Research Notebook

Andy Kapoor

Advisor: Dr. Mary von dem Bussche

Bio 496

Started: 12/23/23

Table of Contents

AlphaFold Setup and Concurrent Runs ……………………………………………………………**3**

- ☐ *23 December 2023 - 29 December 2023*
- ☐ Setup of AlphaFold/ColabFold in Microsoft Azure VM (Machine Learning Studio) with HyperDrive Python Jupyter script for concurrent modeling of wt and mutant instances

Rosetta Setup and Concurrent Runs……………………………………………………………..**9**

- ☐ *29 December 2023 - 1 January 2024*
- ☐ After compilation error in Azure, Rosetta setup in Ubuntu VM with Oracle VirtualBox with GNU Screens installed to allocate one query per core of the CPU. Parameterization outlined and 3 fragment files downloaded from Robot Rosetta (Robetta) fragment server.

AutoDock Vina Binding for WT and Mutant MDH…………………………………………….**14**

- ☐ *1 January 2024 - 23 January 2024*
- ☐ AutoDock Vina setup in Windows personal machine. Parameters formatted with PyMol and MGLTools to be delineated in configuration file for commandline binding analysis.

References…………………………………………………………………………………….**18**

12/23/23

AlphaFold in Microsoft Azure Virtual Machine

Objective: The purpose of this section is to configure AlphaFold/ ColabFold in the Microsoft Azure Virtual Machine, with HyperDrive enabled for concurrent modeling of both protein queries (wild type and mutant MDH). I will use a Jupyter Notebook Script derived from Dr. Colby T. Ford's GitHub repository.

Procedures:

Microsoft Azure VM setup:

1. Login to Microsoft Azure after registering for an account with the allocated VM type, and enter the Machine Learning Studio
2. Under the "Manage" tab, select "Compute" to create a new compute instance for the modeling queries
3. Create a new compute instance with CPU VM with the allocated VM type (Standard DCsv3 and DCdsv3-series used, specifically Standard_DC48ds_v3 VM with 48 physical cores, 384 GB Memory, and 2400 GiB SSD)
4. Name the compute instance, and start it. Enter it, and modify the Jupyter Notebook with the HyperDrive script.

Docker Image Creation for AlphaFold:

```
docker build -t alphafold2_aml --build-arg IMAGE_VERSION=$(git rev-parse --short HEAD) .
# docker run --name alphafold2_aml --rm -p 8787:8787 alphafold2_aml
# docker exec -it alphafold2_aml /bin/bash

az login
az account set --subscription <SUBSCRIPTION_ID>
az acr login --name <CONTAINER_REGISTRY_NAME>
docker tag mmae_aml <CONTAINER_REGISTRY_NAME>.azurecr.io/alphafold2_aml:$(git rev-parse --short
HEAD)
docker push <CONTAINER_REGISTRY_NAME>.azurecr.io/alphafold2_aml:$(git rev-parse --short HEAD)
```

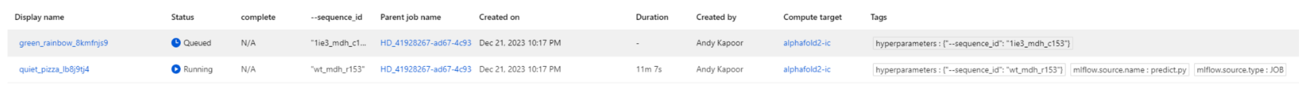
Data:

Contents of AlphaFold sequences.fasta:

```
>wt_mdh_r153
MKVAVLGAAGGIGQALALLLKTQLPSGSELSLYDIAPVTPGVAVDLSHIPTAVKIKGFSGEDATPALEGADVV
LISAGVARKPGMDRSDLFNVNAGIVKNLVQQVAKTCPKACIGIITNPVNTTVAIAAEVLKKAGVYDKNKLF
GVTTLDIIRSNTFVAELKGKQPGEVEVPVIGGHSGVTILPLLSQVPGVSFTEQEVADLTKRIQNAGTEVVEAK
AGGGSATLSMGQAAARFGLSLVRALQGEQGVVECAYVEGDGQYARFFSQPLLLGKNGVEERKSIGTLSAFE
QNALEGMLDTLKKDIALGQEFVNK
>1ie3_mdh_c153
MKVAVLGAAGGIGQALALLLKTQLPSGSELSLYDIAPVTPGVAVDLSHIPTAVKIKGFSGEDATPALEGADVV
LISAGVARKPGMDRSDLFNVNAGIVKNLVQQVAKTCPKACIGIITNPVNTTVAIAAEVLKKAGVYDKNKLF
GVTTLDIICSNTFVAELKGKQPGEVEVPVIGGHSGVTILPLLSQVPGVSFTEQEVADLTKRIQNAGTEVVEAK
```

AGGGSATLSMGQAAARFGLSLVRALQGEQGVVECAYVEGDGQYARFFSQPLLLGKNGVEERKSIGTLSAFE
QNALEGMLDTLKKDIALGQEFVNK

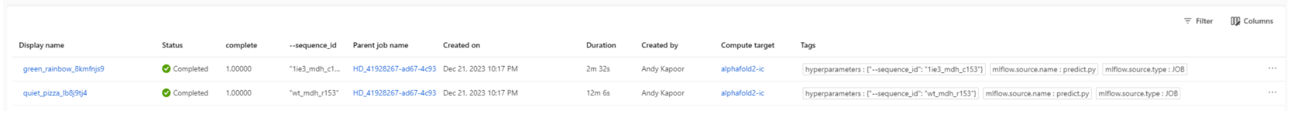
AlphaFold Queued Runs:



Display name	Status	complete	--sequence_id	Parent job name	Created on	Duration	Created by	Compute target	Tags
green_rainbow_8kmfnjs9	Queued	N/A	"1ie3_mdh_c1...	HD_41928267-ad67-4c93	Dec 21, 2023 10:17 PM	-	Andy Kapoor	alphafold2-ic	hyperparameters : {"--sequence_id": "1ie3_mdh_c153"}
quiet_pizza_lb8j9tj4	Running	N/A	"wt_mdh_r153"	HD_41928267-ad67-4c93	Dec 21, 2023 10:17 PM	11m 7s	Andy Kapoor	alphafold2-ic	hyperparameters : {"--sequence_id": "wt_mdh_r153"} mlflow.source.name : predict.py mlflow.source.type : JOB

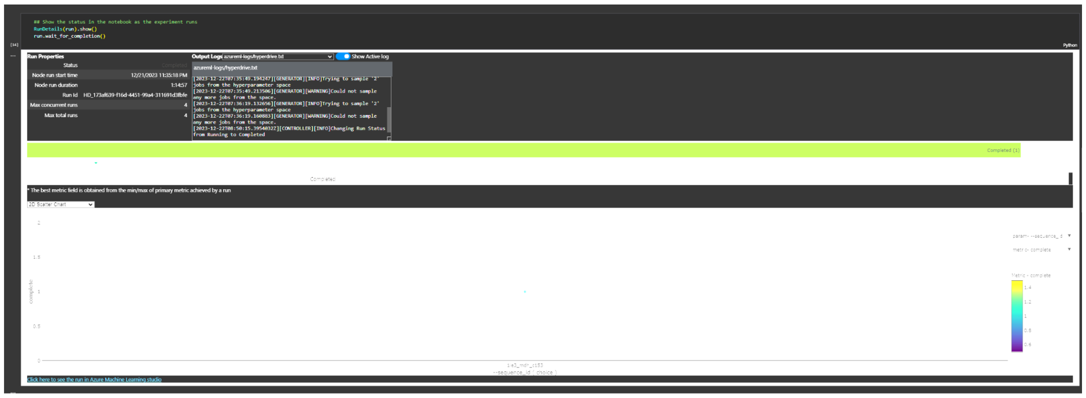
AlphaFold Completed Runs:

Child runs
complete
1.0000
complete
Display name (by created time)
--sequence_id
complete
wt_mdh_r153
1ie3_mdh_c153
1
complete
1
1
1
1
1



Filter Columns

Display name	Status	complete	--sequence_id	Parent job name	Created on	Duration	Created by	Compute target	Tags
green_rainbow_8kmfnjs9	Completed	1.00000	"1ie3_mdh_c1...	HD_41928267-ad67-4c93	Dec 21, 2023 10:17 PM	2m 32s	Andy Kapoor	alphafold2-ic	hyperparameters : {"--sequence_id": "1ie3_mdh_c153"} mlflow.source.name : predict.py mlflow.source.type : JOB
quiet_pizza_lb8j9tj4	Completed	1.00000	"wt_mdh_r153"	HD_41928267-ad67-4c93	Dec 21, 2023 10:17 PM	12m 6s	Andy Kapoor	alphafold2-ic	hyperparameters : {"--sequence_id": "wt_mdh_r153"} mlflow.source.name : predict.py mlflow.source.type : JOB



```
## Show the status in the notebook as the experiment runs
RunDetails(run).show()
run.wait_for_completion()
```

[14] Python

Run Properties	
Status	Completed
Node run start time	12/21/2023 11:35:18 PM
Node run duration	1:14:57
Run Id	HD_173af639-f16d-4451-99a4-311691d3fbfe
Max concurrent runs	4
Max total runs	4

Output Logs azureml-logs/hyperdrive.txt Show Active log

azureml-logs/hyperdrive.txt

```
[2023-12-22T07:35:49.194247][GENERATOR][INFO]Trying to sample '2'
jobs from the hyperparameter space
[2023-12-22T07:35:49.213506][GENERATOR][WARNING]Could not sample
any more jobs from the space.
[2023-12-22T07:36:19.132656][GENERATOR][INFO]Trying to sample '2'
jobs from the hyperparameter space
[2023-12-22T07:36:19.160883][GENERATOR][WARNING]Could not sample
any more jobs from the space.
[2023-12-22T08:50:15.3954032Z][CONTROLLER][INFO]Changing Run Status
from Running to Completed
```

Completed (1)

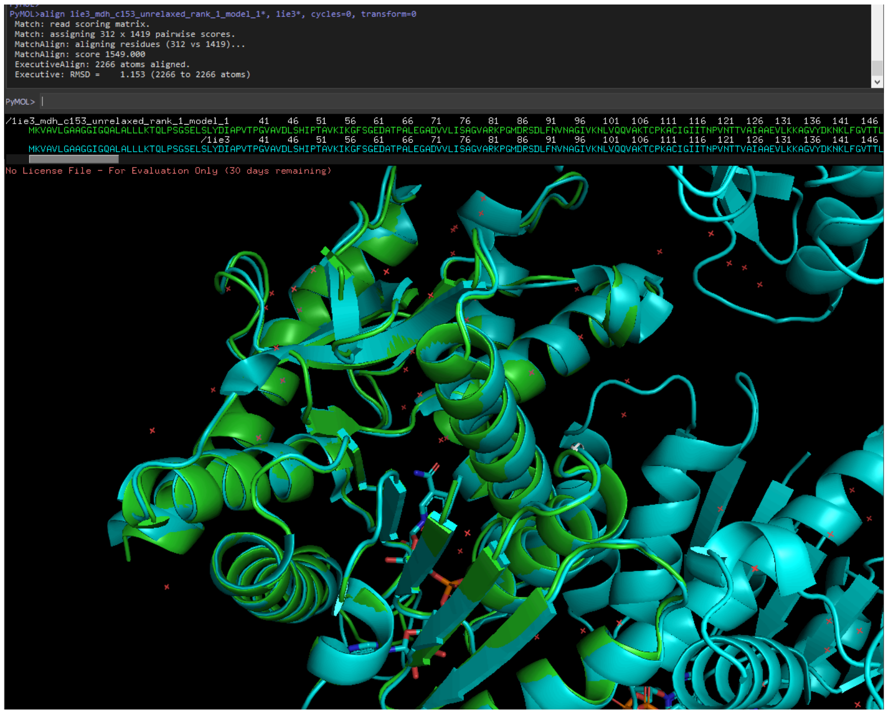
Completed

* The best metric field is obtained from the min/max of primary metric achieved by a run

2D Scatter Chart

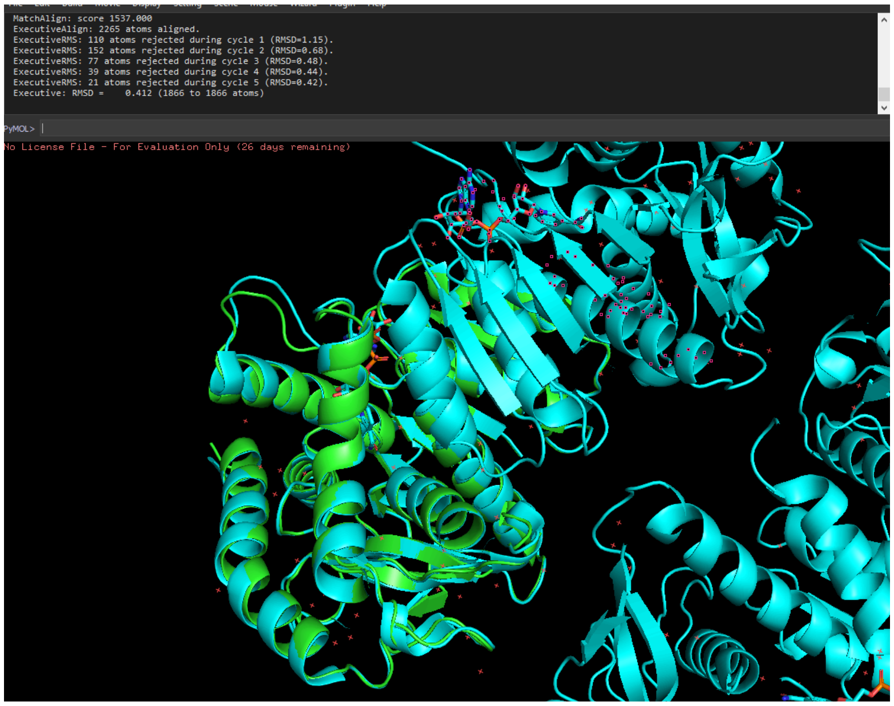
Click here to see the run in Azure Machine Learning studio

AlphaFold Mutant Protein Overlayed With Incorrect PDB Reference Structure:



```
PyMOL>align 1ie3_mdh_c153_unrelaxed_rank_1_model_1*, 1ie3*, cycles=0, transform=0
 Match: read scoring matrix.
 Match: assigning 312 x 1419 pairwise scores.
 MatchAlign: aligning residues (312 vs 1419)...
 MatchAlign: score 1549.000
 ExecutiveAlign: 2266 atoms aligned.
 Executive: RMSD =    1.153 (2266 to 2266 atoms)

PyMOL> |

/1ie3_mdh_c153_unrelaxed_rank_1_model_1      41   46   51   56   61   66   71   76   81   86   91   96   101  106  111  116  121  126  131  136  141  146
    MKVAVLGAAGGIGQALALLLKTQLPSGSELSLYDIAPVTPGVAVDLSHIPTAVKIKGFSGEDATPALEGADVVLISAGVARKPGMDRSDLFNVNAGIVKNLVQQVAKTCPKACIGIITNPVNTTVAIAAEVLKKAGVYDKNKLFGVTTL
                                       /1ie3      41   46   51   56   61   66   71   76   81   86   91   96   101  106  111  116  121  126  131  136  141  146
    MKVAVLGAAGGIGQALALLLKTQLPSGSELSLYDIAPVTPGVAVDLSHIPTAVKIKGFSGEDATPALEGADVVLISAGVARKPGMDRSDLFNVNAGIVKNLVQQVAKTCPKACIGIITNPVNTTVAIAAEVLKKAGVYDKNKLFGVTTL
No License File - For Evaluation Only (30 days remaining)
```

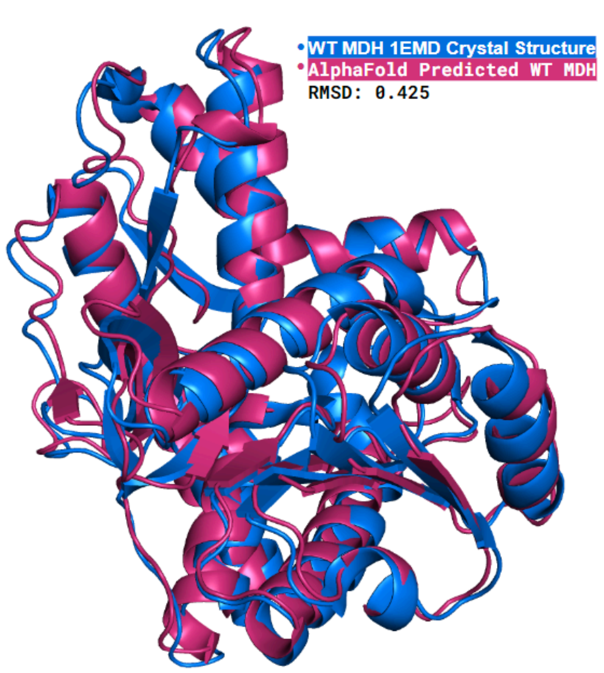
AlphaFold Wild Type Protein Overlayed With Incorrect PDB Reference Structure:

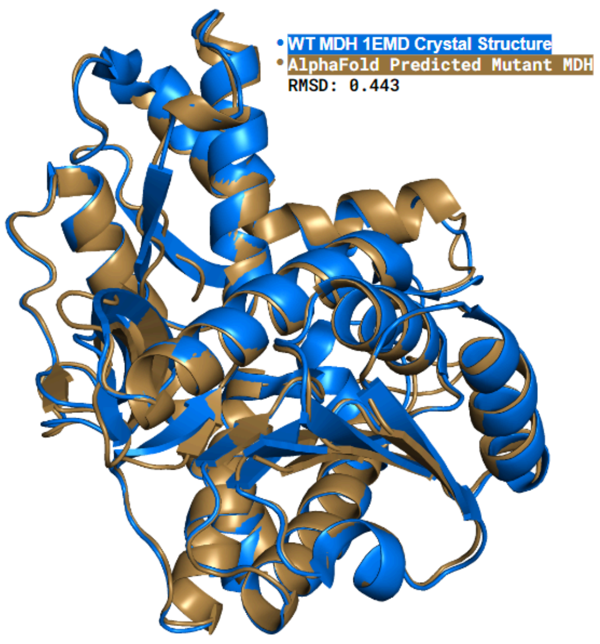


```
 MatchAlign: score 1537.000
 ExecutiveAlign: 2265 atoms aligned.
 ExecutiveRMS: 110 atoms rejected during cycle 1 (RMSD=1.15).
 ExecutiveRMS: 152 atoms rejected during cycle 2 (RMSD=0.68).
 ExecutiveRMS: 77 atoms rejected during cycle 3 (RMSD=0.48).
 ExecutiveRMS: 39 atoms rejected during cycle 4 (RMSD=0.44).
 ExecutiveRMS: 21 atoms rejected during cycle 5 (RMSD=0.42).
 Executive: RMSD =    0.412 (1866 to 1866 atoms)

PyMOL> |
No License File - For Evaluation Only (26 days remaining)
```

AlphaFold Wild Type Protein Overlayed With Correct PDB Reference Structure (1EMD):

- wt_mdh_r153_unrelaxed_rank_1_model_1.pdb from AlphaFold had an RMSD of 0.425 (1794 to 1794 atoms) in pyMOL Molecular Visualization using “orient” and “align 1emd, wt_mdh_r153_unrelaxed_rank_1_model_1" commands. Mutant (R153C) MDH from AlphaFold with filename mdh_c153_unrelaxed_rank_1_model_1.pdb had RMSD 0.443 (1810 to 1810 atoms) using "orient" and "align 1emd, mdh_c153_unrelaxed_rank_1_model_1" commands.





Conclusion: AlphaFold was successfully setup in the Microsoft Azure Virtual Machine with HyperDrive able to concurrently feed the FASTA sequences into the program through the Jupyter Notebook. Products of AlphaFold were overlaid in PyMol with the crystal reference structure 1EMD from the Protein Data Base. The Root Mean Squared Deviation scores indicate high accuracy of the program when the product's similarity is compared to the reference. Rosetta AbInitio was unable to be compiled in the Azure VM, prompting searching for an alternative to preserve the 'concurrent' modeling aspect to maintain high efficiency and throughput of the programs. Inability possible due to corrupted archive, filesystem limitations, permission issues, or version incompatibility. Integrity of the files and commands were confirmed with root user, but Rosetta could not be compiled in the Azure environment.

***Note: Failed** Adapted Script for Rosetta in Azure VM:*

Cell 1: Azure ML Compute Setup

```
from azureml.core.compute import ComputeTarget, AmlCompute
from azureml.core.compute_target import ComputeTargetException

cluster_name = "rosetta-cluster"

try:
    # Check for existing compute target
    training_cluster = ComputeTarget(workspace=ws, name=cluster_name)
    print('Found existing cluster.')
except ComputeTargetException:
    # If it doesn't already exist, create it
    try:
        compute_config = AmlCompute.provisioning_configuration(vm_size='Standard_DC48ds_v3', max_nodes=4)
        training_cluster = ComputeTarget.create(ws, cluster_name, compute_config)
        training_cluster.wait_for_completion(show_output=True)
    except Exception as ex:
        print(ex)
```

Cell 2: Environment and Script Configuration

```
from azureml.core import Experiment, ScriptRunConfig, Environment
from azureml.core.conda_dependencies import CondaDependencies

# Create a Python environment for the experiment
rosetta_env = Environment("rosetta-environment")
rosetta_env.docker.base_image = "YOUR_BASE_IMAGE"  # Replace with the appropriate base image for Rosetta
rosetta_env.python.user_managed_dependencies = True

# Create a script config
script_config = ScriptRunConfig(source_directory=".",
                                script='run_rosetta.py',
                                arguments=['--fasta_file', 'input.fasta',
                                           '--frag3_file', 'frag_3mers',
                                           '--frag9_file', 'frag_9mers'],
                                environment=rosetta_env,
                                compute_target=training_cluster)
```

Cell 3: Rosetta Abinitio Prediction Script

```
%%writefile run_rosetta.py
# Import necessary libraries
import argparse
from azureml.core import Run

# Set up argument parser
parser = argparse.ArgumentParser()
parser.add_argument("--fasta_file", type=str, help="Path to FASTA file")
parser.add_argument("--frag3_file", type=str, help="Path to 3-mer fragment file")
parser.add_argument("--frag9_file", type=str, help="Path to 9-mer fragment file")
args = parser.parse_args()
run = Run.get_context()

# TODO: Add code to run Rosetta AbInitio with the provided files
run.complete()
```

Docker Image Creation for Rosetta AbInitio:

```
git clone https://github.com/Metaphorme/Rosetta2Go.git
cd Rosetta2Go
```

```
git checkout 3.13
chmod +x build4docker.sh
./build4docker.sh
# update notebook environment configuration:
rosetta_env = Environment("rosetta-environment")
rosetta_env.docker.base_image = "rosetta2go:3.13"
rosetta_env.python.user_managed_dependencies = True
```

Compile Rosetta:

tar -xvjf rosetta.binary.linux.release-362.tar.bz2 **or** lbzip2 -dc rosetta.binary.linux.release-362.tar.bz2 | tar xvf -

12/29/23

Rosetta AbInitio in Oracle VirtualBox Ubuntu Virtual Machine

Objective: The purpose of this section is to outline the alternative to setting up Rosetta in the Azure VM. Due to incompatibility between the program and the environment, Rosetta will be set up in a personal Ubuntu environment.

Procedures:

Downloading and Installing Ubuntu VM:

1. New system with sufficient computing power was built for Ubuntu Virtual Machine
   a. Ryzen 9 5900X
   b. Asus RTX 3060
   c. 32 GB DDR4 RAM
   d. 1 TB SSD Storage
   e. 1200 W Gold Rated PSU
2. Oracle VirtualBox downloaded
3. New Virtual Machine setup initiated
4. Destination folder automatically set
5. Ubuntu 24.04.4 Desktop ISO Image downloaded from Canonical Ubuntu
6. ISO selected in Oracle VirtualBox setup
7. CPU and Memory allocated

Compiling Rosetta AbInitio:

1. Register for account at University of Washington COMOTION
2. Download Rosetta Academic License and upload to the virtual machine
3. Compile:

```
sudo apt install zlibig-dev scons build-essential -y
tar -xvzf rosetta_src_<version>_bundle.tgz
cd {ROSETTA}/main/source
./scons.py -j<number_of_processors_to_use> mode=release bin
```

Running Rosetta AbInitio:

1. Parameters:
   a. Download Reference PDB Structure
   b. Retreive FASTA Sequence
   c. Register with old.robetta fragment server and submit FASTA queries to Fragment Libraries to receive fragment files
      i. Download 09_05.200_v1_3, 03_05.200_v1_3, and .psipred_ss2 files.
   d. Create Flag Files:
      i. Delineate locations of files, and inherent parameters of Rosetta

WT Flag File	Mutant Flag File
-database /home/modelingvm/Rosetta/main/database -in:file:native ./wt_1emd.pdb -in:file:fasta ./wt_rcsb_pdb_1EMD.fasta -in:file:frag3 ./wt_aat000_03_05.200_v1_3 -in:file:frag9 ./wt_aat000_09_05.200_v1_3 -psipred_ss2 ./wt_t000_.psipred_ss2 -nstruct 50 -abinitio:relax -use_filters true -abinitio::increase_cycles 10 -abinitio::rg_reweight 0.5 -abinitio::rsd_wt_helix 0.5 -abinitio::rsd_wt_loop 0.5 -relax::fast -out:file:silent ./50x_wt_fold_silent.out	-database /home/modelingvm/Rosetta/main/database -in:file:native ./mut_1emd.pdb -in:file:fasta ./mut_rcsb_pdb_1EMD.fasta -in:file:frag3 ./mut_aat000_03_05.200_v1_3 -in:file:frag9 ./mut_aat000_09_05.200_v1_3 -psipred_ss2 ./mut_t000_.psipred_ss2 -nstruct 50 -abinitio:relax -use_filters true -abinitio::increase_cycles 10 -abinitio::rg_reweight 0.5 -abinitio::rsd_wt_helix 0.5 -abinitio::rsd_wt_loop 0.5 -relax::fast -out:file:silent ./50x_mut_fold_silent.out

Formatting of flag files to produce 50 structures each of wt and mutant MDH. All data reported in silent file.

2. Download and install GNU screens to separate modeling queries:

```
sudo apt-get update
sudo apt-get installs screen
screen -S session_name
screen -ls
screen -r session_name
```

3. Run Rosetta AbInitio:

```
modelingvm@modelingvm:~$ Rosetta/main/source/bin/AbinitioRelax.default.linuxgccrelease @mut_flags.txt
modelingvm@modelingvm:~$ Rosetta/main/source/bin/AbinitioRelax.default.linuxgccrelease @wt_flags.txt
```

4. Extracting the pdbs (wt)

```
grep SCORE 50x_wt_fold_silent.out | sort -nk 2 | awk {'print $32'} | more
grep SCORE 50x_wt_fold_silent.out | sort -nk 2 | awk {'print $32'} > wt_temp
cat wt_temp | awk '{print}' ORS=" " > ./wt_liststring
xargs Rosetta/main/source/bin/extract_pdbs.default.linuxgccrelease -in::file::silent ./50x_wt_fold_silent.out -out:pdb
-in:file:tags < wt_liststring
grep SCORE 50x_wt_fold_silent.out | awk '{print $2 "\t" $32}' | sort -nk 1 > ./wt_score_against_pdbs.txt
```

5. Extracting the pdbs (mut)

```
grep SCORE 50x_mut_fold_silent.out | sort -nk 2 | awk {'print $32'} | more
grep SCORE 50x_mut_fold_silent.out | sort -nk 2 | awk {'print $32'} > mut_temp
cat mut_temp | awk '{print}' ORS=" " > ./mut_liststring
xargs Rosetta/main/source/bin/extract_pdbs.default.linuxgccrelease -in::file::silent ./50x_mut_fold_silent.out -out:pdb
-in:file:tags < mut_liststring
grep SCORE 50x_mut_fold_silent.out | awk '{print $2 "\t" $32}' | sort -nk 1 > ./mut_score_against_pdbs.txt
```

6. Make VALUES_LIST files for 50x mut and wt:

```
grep SCORE 50x_mut_fold_silent.out | awk '{print $2 "\t" $27}' | sort -nk 2 > 50x_mut_VALUELIST.txt
grep SCORE 50x_wt_fold_silent.out | awk '{print $2 "\t" $27}' | sort -nk 2 > 50x_wt_VALUELIST.txt
```

7. Overlay best Rosetta Outputs in PyMol

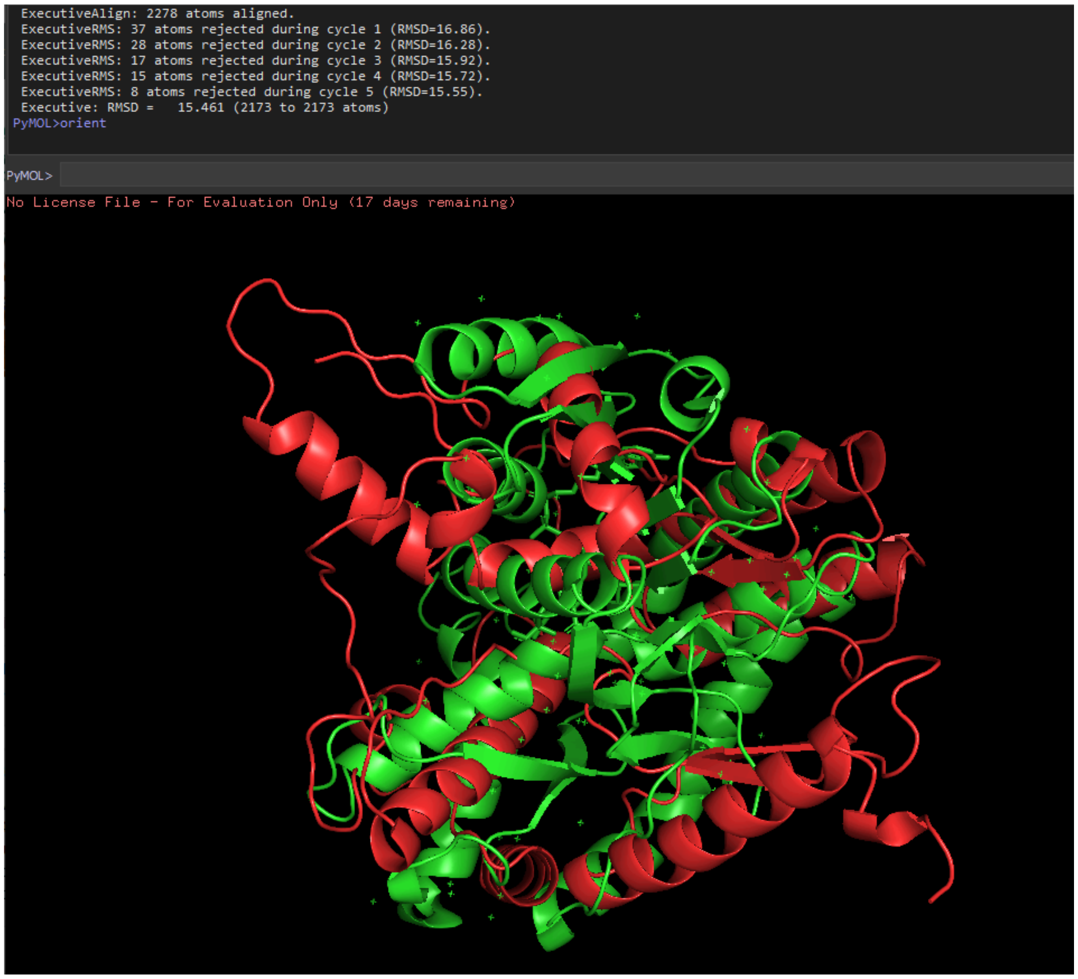
Analysis of RMSD from 50x Rosetta Output:

1. Given 50x_wt_VALUELIST.txt and 50x_mut_VALUELIST.txt, create stripchart in R to show distribution of the RMSD values for all 100 runs (50 each):

~~~
```{r}
MutValues$Type <- 'Mutant'
WTValues$Type <- 'Wild Type'
combinedData <- rbind(MutValues, WTValues)

ggplot(combinedData, aes(x = Type, y = rms, color = Type)) +
  geom_jitter(width = 0.12, height = 0.1) +
  labs(title = "50x RMSD Comparison: R153C Mutant vs. Wild Type MDH",
      x = "Type",
      y = "RMSD") +
  scale_color_brewer(palette = "Set1") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_y_continuous(breaks = seq(floor(min(combinedData$rms)-5), ceiling(max(combinedData$rms)+2),
by = 2))
```
~~~

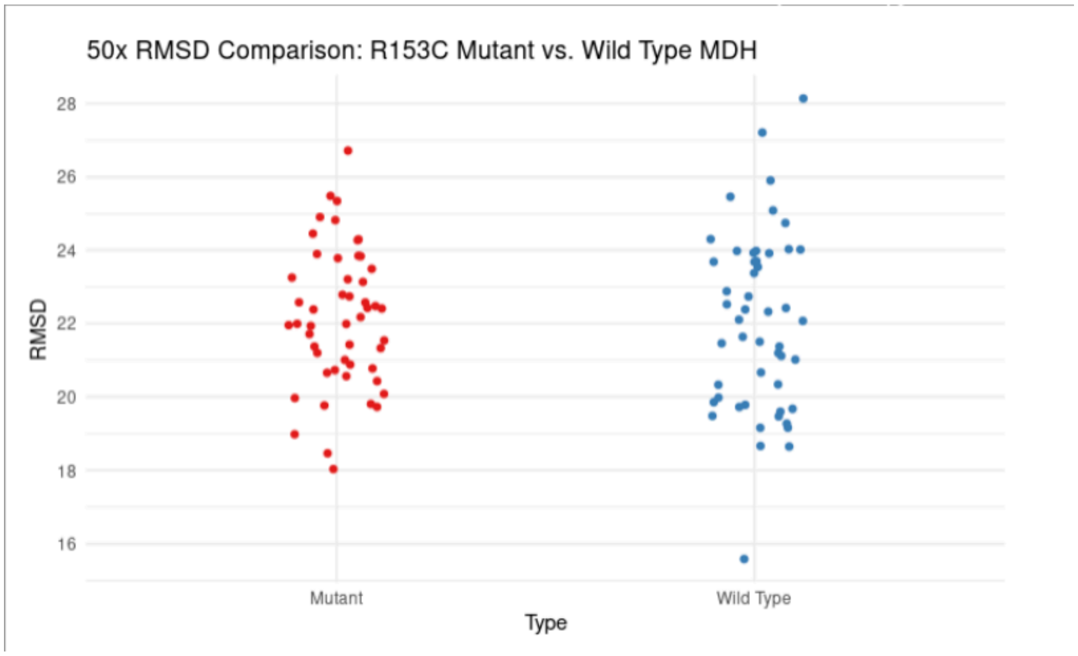
Data:

Rosetta AbInitio 7x Trial Best Output Overlayed with Reference Structure (RMSD:16):



```
 ExecutiveAlign: 2278 atoms aligned.
 ExecutiveRMS: 37 atoms rejected during cycle 1 (RMSD=16.86).
 ExecutiveRMS: 28 atoms rejected during cycle 2 (RMSD=16.28).
 ExecutiveRMS: 17 atoms rejected during cycle 3 (RMSD=15.92).
 ExecutiveRMS: 15 atoms rejected during cycle 4 (RMSD=15.72).
 ExecutiveRMS: 8 atoms rejected during cycle 5 (RMSD=15.55).
 Executive: RMSD =   15.461 (2173 to 2173 atoms)
PyMOL>orient
```

Rosetta AbInitio Strip Chart of 50x runs for wt and mutant MDH:



Conclusions: To keep the vital aspect of the project and increase the efficiency of computational modeling by concurrently modeling the proteins, Rosetta was set up in an Ubuntu VM with GNU screens installed. GNU screens allow allocation of one query to one core of the CPU, overcoming the limitation of Rosetta's inherent lack of multicore processing. The products of Rosetta were unusable in AutoDock analysis, yielding RMSD values well above the required threshold (0-2 for significant similarity).

1/1/24

AutoDock Vina Enzyme-Substrate Binding

Objective: The goal of the AutoDock Binding simulations was to assess the binding of MDH with oxaloacetate before and after the simple missense R153C mutation. The two proteins from AlphaFold would be used as the protein PDB model paired with a downloaded oxaloacetate file from PubChem. Docking would occur with both models from AlphaFold since it produced the only viable models within the RMSD range necessary (RMSD 0-2).

Procedures:

Preparing the PDB Outputs from AlphaFold:

1. Import protein.pdb and Edit → delete water (ensures accurate affinity)
2. Edit → add hydrogens (all hydrogens, noBondOrder (fine since auto dock uses atom type, not bond order) renumber atoms: No
3. Edit → charges → Kollman charges (known was 2.0)
4. Grid → macromolecules → choose protein
5. Save it as the new file format

Preparing the ligand from PubChem:

1. Open ligand sdf in pyMOL
2. Export as pdb

Setup in AutoDock Tools:

1. Download and open AutoDock Tools/ MGLTools in Ubuntu VM or windows machine to use graphical user interface
2. Ligand → choose → ligand → message reports accurate setup of oxaloacetate
3. Ligand → output → as pdbqt file
4. Setting grid box:
   a. Right-click protein under All Molecules → show sequences → click residues of interest one after another (81,87,153) (now they're highlighted in yellow)
5. Adjust offset and dimensions to put the residues in the grid space (leaving extra space for rotations)
6. Import all pdb structures and export as .pdbqt format for docking analysis

Setup Config File:

1. Center and size will be x,y, and z centers from the grid box Grid Options Window. The size parameters are the number of points for x,y, and z from the Grid Options Window. Format:

```
receptor = protein.pdbqt
ligand = ligand.pdbqt

center_x =  2
center_y =  6
```

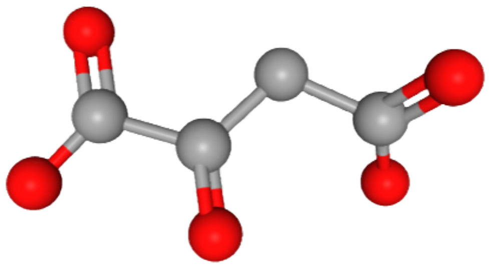
```
center_z = -7

size_x = 25
size_y = 25
size_z = 25

energy_range = 4
```

__Running AutoDock In CommandLine:__

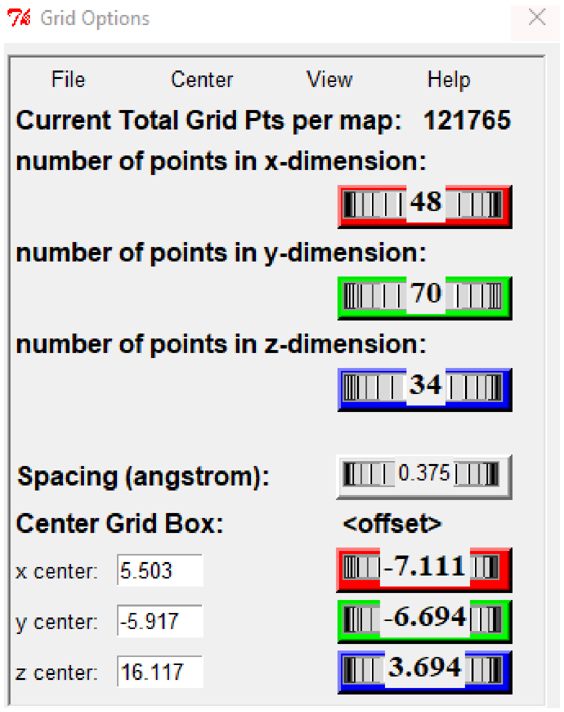
1. Locate files
   a. C:\Users\Andy\Downloads>"C:\Program Files (x86)\The Scripps Research Institute\Vina\vina.exe" --receptor protein.pdbqt --ligand ligand.pdbqt --config config.txt --log Vina_WT_Log.txt --out Vina_WT_Output.pdbqt

Data:

Oxaloacetate structure downloaded from PubChem in sdf format:

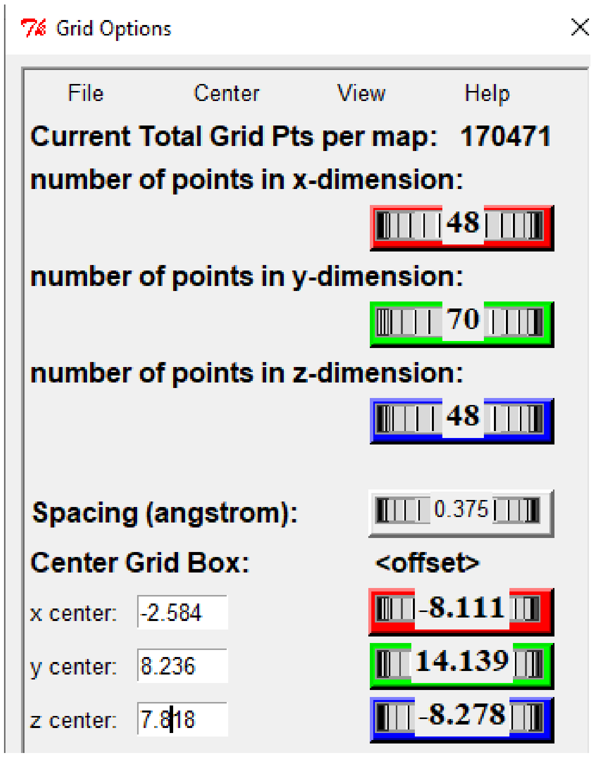


Wild Type grid box grid options window parameters:



Current Total Grid Pts per map: 121765
number of points in x-dimension: 48
number of points in y-dimension: 70
number of points in z-dimension: 34
Spacing (angstrom): 0.375
Center Grid Box: <offset>
x center: 5.503 -7.111
y center: -5.917 -6.694
z center: 16.117 3.694

Mutant grid box grid options window parameters:


Grid Options

File Center View Help

Current Total Grid Pts per map: 170471

number of points in x-dimension: 48

number of points in y-dimension: 70

number of points in z-dimension: 48

Spacing (angstrom): 0.375

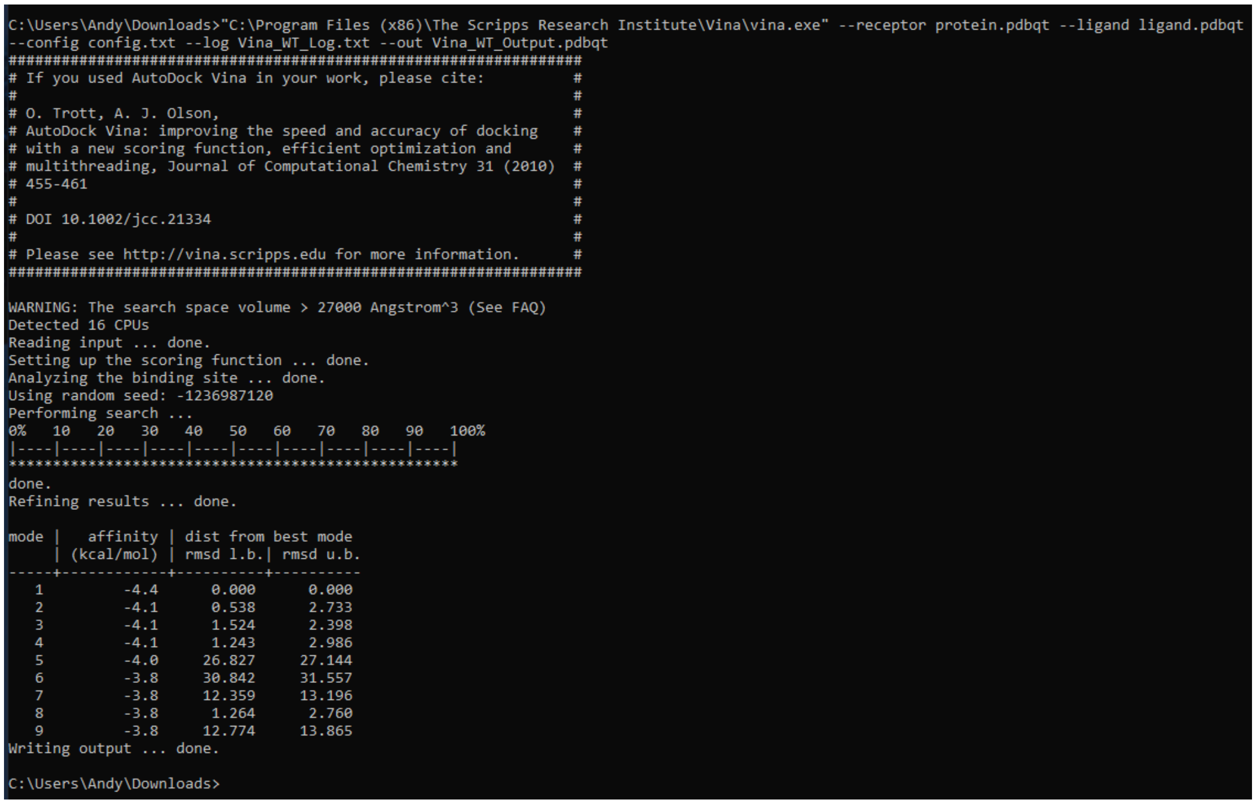
Center Grid Box: <offset>

x center: -2.584 -8.111

y center: 8.236 14.139

z center: 7.818 -8.278

Commandline results of AutoDock binding:


```
C:\Users\Andy\Downloads>"C:\Program Files (x86)\The Scripps Research Institute\Vina\vina.exe" --receptor protein.pdbqt --ligand ligand.pdbqt
--config config.txt --log Vina_WT_Log.txt --out Vina_WT_Output.pdbqt
#################################################################
# If you used AutoDock Vina in your work, please cite:          #
#                                                               #
# O. Trott, A. J. Olson,                                        #
# AutoDock Vina: improving the speed and accuracy of docking    #
# with a new scoring function, efficient optimization and       #
# multithreading, Journal of Computational Chemistry 31 (2010)  #
# 455-461                                                       #
#                                                               #
# DOI 10.1002/jcc.21334                                         #
#                                                               #
# Please see http://vina.scripps.edu for more information.      #
#################################################################

WARNING: The search space volume > 27000 Angstrom^3 (See FAQ)
Detected 16 CPUs
Reading input ... done.
Setting up the scoring function ... done.
Analyzing the binding site ... done.
Using random seed: -1236987120
Performing search ...
0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***************************************************
done.
Refining results ... done.

mode |   affinity | dist from best mode
     | (kcal/mol) | rmsd l.b.| rmsd u.b.
-----+------------+----------+----------
   1         -4.4      0.000      0.000
   2         -4.1      0.538      2.733
   3         -4.1      1.524      2.398
   4         -4.1      1.243      2.986
   5         -4.0     26.827     27.144
   6         -3.8     30.842     31.557
   7         -3.8     12.359     13.196
   8         -3.8      1.264      2.760
   9         -3.8     12.774     13.865
Writing output ... done.

C:\Users\Andy\Downloads>
```

Reformatted results of AutoDock binding for wt and mutant MDH:

Wild Type Docking Results				**Mutant Docking Results**			
Mode	Affinity	RMSD Lower Bound	RMSD Upper Bound	Mode	Affinity	RMSD Lower Bound	RMSD Upper Bound
1	-4-4	0.000	0.000	**1**	-5.8	0.000	0.000
2	-4.1	0.538	2.733	**2**	-5.1	24.884	25.615
3	-4.1	1.524	2.398	**3**	-5.0	35.368	36.027
4	-4.1	1.243	2.986	**4**	-5.0	24.334	24.923
5	-4.0	26.827	27.144	**5**	-4.9	20.821	21.230
6	-3.8	30.842	31.557	**6**	-4.9	30.581	31.115
7	-3.8	12.359	13.196	**7**	-4.8	30.528	31.200
8	-3.8	1.264	2.760	**8**	-4.8	24.854	25.587
9	-3.8	12.774	13.865	**9**	-4.8	20.865	21.455

Conclusions: The use of MGLTools and PyMol to reformat and input the objects into AutoDock Vina for binding analysis was largely successful, only encountering few errors with file naming of the mutant instance. Grid box parameters were set around the Arginine residues involved in substrate coordination, as they were the residues of interest in the study, not the full active site. Enough space was provided around the residues to let AutoDock attempt binding in many conformations, as would be expected in the protein. The deviation in RMSD of the mutant results indicate a large amount of energetically unfavorable conformations between MDH and the substrate, supported by laboratory data reporting more flexible binding and incorrect binding between MDH and many substrates. Mutant affinity values show oxaloacetate binding tighter to MDH, unexpected, but explainable as MDH may potentially be binding so tight it inhibits proper enzymatic activity/ enzymatic chemistry.

References

Accurate Prediction of Protein Structures and Interactions Using a Three-Track Neural Network | Science. https://www.science.org/doi/10.1126/science.abj8754. Accessed 27 Oct. 2023.

Baek, Minkyung, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N. Kinch, R. Dustin Schaeffer, Claudia Millán, Hahnbeom Park, Carson Adams, Caleb R. Glassman, Andy DeGiovanni, Jose. pp. 871–76. *PubMed Central*, https://doi.org/10.1126/science.abj8754.

Bell, Jessica K., et al. "Structural Analyses of a Malate Dehydrogenase with a Variable Active Site *." *Journal of Biological Chemistry*, vol. 276, no. 33, Aug. 2001, pp. 31156–62. *www.jbc.org*, https://doi.org/10.1074/jbc.M100902200.

Bitencourt-Ferreira, Gabriela, et al. "Docking with AutoDock4." *Methods in Molecular Biology (Clifton, N.J.)*, vol. 2053, 2019, pp. 125–48. *PubMed*, https://doi.org/10.1007/978-1-4939-9752-7_9.

Bonneau, R., et al. "Rosetta in CASP4: Progress in Ab Initio Protein Structure Prediction."

Ford, C. (2022). azureml-alphafold2 [Software]. GitHub, from https://github.com/colbyford/azureml-alphafold2

H. Pereira, Andria V. Rodrigues, Alberdina A. van Dijk, Ana C. Ebrecht, Diederik J. Opperman, Theo Sagmeister, Christoph Buhlheller, Tea Pavkov-Keller, Manoj K Rathinaswamy, et al. "Accurate Prediction of Protein Structures and Interactions Using a 3-Track Neural Network." Science (New York, N.Y.), vol. 373, no. 6557, Aug. 2021,

Hung, Chih-Hung, et al. "Crystal Structures and Molecular Dynamics Simulations of Thermophilic Malate Dehydrogenase Reveal Critical Loop Motion for Co-Substrate Binding." *PLoS ONE*, vol. 8, no. 12, Dec. 2013, p. e83091. *PubMed Central*, https://doi.org/10.1371/journal.pone.0083091.

Jumper, John, et al. "Highly Accurate Protein Structure Prediction with AlphaFold." *Nature*, vol. 596, no. 7873, 7873, Aug. 2021, pp. 583–89. *www.nature.com*, https://doi.org/10.1038/s41586-021-03819-2.

Koehler Leman, Julia, et al. "Macromolecular Modeling and Design in Rosetta: Recent Methods and Frameworks." *Nature Methods*, vol. 17, no. 7, July 2020, pp. 665–80. *PubMed Central*, https://doi.org/10.1038/s41592-020-0848-2.

Lyskov, Sergey, and Jeffrey J. Gray. "The RosettaDock Server for Local Protein–Protein Docking." Nucleic Acids Research, vol. 36, no. Web Server issue, July 2008, pp. W233–38. PubMed Central, https://doi.org/10.1093/nar/gkn216.

Moretti, Rocco, et al. "Web‐accessible Molecular Modeling with Rosetta: The Rosetta Online Server That Includes Everyone (ROSIE)." *Protein Science : A Publication of the Protein Society*, vol. 27, no. 1, Jan. 2018, pp. 259–68. *PubMed Central*, https://doi.org/10.1002/pro.3313.

Mirdita M, Schütze K, Moriwaki Y, Heo L, Ovchinnikov S, Steinegger M. ColabFold: making protein folding accessible to all. Nat Methods. 2022 Jun;19(6):679-682. doi: 10.1038/s41592-022-01488-1. Epub 2022 May 30. PMID: 35637307;

PMCID: PMC9184281.

Murthy, Mukundh. “Ab Initio Protein Folding.” *Medium*, 11 May 2020, https://mukundh-murthy.medium.com/ab-initio-protein-folding-be27509d134a. ---. “Ab Initio Protein Folding.” *Medium*, 11 May 2020, https://mukundh-murthy.medium.com/ab-initio-protein-folding-be27509d134a.

Ó Conchúir S, Barlow KA, Pache RA, Ollikainen N, Kundert K, O'Meara MJ, Smith CA, Kortemme T. A Web Resource for Standardized Benchmark Datasets, Metrics, and Rosetta Protocols for Macromolecular Modeling and Design. PLoS One. 2015 Sep 3;10(9):e0130433. doi: 10.1371/journal.pone.0130433. PMID: 26335248; PMCID: PMC4559433.

O. Trott, A. J. Olson, AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization and multithreading, Journal of Computational Chemistry 31 (2010) 455-461

Proteins, vol. Suppl 5, 2001, pp. 119–26. *PubMed*, https://doi.org/10.1002/prot.1170. dinesh-supreme. “Why Structure Prediction Matters.” *DNASTAR*, 21 July 2020, https://www.dnastar.com/blog/protein-analysis-modeling/why-structure-prediction-matters/.

Pricing - Linux Virtual Machines | Microsoft Azure. https://azure.microsoft.com/en-us/pricing/details/virtual-machines/linux/.

Rämisch, Sebastian. "De Novo Structure Prediction." RosettaCommons, n.d., https://www.rosettacommons.org/demos/latest/tutorials/denovo_structure_prediction/Denovo_structure_prediction.

Structural Analyses of a Malate Dehydrogenase with a Variable Active Site - PubMed. https://pubmed.ncbi.nlm.nih.gov/11389141/. Accessed 10 Nov. 2023.

Takahashi-Íñiguez T, Aburto-Rodríguez N, Vilchis-González AL, Flores ME. Function, kinetic properties, crystallization, and regulation of microbial malate dehydrogenase. J Zhejiang Univ Sci B. 2016 Apr;17(4):247–61. doi: 10.1631/jzus.B1500219. PMCID: PMC4829630.