HIGH PERFORMANCE RESEARCH COMPUTING

ACES: AlphaFold Protein Structure Prediction

October 7, 2025

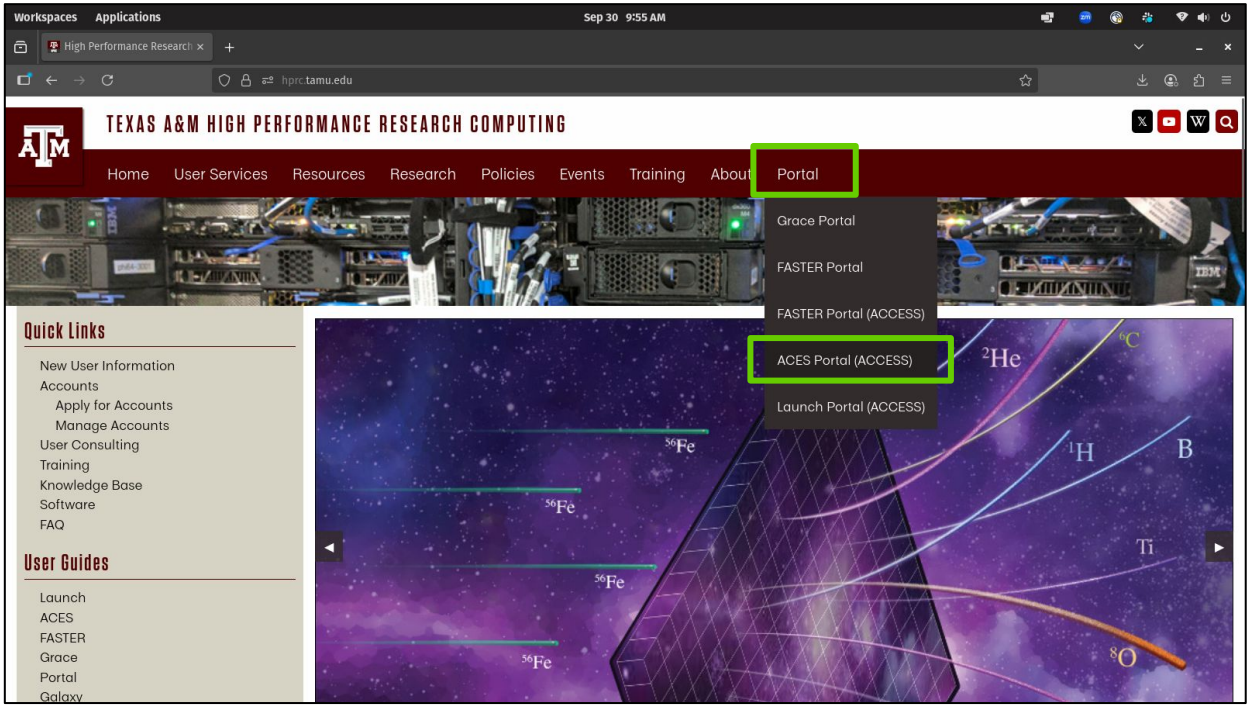
Michael Dickens

High Performance
Research Computing
DIVISION OF RESEARCH

ACES: AlphaFold Protein Structure Prediction

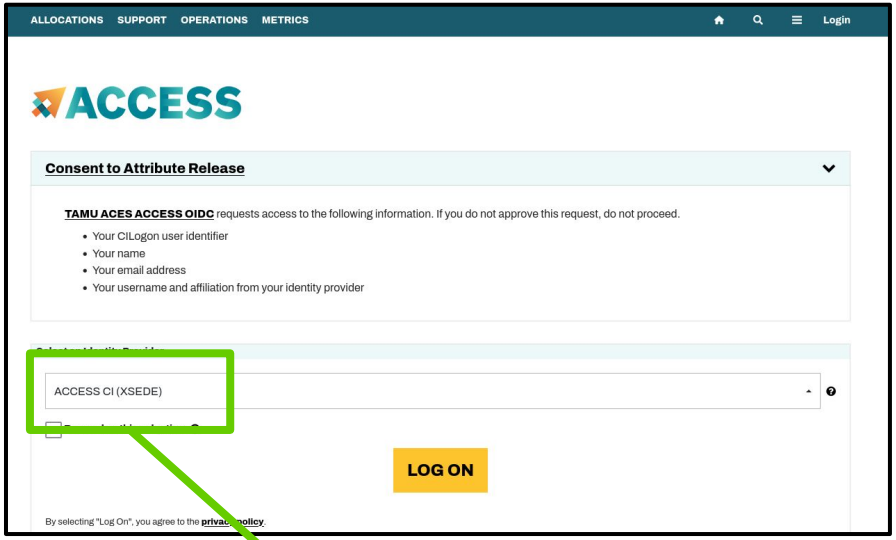
- ACES login
- AlphaFold history
- Running AlphaFold2
  - AlphaFold2 ParaFold workflow
  - AlphaFold2 confidence metrics
  - AlphaFold2 Resource Usage
- Running AlphaFold3
- HPRC cluster utilities
- Visualization of AlphaFold3 results

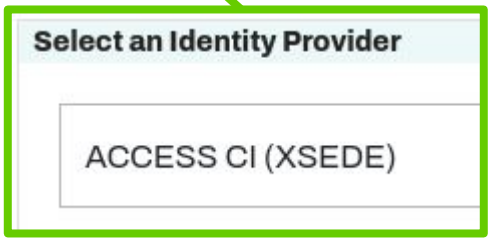
Accessing the HPRC ACES Portal



HPRC webpage: hprc.tamu.edu

Login and Two-factor Authentication

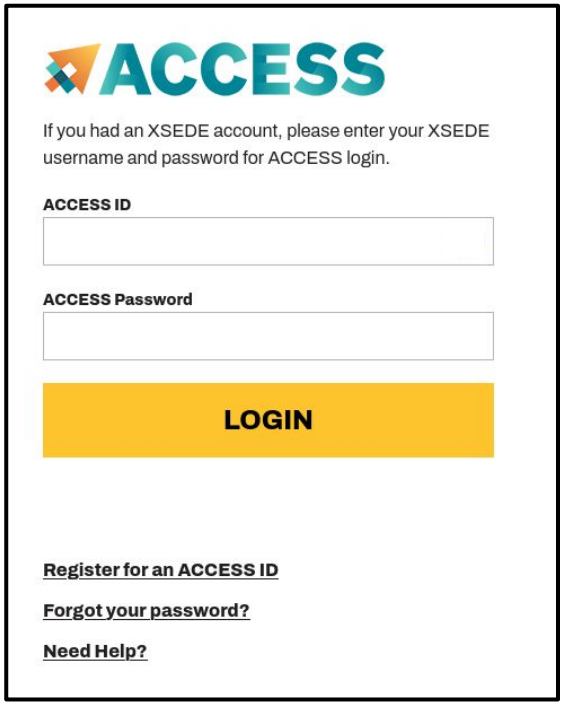




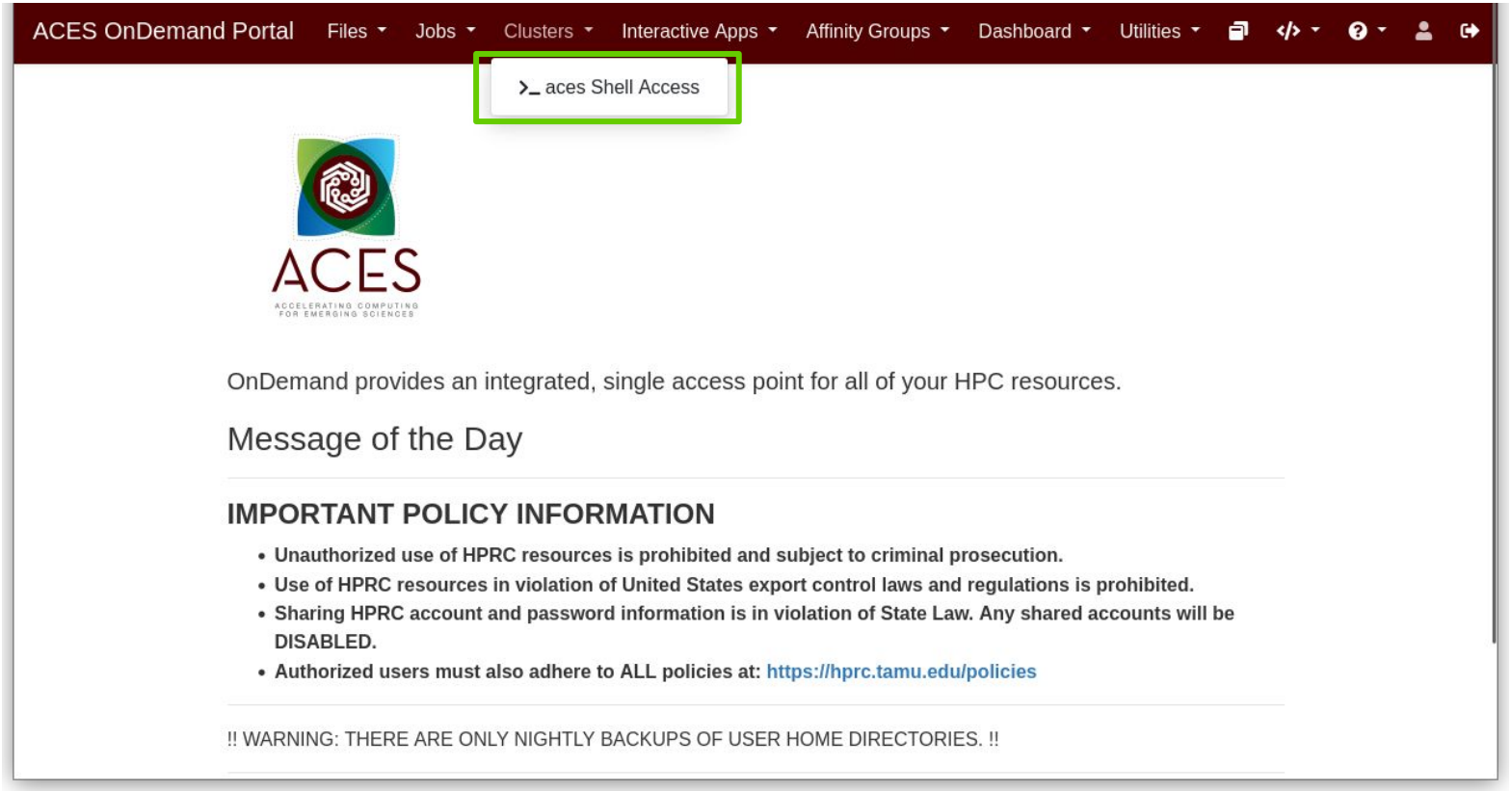
Select the Identity Provider appropriate for your account that you used when you registered

- ACCESS CI (XSEDE)
- search for your institution

Log-in using your ACCESS credentials.



Shell Access via the Portal



AlphaFold History

- An Artificial Intelligence program developed by DeepMind
- 2018 AlphaFold 1 placed 1st at CASP 13
- 2020 AlphaFold 1 code released as open source
- 2020 AlphaFold2 placed 1st at CASP 14
- 2021 AlphaFold publication in Nature
  - Highly accurate protein structure prediction with AlphaFold
- 2021 AlphaFold2 code released as open source on GitHub
- 2024 AlphaFold3 available on the DeepMind AlphaFold Server
  - 20 jobs per day allowed for academic researchers
- 2024 AlphaFold3 code available on github
- 2025 AlphaFold3 available on HPRC ACES cluster

Selection and Limitations of Resources

Resource Limitations

- AlphaFold
  - Currently AlphaFold(2,3) can only utilize one GPU.
  - AlphaFold2 minimum amino acid length: 16
  - AlphaFold2 maximum amino acid lengths are as follows:
    - 2,700 proteomes / Swiss-Prot
    - 1,280 all other UniProt
- About 90% of processing is done on CPU and the other 10% on GPU when using DeepMind's workflow in a single job script.
- ACES job script configuration
  - In your job script, request only ½ of the cores and memory when using 1 x H100 on a GPU node that has 2 x H100 installed so the other H100 GPU on that node is available for other jobs.

Running AlphaFold 2 on ACES using ParaFold

ParaFold and AlphaFold2

- ParaFold is a modified version of the AlphaFold2 workflow for HPC
- Github repo called ParallelFold
  - https://github.com/Zuricho/ParallelFold
- ParaFold separates the CPU and GPU processing into two jobs
  - ParaFold runs the multiple sequence alignment steps in a non-GPU job.
  - ParaFold runs the sequence prediction step in a GPU job.
- AlphaFold2 standard workflow requires a GPU node to run both alignment steps and the GPU sequence prediction step
  - GPU remains idle for 90% of the processing time while the alignment steps are running
- AlphaFold2 and ParaFold workflows both run the four sequence alignment steps sequentially and not in parallel

AlphaFold2 Databases for Structure Predictions on ACES

`/scratch/data/bio/alphafold/2.3.2`

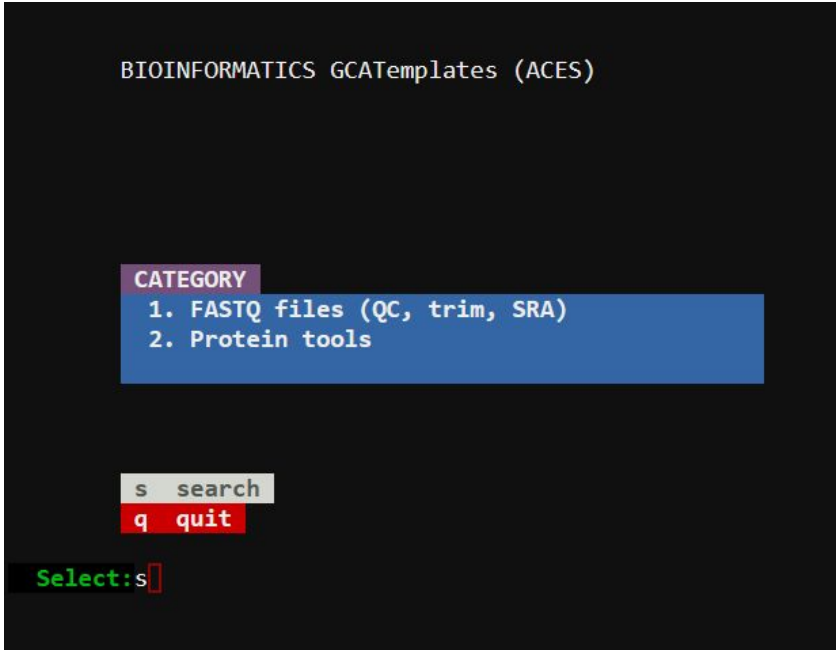
Database	Size	File Count	monomer	multimer
bfd	1.8T	7	✔	✔
mgnify	120G	2	✔	✔
params	5.3G	17	✔	✔
pdb70	56G	10	✔	-
pbd_mmcif	264G	211,106	✔	✔
pdb_seqres	257M	2	-	✔
uniprot	114G	2	-	✔
uniref30	467G	15	✔	✔
uniref90	77G	2	✔	✔
small_bfd	17G	2	✔	✔
example_data	6K	5	✔	✔
TOTAL	**2.9T**	**211,170**		

Finding AlphaFold2 Template Job Scripts using GCATemplates on ACES

- Genomic Computational Analysis Templates are job scripts that use examples input data, which you can run for demo purposes.

```
mkdir $SCRATCH/af_demo
cd $SCRATCH/af_demo
gcatemplates
```

- Type **s** for search, then enter **parafold** to search for the parafold template scripts, and select the **parafold monomer_ptm reduced_dbs** script.
- Review the script.



Example AlphaFold2 (ParaFold) Job Script

```
#!/bin/bash
#SBATCH --job-name=parafold-cpu     # job name
#SBATCH --time=7-00:00:00           # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1         # tasks (commands) per compute node
#SBATCH --cpus-per-task=8           # CPUs (threads) per command
#SBATCH --mem=32G                   # total memory per node
#SBATCH --output=stdout.%x.%j       # save stdout to file
#SBATCH --error=stderr.%x.%j        # save stderr to file

module purge
module load GCC/11.3.0  OpenMPI/4.1.4 AlphaFold/2.3.2-CUDA-11.8.0
module load ParaFold/2.0-CUDA-11.8.0

ALPHAFOLD_DATA_DIR=/scratch/data/bio/alphafold/2.3.2
protein_fasta=/scratch/data/bio/alphafold/example_data/T1083_T1084_multimer.fasta

# First, run CPU-only steps to get multiple sequence alignments
# Second, run GPU steps as a separate job after the first part completes successfully

run_alphafold.sh -d $ALPHAFOLD_DATA_DIR -o pf_output_dir -p multimer -i $protein_fasta -t 2025-1-1 -f && \
sbatch --job-name=parafold-gpu --time=2-00:00:00 --ntasks-per-node=1 --cpus-per-task=24 --mem=122G \
--gres=gpu:h100:1 --partition=gpu --output=stdout.%x.%j --error=stderr.%x.%j \
--dependency=afterok:$SLURM_JOBID<<EOF
#!/bin/bash
module purge
module load GCC/11.3.0  OpenMPI/4.1.4 AlphaFold/2.3.2-CUDA-11.8.0
module load ParaFold/2.0-CUDA-11.8.0 AlphaPickle/1.4.1
run_alphafold.sh -g -u 0 -d $ALPHAFOLD_DATA_DIR -o pf_output_dir -p multimer -i $protein_fasta -t 2025-1-1
# graph pLDDT and PAE .pkl files
run_AlphaPickle.py -od pf_output_dir/T1083_T1084_multimer
EOF
```

Submit and Monitor the Job

- Run the cpuavail utility to see cluster usage status.

```
[userid@aces-login3 ~]$ cpuavail
```

- Submit the job script to the Slurm scheduler.

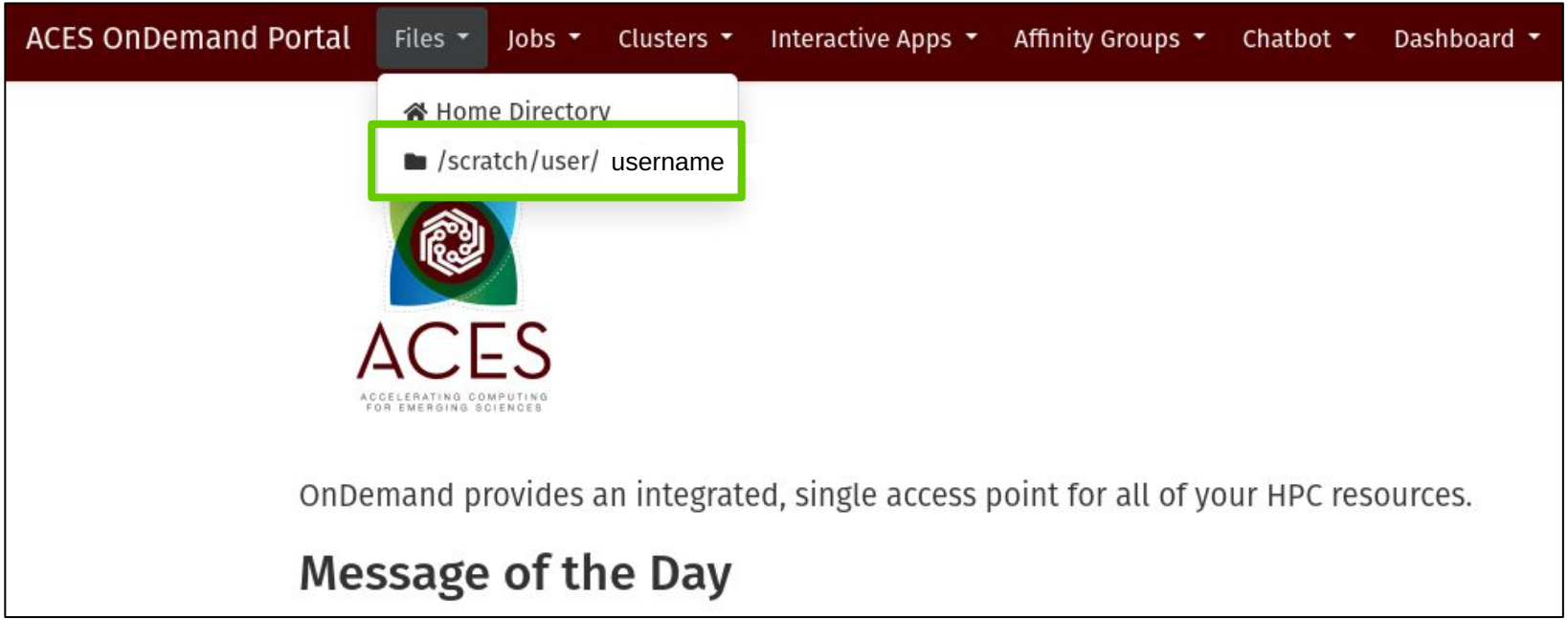
```
[userid@aces-login3 ~]$ sbatch run_parafold_alphafold_2.3.2_monomer_ptm_reduced_dbs_aces.sh

Submitted batch job 245660
```

- Monitor the job status. The GPU job will launch after the CPU job has successfully completed

```
[userid@aces-login3 ~]$ squeue --me

JOBID   NAME         USER      PARTITION  NODES  CPUS  STATE   TIME   TIME_LEFT   START_TIME          REASON  NODELIST
245660  parafold-cpu username  cpu        1      10    RUNNING 6.59   6-23:53:01  2024-09-17T15:49    None    ac047
```

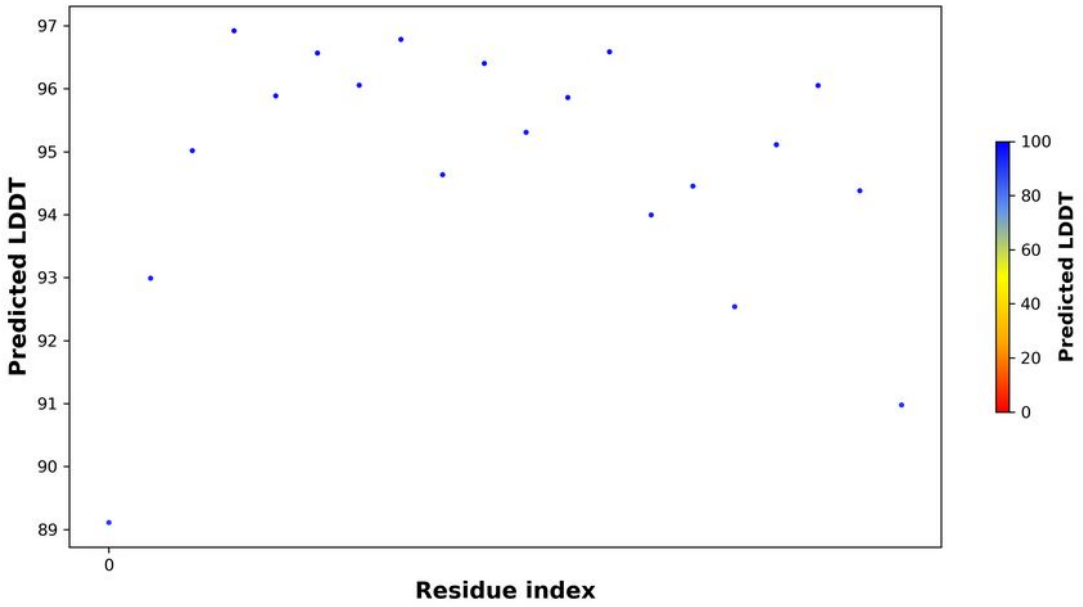
- completes in about 30 minutes, so we will review a completed job

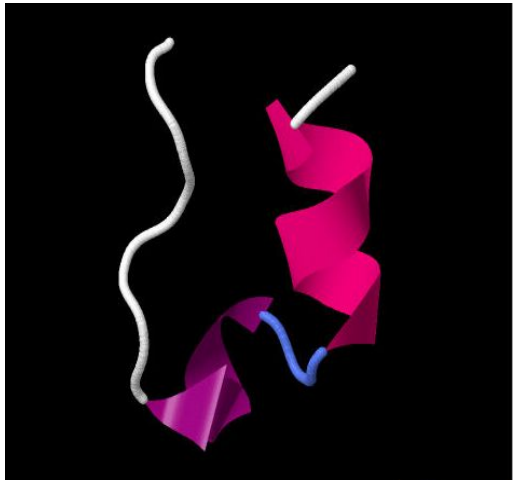
Files App for Viewing Images



AlphaFold2 Confidence Metrics

Visualize AlphaFold2 pLDDT Scores

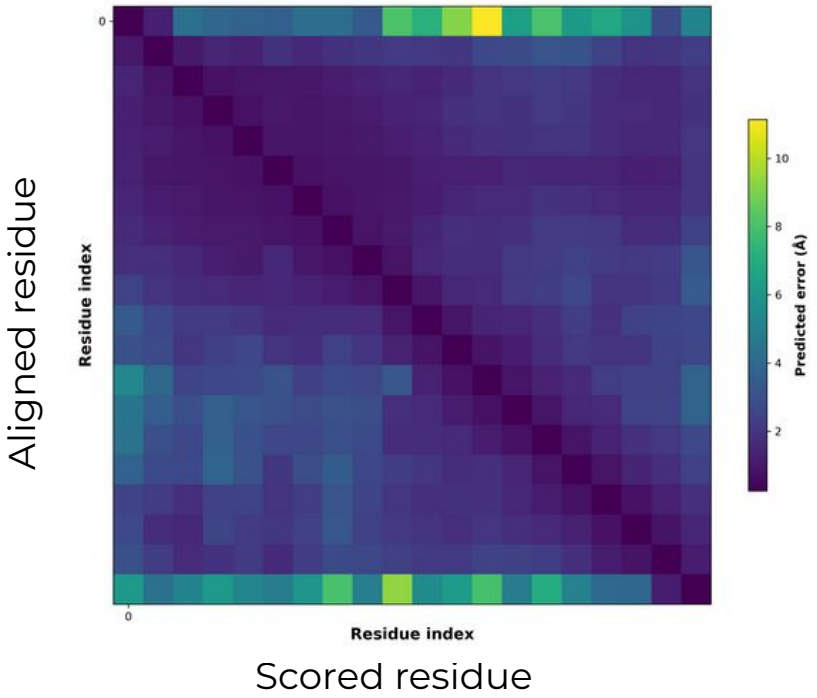




> 90 = Very high
70 - 90 = Confident
50 - 70 = Low
< 50 = Very low

/scratch/training/parafold/out_1L2Y_monomer_ptm_reduced_dbs/1L2Y/ranked_0_pLDDT.png

Visualize AlphaFold2 PAE Results (monomer_ptm)

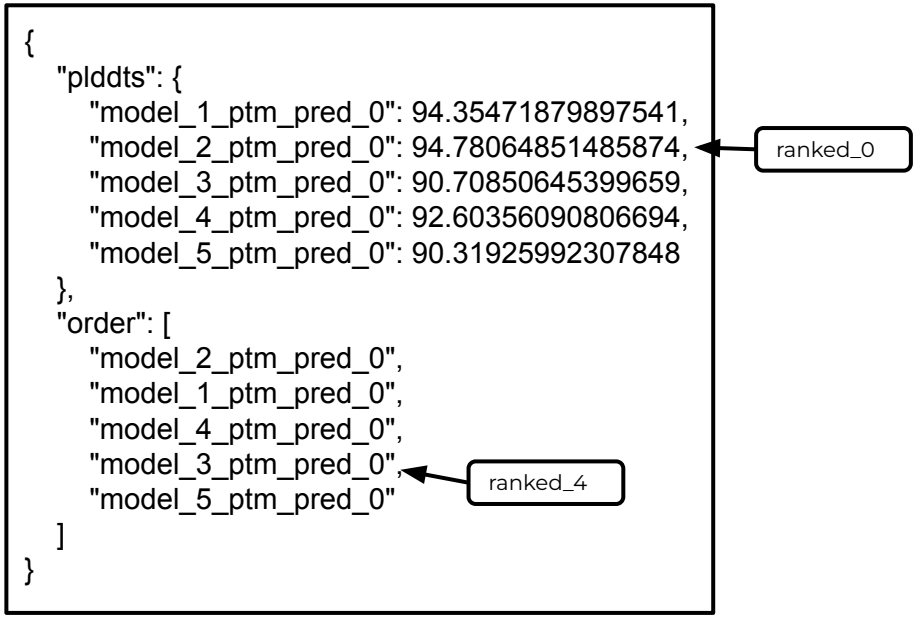


- Low Predicted Aligned Error (PAE) value has a higher confidence of accuracy
- Must use monomer_ptm or multimer as model_preset to create PAE image
- The color at position (x, y) indicates AlphaFold's expected position error at residue x, when the predicted and true structures are aligned on residue y.

/scratch/training/parafold/out_1L2Y_monomer_ptm_reduced_dbs/1L2Y/ranked_0_PAE.png

Model Scores

See which model has the top rank based on pLDDT score.



/scratch/training/parafold/monomer_ptm/out_parafold_1L2Y_monomer_ptm/1L2Y/ranking_debug.json

AlphaFold2 Job Resource Monitoring

ParaFold Workflow

- The ParaFold module uses the same AlphaFold2 installation as the AlphaFold2 module.
- ParaFold divides the AlphaFold2 workflow into two steps which can be run as two separate jobs:
  - CPU-only: processing the CPU steps to generate multiple sequence alignments
  - GPU: processing the GPU steps to generate predictions
- Test run of multimer (T1083_T1084_multimer.fasta) with full_dbs
- Runtimes for the same job script varied +- 1 hour; TM-scores also vary.

AlphaFold 2.3.2	Runtime	Highest Scoring Model	TM-score**
ParaFold	3 hrs 10 min*	model_1_multimer_v3_pred_4	0.892
DeepMind	2 hrs 45 min	model_1_multimer_v3_pred_1	0.883

***** combined time for the separate CPU 3 hour job and GPU 10 min job
****** measure of similarity between two protein structures

https://github.com/Zuricho/ParallelFold

Graphing Confidence Scores with AlphaPickle

AlphaPickle can be used to create graphs for pLDDT and PAE scores for AlphaFold2.

- Graphing PAE scores is only available for the **monomer_ptm** and **multimer** model presets.
- Load the AlphaPickle module at the beginning of the job script.
- Run AlphaPickle at the end, specifying the output directory used in the run_alphafold.py command.

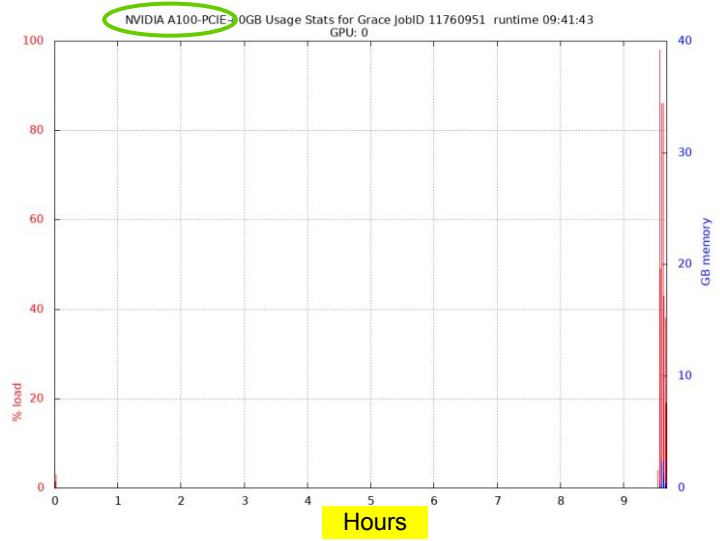
```
module load GCC/11.3.0  OpenMPI/4.1.4 AlphaFold/2.3.2-CUDA-11.8.0
module load ParaFold/2.0-CUDA-11.8.0 AlphaPickle/1.4.1



run_AlphaPickle.py -od pf_output_dir/T1083_T1084_multimer
```

- pLDDT: scale from 0 - 100 of per-residue estimate of prediction confidence
- PAE: Predicted Alignment Error

Comparison of DeepMind vs ParaFold Workflows

- AlphaFold2 DeepMind's one-job workflow (1 GPU job) vs ParaFold's two-job workflow (1 CPU-only job + 1 GPU job) for the same monomer_ptm full_dbs analysis
- The ParaFold workflow significantly reduces GPU idle time and SUs charged.
  - DeepMind Workflow: **1639** SUs ParaFold Workflow: **226** SUs (cpu+gpu jobs)

AlphaFold 2.3.1



AlphaFold 2.3.2

The first job of the ParaFold workflow (CPU-only) completed in 7.5 hours

reduced_dbs

- ParaFold supports the db_preset option
  - -c <db_preset> Choose database reduced_dbs or full_dbs (default: 'full_dbs')

- small_bfd_database is a subset of BFD and is generated by taking the first non-consensus sequence from every cluster in BFD.

- runtime is significantly quicker but with lower accuracy

Running AlphaFold3 on ACES

ALPHAFOLD 3 MODEL PARAMETERS TERMS OF USE

Last Modified: 2024-11-09

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. We make the trained model parameters and output generated using those available free of charge for certain non-commercial uses, in accordance with these terms of use and the AlphaFold 3 Model Parameters Prohibited Use Policy.

Key things to know when using the AlphaFold 3 model parameters and output

1. The AlphaFold 3 model parameters and output are **only** available for non-commercial use by, or on behalf of, non-commercial organizations (*i.e.*, universities, non-profit organizations and research institutes, educational, journalism and government bodies). If you are a researcher affiliated with a non-commercial organization, provided **you are not a commercial organisation or acting on behalf of a commercial organisation,** this means you can use these for your non-commercial affiliated research.
2. You **must not** use nor allow others to use:
    i. AlphaFold 3 model parameters or output in connection with **any commercial activities, including research on behalf of commercial organizations;** or
    ii. AlphaFold 3 output to **train machine learning models** or related technology for **biomolecular structure prediction** similar to AlphaFold 3.
3. You ***must not*** **publish or share AlphaFold 3 model parameters**, except sharing these within your organization in accordance with these Terms.
4. You ***can*** **publish, share and adapt AlphaFold 3** ***output*** in accordance with these Terms, including the requirements to provide clear notice of any modifications you make and that ongoing use of AlphaFold 3 output and derivatives are subject to the AlphaFold 3 Output Terms of Use.

https://github.com/google-deepmind/alphafold3/blob/main/WEIGHTS_TERMS_OF_USE.md

AlphaFold2 vs AlphaFold3

The new AlphaFold model demonstrates substantially improved accuracy over many previous specialized tools: far greater accuracy for protein–ligand interactions compared with state-of-the-art docking tools, much higher accuracy for protein–nucleic acid interactions compared with nucleic-acid-specific predictors and substantially higher antibody–antigen prediction accuracy compared with AlphaFold-Multimer v.2.37,8.

(from AlphaFold3 Abstract)

Abramson, J., Adler, J., Dunger, J. *et al.* Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature* 630, 493–500 (2024). https://doi.org/10.1038/s41586-024-07487-w

AlphaFold3 requires non-commercial users to agree to the terms of use in a Google form in order to download the model parameters file.

Create a New Working Directory

Create a working directory.

```
mkdir $SCRATCH/af3_demo
```

```
cd $SCRATCH/af3_demo
```

You can save your af3.bin.zst file in your $SCRATCH directory.

AlphaFold3 uses .json input format not FASTA

The Singularity image on ACES was built using AlphaFold3 version 3.0.1

alphafold3jobscript

The **`alphafold3jobscript`** utility is available on ACES which will generate one job script that will run AlphaFold3 in two steps:

1. run the four multiple sequence alignment steps in parallel in a non-GPU job
    a. each of the four parallel alignment steps will utilize 3 or 4 cores
2. run the structure prediction step in a separate GPU job launched from within the same job script

alphafold3jobscript

alphafold3jobscript --help

```
Synopsis:
       alphafold3jobscript is a script to create an AlphaFold3 job script to run first a CPU-only job for the
        sequence alignment step and a second GPU job for the prediction step.

Required:
       --json_path /full/path/to/input.json         # full path to your input.json file
       --model_dir /full/path/to/model/dir          # full path to the directory containing your af3.bin.zst model file

Optional:
       --max_template_date date          # format YYYY-MM-DD    (default: 2025-01-01)
       --num_recycles int                # (default: 3)
       --output_dir /full/path/to/dir    # (default: $PWD/output_NAME_JOBID)
       --gpu_type type                   # a30, h100   (default: first available)

Example Command:
       alphafold3jobscript --json_path /full/path/to/my/alphafold_input.json --model_dir /full/path/to/my/model/dir/
```

Example input file

/scratch/data/bio/alphafold3/examples/alphafold_input_2pv7.json

Submit and Monitor the Job

- Submit the job script to the Slurm scheduler.
  - completes in about 10 - 15 minutes
    - 8 minutes for the CPU job
    - 2 minutes for the GPU job

```
[userid@aces-login3 ~]$ sbatch run_alphafold_3.0.1_aces.sh

Submitted batch job 1008938
```

- Monitor the job status.

```
[userid@aces-login3 ~]$ squeue --me

JOBID     NAME             USER      PARTITION  NODES  CPUS  STATE    TIME   TIME_LEFT   START_TIME          REASON  NODELIST
1008938   alphafold3-cpu   userid    cpu        1      32    RUNNING  2.59   23:57:01    2025-04-3T16:11     None    ac046
```

Example alphafold_input_2pv7.json


```
{
  "name": "2PV7",
  "sequences": [
        {
        "protein": {
        "id": ["A", "B"],
        "sequence":
"GMRESYANENQFGFKTINSDIHKIVIVGGYGKLGGLFARYLRASGYPISILDREDWAVAESILANADVVIVSVPINLTLE
TIERLKPYLTENMLLADLTSVKREPLAKMLEVHTGAVLGLHPMFGADIASMAKQVVVRCDGRFPERYEWLLEQIQIW
GAKIYQTNATEHDHNMTYIQALRHFSTFANGLHLSKQPINLANLLALSSPIYRLELAMIGRLFAQDAELYADIIMDKSE
NLAVIETLKQTYDEALTFFENNDRQGFIDAFHKVRDWFGDYSEQFLKESRQLLQQANDLKQG"
        }
        }
  ],
   "modelSeeds": [1],
   "dialect": "alphafold3",
   "version": 1
}
```


https://github.com/google-deepmind/alphafold3/blob/main/docs/input.md

ACES Cluster Utilities

A number of cluster utilities are available to help you query resources from the command line, such as available nodes, GPUs, cores, memory, template job scripts, and shared conda and Python environments.

myjob	**gpuavail***
maxconfig	**cpuavail***
gcatemplates	**envsavail***
jobstats	**venvavail***
toolchains*	**maintenance**

Use the **-h** or **--help** flag with any utility to see available options.

* also available on the portal

Show Your Job Details using myjob

The myjob command

- can be used to see detailed information related to your job.
  - Status (PENDING, RUNNING, COMPLETED, FAILED, …)
  - Node List
  - Submit time, Start time, End time, Total runtime
  - CPU Efficiency
  - Memory Utilized, Memory Efficiency
- will advise you if your job is PENDING due to a scheduled maintenance.
- will advise you if your job FAILED due to CRLF characters in the job script and provide a link to the HPRC documentation on how to resolve this issue.
- will advise you if your job FAILED due to file or disk quota being reached.
  - will show you the directory in your $HOME directory that has the most files when $HOME file quota is reached.

https://hprc.tamu.edu/kb/Software/useful-tools/myjob

Show Your Job Details using myjob

```
[userid@aces-login3 ~]$ myjob 1008938

              Job ID: 1008938
             Cluster: aces
          User/Group: /username
               State: COMPLETED (exit code 0)
           Partition: cpu
          Node Count: 1
            NodeList: ac046
      Cores per node: 32
        CPU Utilized: 00:53:56
      CPU Efficiency: 22.13% of 4:03:44 core-walltime
         Submit time: 2025-04-03 15:53:22
          Start time: 2025-04-03 16:11:01
            End time: 2025-04-03 16:18:38
 Job Wall-clock time: 00:07:37
     Memory Utilized: 552.50 MB
   Memory Efficiency: 0.54% of 100.00 GB
            Job Name: 0.54% of 100.00 GB
Job Submit Directory: /scratch/user/userid/af3_demo
         Submit Line: sbatch run_alphafold_3.0.1_2PV7_aces.sh
```

PENDING Job due to a Scheduled Maintenance

```
[userid@aces-login3 ~]$ myjob 1320633

               Job ID: 1320633
              Cluster: aces
           User/Group: userid/userid
              Account: 123456789101
                State: PENDING
               Reason: ReqNodeNotAvail, Reserved for maintenance
          Submit time: 2024-10-14 10:09:59
            Partition: cpu
           Node Count: 1
             NodeList: None assigned
                Cores: 1
                 Note: Efficiency not available for jobs in the PENDING state.
             Job Name: picard
Job Submit Directory: /scratch/user/userid/myproject/picard
          Submit Line: sbatch run_bwa_samtools_pilon_faster.sh
                 Note: job is PENDING due to runtime overlapping with maintenance window.
                 Note: maintenance will begin in  22 hours, and 49 minutes.
```

Viewing Maximum Available Resources

The **maxconfig** command will show the recommended Slurm parameters for the maximum available resources (cores, memory, time) per node for a specified accelerator or partition (default ACES partition: cpu).

```
[userid@aces-login3 ~]$ maxconfig

  ACES partitions:   cpu  gpu  gpu_debug  pvc  bittware  memverge  nextsilicon  nec
  ACES GPUs in gpu partition:  a30:2  h100:2  h100:4  h100:8  pvc:2  pvc:4  pvc:6  pvc:8  ve:8

  Showing max parameters (cores, mem, time) for partition cpu

  CPU-billing * hours * nodes =   SUs
           96 *    72 *     1 = 6,912

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=3-00:00:00
#SBATCH --nodes=1          # max 64 nodes for partition cpu
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=96
#SBATCH --mem=488G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

https://hprc.tamu.edu/kb/Software/useful-tools/maxconfig

Viewing Maximum Available Resources

See the recommended Slurm parameters for requesting 1 x H100 GPU with ¼ the total CPUs and memory since there are 4 x H100s per node.

```
[userid@aces-login3 ~]$ maxconfig -g h100 -G 1

  ACES partitions:   cpu  gpu  gpu_debug  pvc  bittware  memverge  nextsilicon  nec
  ACES GPUs in gpu partition:  a30:2  h100:2  h100:4  h100:8  pvc:2  pvc:4  pvc:6  pvc:8  ve:8

  Showing 1/8 of total cores and memory for using 1 x h100 GPU

  (CPU-billing + (GPU-billing * GPU-count)) * hours * nodes =   SUs
  (         13 + (        128 *         1)) *    48 *     1 = 6,768

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=2-00:00:00
#SBATCH --partition=gpu
#SBATCH --nodes=1          # max 8 nodes for partition gpu
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=12
#SBATCH --mem=61G
#SBATCH --gres=gpu:h100:1
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

https://hprc.tamu.edu/kb/Software/useful-tools/maxconfig

Checking GPU Configuration & Availability on ACES

- Use the command line (shell) to see the current GPU configuration and availability.
- The GPU configuration can change since ACES is a composable resource cluster.
- If there are no GPUs in the AVAILABILITY output, it means that a GPU job that you submit may take a while to start.
- AlphaFold does not support running on PVC GPUs.

```
[userid@aces-login3 ~]$ gpuavail

    CONFIGURATION
NODE           NODE
TYPE           COUNT
--------------------
gpu:pvc:4      16
gpu:h100:2     10
gpu:a30:2      2
gpu:h100:4     2
gpu:pvc:2      1


                AVAILABILITY
NODE    GPU     GPU    GPUs     CPUs    GB MEM
NAME    TYPE    COUNT  AVAIL    AVAIL   AVAIL
-----------------------------------------------
ac065   a30     2      2        96      488
ac041   h100    4      1        87      421
ac045   h100    2      1        88      422
ac051   pvc     2      2        96      488
```

https://hprc.tamu.edu/kb/Software/useful-tools/gpuavail

Check non-GPU node Availability

Use the **cpuavail** command to see non-GPU nodes readily available for jobs.

```
[userid@aces-login3 ~]$ cpuavail

     CONFIGURATION                          AVAILABILITY
  NODE          NODE                  NODE     CPUs       GB MEM
  TYPE          COUNT                 NAME     AVAIL      AVAIL
  --------------------                --------------------------
  CPU-only           59               ac006         8          196
  GPU                38               ac007         6          86
  other              14               ac017         8          88
                                      ac021         44         4
                                      ac022         4          190
                                      ac042         54         214
                                      ac043         12         92
                                      ac052         60         244
                                      ac053         64         248
                                      ac063         12         228
                                      ac073         8          88
                                      ac080         1          121
```

https://hprc.tamu.edu/kb/Software/useful-tools/cpuavail

ACES Cluster Maintenance

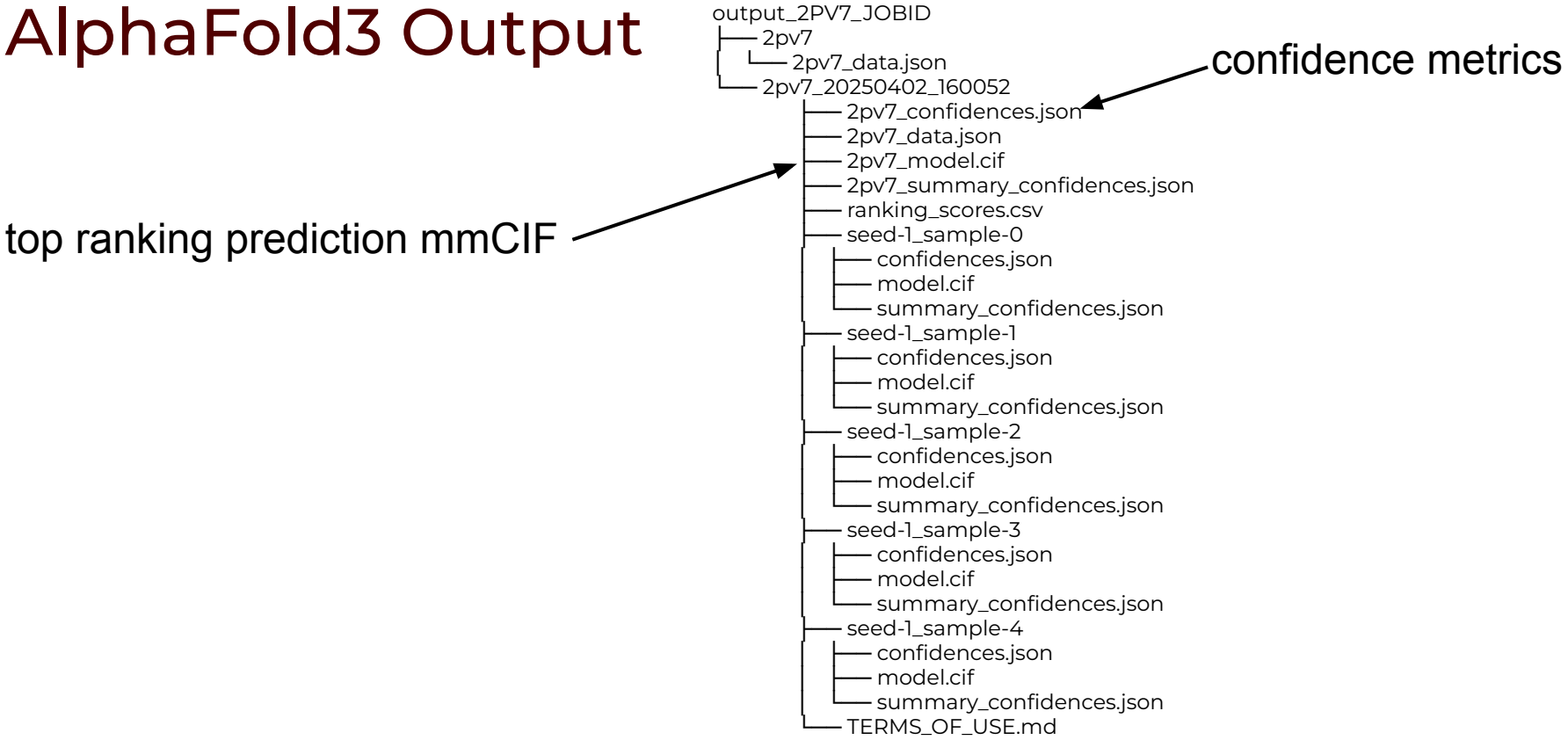
- You can use the maintenance command to see if there is a scheduled cluster maintenance.

```
[userid@aces-login3 ~]$ maintenance

 The scheduled 11 hour ACES maintenance will start in:

   3 days    16 hours    41 minutes

 Scheduled jobs will not start if they overlap with this maintenance window.
```

A 7-day job submitted at the time of the above message will remain queued and will not start until after the maintenance is complete.

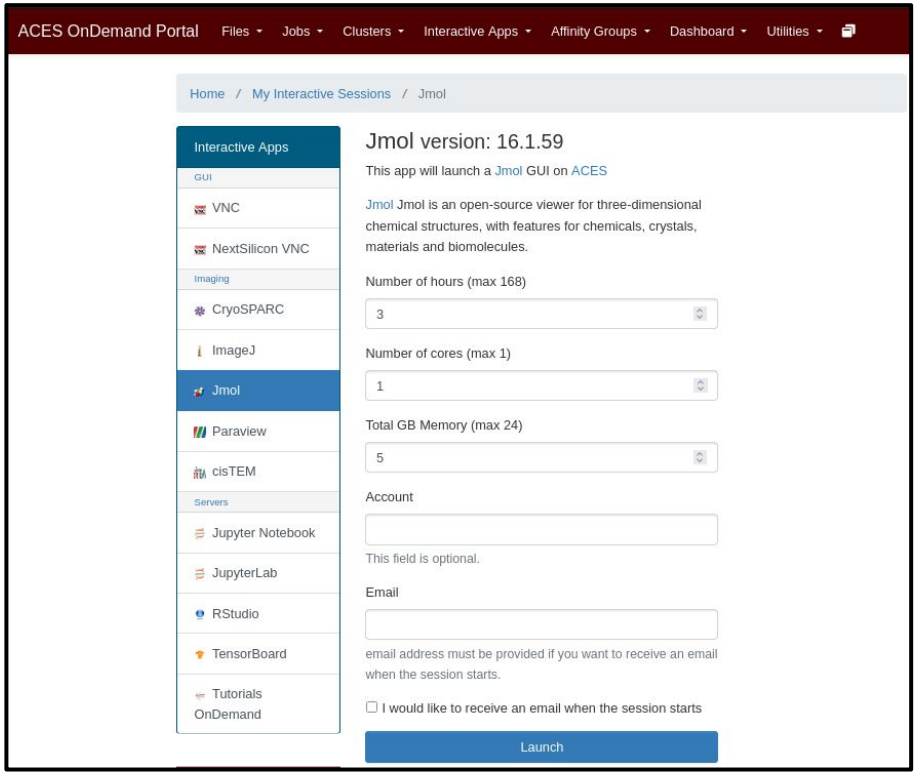
AlphaFold3 Results Visualization

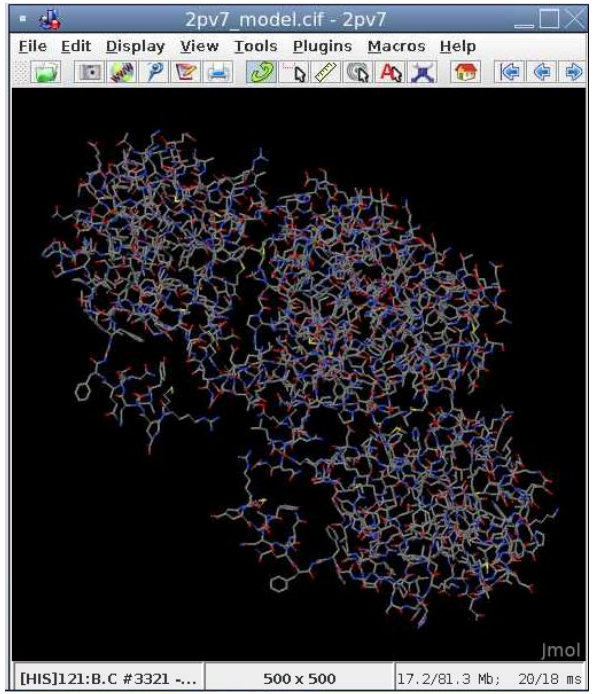
AlphaFold3 Output



https://github.com/google-deepmind/alphafold3/blob/main/docs/output.md

Visualize AlphaFold3 Results with Jmol on the ACES Portal



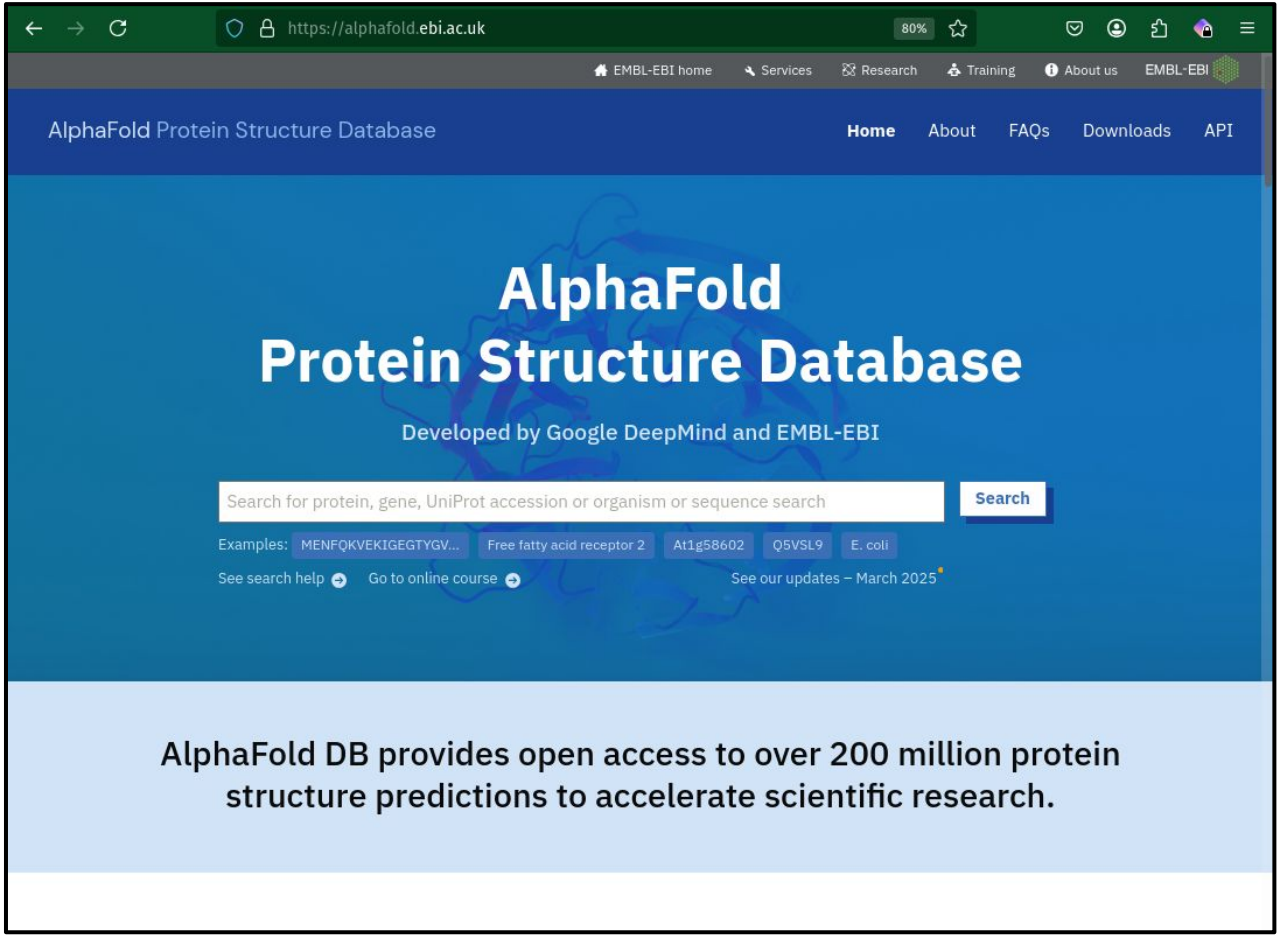


/scratch/training/alphafold/output_2PV7/2pv7_20250403_130646/2pv7_model.cif

Structure Database and References

DeepMind and EMBL's European Bioinformatics Institute (EMBL-EBI) have partnered to create AlphaFold DB to make these predictions freely available to the scientific community.

Search for your protein to see if it the structure has already been predicted using AlphaFold.



References

Abramson, J., Adler, J., Dunger, J. et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. Nature 630, 493–500 (2024). https://doi.org/10.1038/s41586-024-07487-w

Tunyasuvunakool, K., Adler, J., Wu, Z. et al. Highly accurate protein structure prediction for the human proteome. Nature 596, 590–596 (2021). https://doi.org/10.1038/s41586-021-03828-1

Jumper, J., Evans, R., Pritzel, A. et al. Highly accurate protein structure prediction with AlphaFold. Nature 596, 583–589 (2021). https://doi.org/10.1038/s41586-021-03819-2

Zhong, B, et al. (2021) ParaFold doi.org/10.48550/arXiv.2111.06340

Arnold, M. J. (2021) AlphaPickle doi.org/10.5281/zenodo.5708709

ACES Documentation

- ACES KnowledgeBase Documentation hprc.tamu.edu/kb
- ACES User Guide hprc.tamu.edu/kb/User-Guides/ACES
- Email your questions to help@hprc.tamu.edu
  - Received emails generate helpdesk tickets.

Let us know when the issue has been resolved so we can close the helpdesk ticket.

HIGH PERFORMANCE RESEARCH COMPUTING
TEXAS A&M UNIVERSITY

https://hprc.tamu.edu

HPRC Helpdesk:

help@hprc.tamu.edu
Phone: 979-845-0219

Take our short course survey!

https://u.tamu.edu/hprc_shortcourse_survey

Help us help you. Please include details in your request for support, such as, **Cluster** (ACES, FASTER, Grace, Launch), NetID (UserID), Job information (**JobID**(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.