Introduction to AlphaFold for 3D Protein Structure Prediction on Grace
AlphaFold for 3D Protein Structure Prediction on Grace

- Resources and Limitations
- Database Files
- Running AlphaFold
  - Google Colab
  - ChimeraX + Google Colab
  - Grace GPU or non-GPU nodes
- Visualization of Results
  - job resource usage
  - view predictions in ChimeraX
  - plotting pLDDT values
- Alternative Workflows
Finding AlphaFold template job scripts using GCATemplates on Grace

- Genomic Computational Analysis Templates have example input data so you can run the script for demo purposes.

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

- Type `s` for search then enter `alphafold` to search for the alphafold 2.3.1 template.

- Review the script and select the `reduced_dbs` script.

- Type `sbatch` run_alphafold_2.3.1_monomer_ptm_reduced_dbs_a100_grace.sh
Resource Limitations

- **AlphaFold**
  - currently AlphaFold can only utilize one GPU
  - about 90% of processing is done on CPU when using DeepMind's workflow

- **AlphaFold on HPRC Grace**
  - sometimes GPU not detected on certain nodes

- **AlphaFold in Google Colab (web browser or ChimeraX app)**
  - no guarantee of available resources in Colab
  - runs as a Jupyter notebook on Google Colab cloud servers
    - 12GB RAM max
    - a notebook can run for up to 12 hours per day
      - 24 hours per day with Colab Pro ($9.99/month)
    - not suitable for large predictions
### AlphaFold Databases on Grace

/scratch/data/bio/alphafold/2.3.2

<table>
<thead>
<tr>
<th>Database</th>
<th>Size</th>
<th>File Count</th>
<th>monomer</th>
<th>multimer</th>
</tr>
</thead>
<tbody>
<tr>
<td>bfd</td>
<td>1.8T</td>
<td>7</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>mgnify</td>
<td>120G</td>
<td>2</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>params</td>
<td>5.3G</td>
<td>17</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>pdb70</td>
<td>56G</td>
<td>10</td>
<td>✔</td>
<td>-</td>
</tr>
<tr>
<td>pbd_mmcif</td>
<td>264G</td>
<td>211,106</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>pdb_seqres</td>
<td>257M</td>
<td>2</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td>uniprot</td>
<td>114G</td>
<td>2</td>
<td>-</td>
<td>✔</td>
</tr>
<tr>
<td>uniref30</td>
<td>467G</td>
<td>15</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>uniref90</td>
<td>77G</td>
<td>2</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>small_bfd</td>
<td>17G</td>
<td>2</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>example_data</td>
<td>6K</td>
<td>5</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>2.9T</strong></td>
<td><strong>211,170</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Resources for Running AlphaFold

- Run as a Jupyter Notebook on Google Colab in web browser.
- Run as a Jupyter Notebook on Google Colab in ChimeraX.
  - ChimeraX Interactive App on HPRC Grace Portal
    - https://portal-grace.hprc.tamu.edu
- Run as a Slurm job script on Grace.

https://hprc.tamu.edu/kb/Software/AlphaFold
ColabFold AlphaFold2 Jupyter Notebook

Enter an amino acid sequence

NLYIQWLKDGGPSSGRPPPS

ColabFold: AlphaFold2 using MMseqs2

Easy to use protein structure and complex prediction using AlphaFold2 and Alphafold2-multimer. Sequence alignments/templates are generated through MMseqs2 and HHsearch. For more details, see bottom of the notebook, checkout the ColabFold GitHub and read our manuscript. Old versions: v1.0, v1.1, v1.2, v1.3

Mirdita M. Schütze K. Moriwaki Y. Heo L. Ovchinnikov S. Steinegger M. ColabFold - Making protein folding accessible to all. bioRxiv, 2021

Input protein sequence(s), then hit Runtime -> Run all

query_sequence: *NLYIQWLKDGGPSSGRPPPS

- Use : to specify inter-protein chainbreaks for modeling complexes (supports homo- and hetero-oligomers). For example PI...SK:PI...SK

jobname: *test

use_amber: 

template_mode: none
ChimeraX

- Can be used to visualize protein structures
- Can be launched using the Grace portal
  - portal-grace.hprc.tamu.edu
- Can be used to run AlphaFold using the daily build version (2022.02.22+)
  - uses Google Colab with limited resources

ChimeraX

This app will launch a UCSF ChimeraX GUI on Grace

UCSF ChimeraX is a program for the interactive visualization and analysis of molecular structures and related data, including density maps, trajectories, and sequence alignments.

chimerax version

ChimeraX/2022.02.22
ChimeraX

- Launch a ChimeraX job on the Grace portal [portal-grace.hprc.tamu.edu](http://portal-grace.hprc.tamu.edu)
  - select Node Type: CPU only
  - AlphaFold runs on Google Colab GPUs so we can use a non-GPU for running ChimeraX
- ChimeraX will be used later for the following
  - run AlphaFold in Google Colab
  - visualize results from an AlphaFold job on Grace
Running AlphaFold on Grace

- Can be run as a job script requesting one GPU
- Shared databases are available: 2.6TB total size

<table>
<thead>
<tr>
<th>AlphaFold 2.2.3</th>
<th>monomer_ptm 20 aa</th>
<th>multimer 98 &amp; 73 aa</th>
</tr>
</thead>
<tbody>
<tr>
<td>A100</td>
<td>36 minutes</td>
<td>4 hours 49 minutes</td>
</tr>
<tr>
<td>A40</td>
<td>35 minutes</td>
<td>-</td>
</tr>
<tr>
<td>RTX 6000</td>
<td>33 minutes</td>
<td>4 hours 44 minutes</td>
</tr>
<tr>
<td>T4</td>
<td>33 minutes</td>
<td>4 hours 45 minutes</td>
</tr>
<tr>
<td>CPU only</td>
<td>40 minutes</td>
<td>6 hours 11 minutes</td>
</tr>
</tbody>
</table>

- Can be run in ChimeraX from the Grace portal
  - using the ChimeraX AlphaFold option
  - all processing done on Google cloud servers
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 Grace partition: long) and show SUs charged.

```
[username@grace ~]$ maxconfig
Grace partitions: short medium long xlong vnc gpu bigmem special gpu-a40
Grace GPUs in gpu partition: a100:2 a40:3 rtx:2 t4:4

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

CPU-billing * hours * nodes = SUs
  48 * 168 * 1 = 8,064

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=7-00:00:00
#SBATCH --nodes=1           # max 64 nodes for partition long
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=48
#SBATCH --mem=360G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```
See the recommended Slurm parameters for requesting 1 x A100 GPU with ½ the total CPUs and memory since there are 2 x A100s per node.

[username@grace ~]$ maxconfig -g a100 -G 1

Grace partitions: short medium long xlong vnc gpu bigmem special gpu-a40
Grace GPUs in gpu partition: a100:2 a40:3 rtx:2 t4:4

Showing 1/2 of total cores and memory for using 1 x a100 GPU

(CPU-billing + (GPU-billing * GPU-count)) * hours * nodes = SUs
(  25 + (   72 *    1)) *  96 *    1 = 9,312

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=4-00:00:00
#SBATCH --nodes=1           # max 32 nodes for partition gpu
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=24
#SBATCH --mem=180G
#SBATCH --gres=gpu:a100:1
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
Grace SUs Charged: GPU vs CPU-only Jobs

show SU rate for 1 day CPU-only

```
maxconfig -d 1
```

\[
\text{CPU-billing} \times \text{hours} \times \text{nodes} = \text{SUs} \\
48 \times 24 \times 1 = 1,152
\]

show SU rate for 1 x A100 GPU for 1 day

```
maxconfig -g a100 -G 1 -d 1
```

\[
\text{(CPU-billing} + (\text{GPU-billing} \times \text{GPU-count})) \times \text{hours} \times \text{nodes} = \text{SUs} \\
(25 + (72 \times 1)) \times 24 \times 1 = 2,328
\]

show SU charge rate

```
maxconfig -h
```

SU rate per GPU:

<table>
<thead>
<tr>
<th>GPU</th>
<th>SUs per hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>a100</td>
<td>72</td>
</tr>
<tr>
<td>a40</td>
<td>48</td>
</tr>
<tr>
<td>rtx</td>
<td>48</td>
</tr>
<tr>
<td>t4</td>
<td>24</td>
</tr>
</tbody>
</table>
AlphaFold with ChimeraX + Google Colab
Maximize ChimeraX Window

Right click and select Maximize if the ChimeraX window is off the screen.
AlphaFold with ChimeraX

- Launch ChimeraX using the HPRC Grace portal
- Select the AlphaFold Structure Prediction option
AlphaFold with ChimeraX

- Enter an amino acid sequence **NLYIQWLKDGPPSSGRPPPS**
  - or paste in Clipboard first then paste in Sequence field.
- Click Predict.
- A Google Colab page will start and prompt you for your Google login.
- Login to your Google account to begin processing.
- Prediction completes in about 1 hour.
- Not ideal for large prediction jobs
AlphaFold Grace Job Scripts
Example AlphaFold Job Script

• multimer
  ○ dbs in red are required for multimer
• AlphaFold can only use one GPU so reserve half the CPU and memory resources so another job can use the other GPU.
  ○ Grace compute nodes have 360GB of available memory and 48 cores.

```
#!/bin/bash
#SBATCH --job-name=alphafold        # job name
#SBATCH --time=2-00:00:00           # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1         # tasks (commands) per compute node
#SBATCH --cpus-per-task=24          # CPUs (threads) per command
#SBATCH --mem=180G                  # total memory per node
#SBATCH --gres=gpu:a100:1           # request 1 A100 GPU
#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/2.3.1-CUDA-11.7.0

ALPHAFOLD_DATA_DIR=/scratch/data/bio/alphafold/2.3.2

# run jobstats in the background (&) to monitor cpu and gpu usage
jobstats &
run_alphafold.py  
  --use_gpu_relax 
  --data_dir=$ALPHAFOLD_DATA_DIR 
  --uniref90_database_path=$ALPHAFOLD_DATA_DIR/uniref90/uniref90.fasta 
  --uniref30_database_path=$ALPHAFOLD_DATA_DIR/uniref30/UniRef30_2023_02 
  --mgnify_database_path=$ALPHAFOLD_DATA_DIR/mgnify/mgy_clusters_2025_05.fa 
  --bfd_database_path=$ALPHAFOLD_DATA_DIR/bfd/bfd_metaclust_clu_complete_id30_c90_final_seq.sorted_opt 
  --pdb_seqres_database_path=$ALPHAFOLD_DATA_DIR/pdb_seqres/pdb_seqres.txt 
  --uniprot_database_path=$ALPHAFOLD_DATA_DIR/uniprot/uniprot.fasta 
  --template_mmcif_dir=$ALPHAFOLD_DATA_DIR/pdb_mmcif/mmcif_files 
  --obsolete_pdb_dir=$ALPHAFOLD_DATA_DIR/pdb_mmcif/obsolete.dat 
  --max_template_date=2024-1-1 
  --db_preset-full_dbs 
  --output_dir=out_alphafold 
  --fasta_paths=/scratch/data/bio/alphafold/example_data/T1083_T1084_multimer.fasta

# run jobstats to create a graph of cpu and gpu usage for this job
jobstats
```
Example AlphaFold Job Script

- **monomer**
  - dbs in **red** required for monomer
- **monomer_p tm**
  - will produce pTM scores that can be graphed using AlphaPickle.
- AlphaFold can only use one GPU so reserve half the CPU and memory resources so another job can use the other GPU.

```
#!/bin/bash
#SBATCH --job-name=alphafold        # job name
#SBATCH --time=2-00:00:00           # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1         # tasks (commands) per compute node
#SBATCH --cpus-per-task=24          # CPUs (threads) per command
#SBATCH --mem=180G                  # total memory per node
#SBATCH --gres=gpu:a100:1           # request 1 A100 GPU
#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/2.3.1-CUDA-11.7.0
ALPHAFOLD_DATA_DIR=/scratch/data/bio/alphafold/2.3.2
# run jobstats in the background (&) to monitor cpu and gpu usage
jobstats &
run_alphafold.py \
  --use_gpu_relax \n  --data_dir=$ALPHAFOLD_DATA_DIR \n  --uniref90_database_path=$ALPHAFOLD_DATA_DIR/uniref90/uniref90.fasta \n  --uniref30_database_path=$ALPHAFOLD_DATA_DIR/uniref30/UniRef30_2023_02  \n  --mgnify_database_path=$ALPHAFOLD_DATA_DIR/mgnify/mgy_clusters_2022_05.fa \n  --bfd_database_path=$ALPHAFOLD_DATA_DIR/bfd/bfd_metaclust_clu_complete_id30_c90_final_seq.sorted_opt \n  --template_mmcif_dir=$ALPHAFOLD_DATA_DIR/pdb_mmcif/mmcif_files \n  --obsolete_pdb_path=$ALPHAFOLD_DATA_DIR/pdb_mmcif/obsolete.dat \n  --pdb70_database_path=$ALPHAFOLD_DATA_DIR/pdb70/pdb70 \
  --model_preset=monomer \n  --max_template_date=2024-1-1 \n  --db_preset=full_dbs \n  --output_dir=out_alphafold \n  --fasta_paths=/scratch/data/bio/alphafold/example_data/T1083.fasta
```

- monomer
  - dbs in red required for monomer
- monomer_ptm
  - will produce pTM scores that can be graphed using AlphaPickle.
- AlphaFold can only use one GPU so reserve half the CPU and memory resources so another job can use the other GPU.
Example AlphaFold Job Script

- monomer + reduced_dbs
- dbs in red required for monomer + reduced_dbs
- Small_bfd_database is a subset of BFD and is generated by taking the first non-consensus sequence from every cluster in BFD.
- AlphaFold can only use one GPU so reserve half the CPU and memory resources so another job can use the other GPU.
#!/bin/bash
#SBATCH --job-name=alphafold        # job name
#SBATCH --time=2-00:00:00           # max job run time dd-hh:mm:ss
#SBATCH --ntasks-per-node=1         # tasks (commands) per compute node
#SBATCH --cpus-per-task=24          # CPUs (threads) per command
#SBATCH --mem=180G                  # total memory per node
#SBATCH --gres=gpu:a100:1           # request 1 A100 GPU
#SBATCH --output=stdout.%x.%j       # save stdout to file
#SBATCH --error=stderr.%x.%j        # save stderr to file

# version 2.3.1 has the unified memory variable already configured
module purge
module load GCC/11.3.0  OpenMPI/4.1.4 AlphaFold/2.3.1-CUDA-11.7.0

- Unified memory can be used to request more than just the total GPU memory for the JAX step in AlphaFold.
  - A100 GPU has 40GB memory.
  - XLA_PYTHON_CLIENT_MEM_FRACTION (configured to 3.0 in the AlphaFold 2.3.1 module)
- this example script has 120 GB of unified preallocated memory:
  - 40 GB from A100 GPU + 80 GB DDR from motherboard.

AlphaFold Results Visualization
Visualize Results with ChimeraX

Click the minimize button to return to ChimeraX or click Reconnect then minimize.
Visualize AlphaFold Google Colab Results with ChimeraX
View PDB Structure if Available

Type open and the protein name 1L2Y
Visualize AlphaFold Grace Results with ChimeraX
AlphaFold Confidence Metrics
AlphaPickle can be used to create graphs for pLDDT and PAE scores.

- 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.

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

https://github.com/mattarnoldbio/alphapickle
Visualize AlphaFold pLDDT Scores

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

You may get different results compared to the image above when using reduced_dbs.
Visualize AlphaFold PAE Results (monomer_ptm)

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

![PAE Image](out_alphafold_2.3.1/1L2Y/ranked_0_PAЕ.png)
Evaluating Models

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

```
cat out_alphafold_2.3.1/IL2Y/ranking_debug.json
```
AlphaFold Job Resource Monitoring
Review CPU usage for a Job

The `seff` command displays CPU and memory resource usage and efficiency.

```
seff 8862586
```

<table>
<thead>
<tr>
<th>Job ID: 8862586</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster: grace</td>
</tr>
<tr>
<td>User/Group: username/username</td>
</tr>
<tr>
<td>State: COMPLETED (exit code 0)</td>
</tr>
<tr>
<td>Nodes: 1</td>
</tr>
<tr>
<td>Cores per node: 24</td>
</tr>
<tr>
<td>CPU Utilized: 08:19:34</td>
</tr>
<tr>
<td>CPU Efficiency: 2.39% of 14-12:55:36 core-walltime</td>
</tr>
<tr>
<td>Job Wall-clock time: 14:32:19</td>
</tr>
<tr>
<td>Memory Utilized: 51.43 GB</td>
</tr>
<tr>
<td>Memory Efficiency: 28.57% of 180.00 GB</td>
</tr>
</tbody>
</table>

Usage stats are not accurate until the job is complete.
Review GPU and CPU usage for a Job

The `jobstats` command monitors GPU and CPU resource usage and create graphs.

```bash
#!/bin/bash
#SBATCH --job-name=my_gpu_job
#SBATCH --time=1:00:00:00
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=24
#SBATCH --mem=180G
#SBATCH --gres=gpu:a100:1
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j

module purge

# run jobstats in the background &
# to monitor resource usage
jobstats &

my_alphafold_command

# run jobstats to create a graph
# of cpu and gpu usage for this job
jobstats
```

- CPU stats are only accurate for jobs using the entire compute node resources (CPUs, memory).
- GPU stats are accurate if using fewer than max CPUs and memory.

https://hprc.tamu.edu/kb/Software/useful-tools/jobstats
AlphaFold Workflow Alternatives
ParallelFold

- ParallelFold (ParaFold) breaks the AlphaFold workflow into two steps:
  - processing of the three CPU steps in parallel
  - processing of the GPU step
- The parallel portion for CPU steps is not implemented yet resulting in similar or longer runtimes than the DeepMind approach.
  - The first three CPU steps, jackhammer, jackhammer and HHblits are supposed to run as three separate processes in parallel but currently this parallelization step is not implemented yet.
  - When the CPU processing is parallelized, it will require 21 cores.
    - 8 cores for each of the two jackhammer steps
    - 5 cores for the HHblits step.
- The AlphaFold CPU steps may be implemented in parallel soon.
  
  https://github.com/Zuricho/ParallelFold
Databases 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.

AlphaFold DB provides open access to 992,316 protein structure predictions for the human proteome and other key proteins of interest, to accelerate scientific research.
References

Article | Open Access | Published: 15 July 2021

Highly accurate protein structure prediction with AlphaFold

John Jumper, Richard Evans, ... Demis Hassabis

Nature 596, 583–589 (2021) | Cite this article

Article | Open Access | Published: 22 July 2021

Highly accurate protein structure prediction for the human proteome

Kathryn Tunyasuvunakool, Jonas Adler, ... Demis Hassabis

Nature 596, 590–596 (2021) | Cite this article

HPRC Resources

- **Free Help**
  - Send an email to help@hprc.tamu.edu if you have any questions regarding Bioinformatics tools usage on HPRC clusters or to schedule a Zoom or in-person visit.
  - First spend some time investigating the error.
    - read log files, stdout file, stderr file, tool manual
    - Google search
    - Google user groups: many are software specific
  - Include details about your issue.
    - Which **cluster** or which Galaxy you are using
    - The **JobID**
    - Which software you are using
    - Which modules you have loaded
    - Commands you used in your job script
    - Error messages you are seeing
- **HPRC NGS data analysis tools Documentation**
  - [https://hprc.tamu.edu/kb/Software/Bioinformatics](https://hprc.tamu.edu/kb/Software/Bioinformatics)

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