



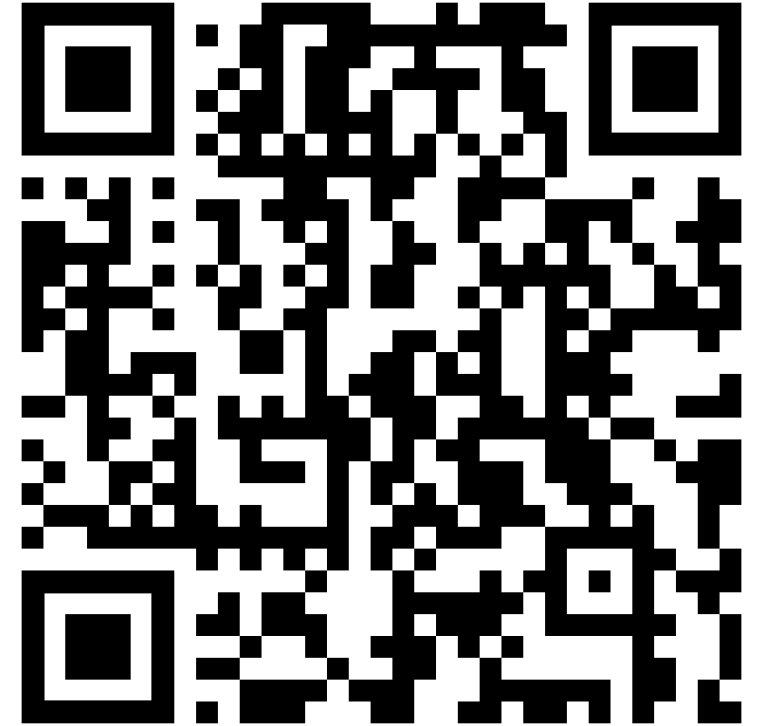
RC Short Course: AlphaFold 3 on Alpine

AlphaFold 3 on Alpine

Date: October 7, 2025

Instructor: Mohal Khandelwal

- Website: <https://www.colorado.edu/rc/>
- Documentation: <https://curc.readthedocs.io>
- Helpdesk: rc-help@colorado.edu
- Survey: <http://tinyurl.com/curc-survey18>



Slides

<https://github.com/ResearchComputing/alphafold3> short course

Meet the User Support Team



Layla
Freeborn



Brandon
Reyes



Andy
Monaghan



Michael
Schneider



John
Reiland



Dylan
Gottlieb



Mohal
Khandelwal



Ragan
Lee

Background: The Protein Folding Problem

- What a protein does (it's biological function) depends on its 3D structure
- Figuring out the what shapes a protein folds into has been a grand challenge in biology for 50+ years
- Can we predict a protein's 3D structure based on its 1D aa sequence?



Notable changes to AlphaFold 3

- The model parameters are a separate download with licensing issues.
- The main `run_alphafold.py` script can now be executed in 2 phases, the CPU/time intensive database search, followed by the GPU-based model inference.
- The database has shrunk from 2+ TB to ~600 GB.
- The input file format has changed to a custom JSON format, instead of FASTA file.
- Overall performance appears to be improved, qualitatively.

AlphaFold 3 Installation

- No customization of the Python code
- The wrapper:
 - Chooses the number of copies of jackhmmer/nhmmer processes to run in parallel based on the number of cores assigned to the job.
 - These are hard-coded to run with 8 threads each.

Database: ~250 GB
file. Pre-installed
and accessible via
***AF3_DATABASES_
DIR***

Containerized
module
Built from Docker
image

run_alphafold.sh
Wrapper script that
handles all inputs
to *run_alphafold.py*
in the container

Model Parameters

- We opted to have users download their own copy of the model parameters.
- We provide a [page](#) with links and guidance on submitting the Google form to request their own copy.
 - So far, no user has reported an issue obtaining their own copy following the guidance here.



AlphaFold 3 | Request to access model parameters

[AlphaFold 3](#) is an AI model developed by [Google DeepMind](#) and [Isomorphic Labs](#). It generates 3D structure predictions of biological molecules, providing model confidence for the structure predictions.

The AlphaFold 3 trained model parameters are available free of charge for non-commercial use, in accordance with the [AlphaFold 3 Model Parameters Terms of Use](#). You may only use the model parameters if received directly from Google. To request access to the AlphaFold 3 trained model parameters, please complete the information below. You must provide accurate and up-to-date information. Access will be granted at Google's sole discretion. We aim to respond to requests within 2 - 3 business days.

The AlphaFold 3 source code can be accessed via this [GitHub repository](#) and is licensed under the [Creative Commons Attribution-Non-Commercial ShareAlike International License, Version 4.0](#).

This form uses the defined terms from the [AlphaFold 3 Model Parameters Terms of Use](#). Please read these carefully. They establish what you can expect from us as you access and use the AlphaFold 3 model parameters and output, and what Google expects from you.

AlphaFold 3 Inputs

- The input **JSON** file format is the biggest change for users.
 - Documentation on the [format](#) is available.
 - There aren't very many examples online.
 - DeepMind does not provide any sort of validation tool for the format.
 - Error messages for bad inputs are not terribly helpful.
- You can provide inputs in one of two ways:
 - Single input file: **--json_path** flag followed by the path to a single JSON file.
 - Multiple input files: **--input_dir** flag followed by the path to a directory of JSON files.

AlphaFold 3 Module

- Loading the module (**ml alphafold/3.0.0**):
- creates a shortcut to the AlphaFold 3 script
- sets environment variables used by the wrapper script:
 - **AF3_IMAGE**: Path to the AlphaFold 3 container image
 - **AF3_CODE_DIR**: Directory containing the AlphaFold 3 codebase
 - **AF3_DATABASES_DIR**: Location of the required AlphaFold 3 reference databases
- redirects temporary files to **/scratch/alpine/\$USER**
 - Can be overridden by resetting TMPDIR after you load the module:

AlphaFold 3 Module

```
-----  
/curc/sw/alpine-modules/udep/bio/alphafold/3.0.0.lua:  
-----  
setenv("AF3_RESOURCES_DIR", "/curc/sw/install/bio/alphafold/3.0.0")  
setenv("AF3_IMAGE", "/curc/sw/install/bio/alphafold/3.0.0/alphafold3.sif")  
setenv("AF3_CODE_DIR", "/curc/sw/install/bio/alphafold/3.0.0/alphafold3-3.0.0")  
setenv("AF3_EXAMPLES", "/curc/sw/install/bio/alphafold/3.0.0/examples")  
setenv("AF3_DATABASES_DIR", "/gpfs/alpine1/datasets/bioinformatics/alphafold3")  
setenv("TMPDIR", "/scratch/alpine/mokh8410")  
set_shell_function("run_alphafold", "bash /curc/sw/install/bio/alphafold/3.0.0/run_alphafold.sh \"$@\"", "bash /curc/sw/install/bio/alphafold/3.0.0/run_alphafold.sh $*")  
help([[AlphaFold 3.0.0 (Containerized)  
  
This module provides access to AlphaFold 3 (DeepMind) via a Singularity container.  
Users must supply the DeepMind-provided model parameters separately.  
  
Usage:  
  
1. MSA Search (CPU):  
  sinteractive --partition=al40 --nodes=1 --ntasks=8 --time=1:00:00  
  ml purge && ml alphafold/3.0.0
```



AlphaFold 3 Workflow

- AlphaFold 3 runs in two stages:
 - Stage 1 (MSA Search): CPU and I/O-intensive; uses jackhmmer and hhmsearch.
 - Stage 2 (Inference): GPU-intensive; performs structure prediction.
- To better utilize limited GPU resources, these stages can be split using flags:
 - `--norun_inference` → Run only the MSA/data pipeline (Stage 1)
 - `--norun_data_pipeline` → Run only the inference step (Stage 2)

Running Your Prediction

- Example input files and scripts are in **`/curc/sw/install/bio/alphafold/3.0.0/examples`**.
- Loading the module stores this path in **`AF3_EXAMPLES`**.
- Copy the folder to a location where you have write permissions

```
$ls $AF3_EXAMPLES
alphafold3_alpine_cpu.sh  alphafold3_alpine_gpu.sh
alphafold3_alpine.sh     fold_protein_2PV7
```

Running Your Prediction

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --time=30:00
#SBATCH --partition=al40
#SBATCH --qos=normal
#SBATCH --gres=gpu:1
#SBATCH --job-name=af3_test
#SBATCH --output=af3_test_%j.out
#SBATCH --ntasks=8
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<your email address>

# Load the AlphaFold 3 module
module purge
module load alphafold/3.0.0

# Set input JSON, output directory, and model parameter path
export INPUT_FILE=/path/to/input/json
export OUTPUT_DIR=/path/to/output
export AF3_MODEL_PARAMETERS_DIR=/path/to/alphafold3/params

# Run AlphaFold 3
run_alphafold --json_path=$INPUT_FILE --output_dir=$OUTPUT_DIR --model_dir=$AF3_MODEL_PARAMETERS_DIR
```



Performance Considerations

- The number of cores/threads used during MSA steps is hardcoded and so AF cannot use more than 8 CPU cores.
- Requesting more cores will **not** improve performance and you may end up waiting longer for your job to start.

```
I1005 11:40:25.187402 23318643615296 subprocess utils.py:68] Launching subprocess  
"/hmmmer/bin/jackhmmmer -o /dev/null -A /tmp/tmpb0qmrqip/output.sto --noali --F1 0.0005 --F2 5e-05 --F3 5e-  
07 --cpu 8 -N 1 -E 0.0001 --incE 0.0001 /tmp/tmpb0qmrqip/query.fasta  
/root/public_databases/uniref90_2022_05.fa"  
I1005 11:42:28.693908 23318639412800 subprocess_utils.py:97] Finished Jackhmmmer in 123.506 seconds
```


Visualizing Outputs

- AlphaPickle: <https://github.com/mattarnoldbio/alphapickle>
- PyMol (free for education): <https://pymol.org/>
- AlphaFold Protein Structure Database website provides a great resource for learning how to interpret visualizations:
<https://alphafold.ebi.ac.uk/entry/Q9Y223#help>

Thank you!

Survey and feedback

<http://tinyurl.com/curc-survey18>

