

# Refactoring plan

## Refactoring plan

### Objective

Want to restructure HDX model code for publication. Need to it be modular (i.e. reuse models and functions) and readable.

### Utility code

- lmfit fitting Model for time-course
- lmfit fitting Model for 2-state melt
- lmfit fitting Model for aggregate fit of kadd
- Arrhenius Linear model for kdeg independent
- global fit to get kdeg

### Main nerd

Nucleotide energetics from reactivity data Tools to extract nucleotide energy from chemical probing experiments.

#### **nerd import**

1. Sample import - from csv, import into sqlite3 db. Includes sequence info, nucleotide info, etc. input: sample sheet output: sqlite3 db
2. Import previously ran shapemapper folder input: shapemapper directory output: sqlite3 db (add directory)
3. Sample import (NMR) input: sample sheet output: sqlite3 db (nmr\_reactions)

#### **nerd fmod\_calc**

1. Run shapemapper input: sample, shapemapper config output: sqlite3 db (with new entry)
2. Run spat input: XX output: XX
3. Run other tools fmod\_calc tools

#### **nerd degradation**

1. NMR deg measurements exponential timecourse input: NMR peak integrals (csv) output: exponential fit
2. Global time-course fits for probing data input: timecourses (rg\_id, specify sites) output: global kdeg fit
3. Store to db (kinetic params) - need to do if want to do arrhenius

#### **nerd adduction**

1. Independent kadd measurements input: NMR peak integrals (csv) output: ODE fit (add values to table)
2. Aggregate kadd from DMS input: kobs values at melted temps output: aggregated kobs values at melted temps (raw data points for arrhenius, add to table)
3. Store to db (kinetic params) - need to do if want to do arrhenius

#### **nerd arrhenius**

1. Linear fit input: array of k values, corresponding temperatures output: LinearModel or params with errors
2. Store to db (Arrhenius)

#### **nerd timecourse**

1. Independent time-course fits input: reaction group id output: non-linear fit or params
2. Refit with constrained deg input: reaction group id output: non-linear fit or params
3. Store to db (kinetic params, kobs)

#### **nerd calc\_energy**

1. 2-state melting fit - K linear curve input: array of kobs values from probe\_kinetic rates output: 2-state melt params
2. Single K calc input: kobs, kadd output: K value

### **New sqlite tables**

#### **sqlite table: probe\_kinetic\_rates**

- Columns:
  - nmr\_reaction\_id
  - k value
  - k error
  - species (DMS, C8, etc.)

#### **sqlite table: nmr\_reactions**

- Columns:
  - reaction type (deg, add)
  - NMR machine (A600, A400, etc.)
  - temperature
  - scans
  - read time
  - probe
  - probe\_conc
  - probe\_solvent
  - substrate
  - substrate\_conc
  - buffer
  - csv directory

#### **sqlite table: arrhenius\_fits**

- Columns:
  - Reaction type (degradation, adduction, kobs, K)
  - Data source (nmr, probe\_global, probe\_free)
  - slope
  - slope\_err
  - intercept
  - intercept\_err
  - rsq
  - dir to LinearModel (contains data)

### **Example Usage**

1. Import NMR deg samples with `nerd import`
2. Run exponential `nerd degradation`
3. Run deg arrhenius `nerd arrhenius`
4. Import DMS samples `nerd import`
5. Run time-course free fits `nerd timecourse`
6. Run global fits for deg `nerd degradation`
7. Run time-course refit with deg `nerd timecourse`

8. Run store kobs values `nerd timecourse store`
9. Run import NMR add samples with `nerd import`
10. Run ODE `nerd adduction`
11. Run add arrhenius `nerd arrhenius`
12. Run aggregate arrhenius on data `nerd arrhenius`
13. Run 2 state melt
14. (optional) Run K calc with given kadd value - HIV or P4P6