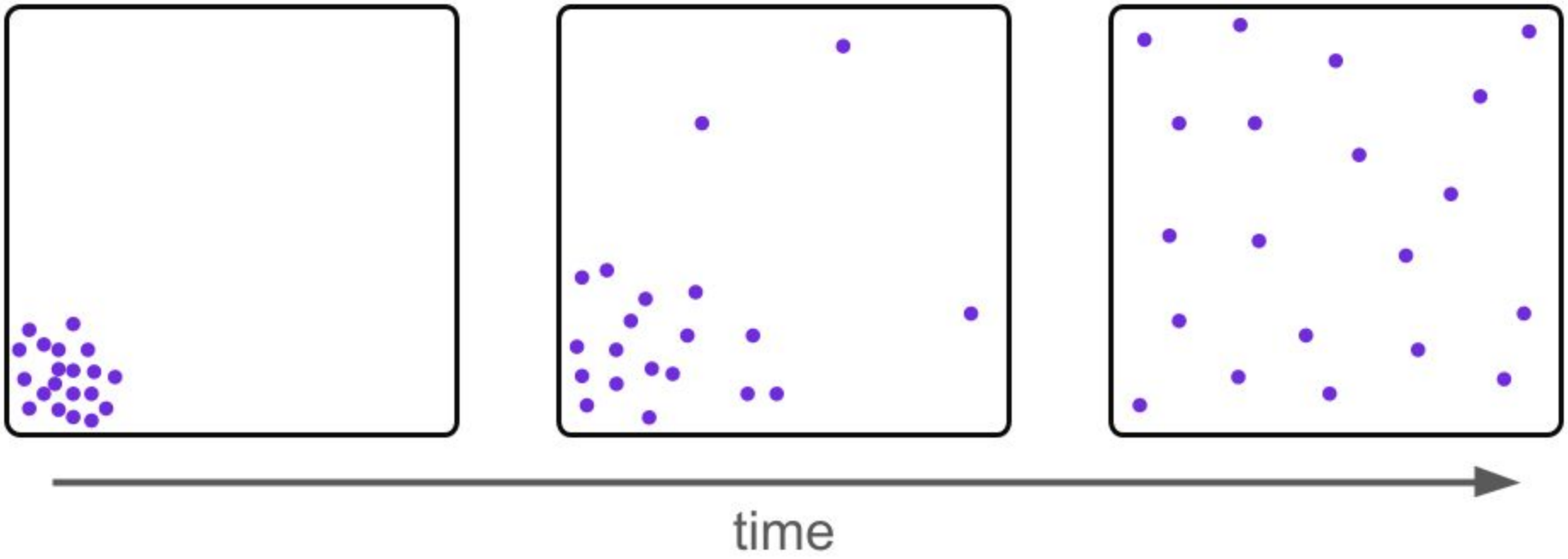
# Diffusion Model (& Score matching)

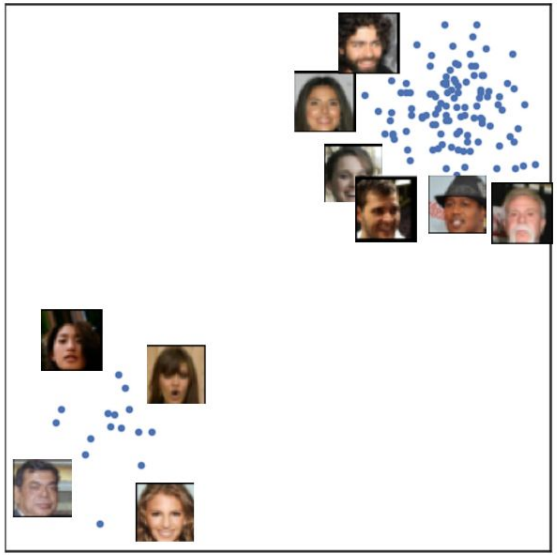



Diffusion Model & Score matching



“A painting of a fox sitting in a field at sunrise in the style of Claude Monet”

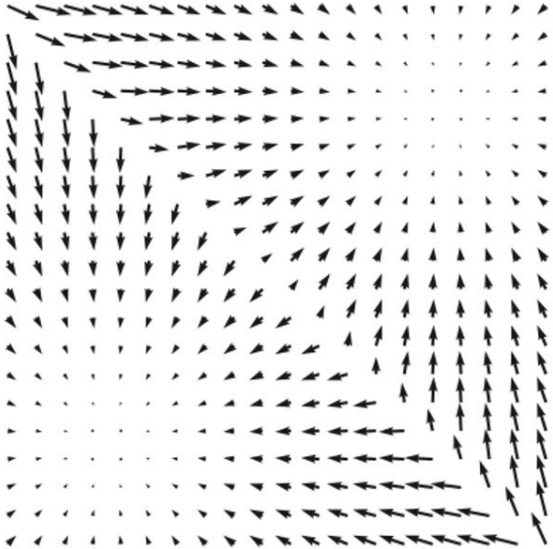

time



Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p(\mathbf{x})$$

score matching



Scores

$$\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

Langevin dynamics

New samples

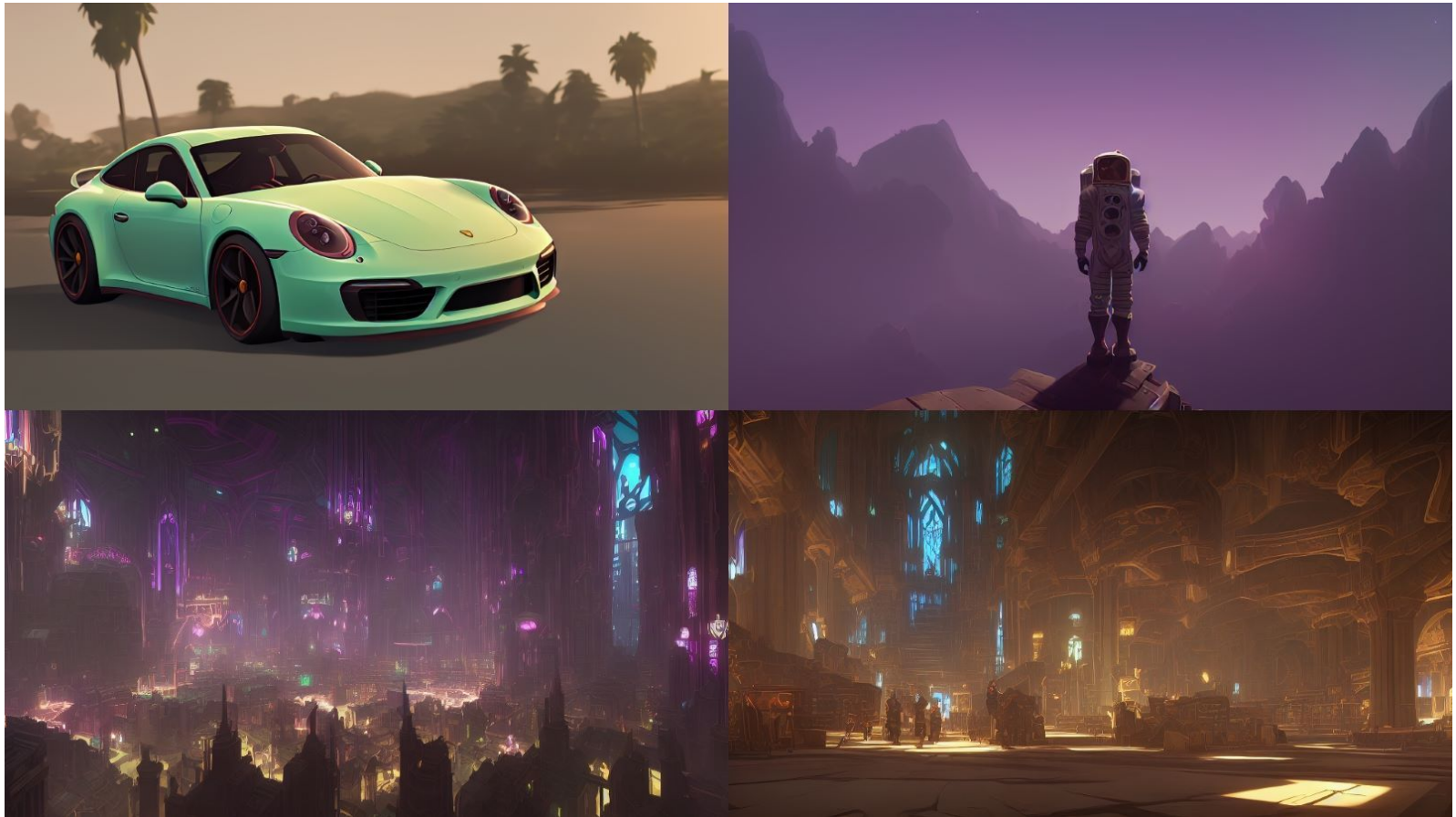
# Example from Imagen

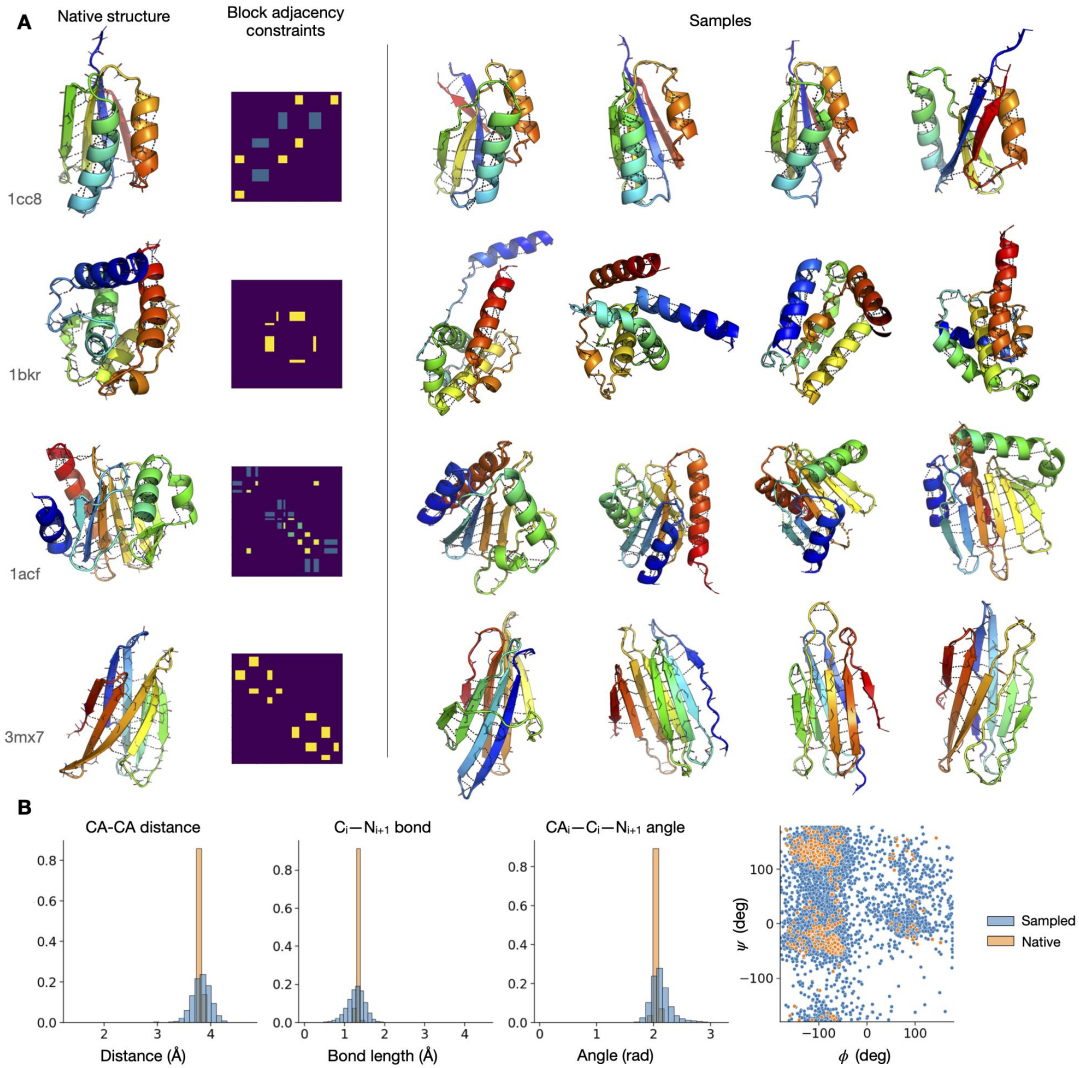

*“A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.“*


*“A blue jay standing on a large basket of rainbow macarons“*


*“A brain riding a rocketship heading towards the moon“*

# Example of Stable diffusion





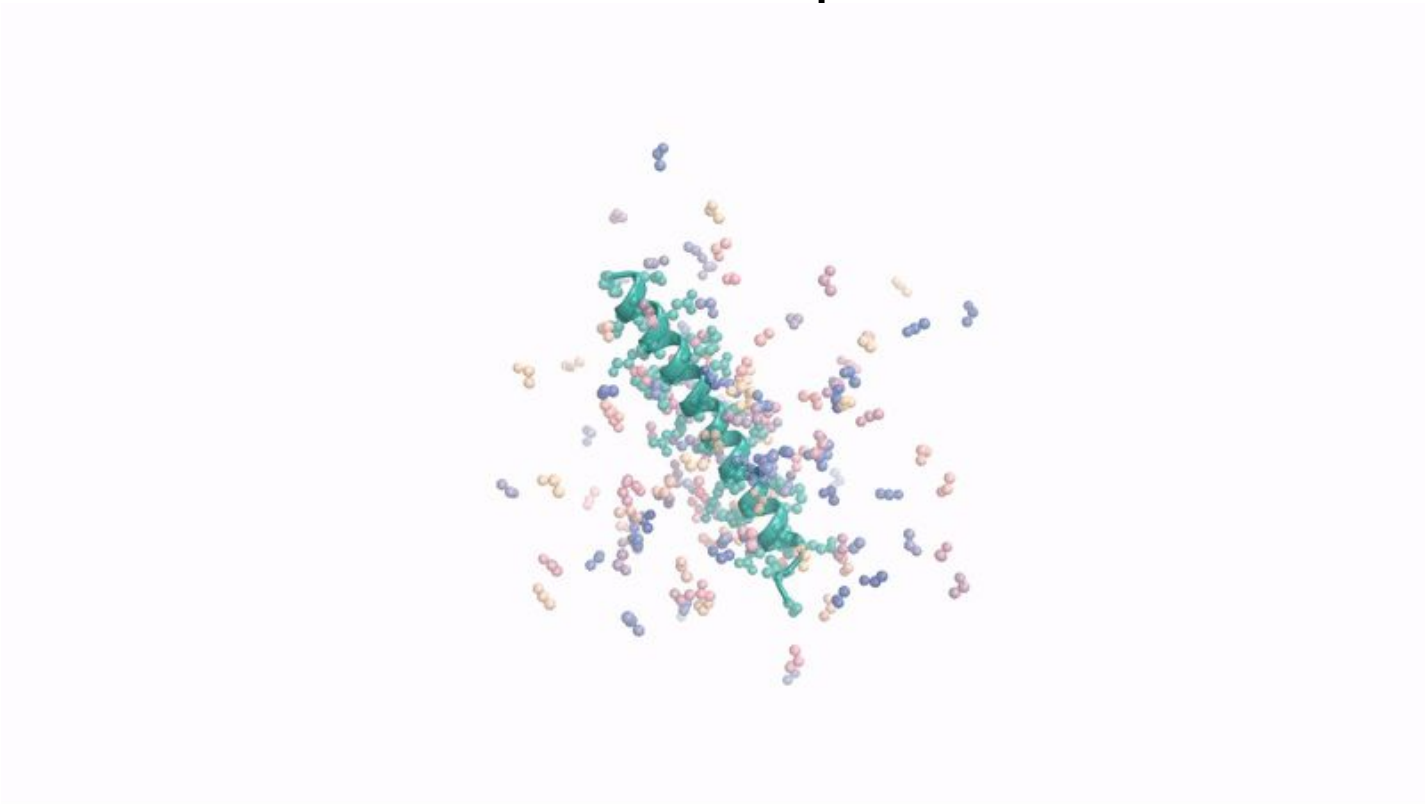

Namrata Anand

Tudor Achim

(arXiv: 2205.15019)

Diffuse BIO

# RFdiffusion: diffusion model for protein



Watson et al (bioarXiv 2023)
INSTITUTE FOR PROTEIN DESIGN, UW

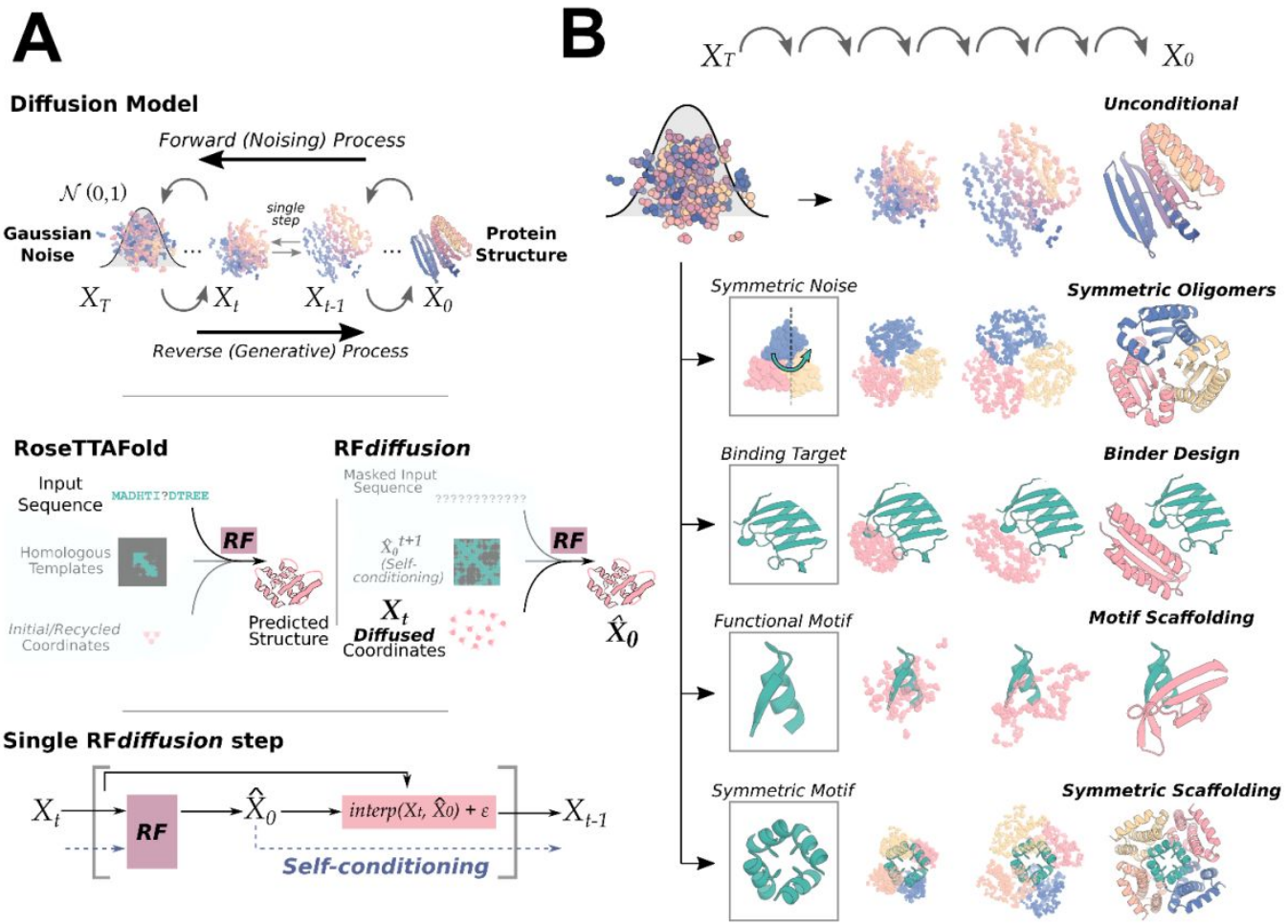
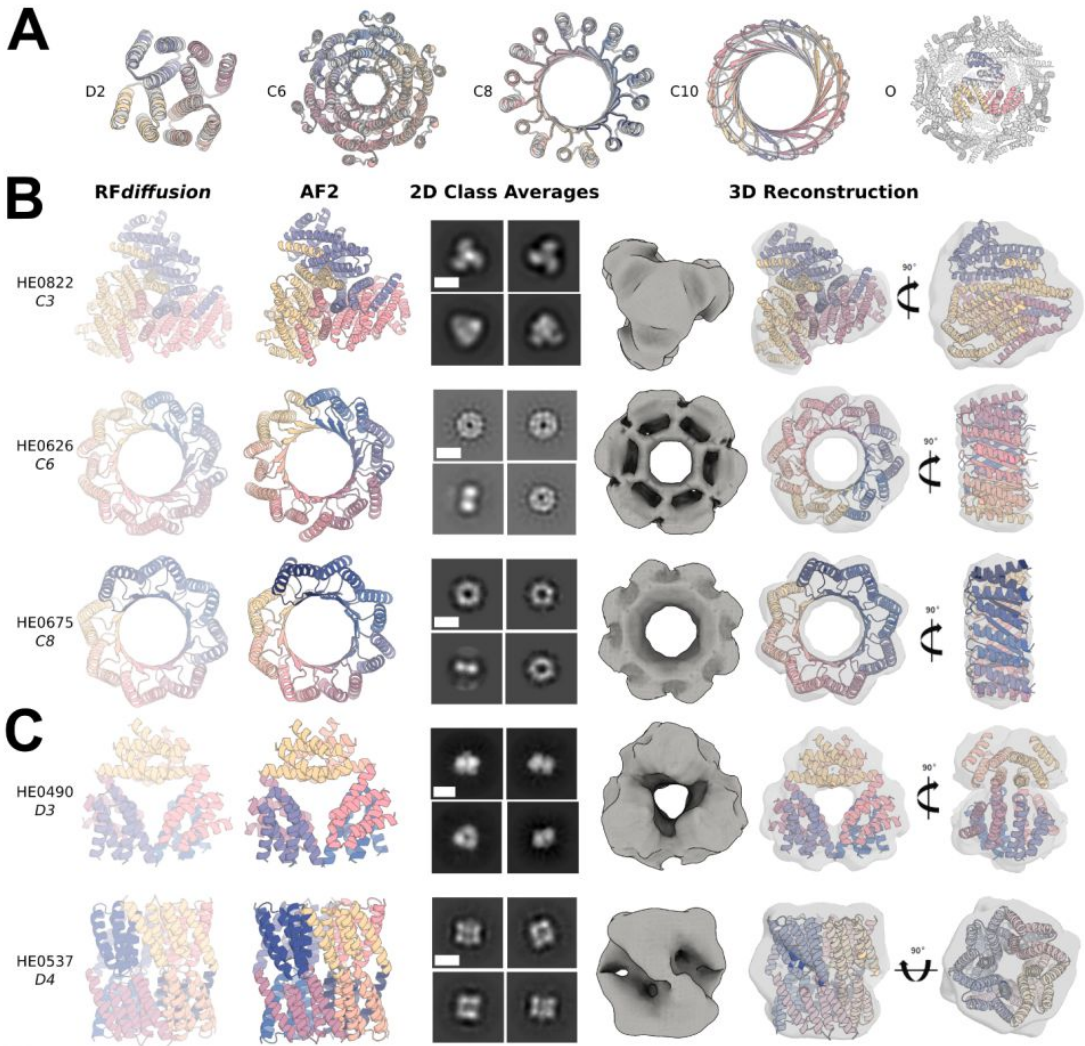
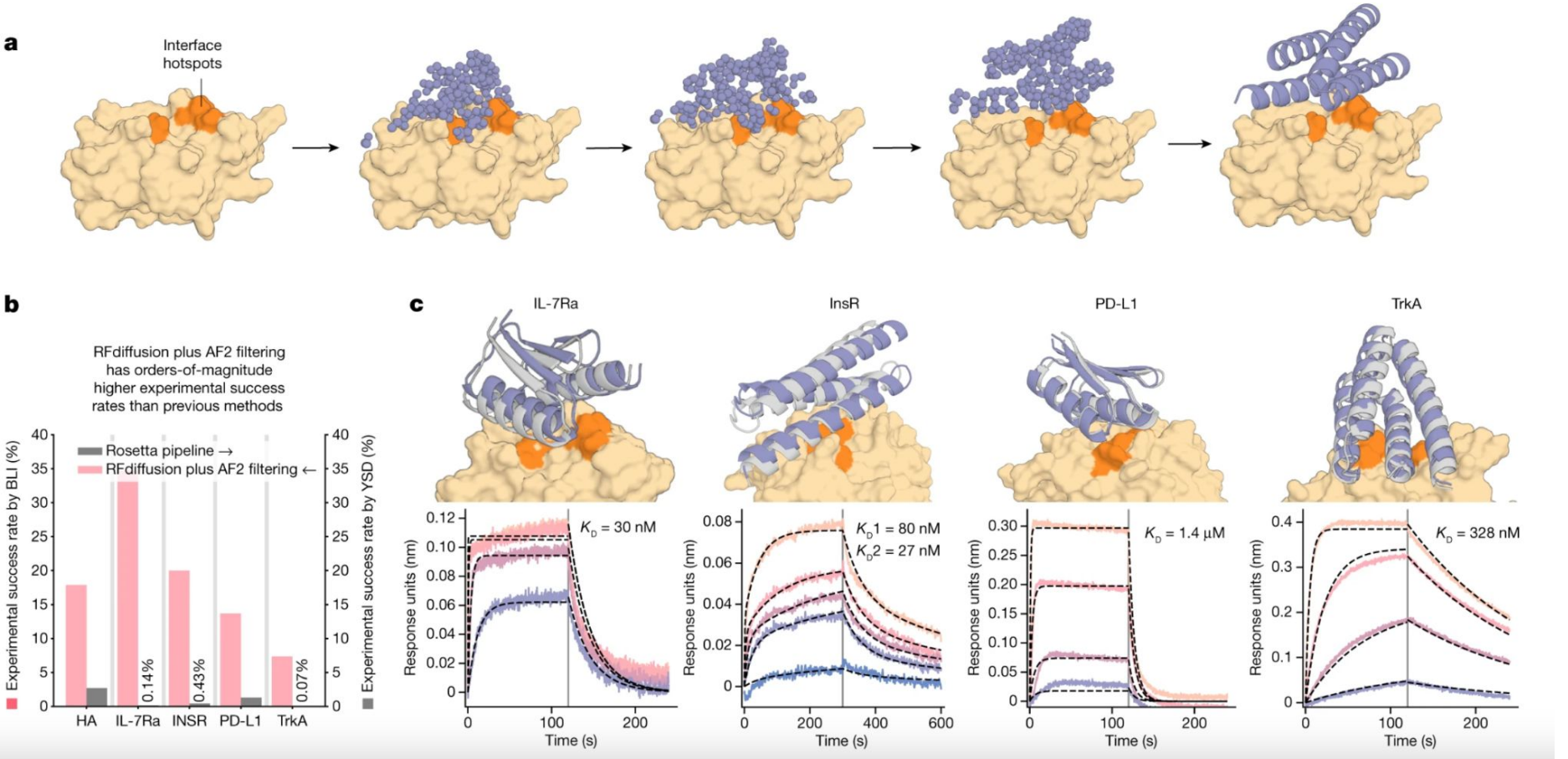



**Figure 1: RF*diffusion* is a** denoising diffusion probabilistic model with RoseTTAFold


A
D2
C6
C8
C10
O
B
RFdiffusion
AF2
2D Class Averages
3D Reconstruction
HE0822
C3
HE0626
C6
HE0675
C8
C
HE0490
D3
HE0537
D4
90°

# Fig. 6: De novo design of protein-binding proteins.

From: [De novo design of protein structure and function with RFdiffusion]



# De novo designed proteins neutralize lethal snake venom toxins

https://doi.org/10.1038/s41586-024-08393-x

Received: 8 May 2024

Accepted: 13 November 2024

Published online: 15 January 2025

Open access

Check for updates

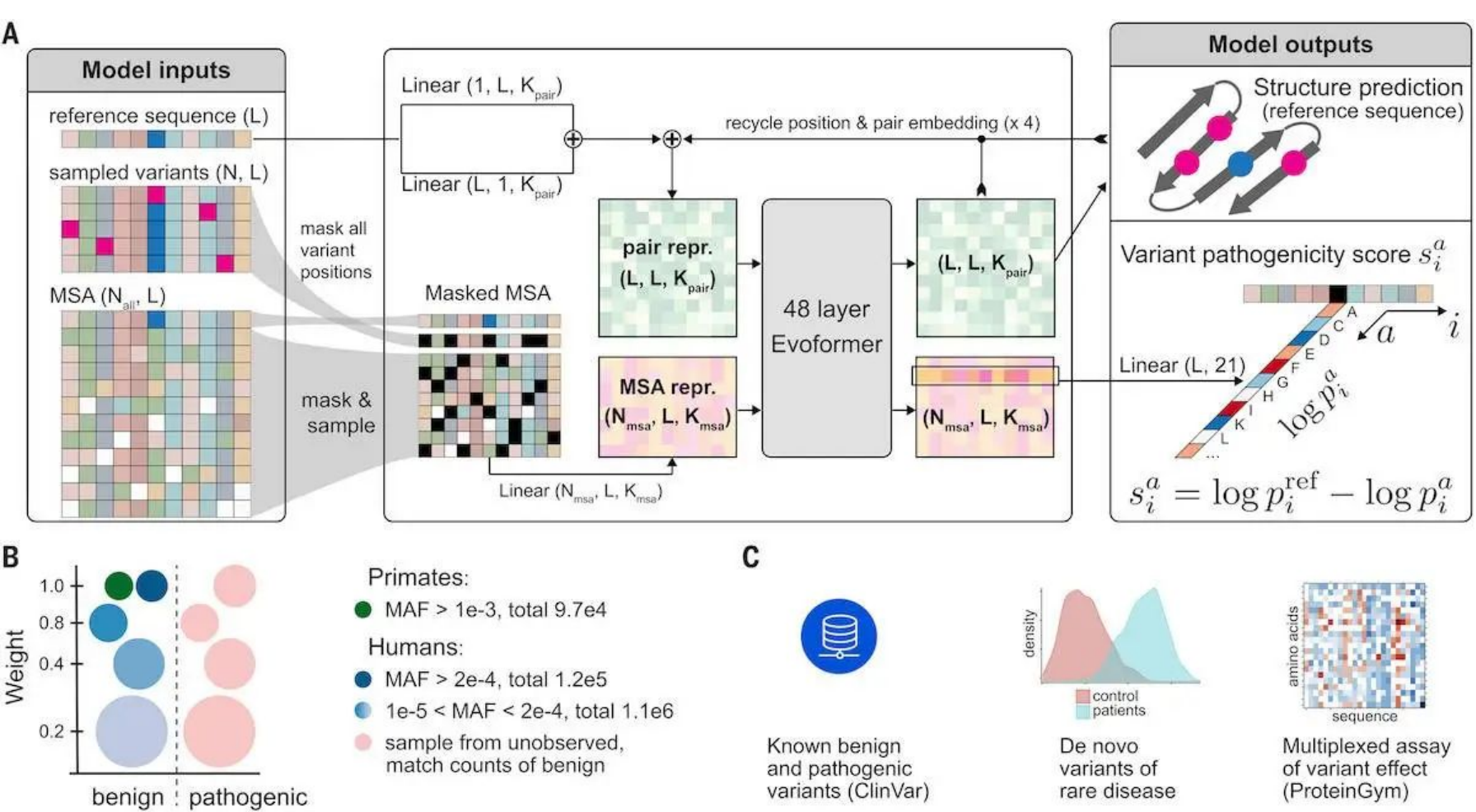
Susana Vázquez Torres[1,2,3], Melisa Benard Valle[4], Stephen P. Mackessy[5], Stefanie K. Menzies[6,7,8], Nicholas R. Casewell[6,7], Shirin Ahmadi[4], Nick J. Burlet[4], Edin Muratspahić[1,2], Isaac Sappington[1,2,3], Max D. Overath[4], Esperanza Rivera-de-Torre[4], Jann Ledergerber[4], Andreas H. Laustsen[4], Kim Boddum[9], Asim K. Bera[1,2], Alex Kang[1,2], Evans Brackenbrough[1,2], Iara A. Cardoso[6], Edouard P. Crittenden[6], Rebecca J. Edge[10], Justin Decarreau[1,2], Robert J. Ragotte[1,2], Arvind S. Pillai[1,2], Mohamad Abedi[1,2], Hannah L. Han[1,2], Stacey R. Gerben[1,2], Analisa Murray[1,2], Rebecca Skotheim[1,2], Lynda Stuart[1,2], Lance Stewart[1,2], Thomas J. A. Fryer[4,11], Timothy P. Jenkins[4]✉ & David Baker[1,2,12]✉

Snakebite envenoming remains a devastating and neglected tropical disease, claiming over 100,000 lives annually and causing severe complications and long-lasting disabilities for many more[1,2]. Three-finger toxins (3FTx) are highly toxic components of elapid snake venoms that can cause diverse pathologies, including severe tissue damage[3] and inhibition of nicotinic acetylcholine receptors, resulting in life-threatening neurotoxicity[4]. At present, the only available treatments for snakebites consist of polyclonal antibodies derived from the plasma of immunized animals, which have high cost and limited efficacy against 3FTxs[5–7]. Here we used deep learning methods to de novo design proteins to bind short-chain and long-chain α-neurotoxins and cytotoxins from the 3FTx family. With limited experimental screening, we obtained protein designs with remarkable thermal stability, high binding affinity and near-atomic-level agreement with the computational models. The designed proteins effectively neutralized all three 3FTx subfamilies in vitro and protected mice from a lethal neurotoxin challenge. Such potent, stable and readily manufacturable toxin-neutralizing proteins could provide the basis for safer, cost-effective and widely accessible next-generation antivenom therapeutics. Beyond snakebite, our results highlight how computational design could help democratize therapeutic discovery, particularly in resource-limited settings, by substantially reducing costs and resource requirements for the development of therapies for neglected tropical diseases.

# Live Demo

- PDB
- uniprot
- ColabFold
- ProteinMPNN
- RFdiffusion

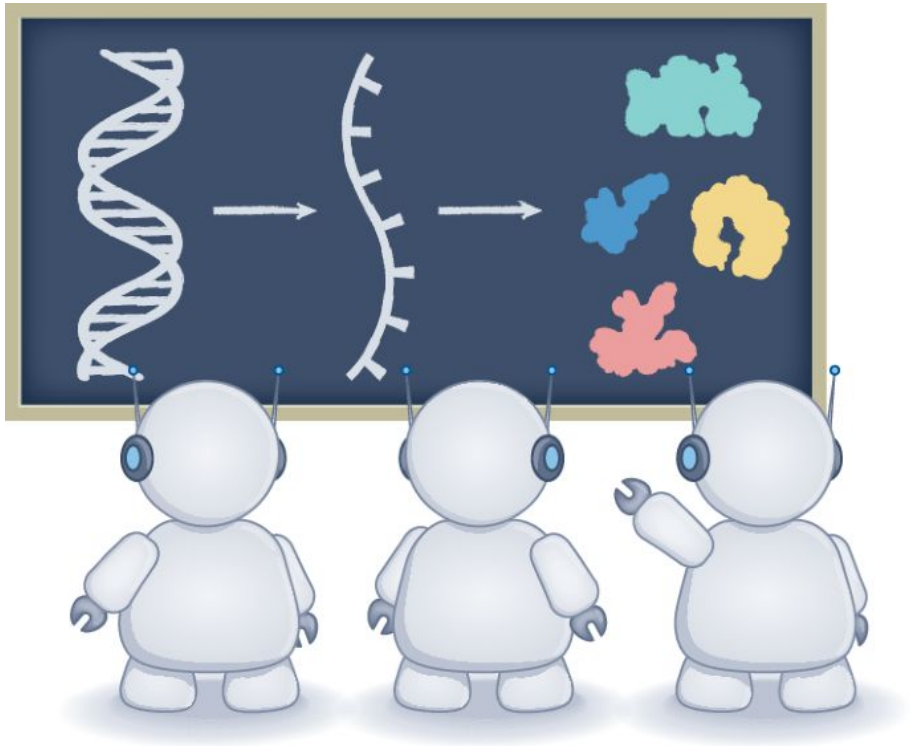

AlphaMissense
AlphaFold Based Tool to
Predict Mutations Causing
Genetic Diseases




(Jun Cheng et al., Science 2023)

# Where are we at AI for Biology?

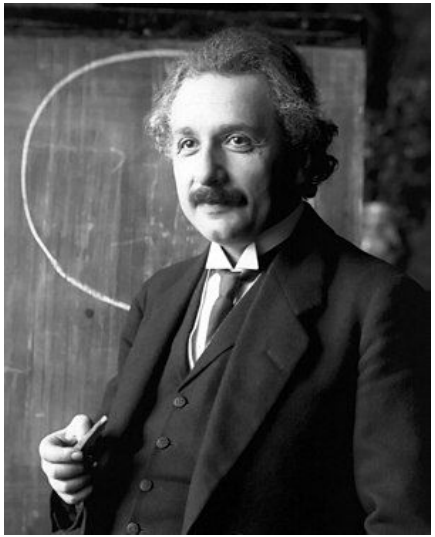
- Where are we at for drug discovery?
- Where are we at for protein dynamics?
- Where are we at for genetics?



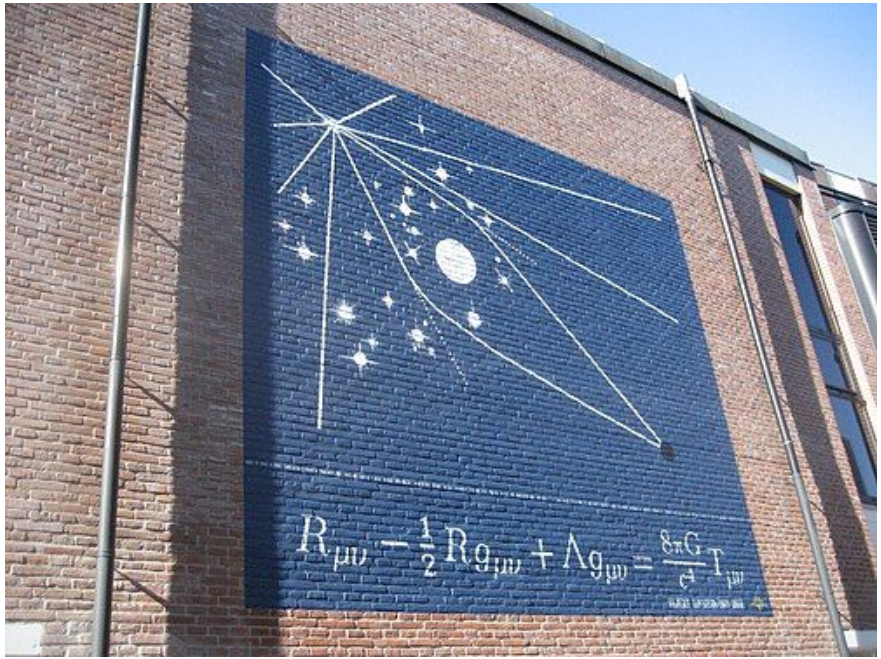
# Some useful Resources

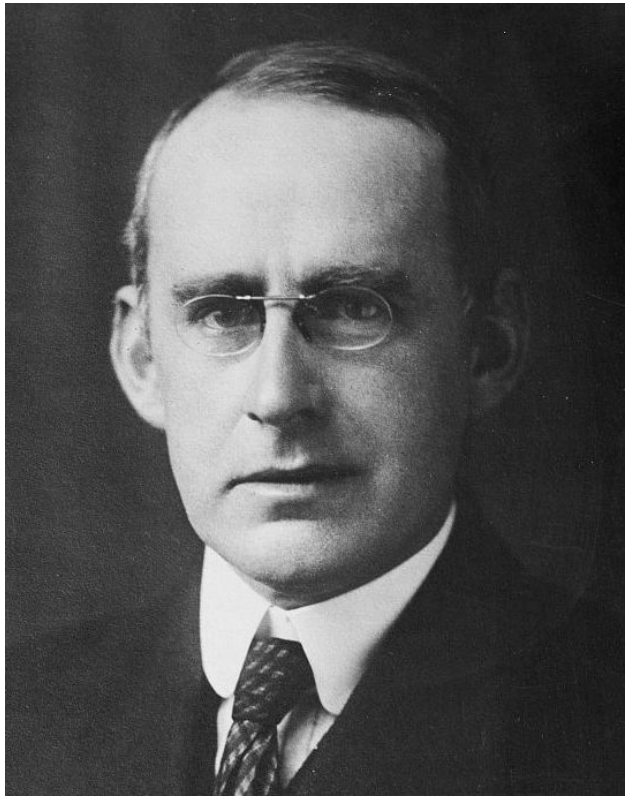
- PDB
- uniprot
- ColabFold
- ProteinMPNN
- RFdiffusion
- Alphafold server
- Foldseek
- Kaggle
- AlphafoldDB

Image Credit: Event Horizon Telescope Collaboration

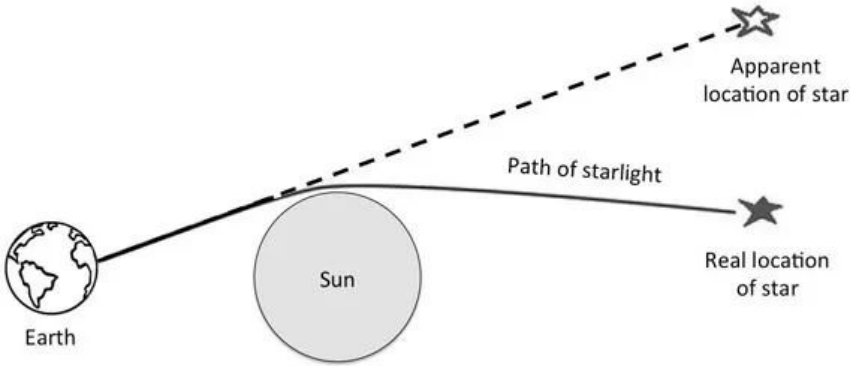


Albert Einstein





Arthur Eddington




Eddington experiment

Karl Schwarzschild

# SCHWARZSCHILD METRIC

describes the shape of spacetime around a spherical source, where $g_{\mu\nu}$ is the spacetime metric:

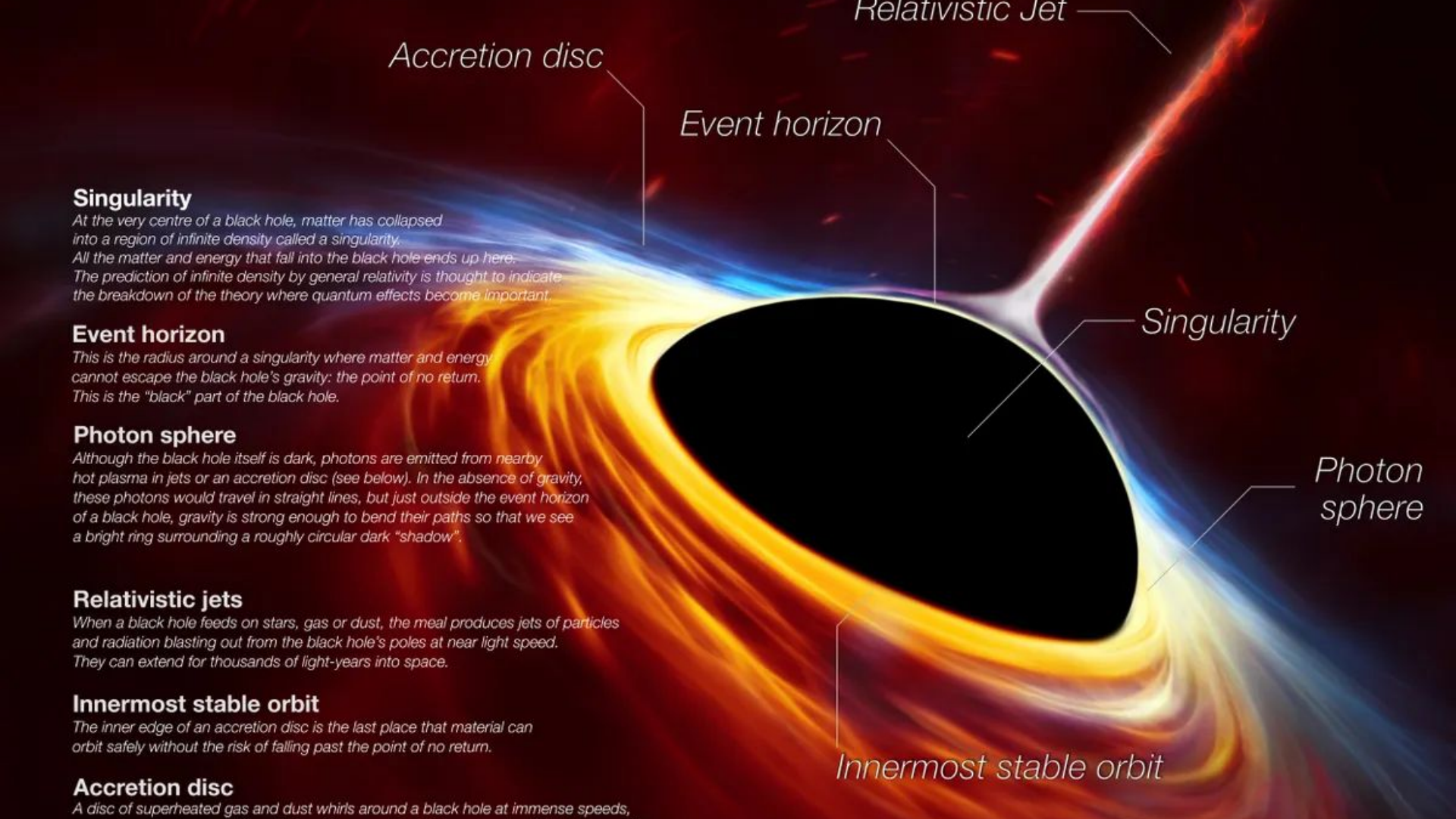
$$g_{\mu\nu} = \begin{pmatrix} g_{tt} & g_{tr} & g_{t\theta} & g_{t\varphi} \\ g_{rt} & g_{rr} & g_{r\theta} & g_{r\varphi} \\ g_{\theta t} & g_{\theta r} & g_{\theta\theta} & g_{\theta\varphi} \\ g_{\varphi t} & g_{\varphi r} & g_{\varphi\theta} & g_{\varphi\varphi} \end{pmatrix}$$

$$= \begin{pmatrix} -\left(1 - \frac{2G_N M}{c^2 r}\right) & 0 & 0 & 0 \\ 0 & \left(1 - \frac{2G_N M}{c^2 r}\right)^{-1} & 0 & 0 \\ 0 & 0 & r^2 & 0 \\ 0 & 0 & 0 & r^2 \sin^2\theta \end{pmatrix}$$

coordinates: $(x^0, x^1, x^2, x^3) = (ct, r, \theta, \varphi)$

Karl Schwarzschild

Mathematics Learning page
By Mamun-Or-Rashid




**Singularity**
*At the very centre of a black hole, matter has collapsed into a region of infinite density called a singularity. All the matter and energy that fall into the black hole ends up here. The prediction of infinite density by general relativity is thought to indicate the breakdown of the theory where quantum effects become important.*

**Event horizon**
*This is the radius around a singularity where matter and energy cannot escape the black hole's gravity: the point of no return. This is the "black" part of the black hole.*

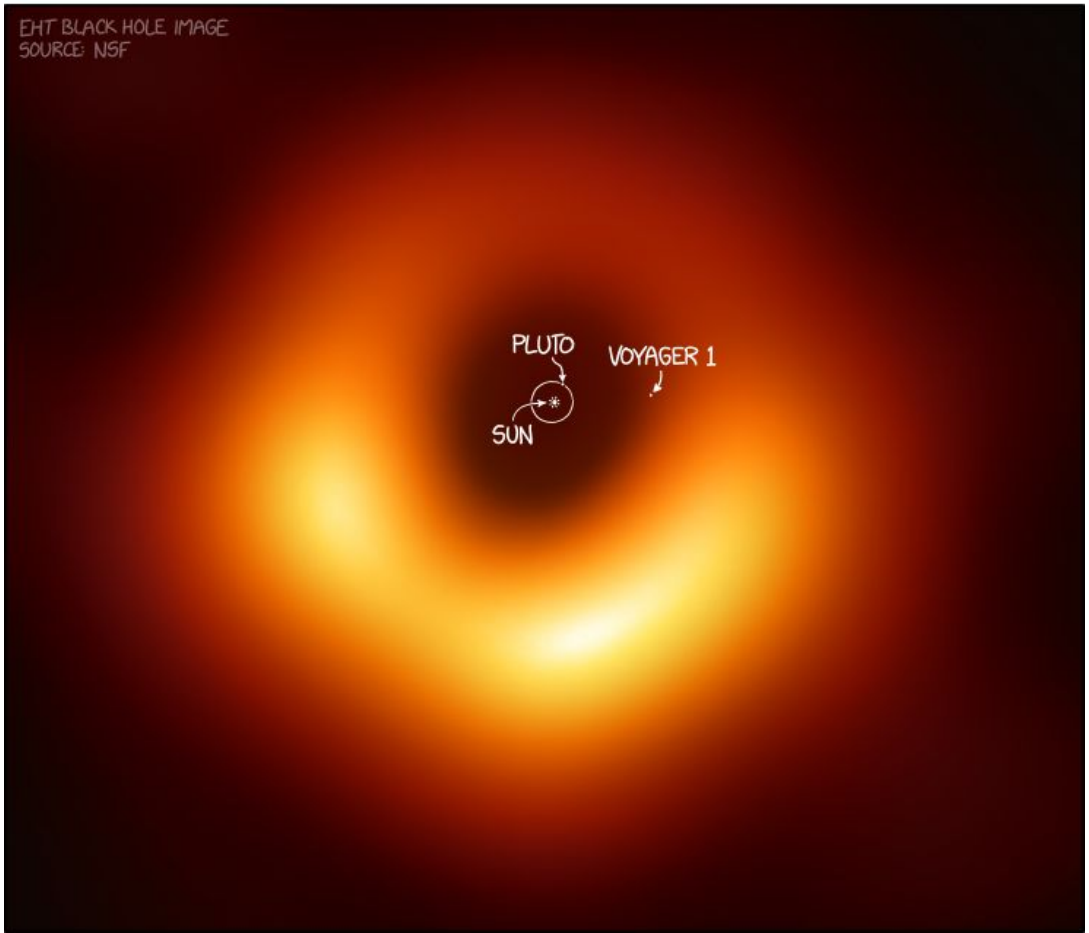
**Photon sphere**
*Although the black hole itself is dark, photons are emitted from nearby hot plasma in jets or an accretion disc (see below). In the absence of gravity, these photons would travel in straight lines, but just outside the event horizon of a black hole, gravity is strong enough to bend their paths so that we see a bright ring surrounding a roughly circular dark "shadow".*

**Relativistic jets**
*When a black hole feeds on stars, gas or dust, the meal produces jets of particles and radiation blasting out from the black hole's poles at near light speed. They can extend for thousands of light-years into space.*

**Innermost stable orbit**
*The inner edge of an accretion disc is the last place that material can orbit safely without the risk of falling past the point of no return.*

**Accretion disc**
*A disc of superheated gas and dust whirls around a black hole at immense speeds,*

SIZE COMPARISON:
THE M87 BLACK HOLE
AND
OUR SOLAR SYSTEM




**~ 40 uas**

# HST best resolution ~ 0.04 as
# ALMA best resolution ~ 0.01 as

# Black Hole Observation

- **Wavelength (230GHz, ~1.3mm)** ✔

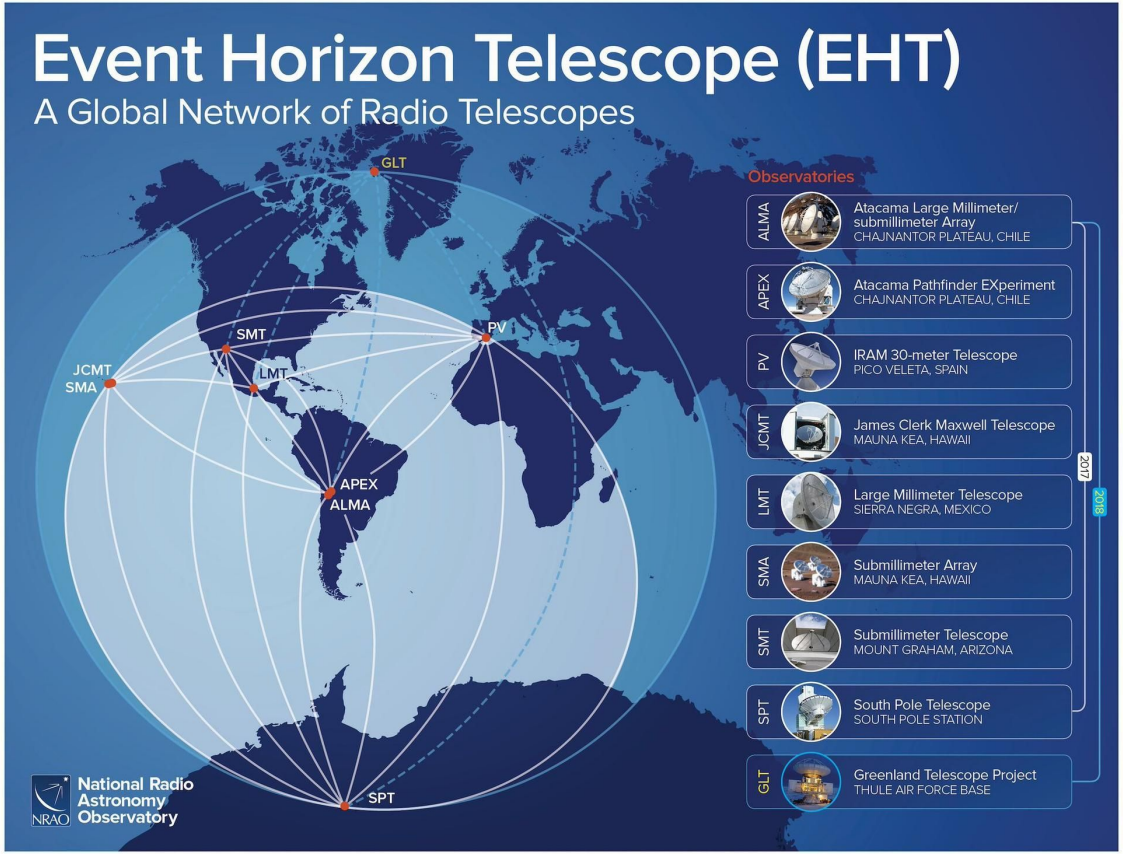
- Right size of the telescope ~ (13 Million meters = diameter of the earth)

**ALMA Telescope** Credits: ESO

# Black Hole Observation

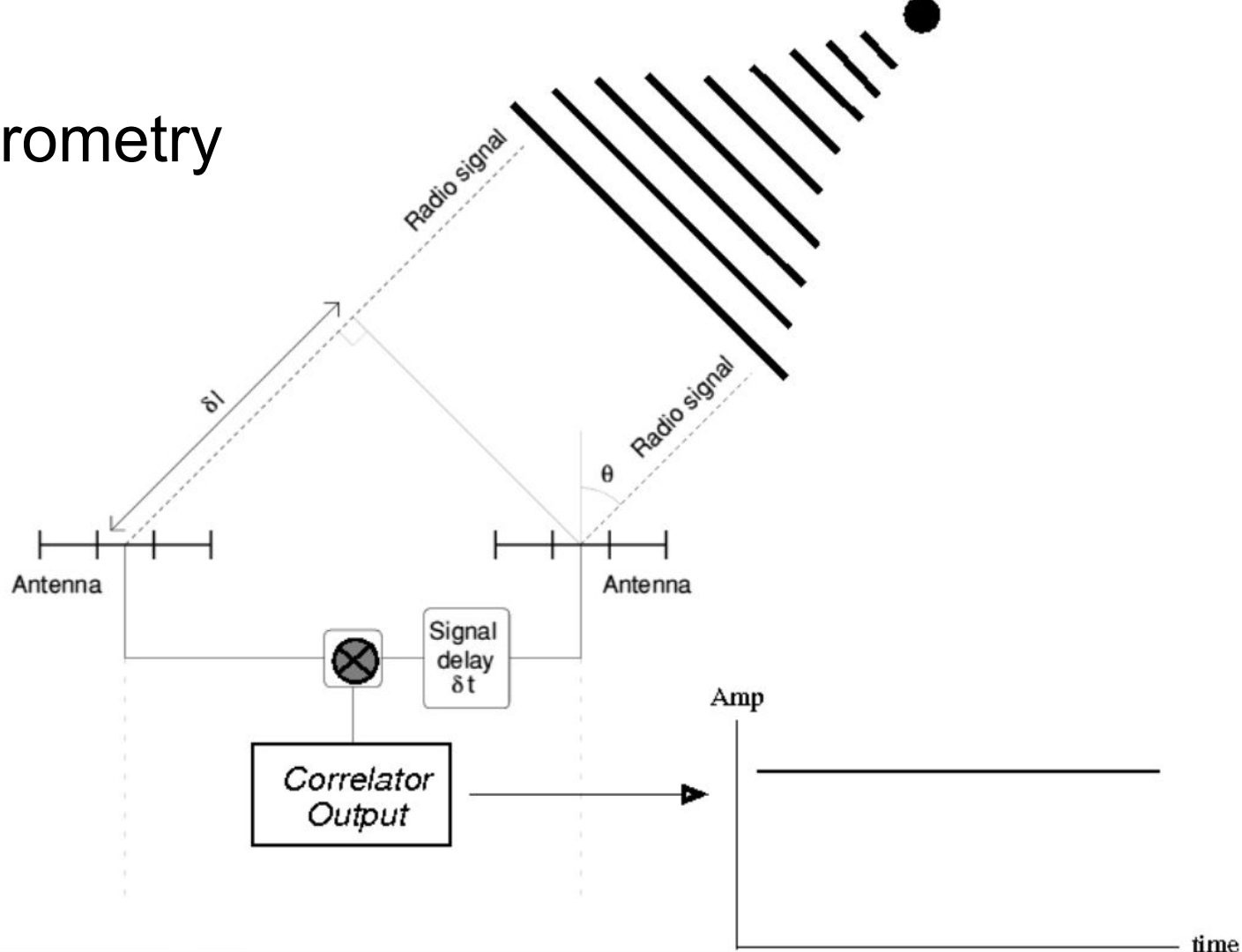
- Wavelength (230GHz, ~1.3mm) ✔

- **Right size of the telescope ~ (13 Million meters = diameter of the earth)** ✔




Credits: EHT

# Radio Interferometry



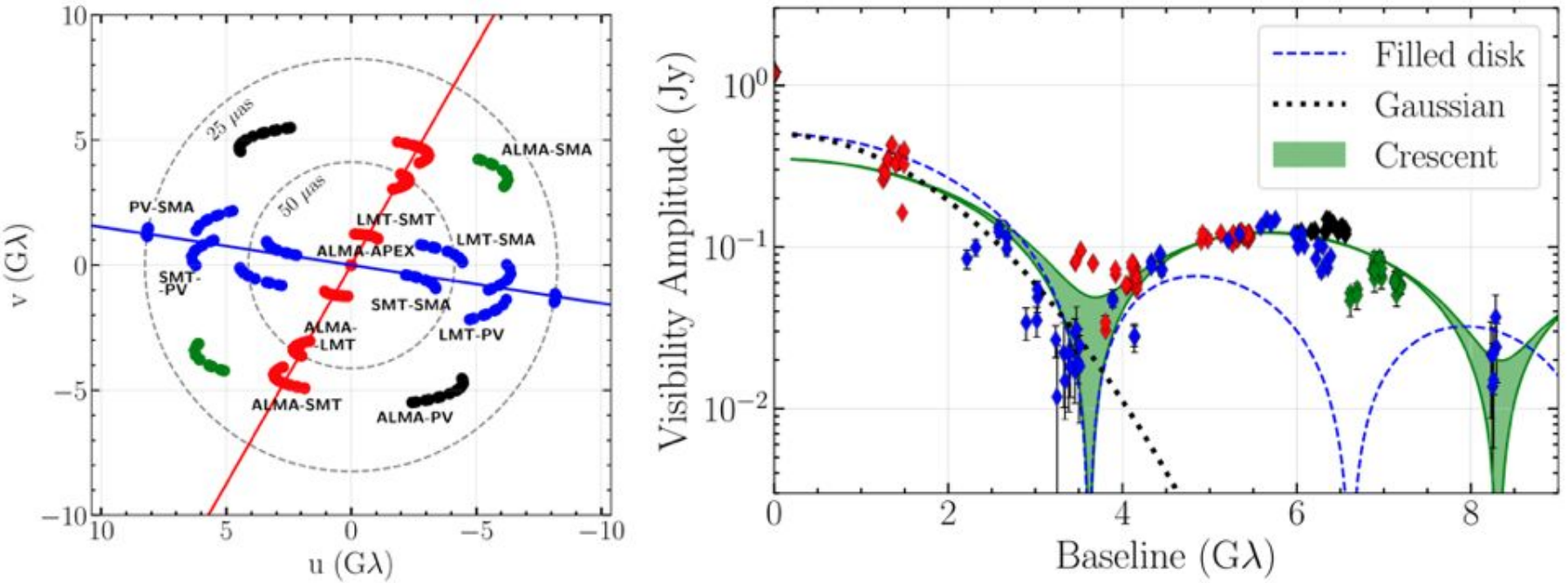



**Figure 1.** ($u$, $v$)-coverage (left panel) and visibility amplitudes (right panel) of M87 for the high-band April 11 data. The ($u$, $v$)-coverage has two primary orientations, east–west in blue and north–south in red, with two diagonal fillers at large baselines in green and black. Note that the Large Millimeter Telescope (LMT) and the Submillimeter Telescope (SMT) participate in both orientations, and that the LMT amplitudes are subject to significant gain errors. There is evidence for substantial depressions in the visibility amplitudes at ∼3.4 Gλ and ∼8.3 Gλ. The various lines in the right panel show the expected behavior of (dotted line) a Gaussian, (dashed line) a filled disk, and (green area) a crescent shape along different orientations. The image of M87 does not appear to be consistent with a filled disk or a Gaussian.

Machine Learning

Traditional Method



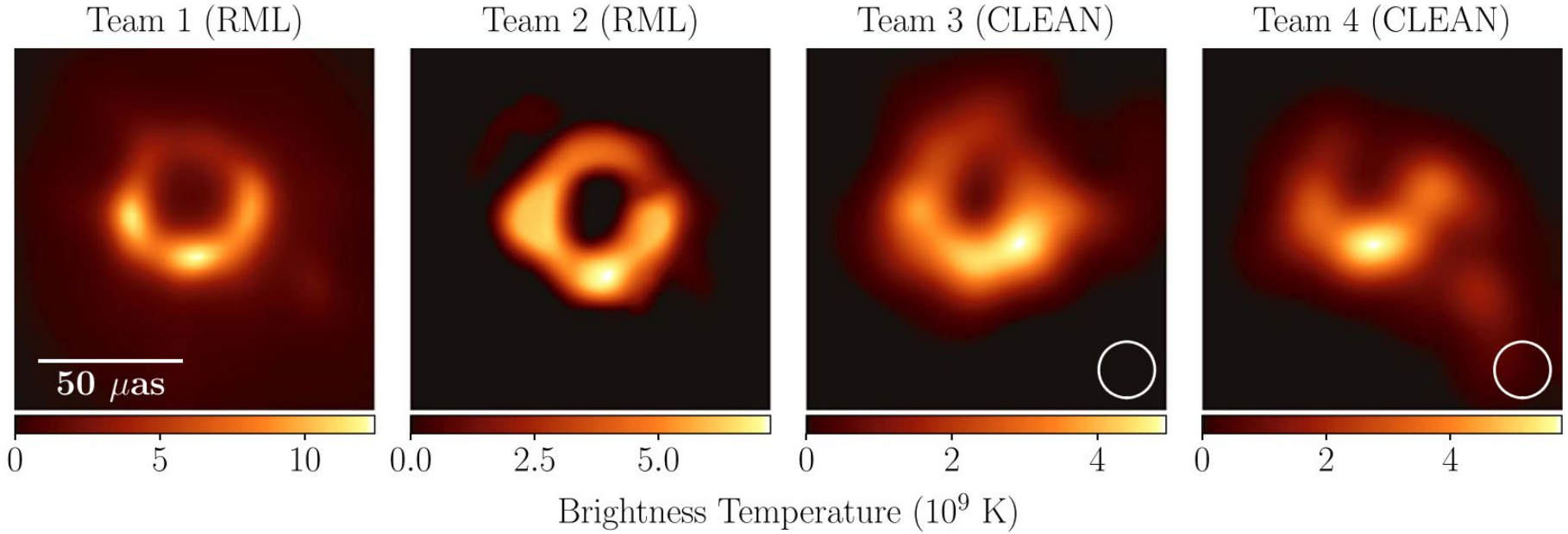



**Figure 4.** The first EHT images of M87, blindly reconstructed by four independent imaging teams using an early, engineering release of data from the April 11 observations. These images all used a single polarization (LCP) rather than Stokes *I*, which is used in the remainder of this Letter. Images from Teams 1 and 2 used RML methods (no restoring beam); images from Teams 3 and 4 used CLEAN (restored with a circular 20 $\mu$as beam, shown in the lower right). The images all show similar morphology, although the reconstructions show significant differences in brightness temperature because of different assumptions regarding the total compact flux density (see Table 2) and because restoring beams are applied only to CLEAN images.

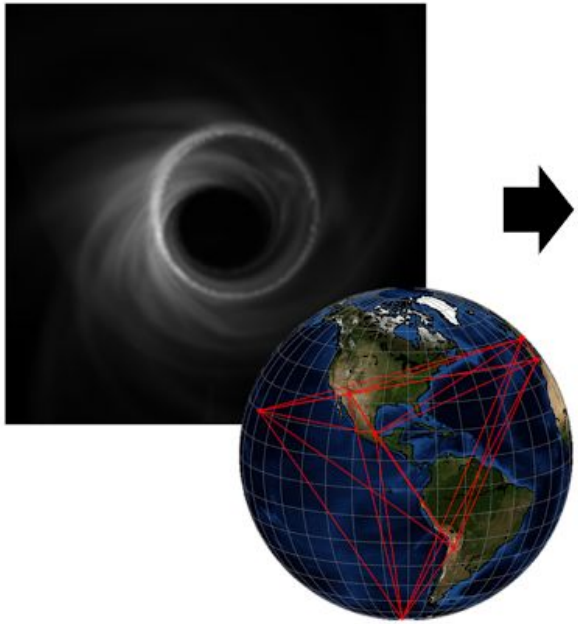
# Image reconstruction (for EHT)

**Regularized Maximum Likelihood (RML)**

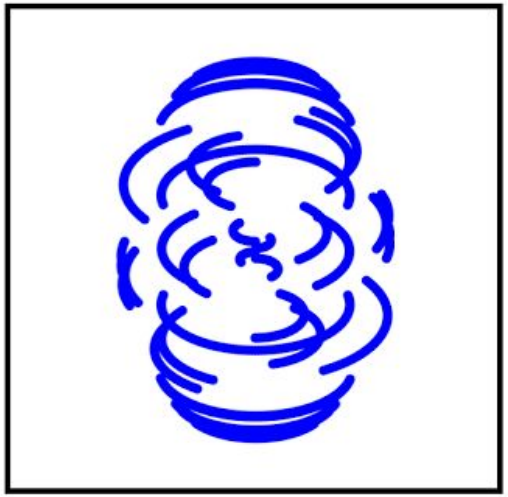
- Used for reconstructing M87* image for EHT
- Prior and forward model (image -> visibility) needed
- Combining with **Normalizing Flow** -> **Deep Probabilistic Imaging**

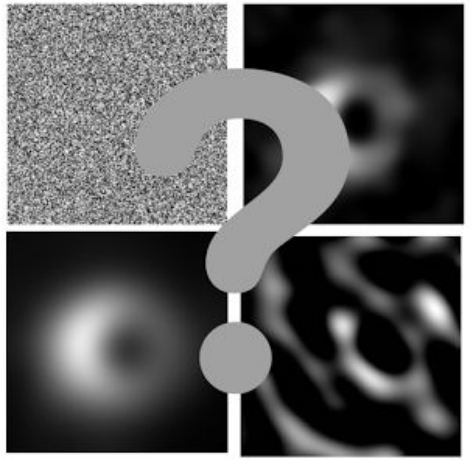
Clean (Deconvolution)

- Used for reconstructing M87* image for EHT
- Also used for most of the Images Reconstruction for Radio Interferometry
- No prior “hard coded”, but require lots of human expertise and experience on hyperparameter tuning…
- Hard to evaluate uncertainty



Measurements

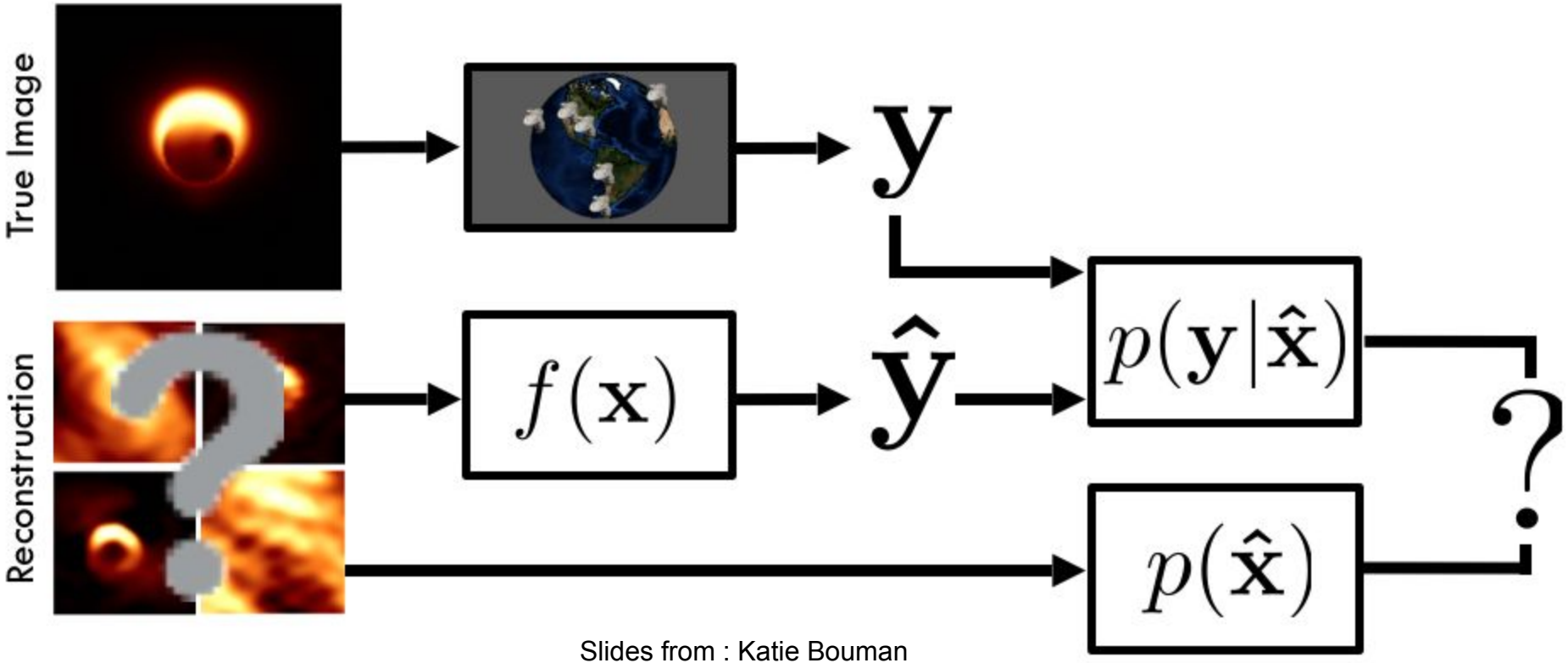




Infinite Number
of Possibilities

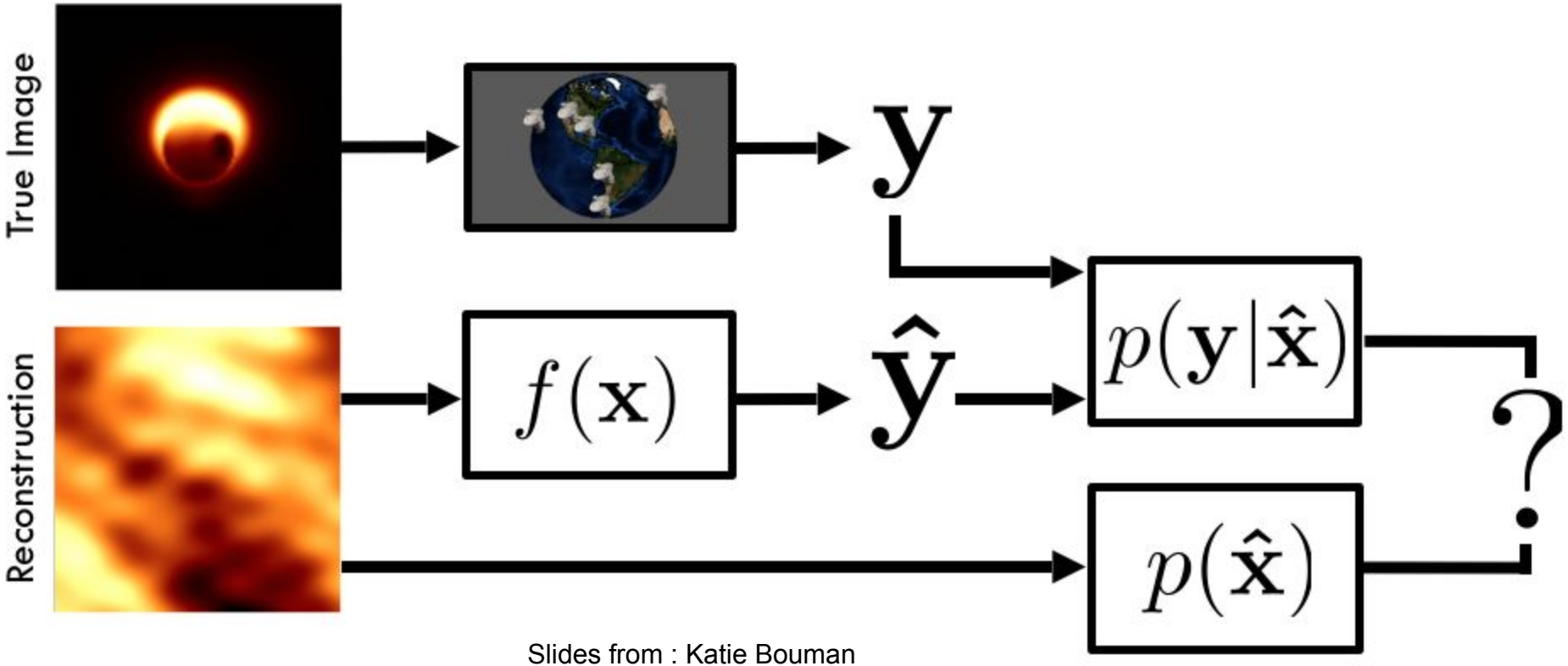
Slides from : Katie Bouman

# Bayesian Model Inversion



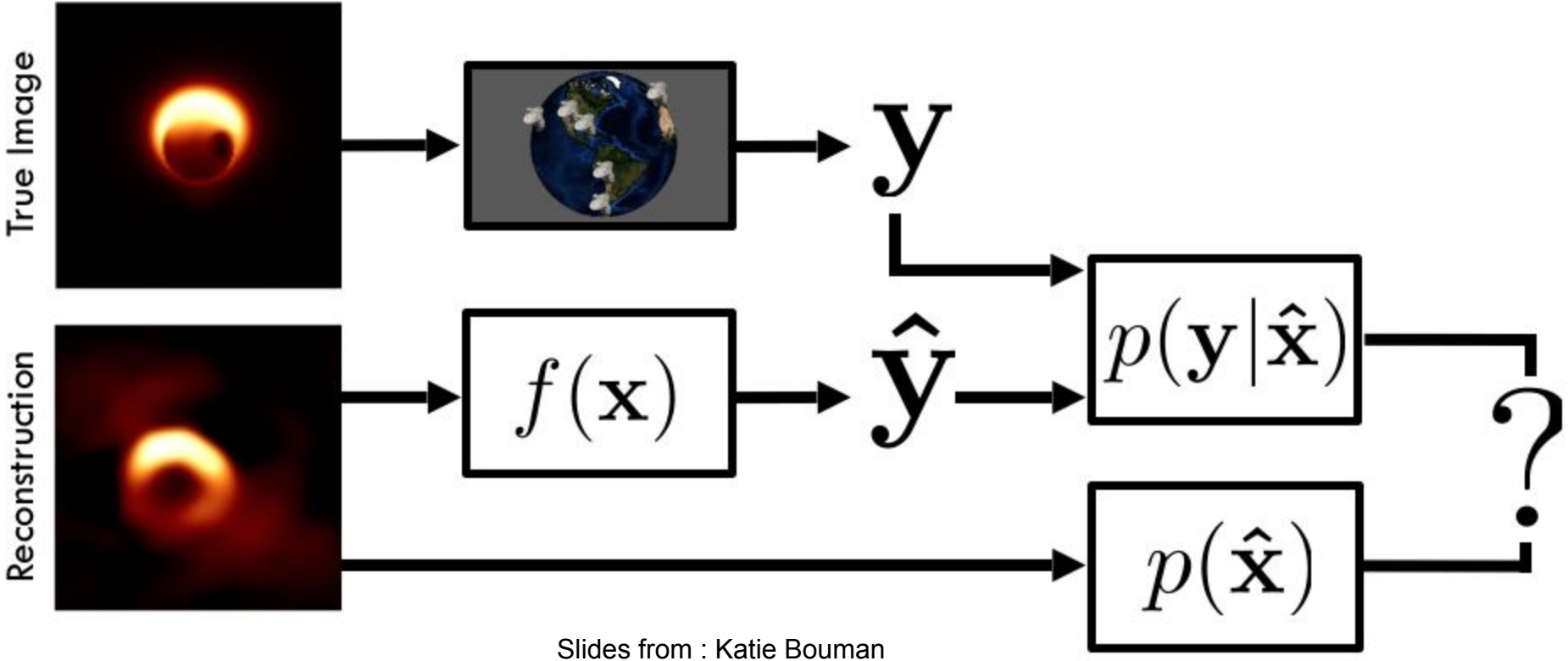

Slides from : Katie Bouman

# Bayesian Model Inversion




Slides from : Katie Bouman

# Bayesian Model Inversion




Slides from : Katie Bouman

Best Image

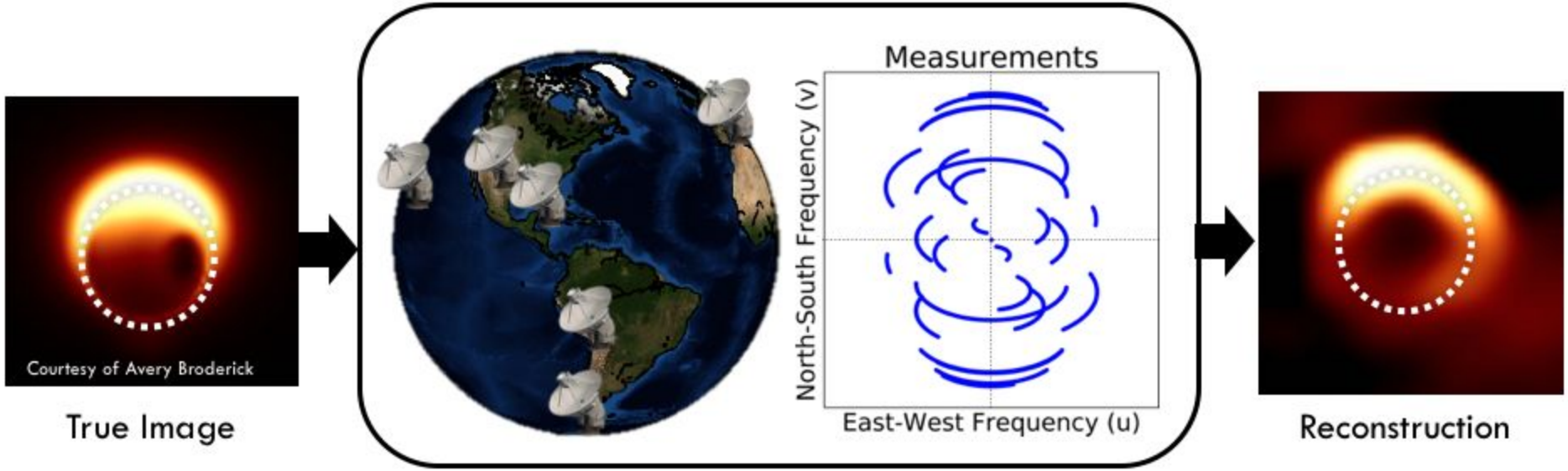
$$\hat{\mathbf{x}}_{\mathrm{MAP}} = \mathrm{argmax}_{\mathbf{x}} \left[ \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) \right]$$

Likelihood

Prior

Slides from : Katie Bouman

# Simulated EHT Image Reconstruction




Slides from : Katie Bouman

Bouman, et al., CVPR, 2016

# EHT-imaging Demo

https://github.com/achael

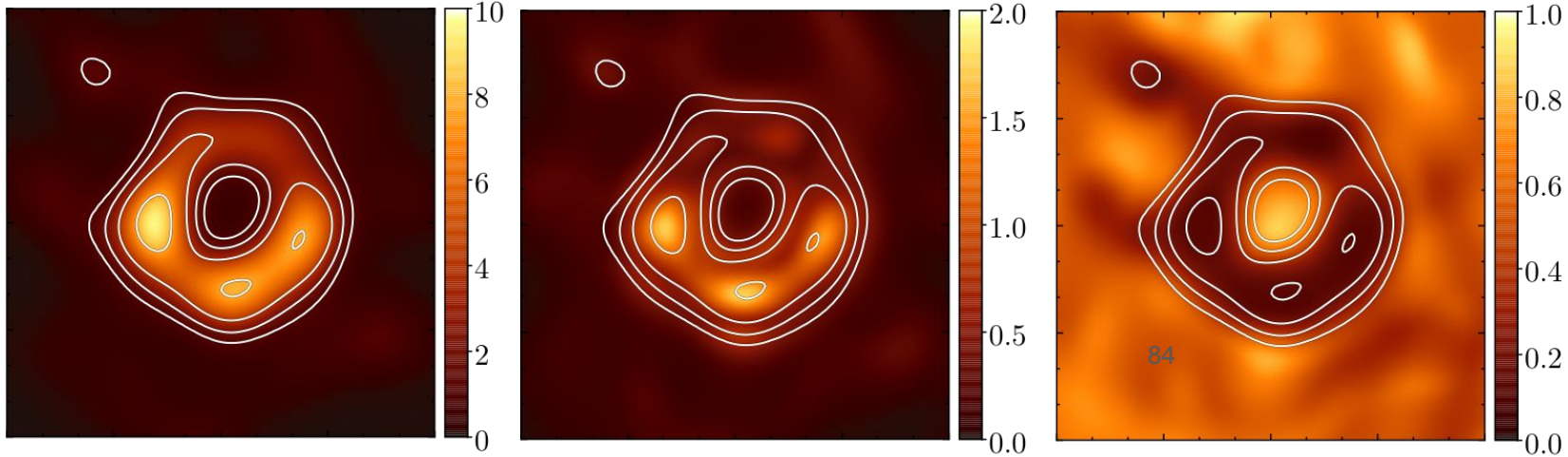
Andrew Chael
(Princeton)

# Uncertainty quantification is important!

Mean Image

Standard Deviation

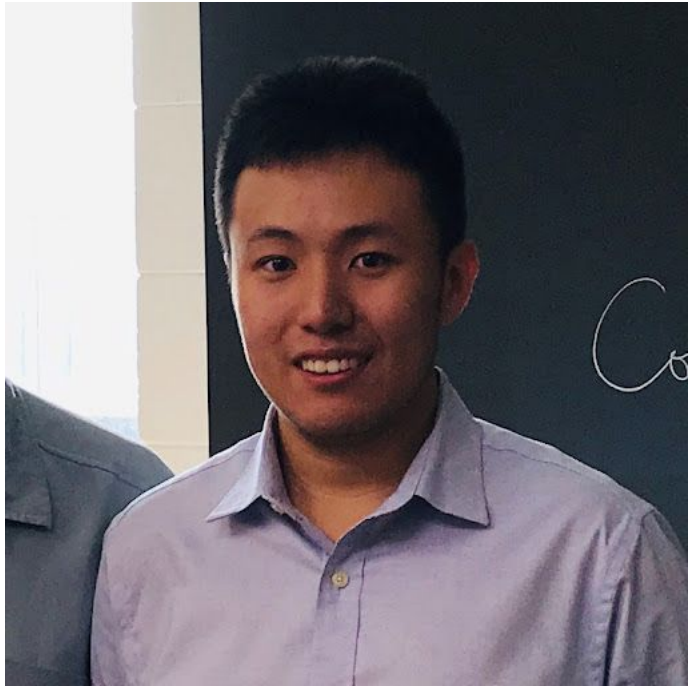
Fractional Standard Deviation



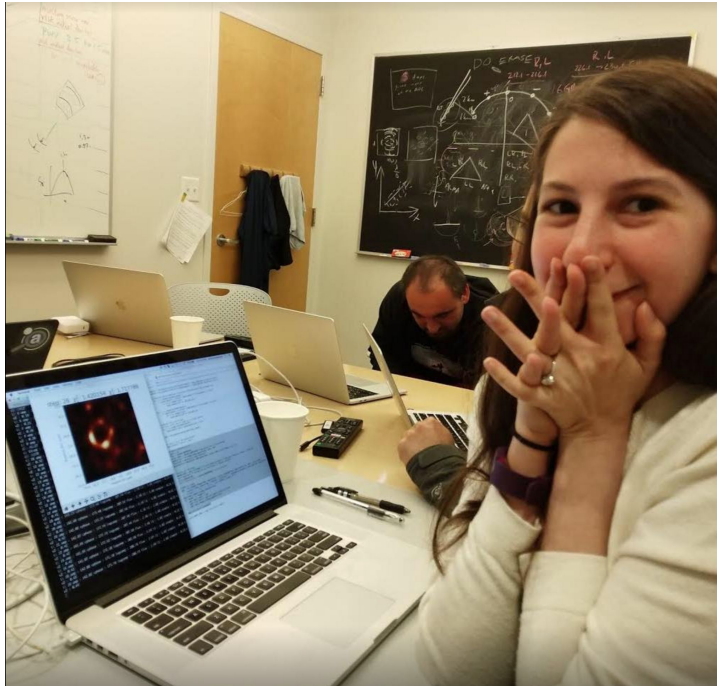

from M87 Paper IV

Slides from : He Sun, Katie Bouman

# Deep Probabilistic Imaging: Uncertainty Quantification and Multi-modal Solution Characterization for Computational Imaging



**He Sun (Caltech)**



**Katie Bouman (Caltech)**

Sun & Bouman (AAAI 2021, 2010.14462)

# Deep Probabilistic Imaging

Big question: Given observable **y**, what's the distribution of image **x**?

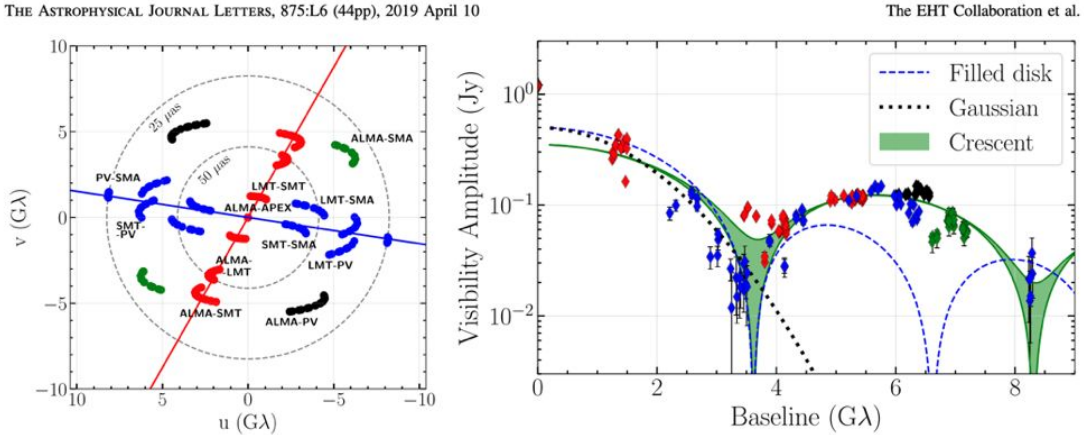
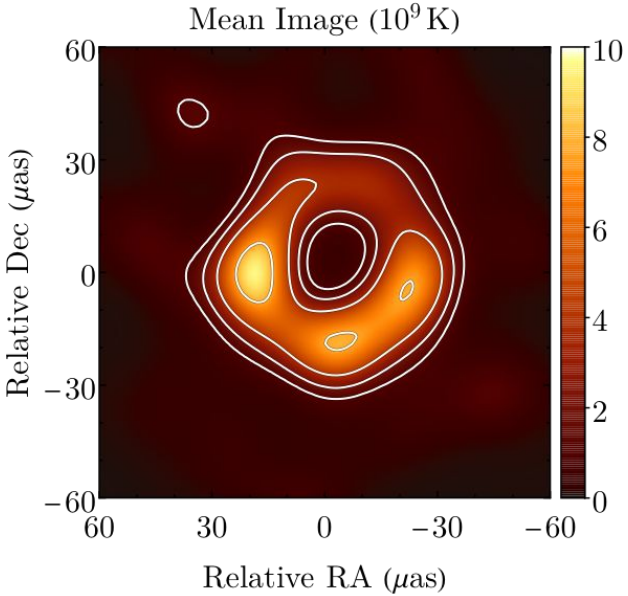



**Figure 1.** ($u$, $v$)-coverage (left panel) and visibility amplitudes (right panel) of M87 for the high-band April 11 data. The ($u$, $v$)-coverage has two primary orientations, east–west in blue and north–south in red, with two diagonal fillers at large baselines in green and black. Note that the Large Millimeter Telescope (LMT) and the Submillimeter Telescope (SMT) participate in both orientations, and that the LMT amplitudes are subject to significant gain errors. There is evidence for substantial depressions in the visibility amplitudes at ∼3.4 Gλ and ∼8.3 Gλ. The various lines in the right panel show the expected behavior of (dotted line) a Gaussian, (dashed line) a filled disk, and (green area) a crescent shape along different orientations. The image of M87 does not appear to be consistent with a filled disk or a Gaussian.



# Regularized Maximum Likelihood (RML)

Best Image

$$\hat{\mathbf{x}}_{\text{MAP}} = \text{argmax}_{\mathbf{x}} \left[\log p(\mathbf{x}|\mathbf{y})\right]$$

$$= \text{argmax}_{\mathbf{x}} \left[\log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x})\right]$$

Likelihood

Prior

Slides from : He Sun, Katie Bouman

# Posterior Exploration

$$x \sim p(\mathbf{x}|\mathbf{y})$$

The posterior can be sampled directly using Markov Chain Monte Carlo (MCMC) method.

"Hybrid Very Long Baseline Interferometry Imaging and Modeling with themis", ApJ 2020



Slides from : He Sun, Katie Bouman

# Posterior Exploration

$$x \sim p(\mathbf{x}|\mathbf{y})$$

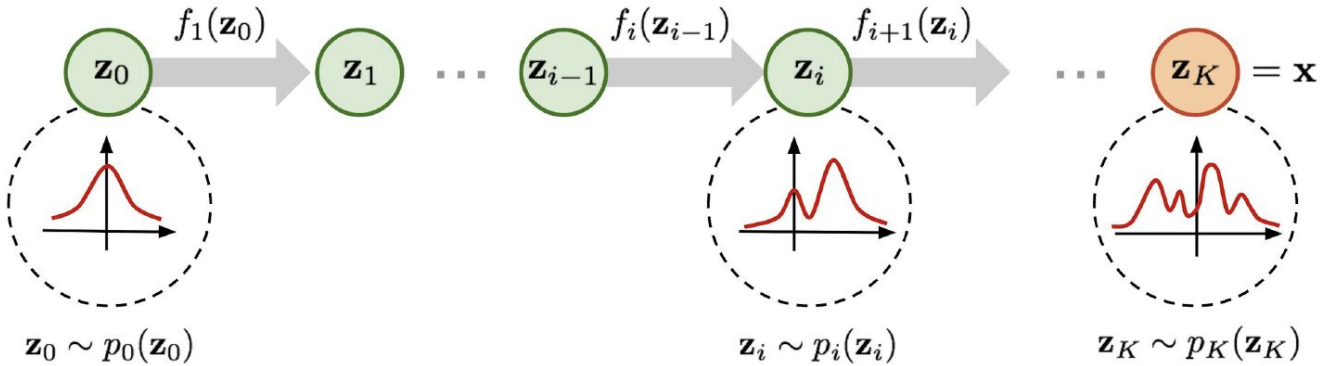
Or we could use a **Neural Networks** (Normalizing Flow)!

Slides from : He Sun, Katie Bouman

# Introduction to Normalizing Flow

## Normalizing Flows

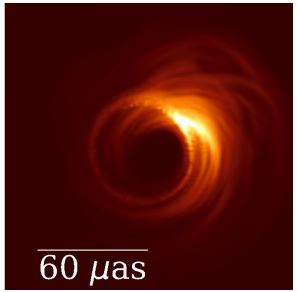
A normalizing flow transforms a simple distribution into a complex one by applying a sequence of invertible transformation functions.



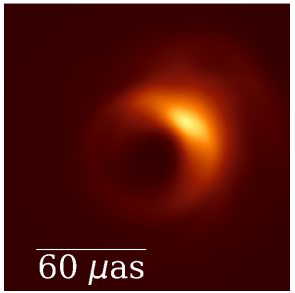
Slides from : He Sun, Katie Bouman

# Characterizing Uncertainty & Multimodal Solutions

Truth

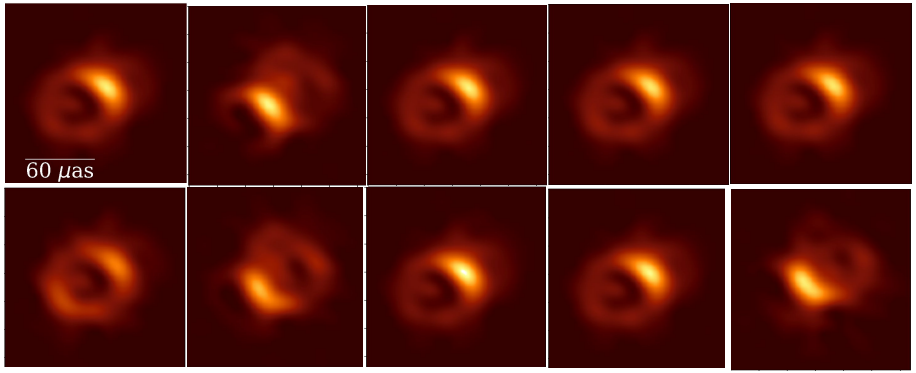



Blurred Truth

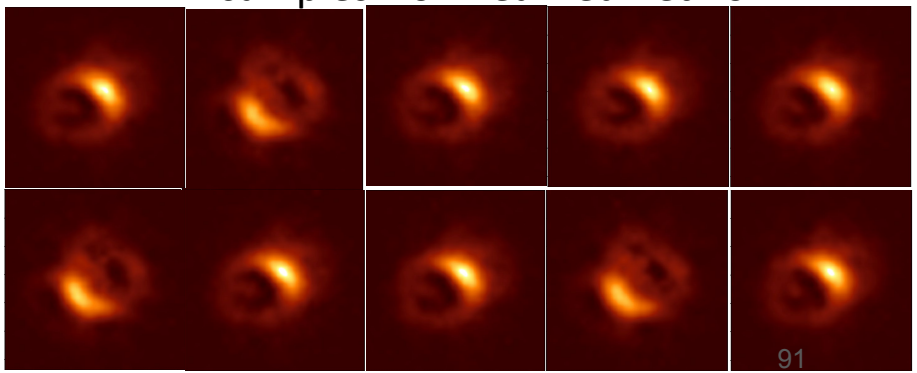




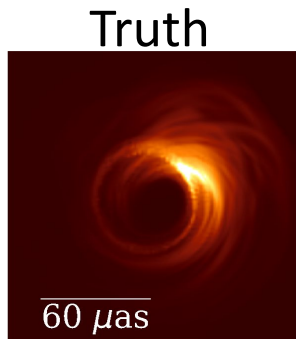
RML results from different initializations

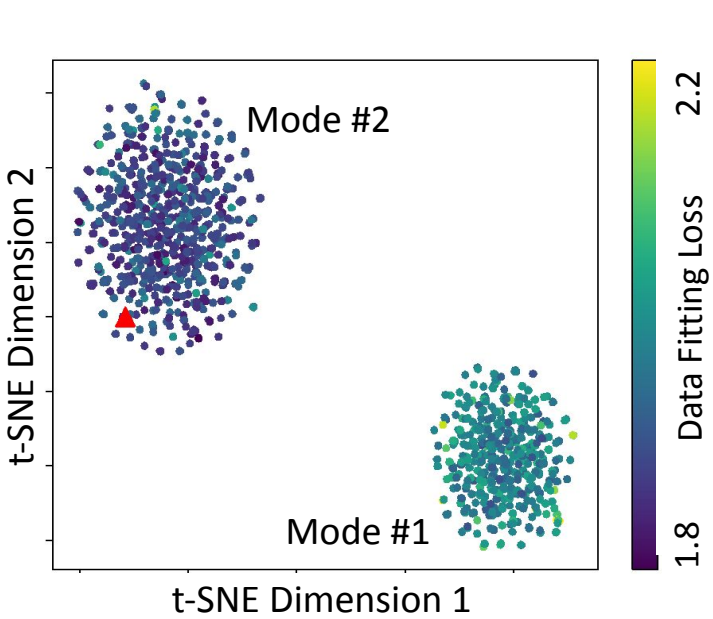


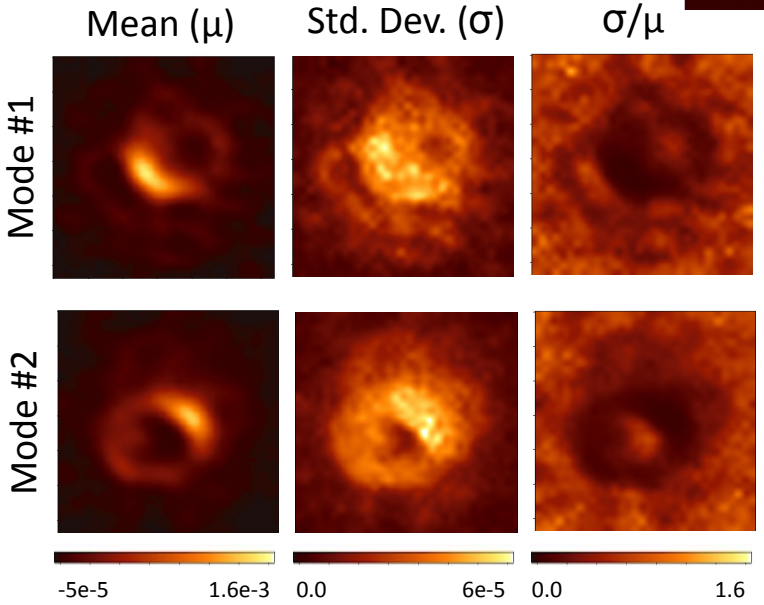

DPI samples from learned network



# Exploring the Posterior

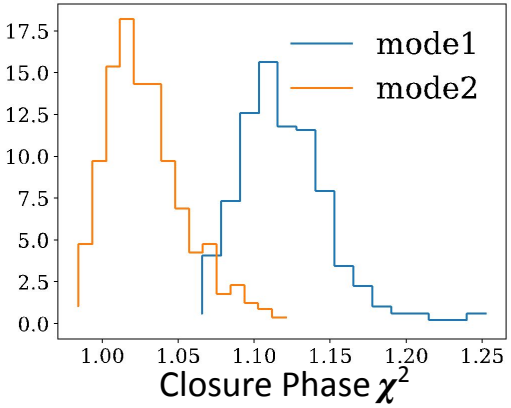


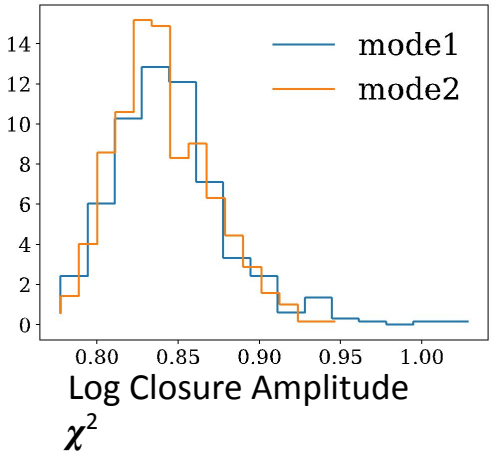



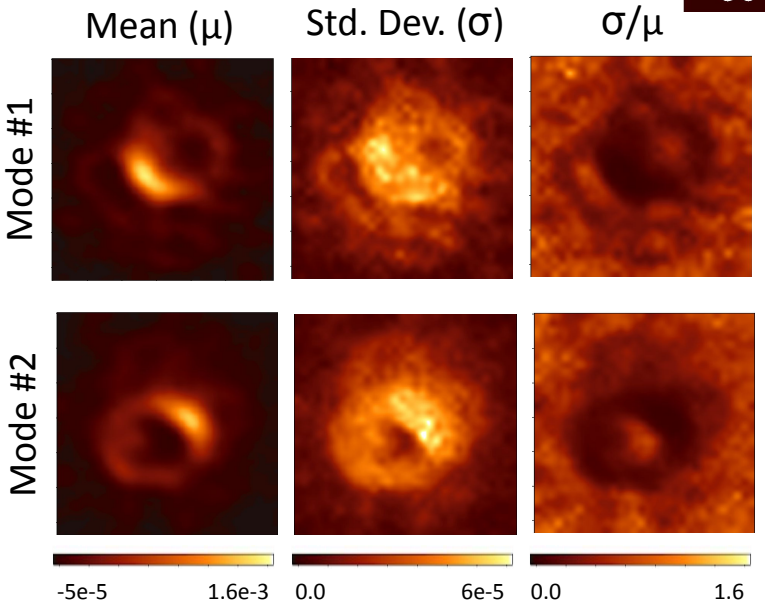

Slides from : He Sun, Katie Bouman

# Exploring the Posterior



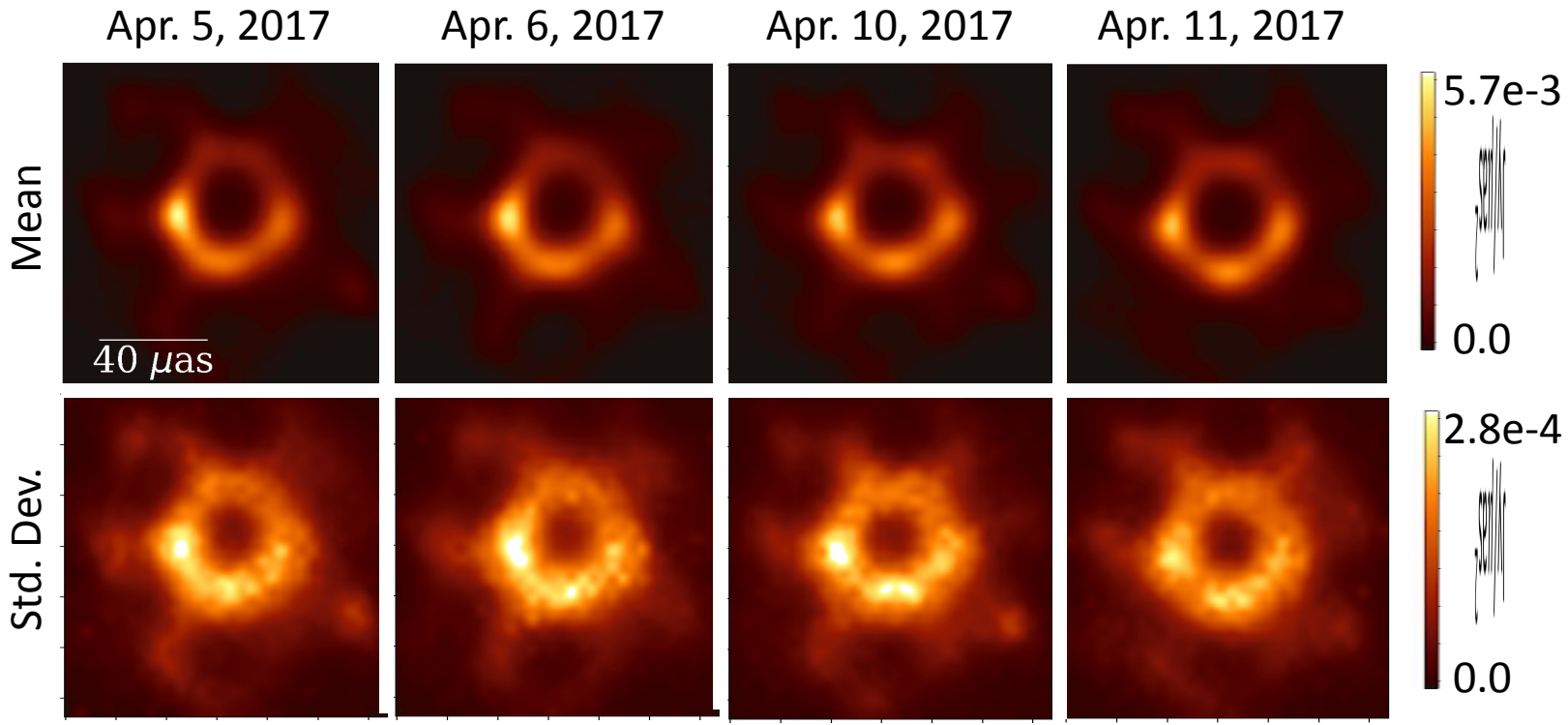







Slides from : He Sun, Katie Bouman

# Real M87 results: closure quantities + vis amp + MEM+TV2




Slides from : He Sun, Katie Bouman

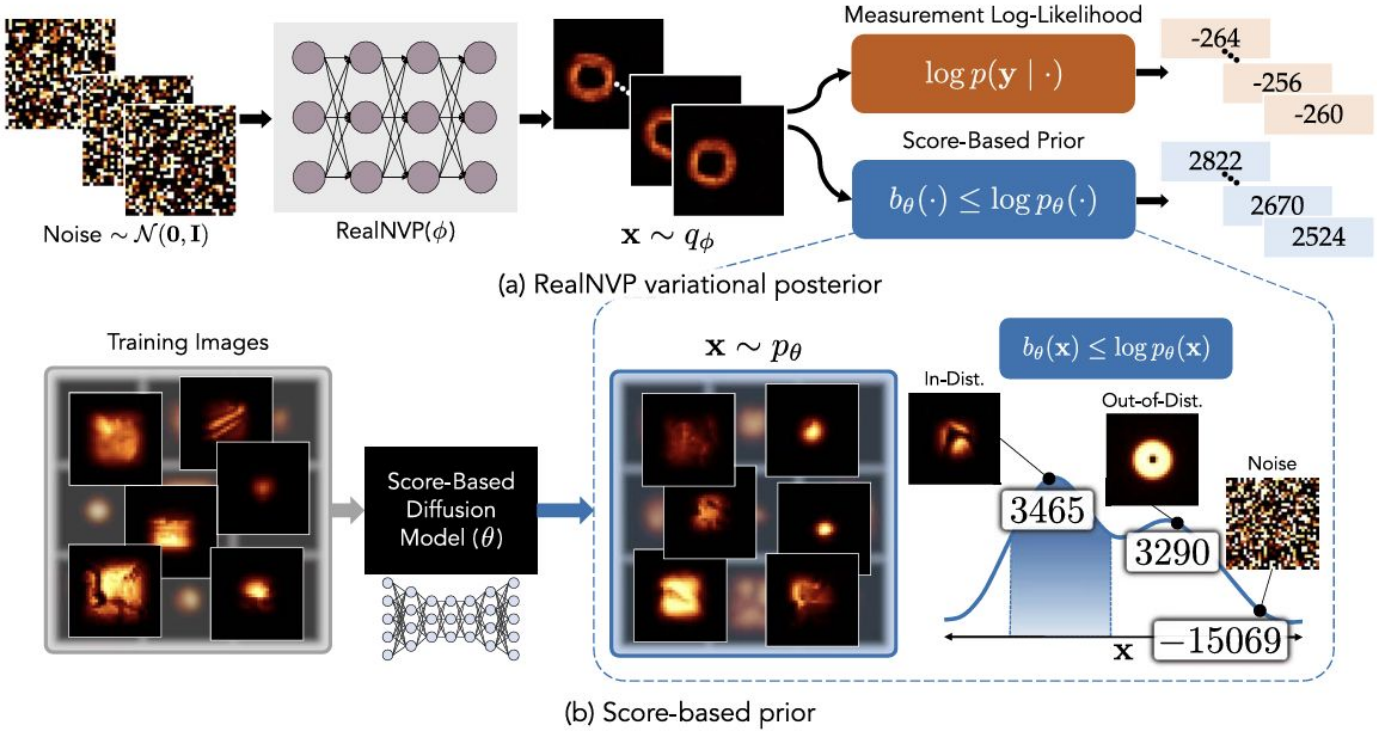
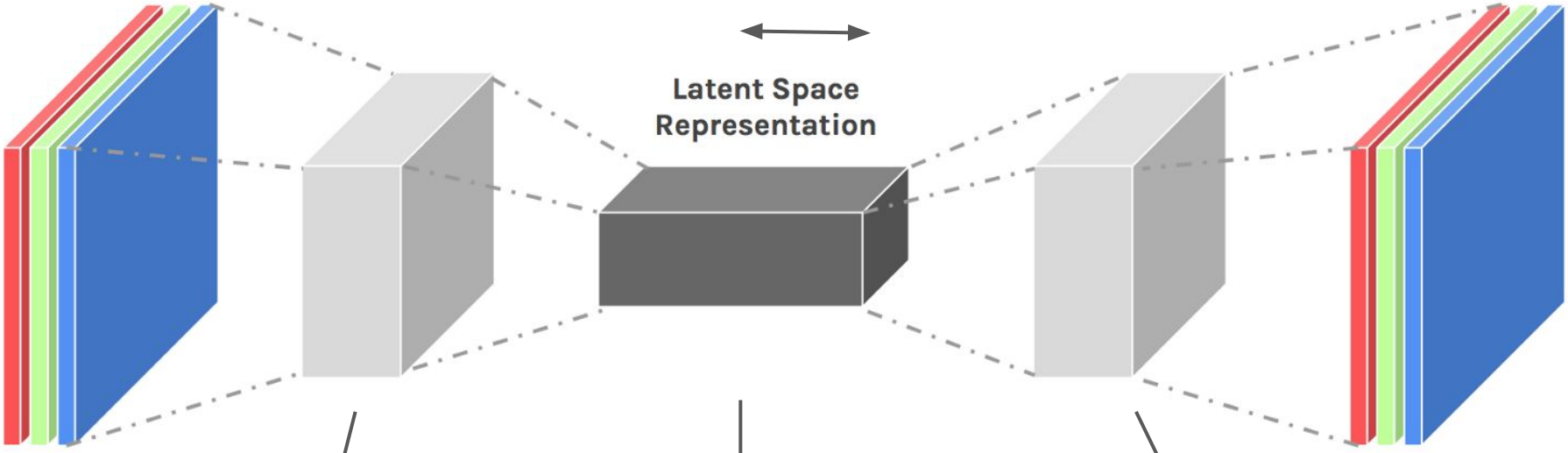
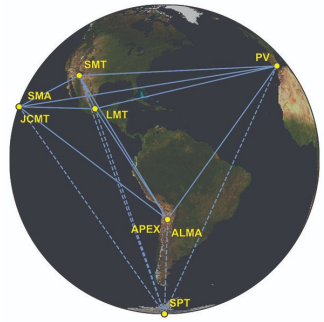


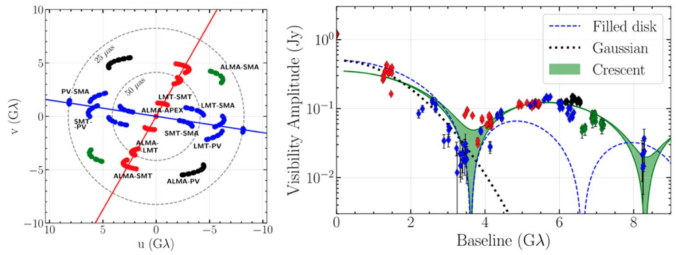

**Figure 1.** Method illustration. The CIFAR-10 prior was used for these examples; images are shown as 32 × 32 pixels on a [0, 1] scale. At a high level, we optimize a variational distribution $q_\phi$ to approximate the image posterior $p_\theta(\cdot \mid \boldsymbol{y})$ given a score-based prior $p_\theta$ and log likelihood based on EHT measurements. Panel (a) illustrates our particular variational distribution: a RealNVP with parameters $\phi$. At each optimization iteration $i$, the measurement log likelihood (Equation (2)) and the log density under the score-based prior of each sample $\boldsymbol{x}$ from $q_\phi = q_{\phi^{(i)}}$ are evaluated. The average gradient is computed with respect to $\phi$ to update $\phi^{(i)}$. In other words, $q_\phi$ is optimized to produce samples that have high probability under both measurement likelihood and prior. Panel (b) zooms in to the score-based prior. A score-based prior is based on a score-based diffusion model, a deep generative model with parameters $\theta$, that is trained on images from a target prior. Once trained, the diffusion model generates samples from a generative image distribution $p_\theta$. There is an analytical formula for computing the ELBO $b_\theta(\boldsymbol{x})$ of the log probability $\log p_\theta(\boldsymbol{x})$ for any image $\boldsymbol{x}$, even for out-of-distribution images and images of pure noise.






Radio Telescope config.

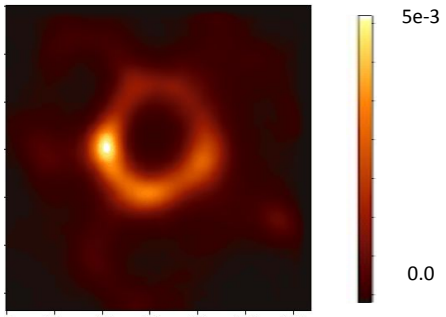



Visibility

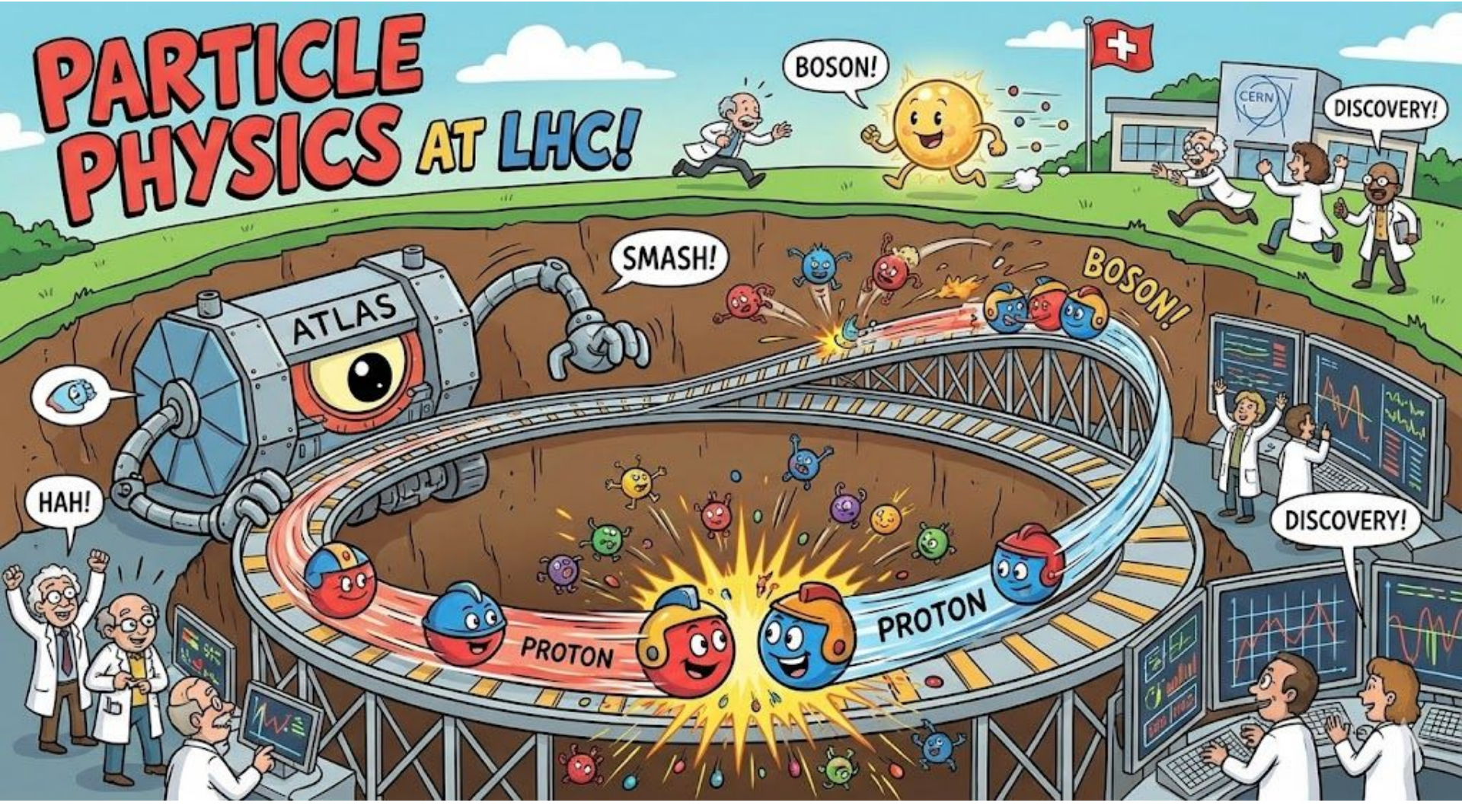
Image Reconstruction
Algo. (e.g. RML, CLEAN)

# Summary

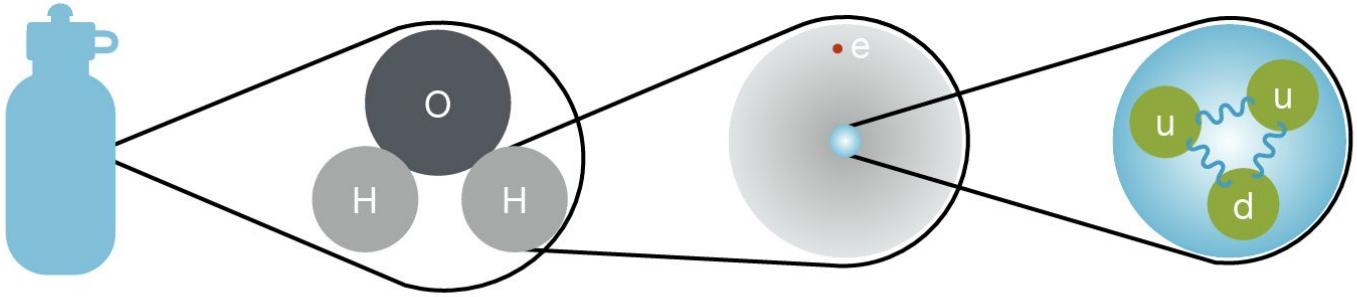
- Deep Probabilistic Imaging (DPI) using variational methods and generative model to approximate the posterior distribution of reconstructed image;
- DPI can capture multiple feasible solutions and quantify the uncertainty;
- DPI is preliminarily tested on EHT simulated data and EHT2017 M87 data.
- Generalizable to other parameter estimation problems



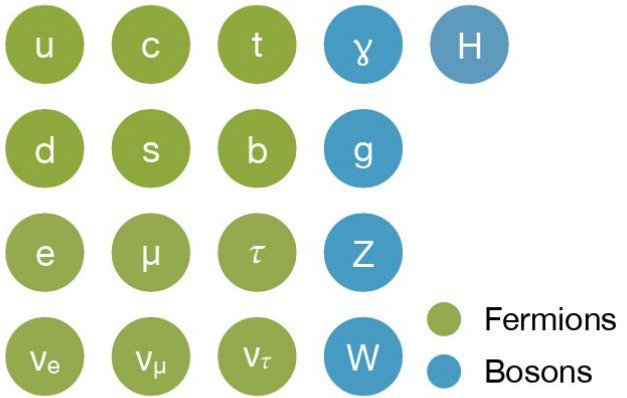

Slides from : He Sun, Katie Bouman


PARTICLE PHYSICS AT LHC!
BOSON!
CERN
DISCOVERY!
SMASH!
ATLAS
BOSON!
HAH!
PROTON
PROTON
DISCOVERY!

# Particle Physics in a Nutshell 1/2




~3x10$^{-1}$m ~3x10$^{-10}$m ~10$^{-10}$m ~10$^{-15}$m

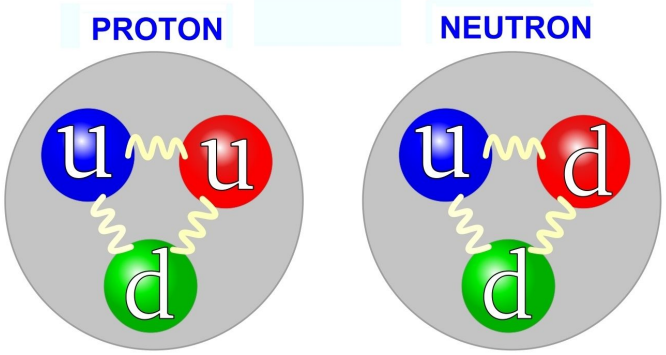



$$-\tfrac{1}{2}\partial_\nu g^a_\mu \partial_\nu g^a_\mu - g_s f^{abc}\partial_\mu g^a_\nu g^b_\mu g^c_\nu - \tfrac{1}{4} g_s^2 f^{abc} f^{ade} g^b_\mu g^c_\nu g^d_\mu g^e_\nu + \tfrac{1}{2} i g_s^2 (\bar{q}^\sigma_i \gamma^\mu q^\sigma_j) g^a_\mu + \bar{G}^a \partial^2 G^a + g_s f^{abc} \partial_\mu \bar{G}^a G^b g^c_\mu - \partial_\nu W^+_\mu \partial_\nu W^-_\mu - M^2 W^+_\mu W^-_\mu - \tfrac{1}{2}\partial_\nu Z^0_\mu \partial_\nu Z^0_\mu - \tfrac{1}{2c_w^2} M^2 Z^0_\mu Z^0_\mu - \tfrac{1}{2}\partial_\mu A_\nu \partial_\mu A_\nu - \tfrac{1}{2}\partial_\mu H \partial_\mu H - \tfrac{1}{2} m_h^2 H^2 - \partial_\mu \phi^+ \partial_\mu \phi^- - M^2 \phi^+ \phi^- - \tfrac{1}{2}\partial_\mu \phi^0 \partial_\mu \phi^0 - \tfrac{1}{2c_w^2} M \phi^0 \phi^0 - \beta_h [\tfrac{2M^2}{g^2} + \tfrac{2M}{g} H + \tfrac{1}{2}(H^2 + \phi^0 \phi^0 + 2\phi^+ \phi^-)] + \tfrac{2M^4}{g^2}\alpha_h - igc_w [\partial_\nu Z^0_\mu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - Z^0_\mu (W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + Z^0_\mu (W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)] - ig s_w [\partial_\nu A_\mu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - A_\nu (W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + A_\mu (W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)] - \tfrac{1}{2} g^2 W^+_\mu W^-_\mu W^+_\nu W^-_\nu + \tfrac{1}{2} g^2 W^+_\mu W^-_\nu W^+_\mu W^-_\nu + g^2 c_w^2 (Z^0_\mu W^+_\mu Z^0_\nu W^-_\nu - Z^0_\mu Z^0_\mu W^+_\nu W^-_\nu) + g^2 s_w^2 (A_\mu W^+_\mu A_\nu W^-_\nu - A_\mu A_\mu W^+_\nu W^-_\nu) + g^2 s_w c_w [A_\mu Z^0_\nu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - 2 A_\mu Z^0_\mu W^+_\nu W^-_\nu] - g\alpha [H^3 + H\phi^0\phi^0 + 2H\phi^+\phi^-] - \tfrac{1}{8} g^2 \alpha_h [H^4 + (\phi^0)^4 + 4(\phi^+\phi^-)^2 + 4(\phi^0)^2 \phi^+\phi^- + 4H^2\phi^+\phi^- + 2(\phi^0)^2 H^2] -$$

...

Slides Credit: Dennis Noll

[1]




Slides Credit: Dennis Noll

# Standard Model of Elementary Particles

three generations of matter (fermions)

interactions / force carriers (bosons)

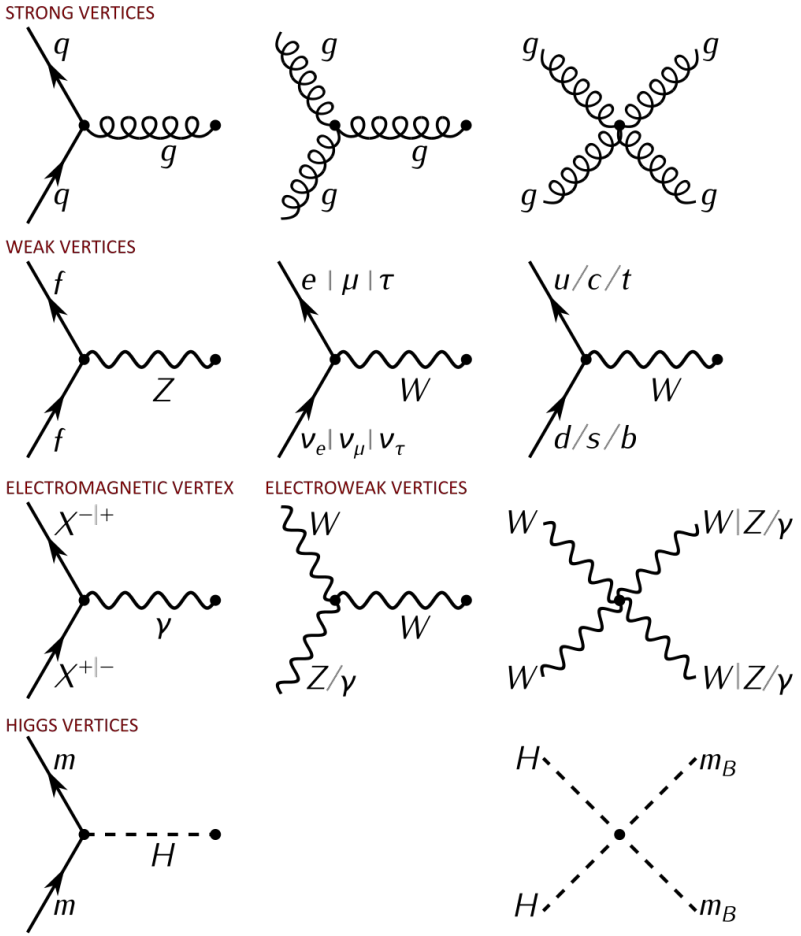
| | I | II | III | | |
|---|---|---|---|---|---|
| mass | ≈2.16 MeV/c² | ≈1.273 GeV/c² | ≈172.57 GeV/c² | 0 | ≈125.2 GeV/c² |
| charge | ⅔ | ⅔ | ⅔ | 0 | 0 |
| spin | ½ | ½ | ½ | 1 | 0 |
| | u up | c charm | t top | g gluon | H higgs |
| QUARKS | ≈4.7 MeV/c² | ≈93.5 MeV/c² | ≈4.183 GeV/c² | 0 | |
| | −⅓ | −⅓ | −⅓ | 0 | |
| | ½ | ½ | ½ | 1 | |
| | d down | s strange | b bottom | γ photon | |
| | ≈0.511 MeV/c² | ≈105.66 MeV/c² | ≈1.77693 GeV/c² | ≈91.188 GeV/c² | |
| | −1 | −1 | −1 | 0 | |
| | ½ | ½ | ½ | 1 | |
| | e electron | μ muon | τ tau | Z Z boson | |
| LEPTONS | <0.8 eV/c² | <0.17 MeV/c² | <18.2 MeV/c² | ≈80.3692 GeV/c² | |
| | 0 | 0 | 0 | ±1 | |
| | ½ | ½ | ½ | 1 | |
| | $\nu_e$ electron neutrino | $\nu_\mu$ muon neutrino | $\nu_\tau$ tau neutrino | W W boson | |

GAUGE BOSONS VECTOR BOSONS

SCALAR BOSONS

# Feynman Diagram

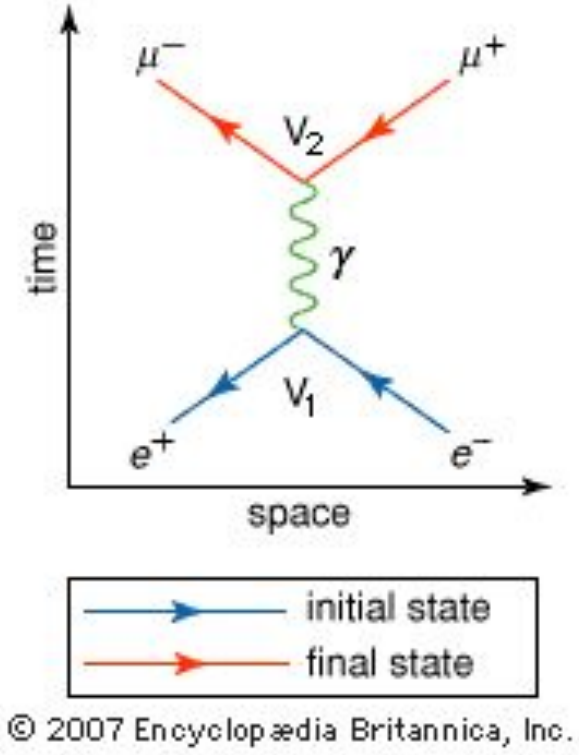
Richard Feynman



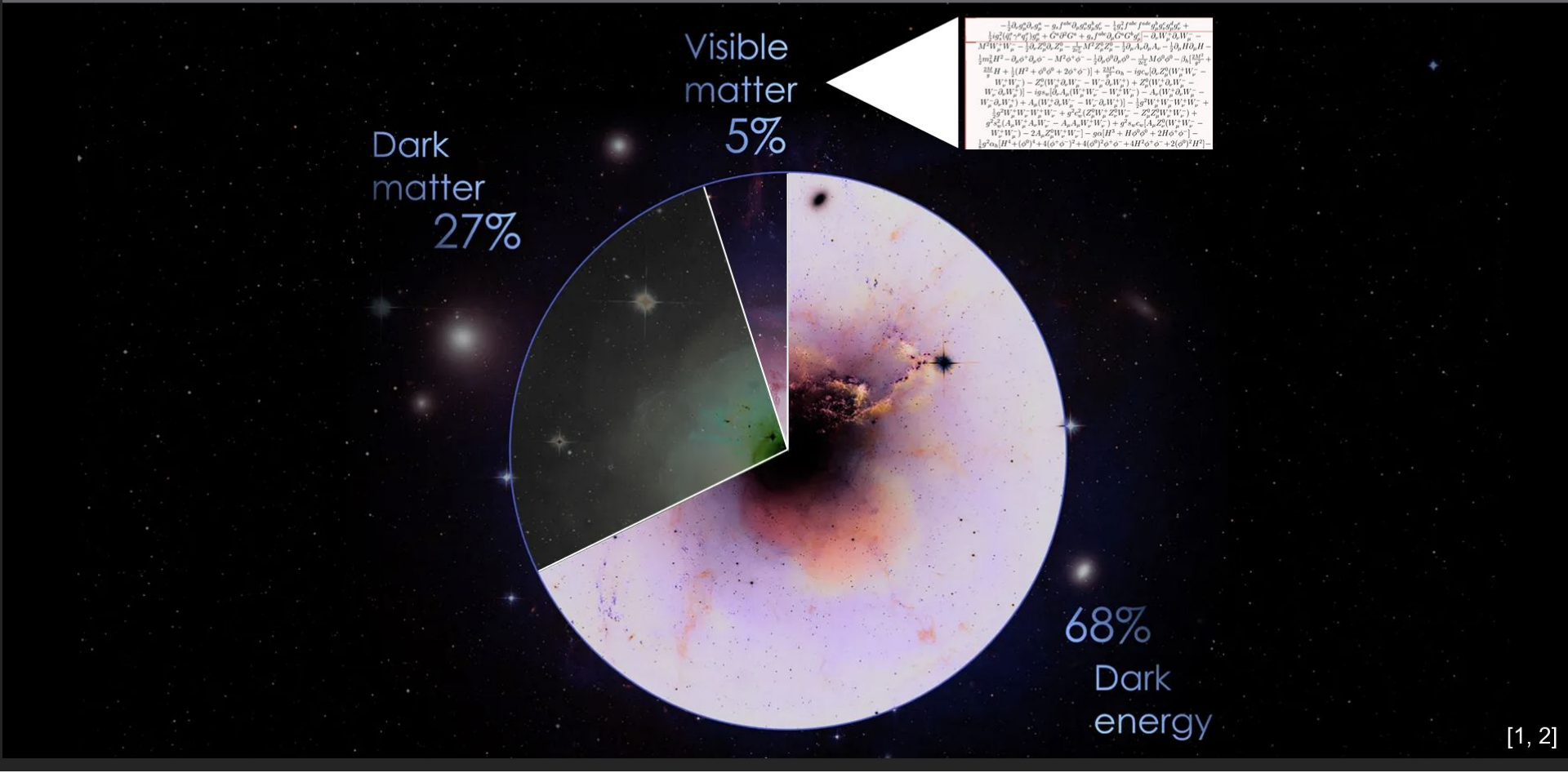
# Feynman Diagram 101

Richard Feynman

$$e^+ + e^- \longrightarrow \mu^- + \mu^+$$

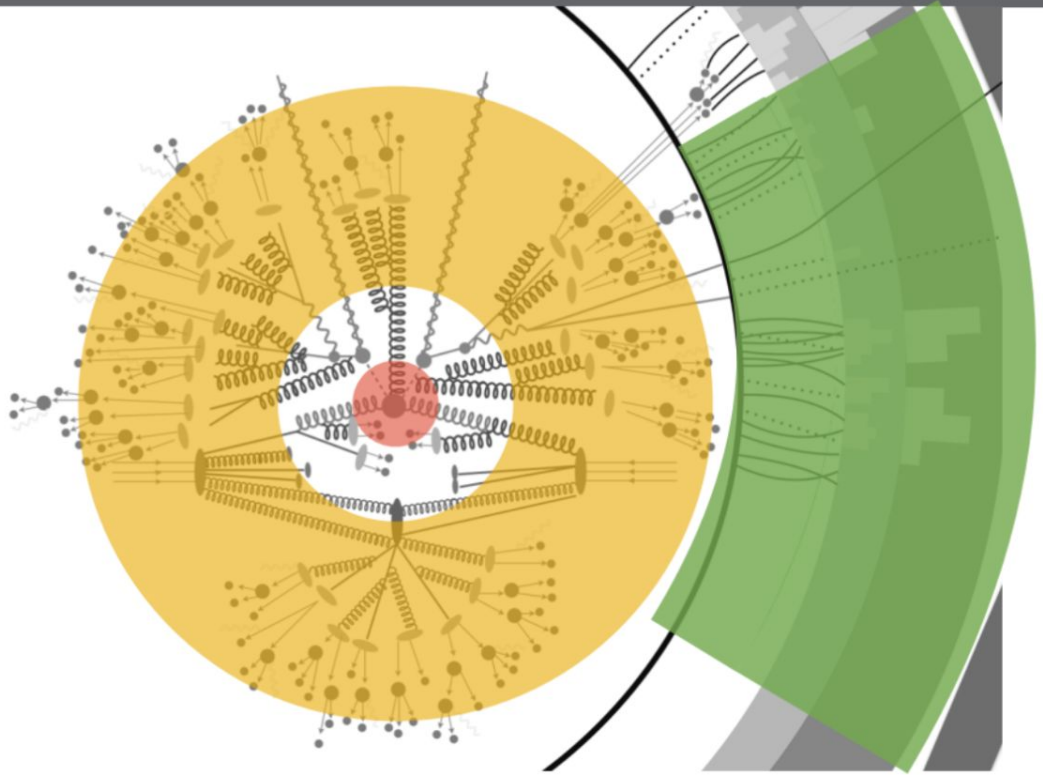


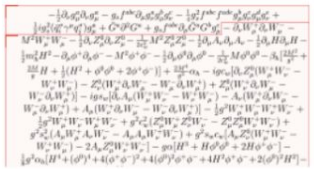
# Particle Physics in a Nutshell 2/2




[1, 2]

# Very Precise Simulations Available … But Expensive!





Hard Process

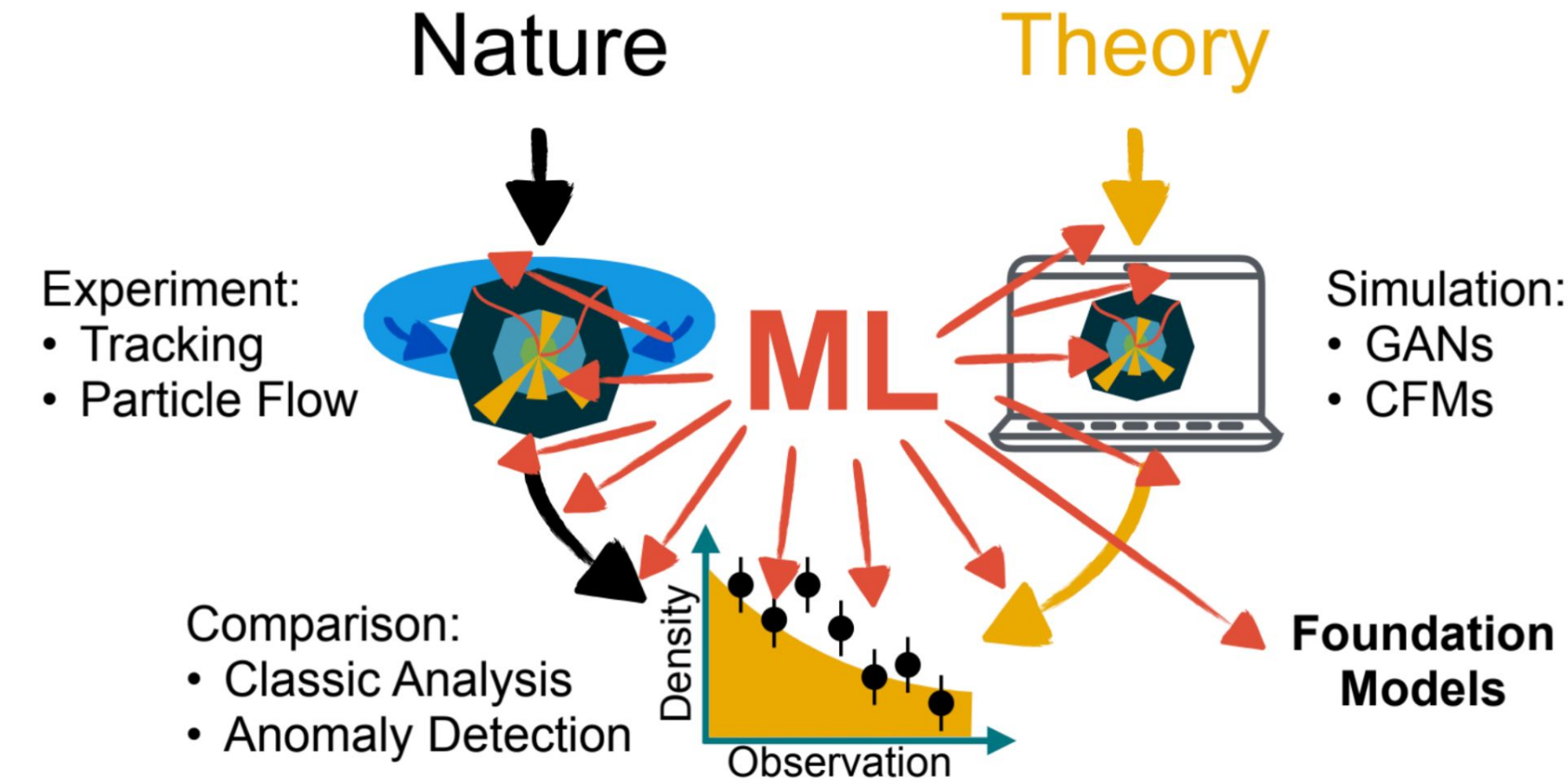
Showering & Hadronization

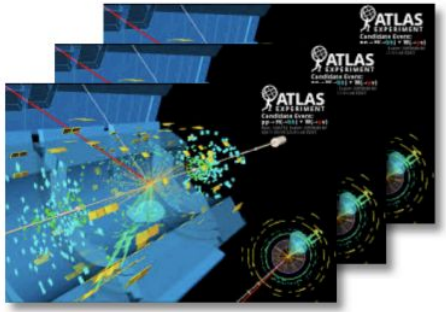
Detector Interaction

~100s/event

>99% of time

[1]

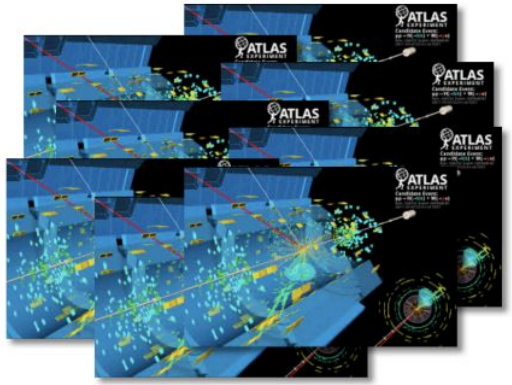

Nature
Theory
Experiment:
• Tracking
• Particle Flow
ML
Simulation:
• GANs
• CFMs
Comparison:
• Classic Analysis
• Anomaly Detection
Density
Observation
Foundation Models



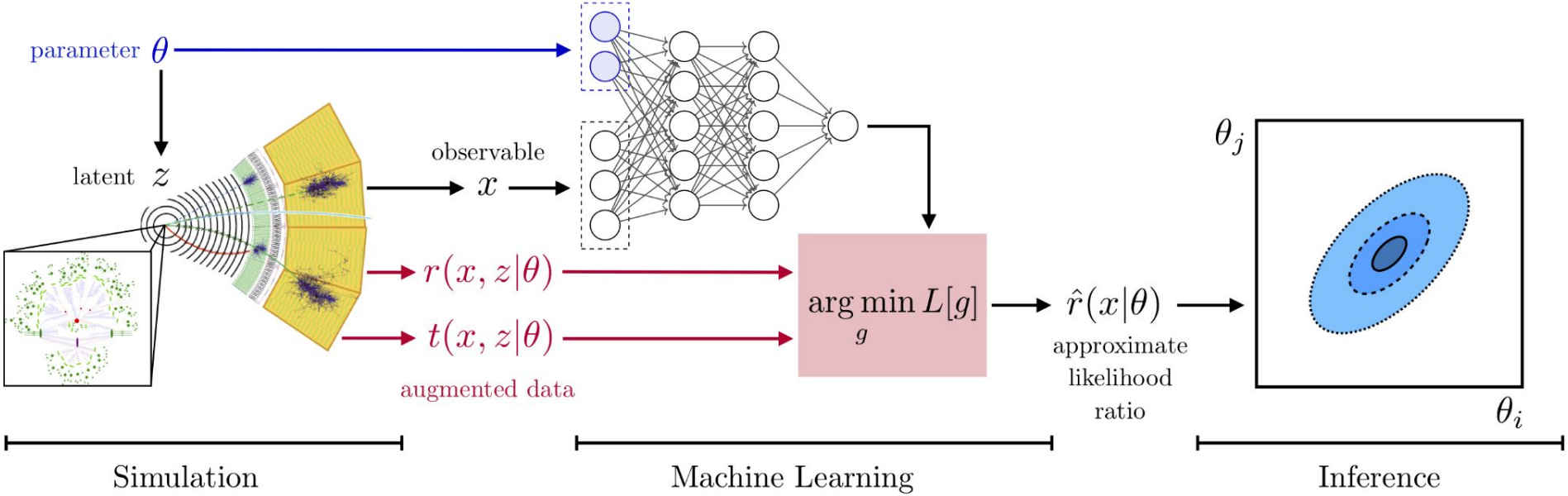
Simulation /
Recorded Data

## Generative ML Model

- GANs
- Variational AEs
- Normalizing Flows
- Diffusion Models
- …

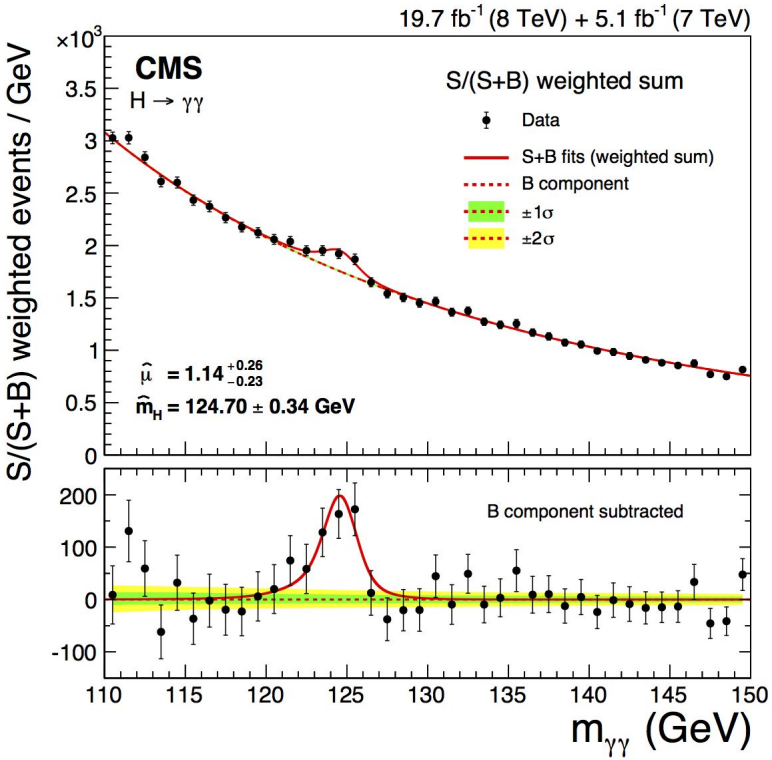


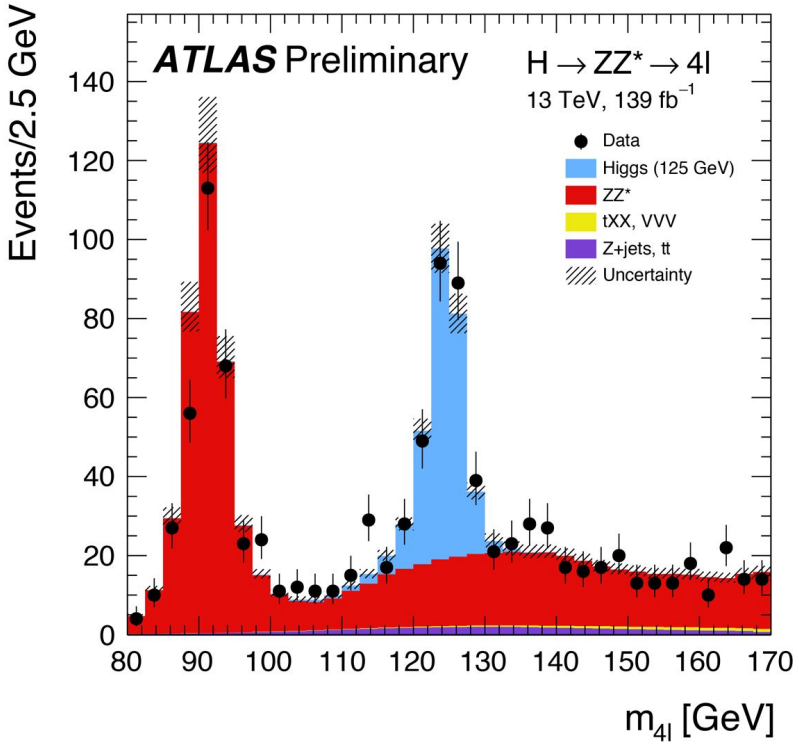
Oversampled /
Interpolated




K. Cranmer, J. Brehmer and G. Louppe

# What analysis looks like…



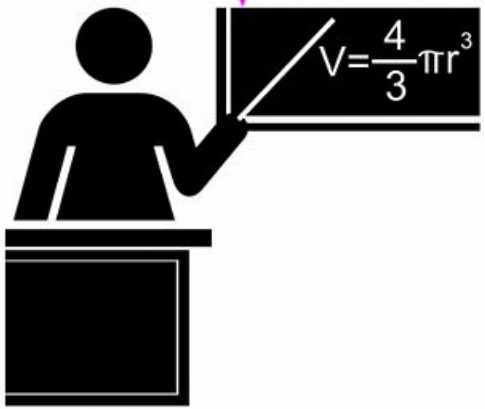



# AI for Math

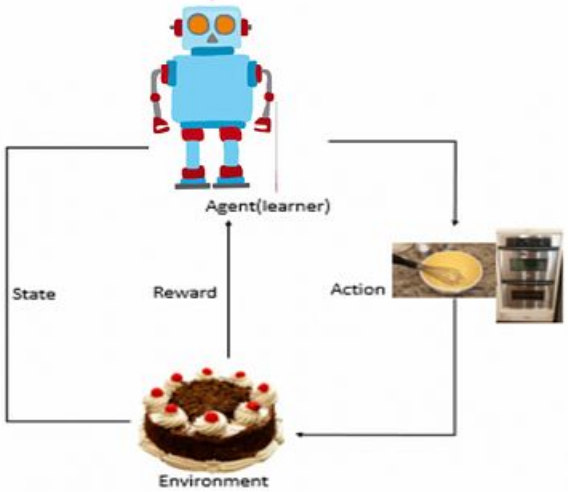
$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{n=0}^{\infty} \frac{(4n)!\ (1103 + 26390\mathrm{n})}{(\mathrm{n}!)^4 396^{4n}}$$

$$\pi = \frac{9801}{1103\sqrt{8}}$$

# Machine Learning





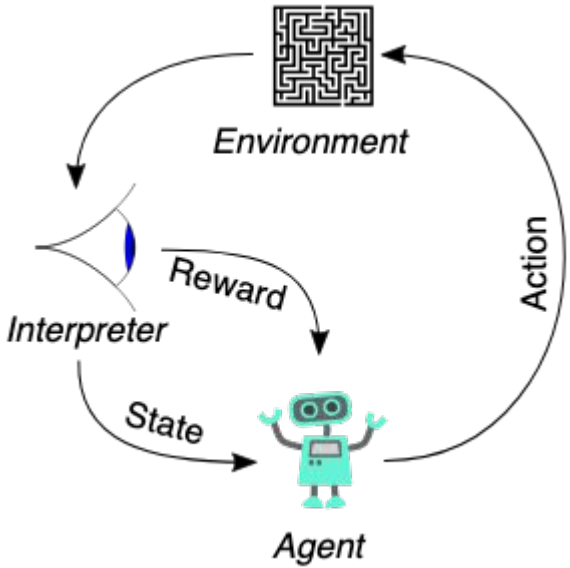



Supervised Learning

Unsupervised Learning

Reinforcement Learning

# What’s Reinforcement Learning

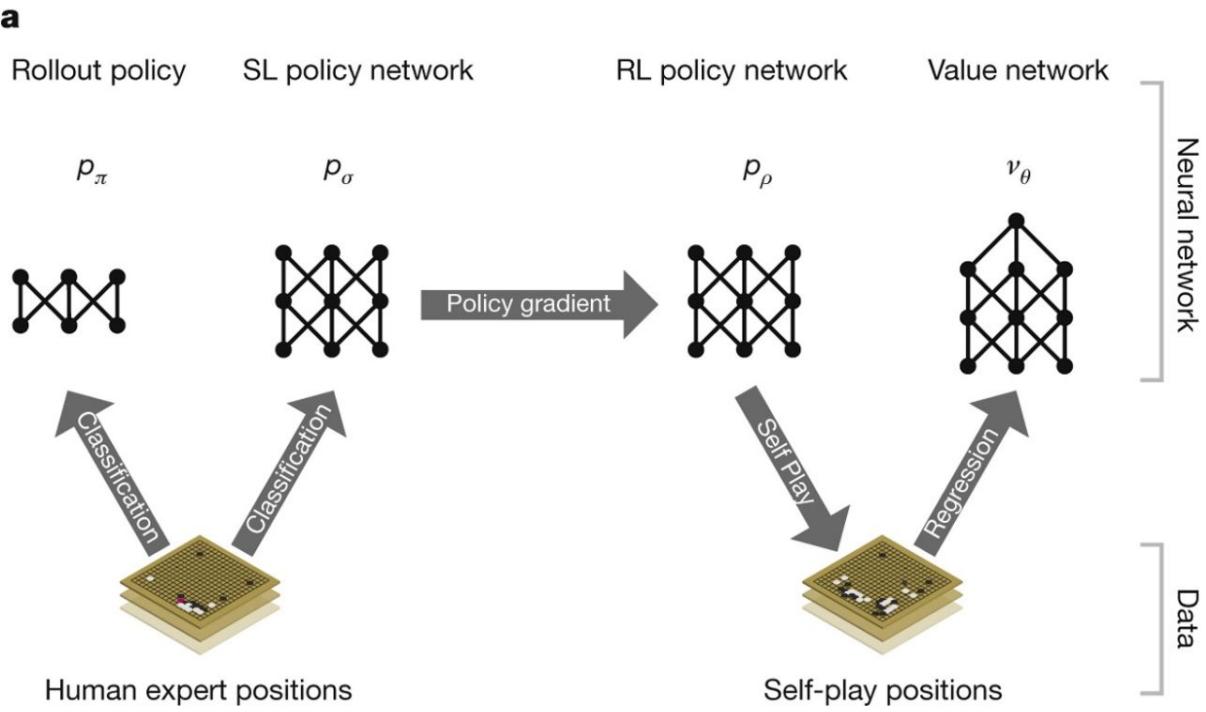


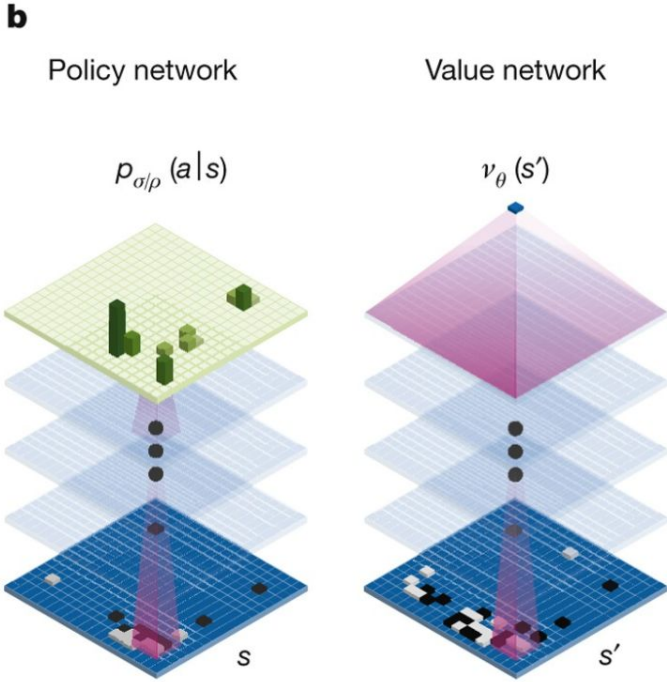
# Successful example of Reinforcement Learning

AlphaGo

LLM with CoT +
Reinforcement Learning

# AlphaGo Fan/Lee (2016)




[Mastering the game of Go with deep neural networks and tree search](#) (DeepMind, Nature 2016)

# Learning from Human Expert Move

# Generating new self-play to prevent overfitting in RL

## Reinforcement learning of value networks

The final stage of the training pipeline focuses on position evaluation, estimating a value function $v^p(s)$ that predicts the outcome from position $s$ of games played by using policy $p$ for both players[28–30]

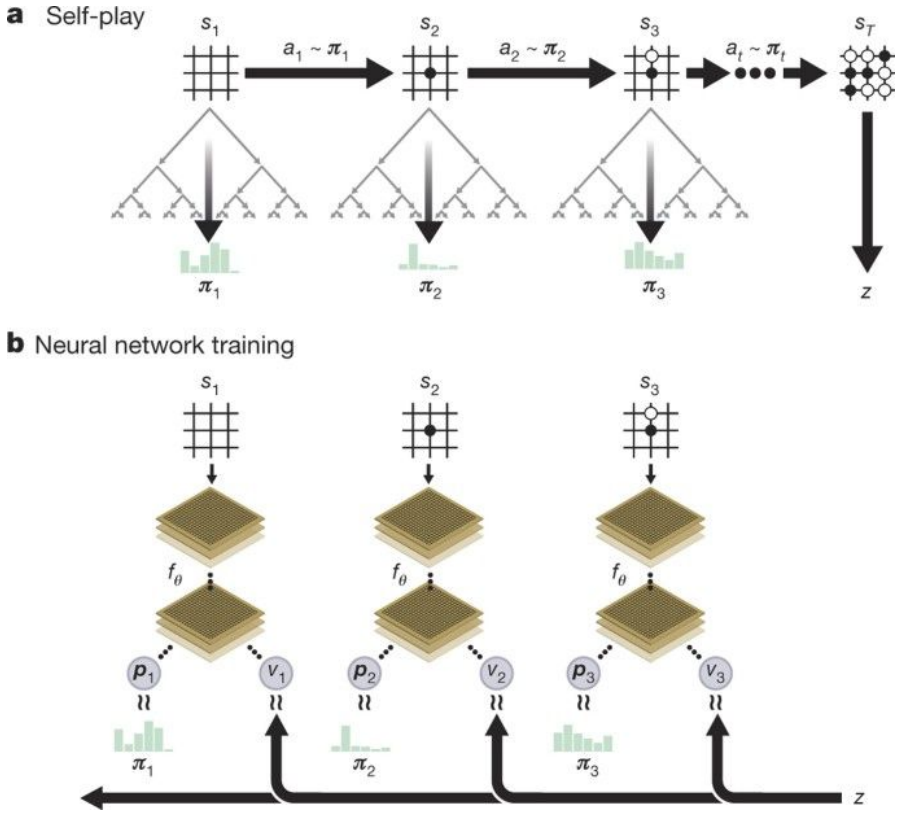
$$v^p(s) = \mathbb{E}[z_t | s_t = s,\ a_{t\ldots T} \sim p]$$

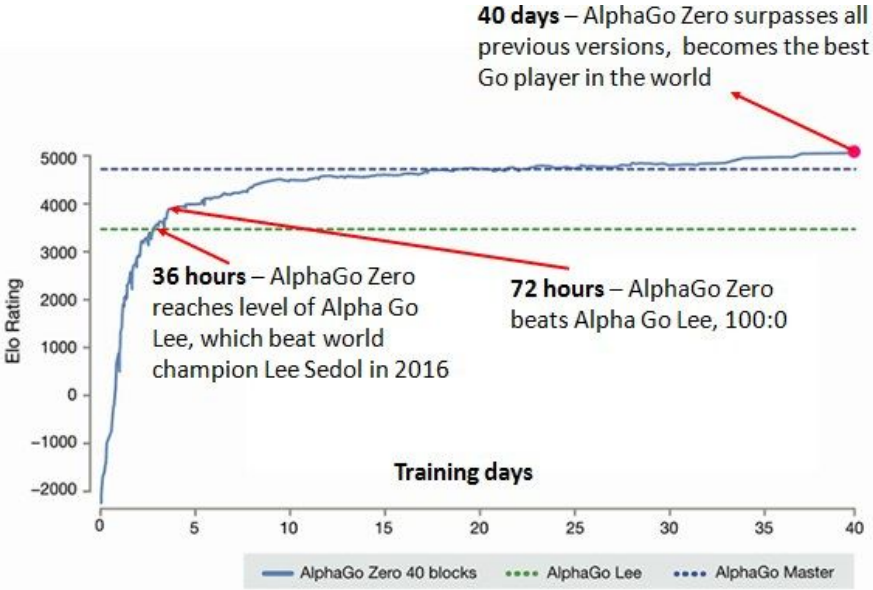
Ideally, we would like to know the optimal value function under perfect play $v^*(s)$; in practice, we instead estimate the value function $v^{p_\rho}$ for our strongest policy, using the RL policy network $p_\rho$. We approximate the value function using a value network $v_\theta(s)$ with weights $\theta$, $v_\theta(s) \approx v^{p_\rho}(s) \approx v^*(s)$. This neural network has a similar architecture to the policy network, but outputs a single prediction instead of a probability distribution. We train the weights of the value network by regression on state-outcome pairs $(s, z)$, using stochastic gradient descent to minimize the mean squared error (MSE) between the predicted value $v_\theta(s)$, and the corresponding outcome $z$

$$\Delta\theta \propto \frac{\partial v_\theta(s)}{\partial \theta}(z - v_\theta(s))$$

The naive approach of predicting game outcomes from data consisting of complete games leads to overfitting. The problem is that successive positions are strongly correlated, differing by just one stone, but the regression target is shared for the entire game. When trained on the KGS data set in this way, the value network memorized the game outcomes rather than generalizing to new positions, achieving a minimum MSE of 0.37 on the test set, compared to 0.19 on the training set. To mitigate this problem, we generated a new self-play data set consisting of 30 million distinct positions, each sampled from a separate game. Each game was played between the RL policy network and itself until the game terminated. Training on this data set led to MSEs of 0.226 and 0.234 on the training and test set respectively, indicating minimal overfitting. Figure 2b shows the position evaluation accuracy of the value network, compared to Monte Carlo rollouts using the fast rollout policy $p_\pi$; the value function was consistently more accurate. A single evaluation of $v_\theta(s)$ also approached the accuracy of Monte Carlo rollouts using the RL policy network $p_\rho$, but using 15,000 times less computation.

# AlphaGO Zero (2017)




[Mastering the game of Go without human knowledge](#) (DeepMind, 2017)

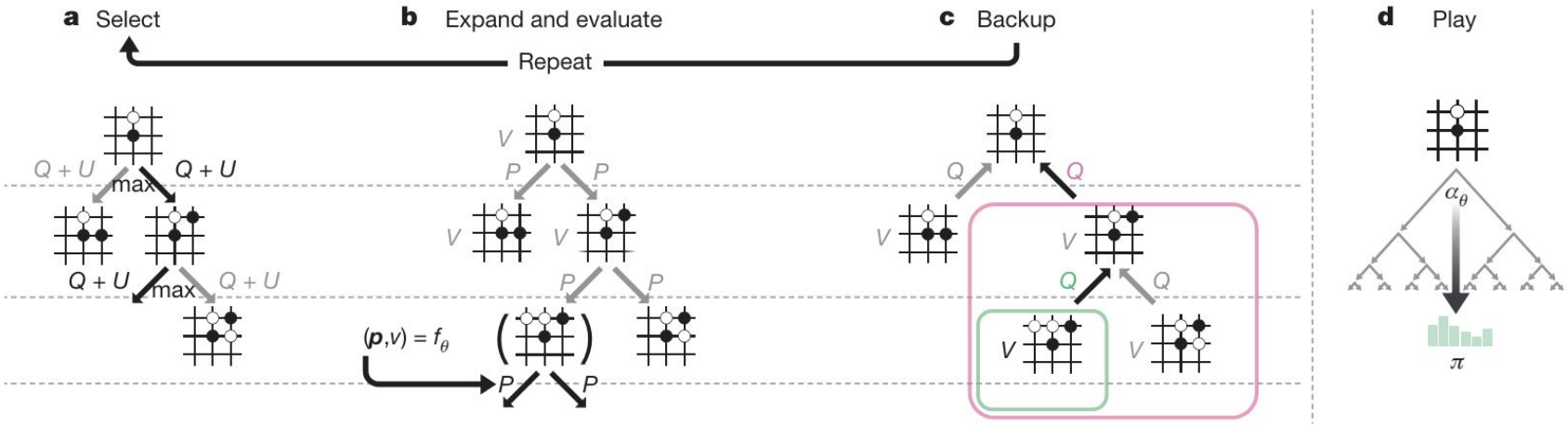



**Figure 2 | MCTS in AlphaGo Zero. a**, Each simulation traverses the tree by selecting the edge with maximum action value $Q$, plus an upper confidence bound $U$ that depends on a stored prior probability $P$ and visit count $N$ for that edge (which is incremented once traversed). **b**, The leaf node is expanded and the associated position $s$ is evaluated by the neural network $(P(s, \cdot), V(s)) = f_\theta(s)$; the vector of $P$ values are stored in the outgoing edges from $s$. **c**, Action value $Q$ is updated to track the mean of all evaluations $V$ in the subtree below that action. **d**, Once the search is complete, search probabilities $\boldsymbol{\pi}$ are returned, proportional to $N^{1/\tau}$, where $N$ is the visit count of each move from the root state and $\tau$ is a parameter controlling temperature.

**Self-play.** The best current player $\alpha_{\theta_*}$, as selected by the evaluator, is used to generate data. In each iteration, $\alpha_{\theta_*}$ plays 25,000 games of self-play, using 1,600 simulations of MCTS to select each move (this requires approximately 0.4 s per search). For the first 30 moves of each game, the temperature is set to $\tau = 1$; this selects moves proportionally to their visit count in MCTS, and ensures a diverse set of positions are encountered. For the remainder of the game, an infinitesimal temperature is used, $\tau \rightarrow 0$. Additional exploration is achieved by adding Dirichlet

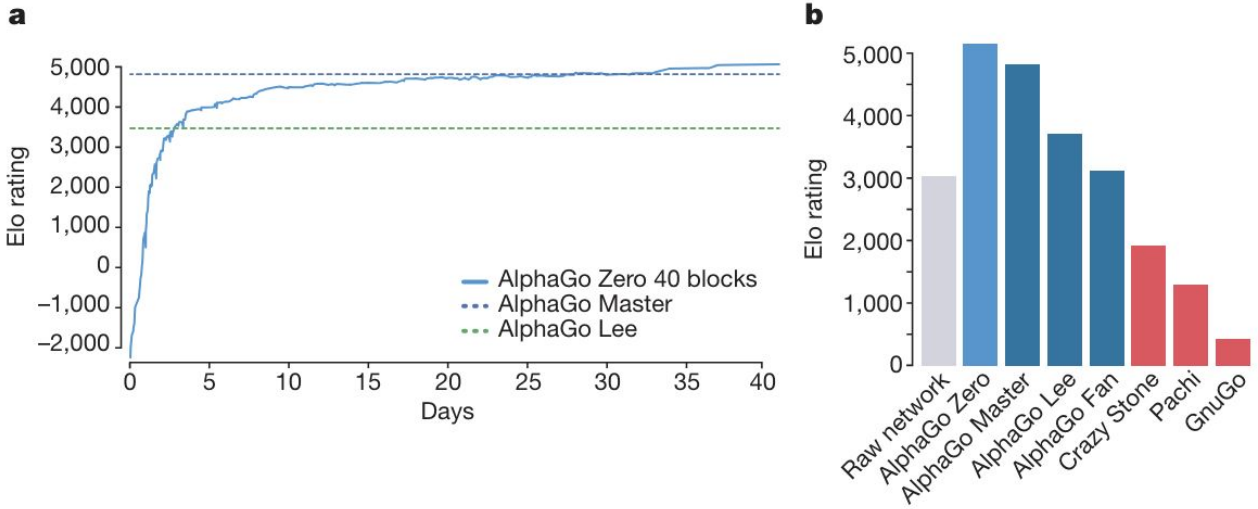



**Figure 6 | Performance of AlphaGo Zero. a**, Learning curve for AlphaGo Zero using a larger 40-block residual network over 40 days. The plot shows the performance of each player $\alpha_{\theta_i}$ from each iteration $i$ of our reinforcement learning algorithm. Elo ratings were computed from evaluation games between different players, using 0.4 s per search (see Methods). **b**, Final performance of AlphaGo Zero. AlphaGo Zero was trained for 40 days using a 40-block residual neural network. The plot shows the results of a tournament between: AlphaGo Zero, AlphaGo Master (defeated top human professionals 60–0 in online games), AlphaGo Lee (defeated Lee Sedol), AlphaGo Fan (defeated Fan Hui), as well as previous Go programs Crazy Stone, Pachi and GnuGo. Each program was given 5 s of thinking time per move. AlphaGo Zero and AlphaGo Master played on a single machine on the Google Cloud; AlphaGo Fan and AlphaGo Lee were distributed over many machines. The raw neural network from AlphaGo Zero is also included, which directly selects the move $a$ with maximum probability $p_a$, without using MCTS. Programs were evaluated on an Elo scale[25]: a 200-point gap corresponds to a 75% probability of winning.

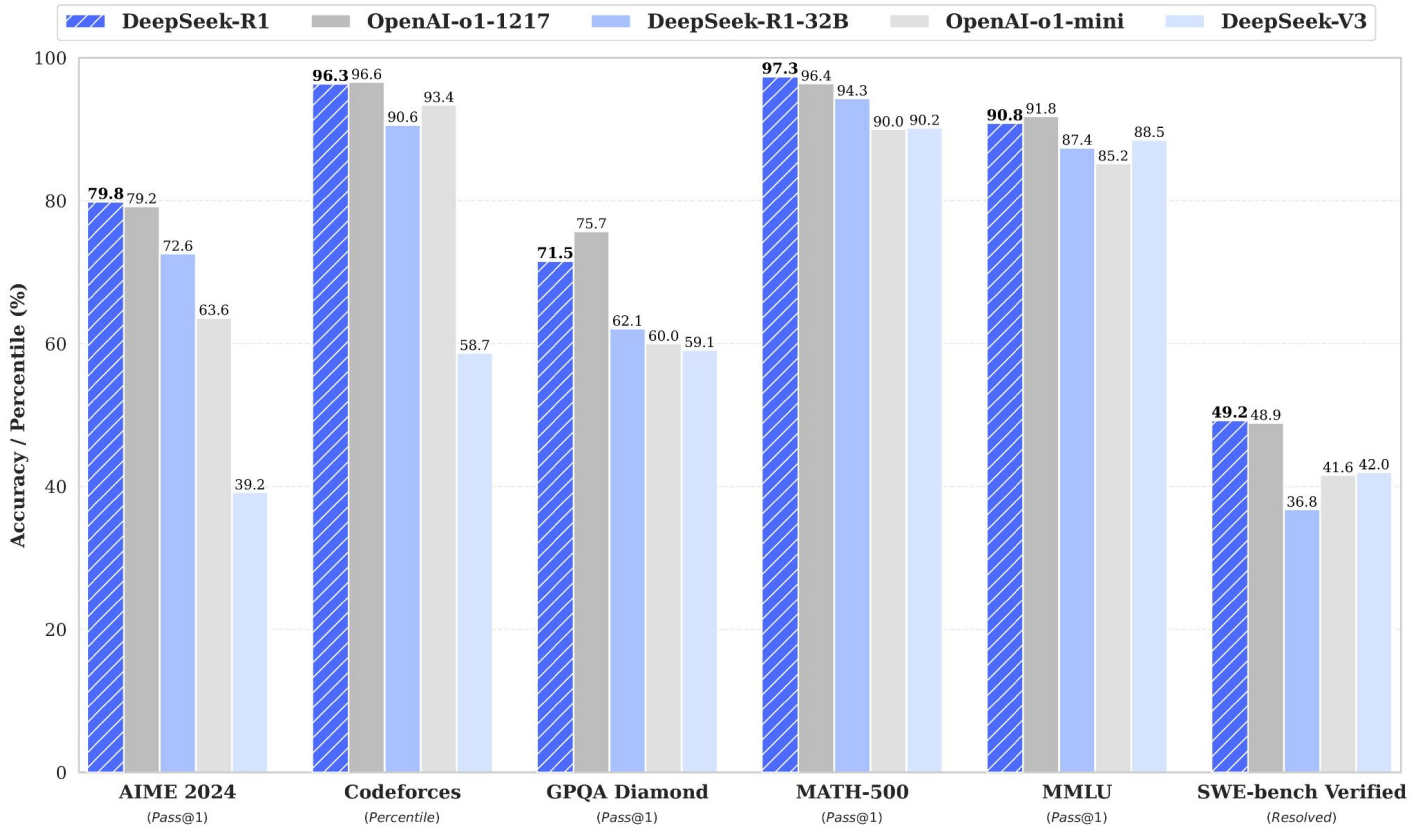
# DeepSeek R1

**Summary of DeepSeek-R1:**

- **Overview**: Launched in January 2025 by Chinese AI startup DeepSeek, **DeepSeek-R1 is an open-source large language model** excelling in advanced reasoning tasks like math, coding, and logic, competing with OpenAI's o1.
- **Technology**: Built on the 671-billion-parameter DeepSeek-V3-Base, it uses **reinforcement learning (RL)** with minimal supervised fine-tuning (SFT) and **Group Relative Policy Optimization (GRPO)** for efficient training, achieving high performance (e.g., 79.8% on AIME, 97.3% on MATH-500).
- **Cost Efficiency**: Trained for ~$6 million using ~2,000 Nvidia H800 chips, far less than the $100 million–$1 billion spent by U.S. competitors. API pricing is significantly lower ($0.55/million input tokens vs. OpenAI's $15).
- **Accessibility**: Available under MIT License, supporting commercial use and model distillation, with six smaller distilled models (1.5B–70B parameters). Powers DeepSeek's chatbot via web, app, and API.
- **Performance**: Outperforms or matches U.S. models like OpenAI's o1 and Meta's Llama in benchmarks, with a top-five ranking on Chatbot Arena.

**Impact on Nvidia Stock (January–May 2025):**

- **Initial Market Shock**: DeepSeek-R1's release on January 20, 2025, triggered a 17% drop in Nvidia's stock on January 27, erasing ~$593 billion in market value, the largest single-day loss in Wall Street history. The cost-efficient model raised fears of reduced demand for Nvidia's high-end GPUs


DeepSeek-R1
OpenAI-o1-1217
DeepSeek-R1-32B
OpenAI-o1-mini
DeepSeek-V3
Accuracy / Percentile (%)
0
20
40
60
80
100
79.8
79.2
72.6
63.6
39.2
96.3
96.6
90.6
93.4
58.7
71.5
75.7
62.1
60.0
59.1
97.3
96.4
94.3
90.0
90.2
90.8
91.8
87.4
85.2
88.5
49.2
48.9
36.8
41.6
42.0
AIME 2024
(Pass@1)
Codeforces
(Percentile)
GPQA Diamond
(Pass@1)
MATH-500
(Pass@1)
MMLU
(Pass@1)
SWE-bench Verified
(Resolved)

### 2.2.1. *Reinforcement Learning Algorithm*

Core technology of DeepSeek

**Group Relative Policy Optimization** In order to save the training costs of RL, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question $q$, GRPO samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model $\pi_\theta$ by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$

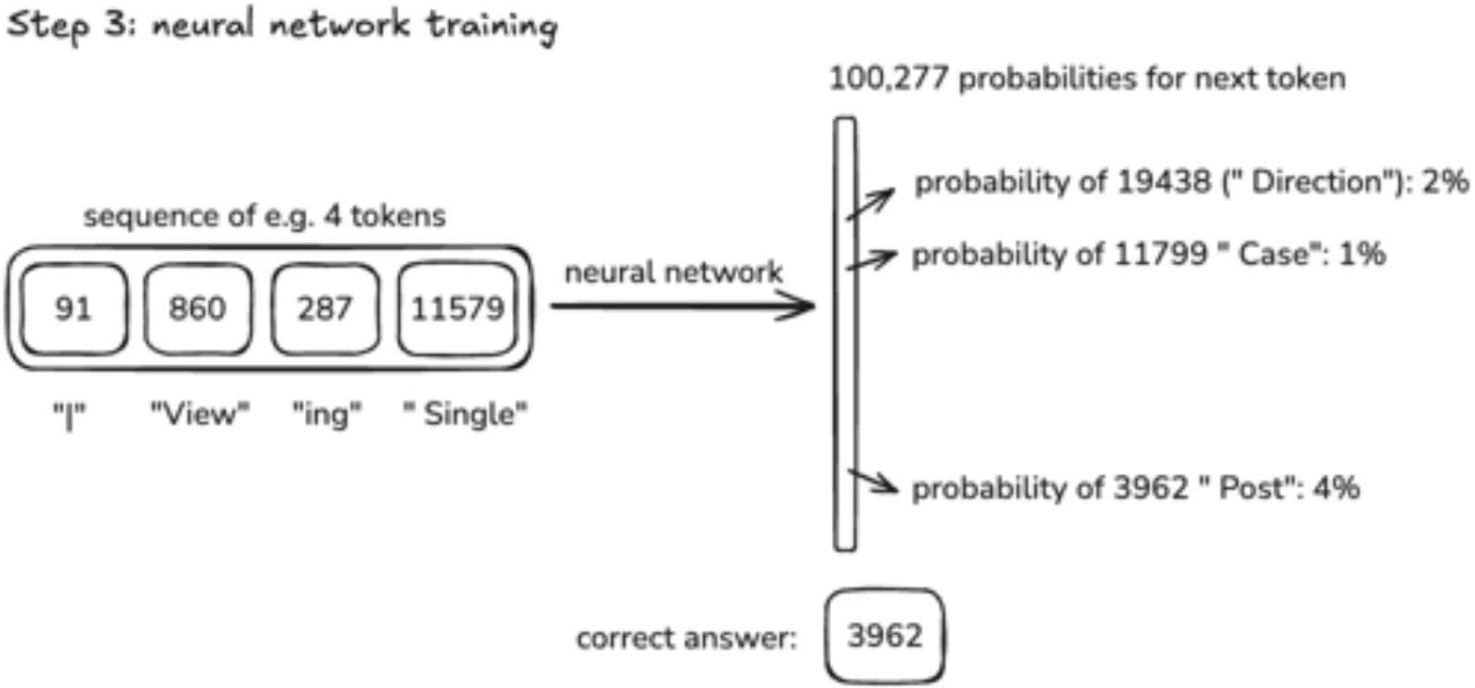
$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right), \tag{1}$$

$$\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right) = \frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{2}$$

where $\varepsilon$ and $\beta$ are hyper-parameters, and $A_i$ is the advantage, computed using a group of rewards $\{r_1, r_2, \ldots, r_G\}$ corresponding to the outputs within each group:

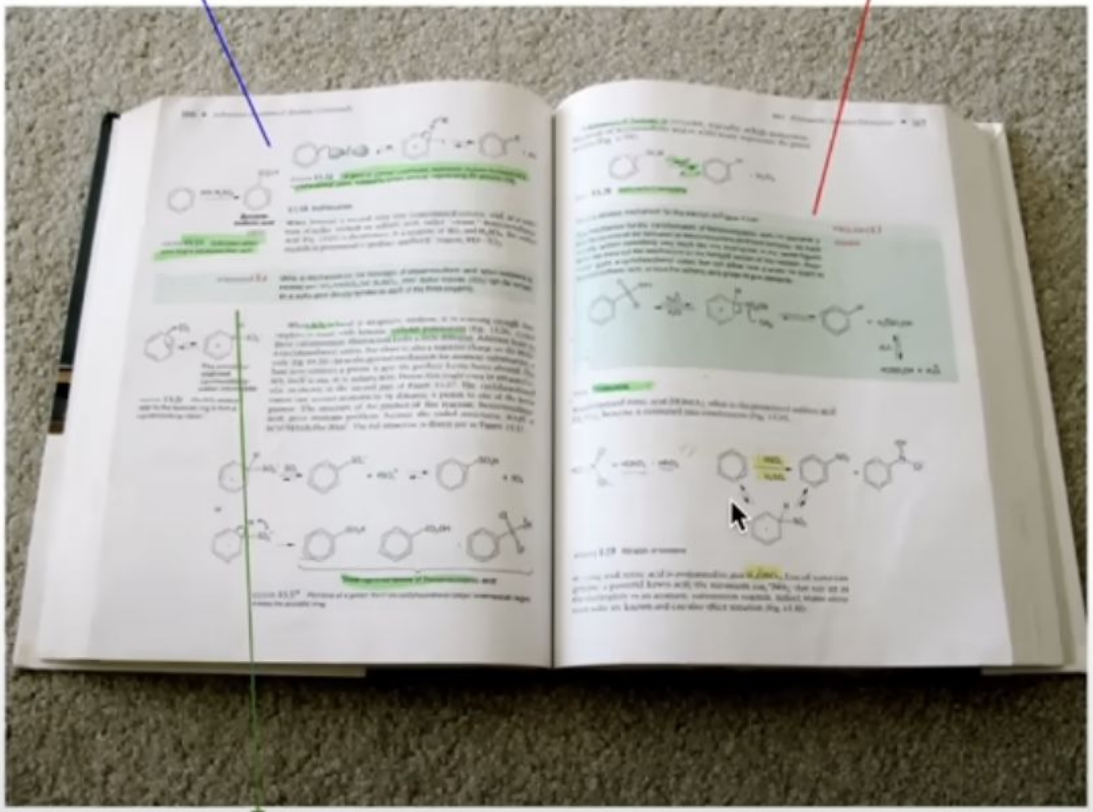
$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{std(\{r_1, r_2, \cdots, r_G\})}. \tag{3}$$

# Next token generation as policy

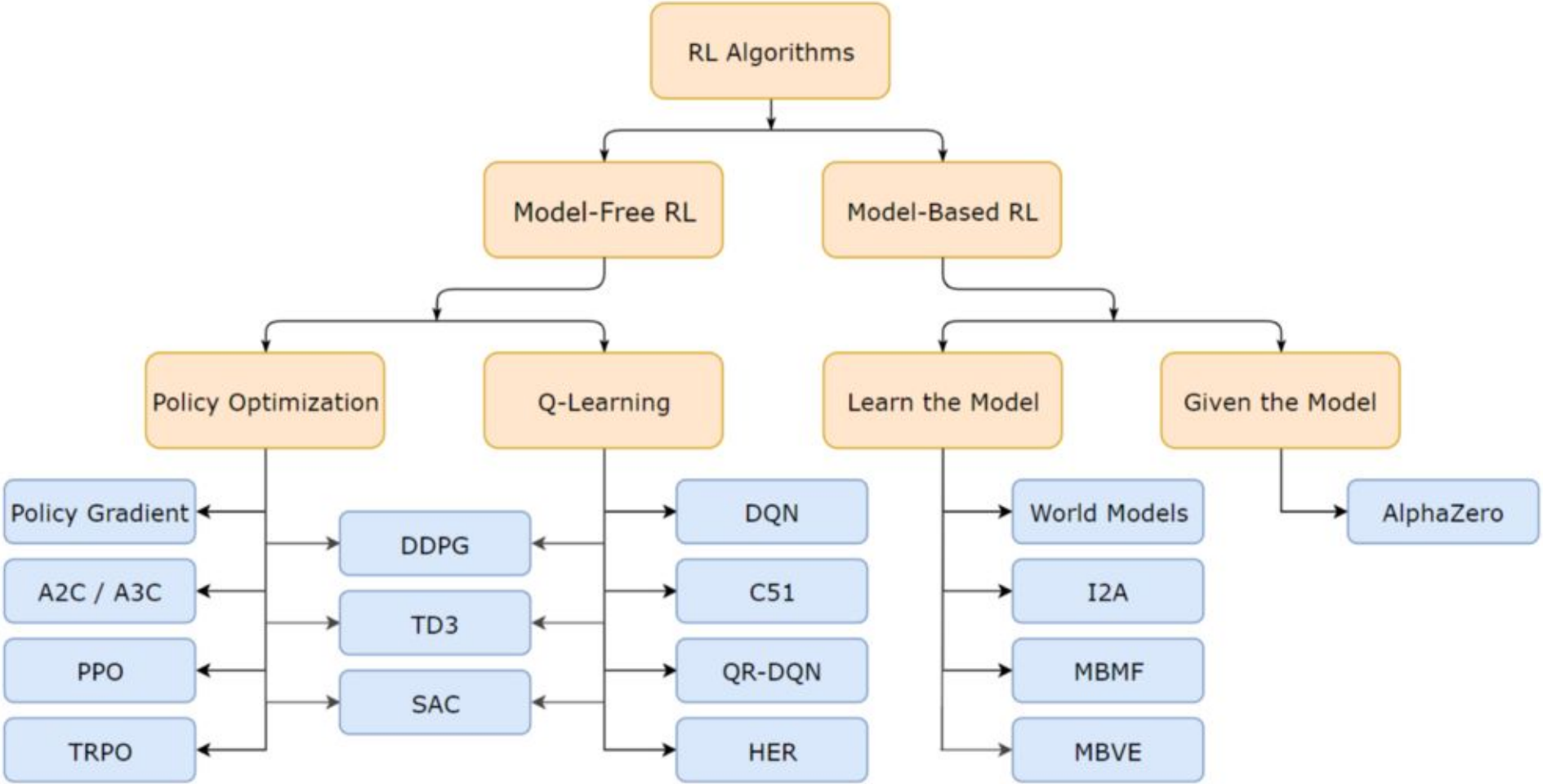


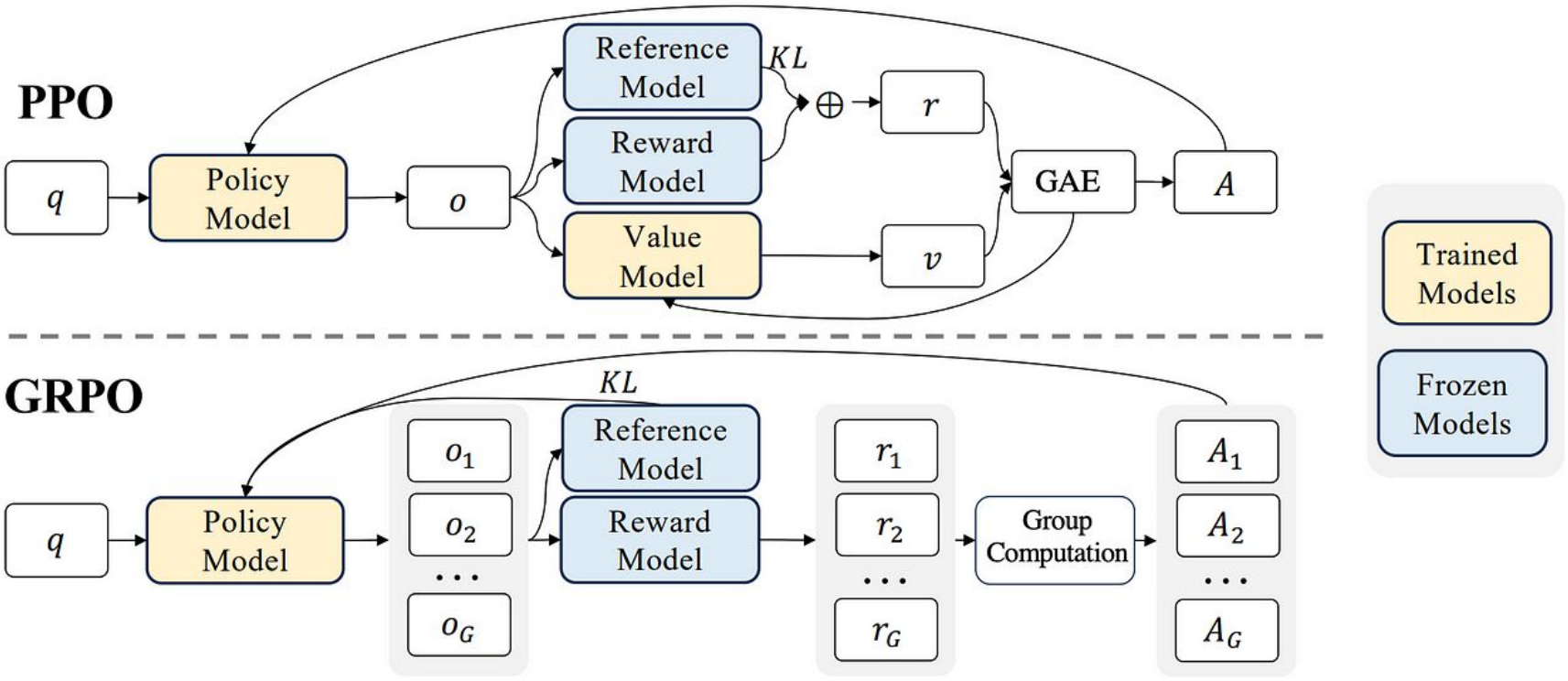
exposition ⇔ pretraining
(background knowledge)

worked problems ⇔ supervised finetuning
(problem + demonstrated solution, for imitation)



practice problems ⇔ reinforcement learning
(prompts to practice, trial & error until you reach the correct answer)


RL Algorithms
Model-Free RL
Model-Based RL
Policy Optimization
Q-Learning
Learn the Model
Given the Model
Policy Gradient
A2C / A3C
PPO
TRPO
DDPG
TD3
SAC
DQN
C51
QR-DQN
HER
World Models
I2A
MBMF
MBVE
AlphaZero


PPO
q
Policy Model
o
Reference Model
Reward Model
Value Model
KL
⊕
r
v
GAE
A
GRPO
KL
q
Policy Model
$o_1$
$o_2$
...
$o_G$
Reference Model
Reward Model
$r_1$
$r_2$
...
$r_G$
Group Computation
$A_1$
$A_2$
...
$A_G$
Trained Models
Frozen Models

### 2.2.1. *Reinforcement Learning Algorithm*

**Group Relative Policy Optimization** In order to save the training costs of RL, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question $q$, GRPO samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model $\pi_\theta$ by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$

$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right), \tag{1}$$

$$\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right) = \frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{2}$$

where $\varepsilon$ and $\beta$ are hyper-parameters, and $A_i$ is the advantage, computed using a group of rewards $\{r_1, r_2, \ldots, r_G\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})}. \tag{3}$$

# PPO



## Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

**Abstract**

We propose a new family of policy gradient methods for reinforcement learning, which alternate between sampling data through interaction with the environment, and optimizing a "surrogate" objective function using stochastic gradient ascent. Whereas standard policy gradient methods perform one gradient update per data sample, we propose a novel objective function that enables multiple epochs of minibatch updates. The new methods, which we call proximal policy optimization (PPO), have some of the benefits of trust region policy optimization (TRPO), but they are much simpler to implement, more general, and have better sample complexity (empirically). Our experiments test PPO on a collection of benchmark tasks, including simulated robotic locomotion and Atari game playing, and we show that PPO outperforms other online policy gradient methods, and overall strikes a favorable balance between sample complexity, simplicity, and wall-time.

Proximal Policy Optimization Algorithms (OpenAI, 2017)

## 2.1 Policy Gradient Methods

Policy gradient methods work by computing an estimator of the policy gradient and plugging it into a stochastic gradient ascent algorithm. The most commonly used gradient estimator has the form

$$\hat{g} = \hat{\mathbb{E}}_t \Big[ \nabla_\theta \log \pi_\theta(a_t \mid s_t) \hat{A}_t \Big] \tag{1}$$

where $\pi_\theta$ is a stochastic policy and $\hat{A}_t$ is an estimator of the advantage function at timestep $t$. Here, the expectation $\hat{\mathbb{E}}_t[\ldots]$ indicates the empirical average over a finite batch of samples, in an algorithm that alternates between sampling and optimization. Implementations that use automatic differentiation software work by constructing an objective function whose gradient is the policy gradient estimator; the estimator $\hat{g}$ is obtained by differentiating the objective

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t \Big[ \log \pi_\theta(a_t \mid s_t) \hat{A}_t \Big]. \tag{2}$$

While it is appealing to perform multiple steps of optimization on this loss $L^{PG}$ using the same trajectory, doing so is not well-justified, and empirically it often leads to destructively large policy updates (see Section 6.1; results are not shown but were similar or worse than the "no clipping or penalty" setting).

## 2.2 Trust Region Methods

In TRPO [Sch+15b], an objective function (the "surrogate" objective) is maximized subject to a constraint on the size of the policy update. Specifically,

$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t \right] \tag{3}$$

$$\text{subject to} \quad \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \leq \delta. \tag{4}$$

Here, $\theta_{\text{old}}$ is the vector of policy parameters before the update. This problem can efficiently be approximately solved using the conjugate gradient algorithm, after making a linear approximation to the objective and a quadratic approximation to the constraint.

The theory justifying TRPO actually suggests using a penalty instead of a constraint, i.e., solving the unconstrained optimization problem

$$\underset{\theta}{\text{maximize}} \, \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t - \beta \, \mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)] \right] \tag{5}$$

# 3 Clipped Surrogate Objective

Let $r_t(\theta)$ denote the probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$, so $r(\theta_{\text{old}}) = 1$. TRPO maximizes a "surrogate" objective

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\Big[r_t(\theta)\hat{A}_t\Big]. \tag{6}$$

The superscript $CPI$ refers to conservative policy iteration [KL02], where this objective was proposed. Without a constraint, maximization of $L^{CPI}$ would lead to an excessively large policy update; hence, we now consider how to modify the objective, to penalize changes to the policy that move $r_t(\theta)$ away from 1.

The main objective we propose is the following:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\Big[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\Big] \tag{7}$$

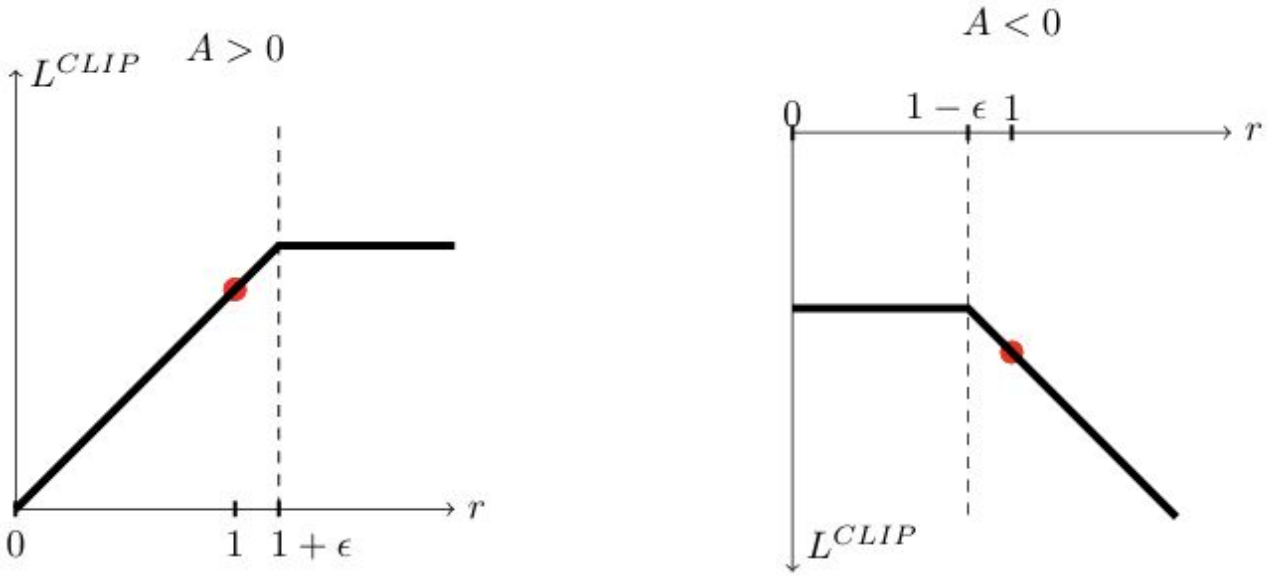
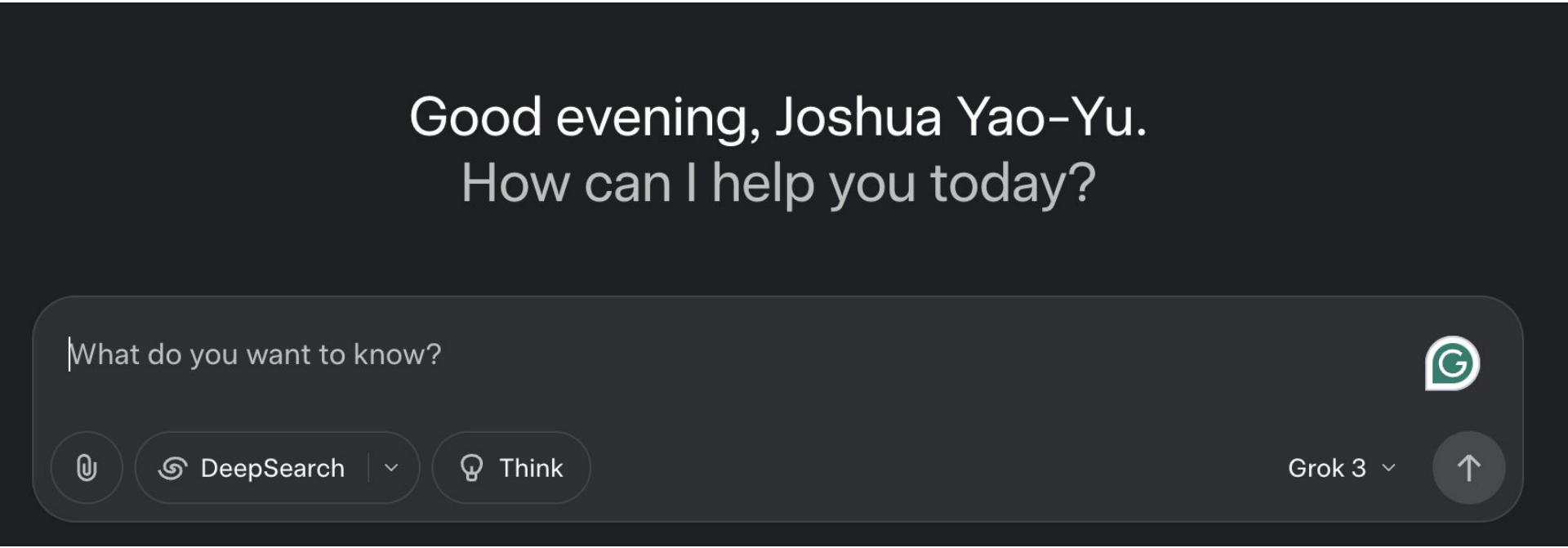



Figure 1: Plots showing one term (i.e., a single timestep) of the surrogate function $L^{CLIP}$ as a function of the probability ratio $r$, for positive advantages (left) and negative advantages (right). The red circle on each plot shows the starting point for the optimization, i.e., $r = 1$. Note that $L^{CLIP}$ sums many of these terms.

# DeepResearch



# Hello, Joshua Yao-Yu

Tell me what you can do

Save me time

Research a topic

Write an essay on the history of chess

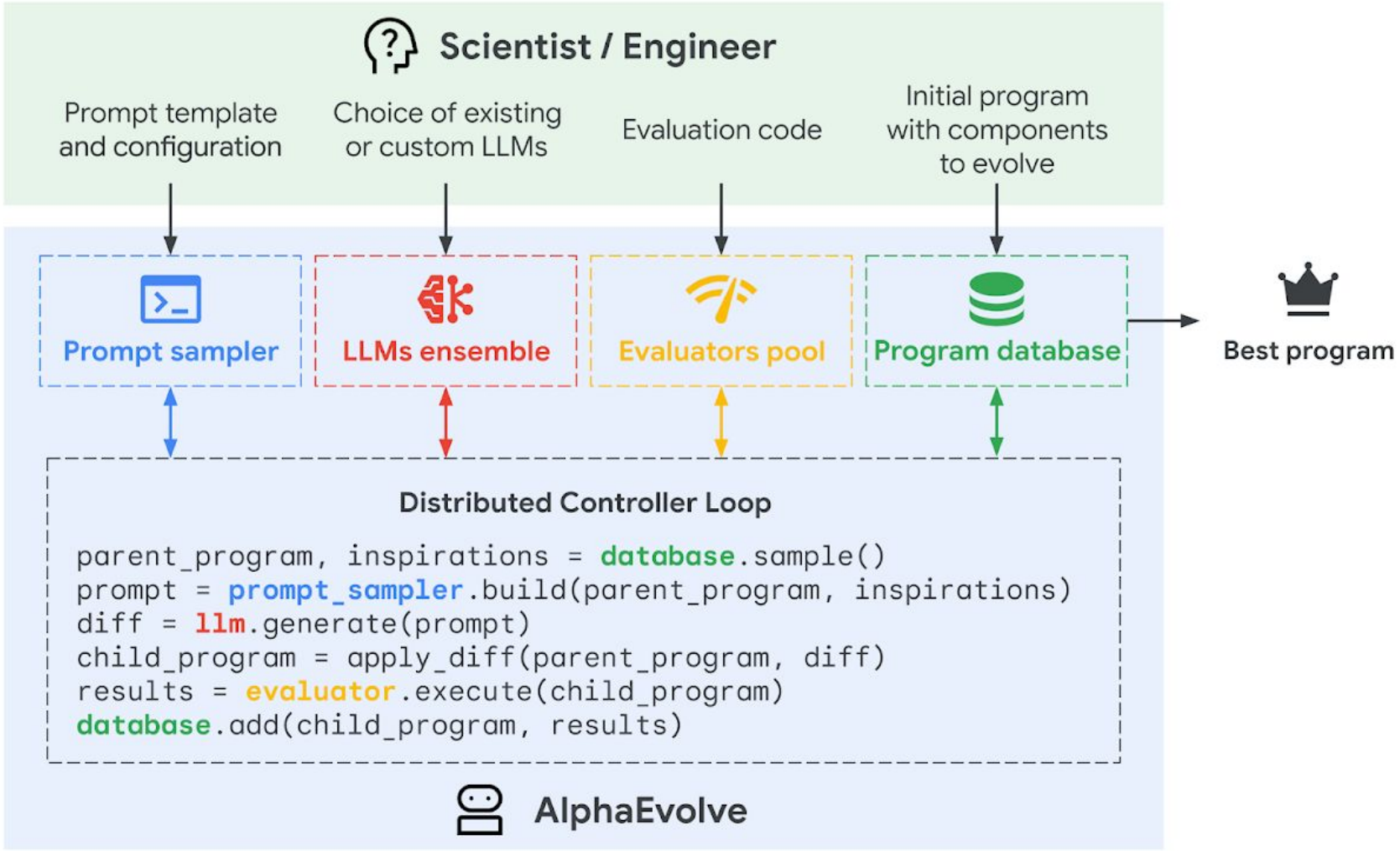
Ask Gemini

Deep Research

Canvas

# AlphaEvolve

- Advancing the frontiers in mathematics and algorithm discovery
- Enhancing AI training and inference
    - AlphaEvolve achieved up to a 32.5% speedup for the FlashAttention kernel implementation in Transformer-based AI models
- Designing better algorithms with large language models


Scientist / Engineer
Prompt template and configuration
Choice of existing or custom LLMs
Evaluation code
Initial program with components to evolve
Prompt sampler
LLMs ensemble
Evaluators pool
Program database
Best program
Distributed Controller Loop
parent_program, inspirations = database.sample()
prompt = prompt_sampler.build(parent_program, inspirations)
diff = llm.generate(prompt)
child_program = apply_diff(parent_program, diff)
results = evaluator.execute(child_program)
database.add(child_program, results)
AlphaEvolve

## Analysis

$$\max_{-1/2\le s\le 1/2}\int_{\mathbb{R}} f_t(t-x)f_t(x)\,dx \ge C\left(\int_{-1/4}^{1/4} f_t(x)\,dx\right)^2$$

1.5098 → **1.5053**

$$\|f*f\|_2^2 \le C'\|f*f\|_1\|f*f\|_\infty$$

0.8892 → **0.8962**

$$\max_{-1/2\le s\le 1/2}\left|\int_{\mathbb{R}} f(t-x)f(x)\,dx\right| \ge C''\left(\int_{-1/4}^{1/4} f(x)\,dx\right)^2$$

1.4581 → **1.4557**

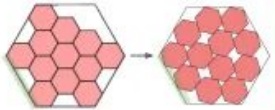
$$A(f)A(\hat{f}) \ge C'''$$

0.3523 → **0.3521**

## Geometry

Hexagon outer edge

4.000 → **3.942**

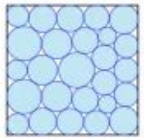


Max distance/min distance

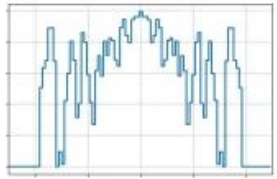
12.890 → **12.889**

Sum of radii

2.6340 → **2.6358**



## Combinatorics



$$\sup_{x\in[-2,2]}\int_{-1}^{1} f(t)g(x+t)\,dt \ge C$$

0.380926 → **0.380924**

$$|A+B| \ll |A|$$

$$|A-B| \gg |A|^C$$

1.1446 → **1.1584**

The Strassen algorithm partitions $A$, $B$ and $C$ into equally sized block matrices

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}, \quad C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},$$

with $A_{ij}, B_{ij}, C_{ij} \in \mathrm{Mat}_{2^{n-1} \times 2^{n-1}}(\mathcal{R})$. The naive algorithm would be:

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} \times B_{11} + A_{12} \times B_{21} & A_{11} \times B_{12} + A_{12} \times B_{22} \\ A_{21} \times B_{11} + A_{22} \times B_{21} & A_{21} \times B_{12} + A_{22} \times B_{22} \end{bmatrix}.$$

This construction does not reduce the number of multiplications: 8 multiplications of matrix blocks are still needed to calculate the $C_{ij}$ matrices, the same number of multiplications needed when using standard matrix multiplication.

The Strassen algorithm defines instead new values:

$$\begin{aligned} M_1 &= (A_{11} + A_{22}) \times (B_{11} + B_{22}); \\ M_2 &= (A_{21} + A_{22}) \times B_{11}; \\ M_3 &= A_{11} \times (B_{12} - B_{22}); \\ M_4 &= A_{22} \times (B_{21} - B_{11}); \\ M_5 &= (A_{11} + A_{12}) \times B_{22}; \\ M_6 &= (A_{21} - A_{11}) \times (B_{11} + B_{12}); \\ M_7 &= (A_{12} - A_{22}) \times (B_{21} + B_{22}), \end{aligned}$$

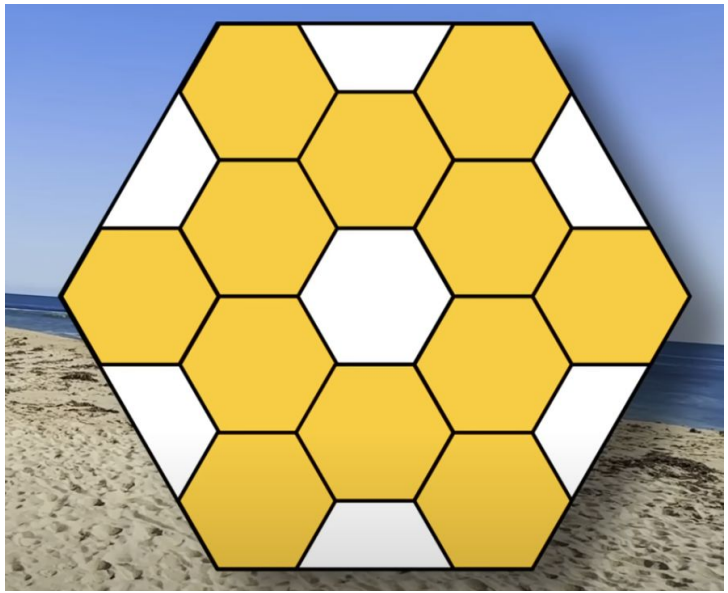
using only 7 multiplications (one for each $M_k$) instead of 8. We may now express the $C_{ij}$ in terms of $M_k$:

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 - M_2 + M_3 + M_6 \end{bmatrix}.$$

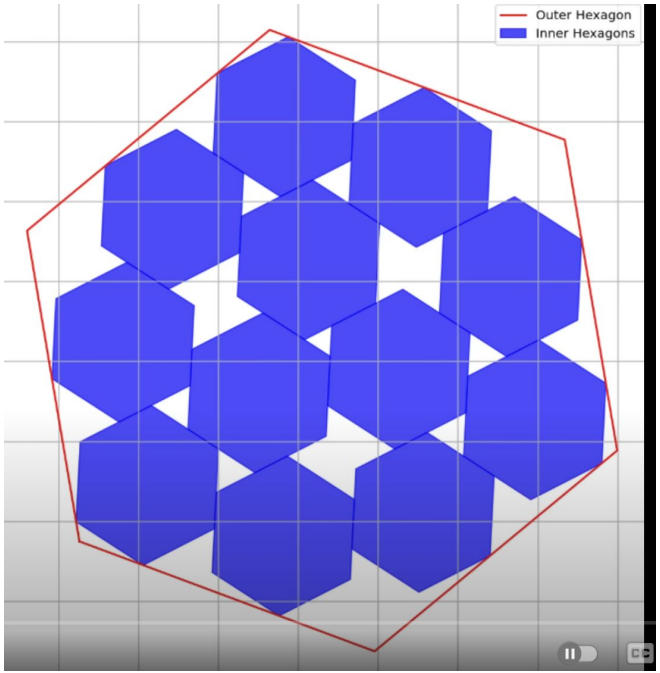
Strassen's 1969 algorithm

## Packing problem:Hexagons in Hexagons

The following pictures show n regular hexagons with side 1 packed inside the smallest known regular hexagon (of side length s ).



side lengths = 4




side lengths = 3.942

# New result distribution

Visualization of results across 67 problems.

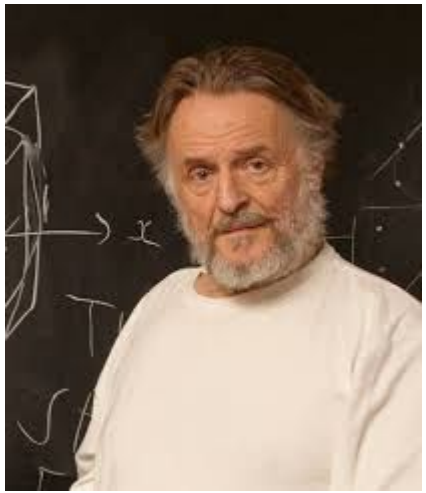
New result | Former new result, got improved upon | Worse than literature bound | Matched known optimal bound | Matched literature bound / N/A

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 |
| 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | | | |

Terry Tao
Blogpost:
[Mathematical exploration and discovery at scale]

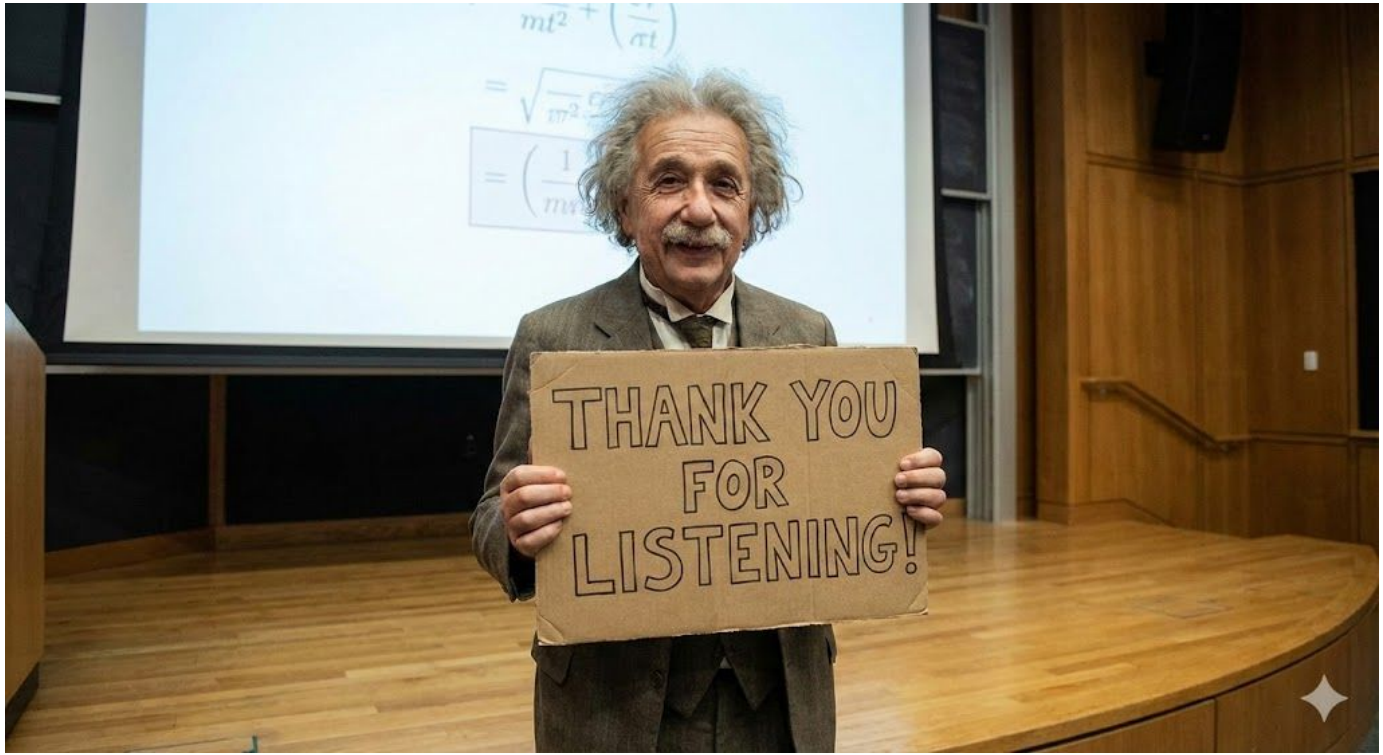
# Bonus (emergent Phenome):

- Game of Life
- Neural Cellular Automata



John Conway

Stephen Wolfram




p.s. We’re hiring! Please feel free to reach out!

Generated by Gemini3/Nanobanaba pro