

## Table of Contents

1. Code	
1.1 Alzymes	2
1.1.1 Alzymes_MAIN	2
1.1.2 __init__	[page]
1.1.3 setup	[page]
1.1.4 initialize	[page]
1.1.5 controller	[page]
1.1.6 plot	[page]
1.2 design_ESMfold	3
1.2.1 prepare_ESMfold	[page]
1.3 design_LigandMPNN	3
1.3.1 prepare_LigandMPNN	[page]
1.4 design_match	4
1.5 design_ProteinMPNN	4
1.5.1 prepare_ProteinMPNN	[page]
1.6 design_RosettaDesign	4
1.6.1 prepare_RosettaDesign	[page]
1.7 design_RosettaRelax	5
1.7.1 prepare_RosettaRelax	[page]
1.8 helper	5
1.8.1 run_command	[page]
1.8.2 get_PDB_in	[page]
1.8.3 save_cat_res_into_all_scores_df	[page]
1.8.4 reset_to_after_index	[page]
1.8.5 wait_for_file	[page]
1.9 main_design	6
1.10 main_running	6
1.10.1 start_controller	[page]
1.10.2 check_running_jobs	[page]
1.10.3 update_potential	[page]
1.10.4 update_potential	[page]
1.10.5 update_scores	[page]
1.10.6 boltzmann_selection	[page]
1.10.7 check_parent_done	[page]
1.10.8 start_parent_design	[page]
1.10.9 start_calculation	[page]
1.10.10 create_new_index	[page]
1.11 main_scripts	8
1.12 main_startup	8
1.13 plotting	8
1.13.1 plot_interface_v_total_score_selection	[page]
1.13.2 plot_interface_v_total_score_generation	[page]
1.13.3 plot_stacked_histogram_by_cat_resi	[page]
1.13.4 plot_stacked_histogram_by_cat_resn	[page]
1.13.5 plot_stacked_histogram_by_generation	[page]
1.14 scoring_efields	8
1.14.1 generate_AMBER_files	[page]
1.14.2 calc_efields_score	[page]
1.14.3 update_efieldsdf	[page]
1.15 setup_system	9

# 1. Code

## 1.1 Alzymes

### Alzymes Project Main Workflow

This script defines the main Alzymes workflow, including setup, initialization, control, and plotting functions. It manages the primary processes and configurations required to execute Alzymes functionalities.

#### Classes

Alzymes_MAIN	Manages the main workflow for Alzymes, including setup, initialization, and various control functions.
--------------	--

#### Functions

__init__()	Initializes an instance of the Alzymes_MAIN class.
setup()	Sets up the Alzymes project environment with specified parameters.
initialize()	Initializes Alzymes with provided configurations.
controller()	Controls the Alzymes project based on scoring and normalization parameters.
plot()	Generates various plots based on Alzymes data.

### 1.1.1 Alzymes\_MAIN

Main class for managing Alzymes workflow, including setup, initialization, control, and plotting functions.

### 1.1.2 \_\_init\_\_

Initializes an instance of the Alzymes\_MAIN class.

### 1.1.3 setup

Sets up the Alzymes project environment with specified parameters.

#### Args

FOLDER_HOME (str)	Path to the main folder.
FOLDER_PARENT (str)	Path to the parent folder.
CST_NAME (str)	Constraint name.
WT (str)	Wild type information.
LIGAND (str)	Ligand data.
DESIGN (str)	Design specifications.
MAX_JOBS (int)	Maximum number of jobs to run concurrently.
N_PARENT_JOBS (int)	Number of parent jobs.
MAX_DESIGNS (int)	Maximum number of designs.
KBT_BOLTZMANN (list)	Boltzmann constant values.
CST_WEIGHT (float)	Constraint weight.
ProteinMPNN_PROB (float)	Probability parameter for ProteinMPNN.
ProteinMPNN_BIAS (float)	Bias parameter for ProteinMPNN.
LMPNN_PROB (float)	Probability parameter for LMPNN.
FOLDER_MATCH (str)	Path to match folder.
ProteinMPNN_T (str)	Temperature for ProteinMPNN.
LMPNN_T (str)	Temperature for LMPNN.
LMPNN_BIAS (float)	Bias parameter for LMPNN.
SUBMIT_PREFIX (str)	Submission prefix.
SYSTEM (str)	System information.
MATCH (str)	Match specifications.
ROSETTA_PATH (str)	Path to Rosetta source.
EXPLORE (bool)	Whether to explore parameter space.
FIELD_TOOLS (str)	Path to FieldTools script.
LOG (str)	Logging level.
PARENT_DES_MED (str)	Parent design method.

#### 1.1.4 initialize

Initializes Alzymes with given parameters.

##### Args

---

FOLDER_HOME (str)	Path to the main folder.
UNBLOCK_ALL (bool)	Flag to unblock all processes.
PRINT_VAR (bool)	Flag to print variables.
PLOT_DATA (bool)	Flag to plot data.
LOG (str)	Logging level.

#### 1.1.5 controller

Controls the Alzymes project based on scoring and normalization parameters.

##### Args

---

HIGHSCORE (float)	High score threshold for evaluation.
NORM (dict)	Normalization values for different scores.

#### 1.1.6 plot

Generates plots based on Alzymes data, including main, tree, and landscape plots.

##### Args

---

main_plots (bool)	Flag to generate main plots.
tree_plot (bool)	Flag to generate tree plot.
landscape_plot (bool)	Flag to generate landscape plot.
print_vals (bool)	Flag to print values on plots.
NORM (dict)	Normalization values for different scores.
HIGHSCORE_NEGBEST (dict)	High score and negative best score for different metrics.

## 1.2 design\_ESMfold

Design ESMfold Module

Manages the structure prediction of protein sequences using ESMfold within the Alzymes project.

##### Functions

---

- prepare_ESMfold	Prepares commands for ESMfold job submission.
-------------------	---

#### 1.2.1 prepare\_ESMfold

Predicts structure of sequence in {index} using ESMfold.

Parameters: index (str): The index of the protein variant to be predicted. cmd (str): Growing list of commands to be executed by run\_design using submit\_job.

##### Returns

---

cmd (str)	Command to be executed by run_design using submit_job.
-----------	--

## 1.3 design\_LigandMPNN

NOTE: NOT WORKING YET!

Integrates LigandMPNN for generating protein sequences adapted to specific ligand contexts.

##### Functions

prepare_LigandMPNN	Executes LigandMPNN steps to adapt protein designs with ligand binding considerations.
--------------------	--

#### Modules Required

helper\_001

### 1.3.1 prepare\_LigandMPNN

Executes the LigandMPNN pipeline for a given protein-ligand structure and generates new protein sequences with potentially higher functional scores considering the ligand context.

Parameters: - parent\_index (str): The index of the parent protein variant. - new\_index (str): The index assigned to the new protein variant. - all\_scores\_df (DataFrame): A DataFrame containing information for protein variants.

## 1.4 design\_match

#### Design Match Module

Provides functionalities for matching protein designs with specific constraints and requirements in the Alzymes workflow.

#### Functions

- prepare_LigandMPNN	Executes the LigandMPNN pipeline for protein-ligand structure adaptation.
----------------------	---

#### Modules Required

- helper\_001

## 1.5 design\_ProteinMPNN

Manages ProteinMPNN design steps to generate protein sequences tailored for specific functional and structural properties in the Alzymes project.

This function assumes the ProteinMPNN toolkit is available and properly set up in the specified location. It involves multiple subprocess calls to Python scripts for processing protein structures and generating new sequences.

#### Functions

prepare_ProteinMPNN	Sets up commands for ProteinMPNN job submission.
---------------------	--

#### Modules Required

helper\_001

### 1.5.1 prepare\_ProteinMPNN

Executes the ProteinMPNN pipeline for a given protein structure and generates new protein sequences with potentially higher functional scores.

#### Args

new_index (str)	The index of the designed variant.
cmd (str)	Growing list of commands to be executed by run_design using submit_job.

#### Returns

cmd (str)	Command to be executed by run_design using submit_job
-----------	---

## 1.6 design\_RosettaDesign

Handles RosettaDesign steps to optimize protein design scores and enhance stability within the Alzymes project.

#### Functions

---

prepare_RosettaDesign	Prepares RosettaDesign commands for job submission.
-----------------------	---

#### Modules Required

helper\_001

### 1.6.1 prepare\_RosettaDesign

Designs protein structure in {new\_index} based on {parent\_index} using RosettaDesign.

#### Args

---

parent_index (str)	Index of the parent protein variant to be designed.
new_index (str)	Index assigned to the resulting design.
input_suffix (str)	Suffix of the input structure to be used for design.

#### Returns

---

cmd (str)	Command to be executed by run_design using submit_job.
-----------	--

## 1.7 design\_RosettaRelax

Executes RosettaRelax to refine protein structures and improve stability in the Alzymes project.

#### Functions

---

prepare_RosettaRelax	Sets up commands for RosettaRelax job submission.
----------------------	---

#### Modules Required

helper\_001

### 1.7.1 prepare\_RosettaRelax

Relaxes protein structure in {index} using RosettaRelax.

Args index (str): The index of the protein variant to be relaxed. cmd (str): collection of commands to be run, this script will append its commands to cmd

Optional parameters: PreMatchRelax (bool): True if ESMfold to be run without ligand (prior to RosettaMatch).

## 1.8 helper

Contains utility functions and supporting routines used across multiple modules within the Alzymes project.

#### Functions

---

- normalize\_scores - one\_to\_three\_letter\_aa - run\_command - get\_PDB\_in - load\_main\_variables - save\_main\_variables - submit\_job - sequence\_from\_pdb - generate\_remark\_from\_all\_scores\_df - save\_cat\_res\_into\_all\_scores\_df - reset\_to\_after\_parent\_design - reset\_to\_after\_index - save\_all\_scores\_df - get\_best\_structures - remove\_intersection\_best\_structures - trace\_mutation\_tree - print\_average\_scores - wait\_for\_file - hamming\_distance - exponential\_func

---

#### Modules Required

setup\_system\_001

### 1.8.1 run\_command

Wrapper to execute .py files in runtime with arguments, and print error messages if they occur.

Parameters: command: The command to run as a list of strings. cwd: Optional; The directory to execute the command in.

capture\_output: Optional; If True, capture stdout and stderr. Defaults to False (This is to conserve memory).

### 1.8.2 get\_PDB\_in

Based on index, find the input PDB files for the Alzymes modules

Parameters: - index: The index of the current design

Output: - PDBfile\_Design\_in: Input file for RosettaDesign - PDBfile\_Relax\_in: Input file for RosettaRelax - PDBfile\_Relax\_ligand\_in: Input file for Ligand to be used in RosettaRelax

### 1.8.3 save\_cat\_res\_into\_all\_scores\_df

Finds the indices and names of the catalytic residue from Saves indices and residues into in row as lists. To make sure these are saved and loaded as list, ",".join() and .split(",") should be used If information is read from an input structure for design do not save cat\_resn

### 1.8.4 reset\_to\_after\_index

This function resets the run back to a chosen index. It removes all later entries from the all\_scores.csv and the home dir. index: The last index to keep, after which everything will be deleted.

### 1.8.5 wait\_for\_file

Wait for a file to exist and have a non-zero size.

## 1.9 main\_design

Main Design Module

Coordinates various design steps, managing the workflow of Rosetta, ProteinMPNN, and other modules within the Alzymes project.

#### Functions

---

- get_ram	Determines RAM allocation for design steps.
- run_design	Runs the selected design steps based on configuration.

#### Modules Required

- helper\_001, design\_match\_001, design\_ProteinMPNN\_001, design\_LigandMPNN\_001, design\_RosettaDesign\_001, design\_ESMfold\_001, design\_RosettaRelax\_001

## 1.10 main\_running

main\_running\_001.py

This module contains the main control functions for managing the Alzymes workflow, including job submission, score updating, and Boltzmann selection. The functions in this file are responsible for the high-level management of the design process, interacting with the Alzymes\_MAIN class to initiate, control, and evaluate design variants.

#### Classes

None

---

#### Functions

start\_controller(self) check\_running\_jobs(self) update\_potential(self, score\_type, index) update\_scores(self) boltzmann\_selection(self) check\_parent\_done(self) start\_parent\_design(self) start\_calculation(self, parent\_index) create\_new\_index(self, parent\_index)

---

### 1.10.1 start\_controller

Runs the main loop to manage the design process until the maximum number of designs is reached.

Continues submitting jobs, monitoring running processes, and updating scores based on Boltzmann selection until `MAX\_DESIGNS` is

achieved. The function pauses or starts new designs based on system resources.  
Parameters: self: An instance of the Alzymes\_MAIN class with setup attributes and properties.

#### 1.10.2 check\_running\_jobs

Checks the current number of running jobs based on the system type.  
Depending on the value of 'SYSTEM', this function counts the active jobs in GRID, BLUEPEBBLE, BACKGROUND\_JOB, or ABBIE\_LOCAL systems.

##### Returns

---

int	Number of running jobs for the specific system.
-----	---

#### 1.10.3 update\_potential

Updates the potential file for a given score type at the specified variant index.  
Creates or appends to a '\_potential.dat' file in 'FOLDER\_HOME/', calculating and updating potentials for the parent variant if necessary.  
Parameters: score\_type (str): Type of score to update (e.g., total, interface, catalytic, efield). index (int): Variant index to update potential data.

#### 1.10.4 update\_potential

Creates a \_potential.dat file in FOLDER\_HOME/  
If latest score comes from Rosetta Relax - then the score for variant will be added to the \_potential.dat of the parent index and the \_potential value of the dataframe for the parent will be updated with the average of the parent and child scores.  
Parameters: - score\_type(str): Type of score to update, one of these options: total, interface, catalytic, efield - index (int): variant index to update

#### 1.10.5 update\_scores

Updates various scores, including total, interface, catalytic, and efield scores for each design variant.  
This function iterates over design variants, updating scores based on files generated by different processes. It also updates sequence information, tracks mutations, and saves the updated DataFrame.  
Parameters: self: An instance of the Alzymes\_MAIN class with setup attributes and properties.

#### 1.10.6 boltzmann\_selection

Selects a design variant based on a Boltzmann-weighted probability distribution.  
Filters variants based on certain conditions (e.g., scores, block status), then computes probabilities using Boltzmann factors with a temperature factor ('KBT\_BOLTZMANN') to select a variant for further design steps.

##### Returns

---

int	Index of the selected design variant.
-----	---------------------------------------

#### 1.10.7 check\_parent\_done

Determines if parent designs are complete based on the number of generated designs and parent jobs.

##### Returns

---

bool	True if parent designs are complete, otherwise False.
------	---

#### 1.10.8 start\_parent\_design

Initiates a new design process for a parent structure by creating a new variant entry.  
Sets up the required files and configuration for designing a parent structure, then calls the design method specified in 'PARENT\_DES\_MED'.  
Parameters: self: An instance of the Alzymes\_MAIN class with setup attributes and properties.

#### 1.10.9 start\_calculation

Decides the next calculation step for the specified design variant index.  
Based on the current design state, this function decides to run ESMfold, RosettaRelax, or a design method for the given index.  
Parameters: parent\_index (int): Index of the variant to evaluate for further calculations.

#### 1.10.10 create\_new\_index

Creates a new design entry in `all\_scores\_df` with a unique index, inheriting attributes from a parent variant. Updates the DataFrame with new index information, saves the updated file, and sets up the directory structure for the new design variant.

Parameters: parent\_index (str): The index of the parent variant or "Parent" for initial designs.

#### Returns

---

int The newly created index for the variant.

### 1.11 main\_scripts

### 1.12 main\_startup

### 1.13 plotting

#### 1.13.1 plot\_interface\_v\_total\_score\_selection

Plots a scatter plot of total\_scores vs interface\_scores and highlights the points corresponding to the selected indices.

Parameters: - ax (matplotlib.axes.Axes): The Axes object to plot on. - total\_scores (list or np.array): The total scores of the structures. - interface\_scores (list or np.array): The interface scores of the structures. - selected\_indices (list of int): Indices of the points to highlight.

#### 1.13.2 plot\_interface\_v\_total\_score\_generation

Plots a scatter plot of total\_scores vs interface\_scores and colors the points according to the generation for all data points, using categorical coloring. Adds a legend to represent each unique generation with its corresponding color.

Parameters: - ax (matplotlib.axes.Axes): The Axes object to plot on. - total\_scores (list or np.array): The total scores of the structures. - interface\_scores (list or np.array): The interface scores of the structures. - generation (pd.Series or np.array): Generation numbers for all data points.

#### 1.13.3 plot\_stacked\_histogram\_by\_cat\_resi

Plots a stacked bar plot of interface scores colored by cat\_resi on the given Axes object, where each bar's segments represent counts of different cat\_resi values in that bin.

Parameters: - ax (matplotlib.axes.Axes): The Axes object to plot on. - all\_scores\_df (pd.DataFrame): DataFrame containing 'cat\_resi' and 'interface\_score' columns. - color\_map (dict): Optional; A dictionary mapping catalytic residue indices to colors. - show\_legend (bool): Optional; Whether to show the legend. Defaults to False.

#### 1.13.4 plot\_stacked\_histogram\_by\_cat\_resn

Plots a stacked bar plot of interface scores colored by cat\_resn on the given Axes object, where each bar's segments represent counts of different cat\_resn values in that bin.

Parameters: - ax (matplotlib.axes.Axes): The Axes object to plot on. - all\_scores\_df (pd.DataFrame): DataFrame containing 'cat\_resn' and 'interface\_score' columns.

#### 1.13.5 plot\_stacked\_histogram\_by\_generation

Plots a stacked bar plot of interface scores colored by generation on the given Axes object, where each bar's segments represent counts of different generation values in that bin.

Parameters: - ax (matplotlib.axes.Axes): The Axes object to plot on. - all\_scores\_df (pd.DataFrame): DataFrame containing 'generation' and 'interface\_score' columns.

### 1.14 scoring\_efields

#### 1.14.1 generate\_AMBER\_files

Uses tleap to create a .parm7 and .rst7 file from a pdb. Requires ambertools. Also requires 5TS.prepi and 5TS.frmod in the INPUT folder

TODO: Add script to generate these if not present.

Parameters: - filename (str): The path to the pdb file to analyse without the file extension.

#### 1.14.2 calc\_efields\_score

Executes the FieldTools.py script to calculate the electric field across the C-H bond of 5TS. Requires a field\_target.dat in the Input folder. Currently hard-coded based on 5TS

TODO: Make this function agnostic to contents of field\_target

Parameters: - pdb\_path (str): The path to the pdb structure to analyse - either from design or relax.

#### Returns

---



---

- bond_field (float)	The total electric field across the 5TS@C9_:5TS@H04 bond in MV/cm. (Currently hard coded to these atoms)
- all_fields (dict)	The components of the electric field across the 5TS@C9_:5TS@H04 bond per residue.

#### 1.14.3 update\_efieldsdf

Adds a new row to "{FOLDER\_HOME}/electric\_fields.csv" containing the electric fields generated by FieldTools.py for all residues in the protein

### 1.15 setup\_system