Getting Started with VIPR

18 November 2025

VIPR Consortium

The VIPR framework (Versatile Inverse Problem Software Framework)

- **Core idea:** Standardized, extensible framework for ML pipelines in scientific data analysis
- **Focus**: Solving inverse problems in materials science and particle physics
- **Goal**: Interoperability (enables sharing of ML components between research facilities)

Versatile Inverse Problem Software Framework

VIPR Core (CLI Foundation)

↓

Plugin Ecosystem

🔌 Reflectometry (available)

🔌 GISAXS (planned)

🔌 Custom Plugins

↓

User Interfaces

💻 CLI | 🌐 Web-App

The VIPR framework (Versatile Inverse Problem Software Framework)

Two ways for usage:

- CLI: VIPR Core + plugins
- Web-App: GUI + API (uses VIPR Core + plugins)

Codebase

vipr-core (CLI): https://codebase.helmholtz.cloud/vipr/vipr-core

vipr-reflectometry-plugin: https://codebase.helmholtz.cloud/vipr/vipr-reflectorch-plugin

vipr-api: https://codebase.helmholtz.cloud/vipr/vipr-api

vipr-frontend: https://codebase.helmholtz.cloud/vipr/vipr-frontend

vipr-framework (overview): https://codebase.helmholtz.cloud/vipr/vipr-framework

Getting started with VIPR core and the reflectometry plugin

- VIPR Core: foundation of VIPR
- CLI program, based on Cement CLI framework (https://builtoncement.com/)
- Can be extended by Cement plugins
- Minimum requirements: Python 3.10 + Access to VIPR codebase
- **Guide**: https://codebase.helmholtz.cloud/vipr/vipr-framework/-/blob/init/VIPR_CLI.md

Installation with Reflectometry Plugin

pip install git+https://codebase.helmholtz.cloud/vipr/vipr-core.git

pip install git+https://codebase.helmholtz.cloud/vipr/vipr-reflectorch-plugin.git

Example Prediction Command

```
vipr --config @vipr_reflectometry/reflectorch/examples/configs/PTCDI-C3.yaml inference run
```

VIPR Core Test with Reflectometry Plugin

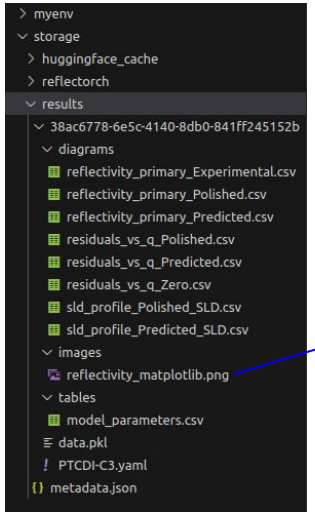
Prediction for XRR (PTCDI-C3 layer on SiO_x/Si)

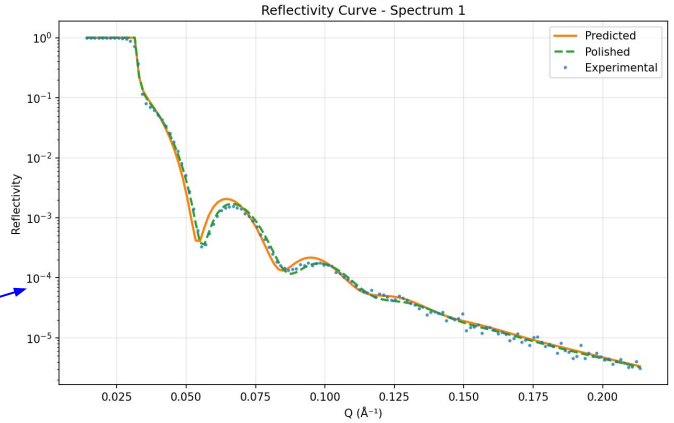
```
vipr --config @vipr_reflectometry/reflectorch/examples/configs/PTCDI-C3.yaml inference run
```

Terminal Output:

```
INFO: LoadModelInferenceStep completed
INFO: Step 3: Normalizing data
INFO: Starting NormalizeInferenceStep
INFO: No normalizer handler specified, using identity normalization
INFO: NormalizeInferenceStep completed
INFO: Step 4: Preprocessing data
INFO: Starting PreprocessInferenceStep
INFO: Interpolating 1 spectra with Reflectorch
INFO: No preprocessor handler specified, using identity preprocessing
INFO: PreprocessInferenceStep completed
INFO: Step 5: Performing prediction
INFO: Starting PredictionInferenceStep
INFO: Prediction config: {'handler': 'reflectorch_predictor', 'parameters': {'calc_polished_sld_prof
rediction': True, 'prior_bounds': {'roughnesses': [[0, 20], [0, 15], [0, 15]], 'slds': [[10, 13], [2
se_q_shift': False}}
INFO: Looking for predictor handler: 'reflectorch_predictor'
INFO: Using predictor handler 'reflectorch_predictor'
INFO: Processing batch of 1 spectra with DataSet interface
INFO: Using scalar q_resolution from config: None
INFO: Converted flat dict prior_bounds to array with 8 bounds
INFO: PredictionInferenceStep completed
INFO: Step 6: Postprocessing results
INFO: Starting PostprocessInferenceStep
INFO: Processing standard reflectorch UI data for batch_size=1
INFO: Stored standard reflectorch UI data with 1 spectra
INFO: No postprocessor handler specified, returning data as is
INFO: PostprocessInferenceStep completed
INFO: Inference workflow completed successfully
INFO: Collecting 87 buffered log entries
INFO: UI Plugin stored its own data with ID: 38ac6778-6e5c-4140-8db0-841ff245152b
```

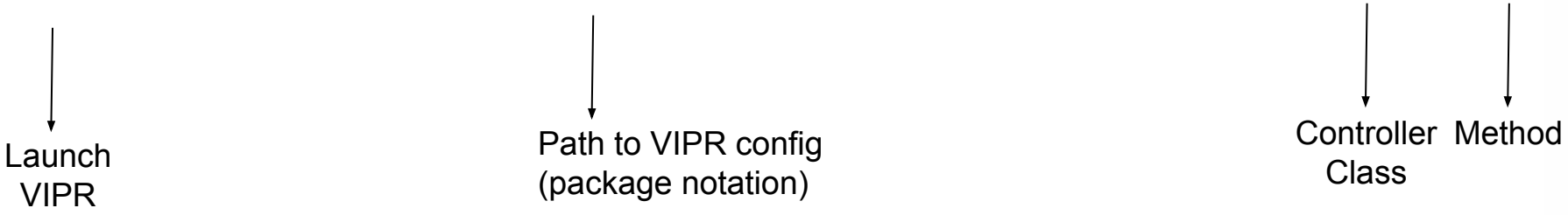
Results Directory:





VIPR CLI command for prediction

```
vipr --config @vipr_reflectometry/reflectorch/examples/configs/PTCDI-C3.yaml inference run
```



Config File Reference Options

Relative Path: --config ./configs/my-config.yaml
Absolute Path: --config /home/user/project/configs/config.yaml
Package Notation: --config @vipr_reflectometry/reflectorch/examples/configs/PTCDI-C3.yaml
(useful for the examples that come with the package)

VIPR config

Example config: https://codebase.helmholtz.cloud/vipr/vipr-framework/-/blob/init/example-config.yaml

- Copy example config YAML to your working directory
- Customize config parameters (e.g., your own `data_path`, filters, model settings)
- Reference your config with CLI command: `--config ./my-config.yaml`
- Run inference: `vipr --config ./my-config.yaml inference run`

example-config.yaml 1.25 KiB

```yaml
vipr:
  inference:
    filters:
      INFERENCE_PREPROCESS_PRE_FILTER:
      - class: vipr_reflectometry.reflectorch.reflectorch_extension.Reflectorch
        enabled: true
        method: _preprocess_interpolate
        parameters: null
        weight: 0
    load_data:
      handler: csv_spectrareader
      parameters:
        column_mapping:
          I: 1
          q: 0
        data_path: '@vipr_reflectometry/reflectorch/examples/data/PTCDI-C3.txt'
    load_model:
      handler: reflectorch
      parameters:
        config_name: b_mc_point_xray_conv_standard_L2_InputQ
    normalize:
      handler: ''
      parameters: {}
    postprocess:
      handler: ''
      parameters: {}
    prediction:
      handler: reflectorch_predictor
      parameters:
        calc_polished_sld_profile: true
        calc_pred_curve: true
        calc_pred_sld_profile: true
        clip_prediction: true
        polish_prediction: true
        prior_bounds:
          roughnesses: [[0, 20], [0, 15], [0, 15]]
          slds: [[10, 13], [20, 21], [20, 21]]
          thicknesses: [[1, 400], [1, 10]]
        q_resolution: null
        upper_phase_sld: 0
        use_q_shift: false
    preprocess:
      handler: ''
      parameters: {}
  result_id: 38ac6778-6e5c-4140-8db0-841ff245152b
```

Getting started with the VIPR web-app

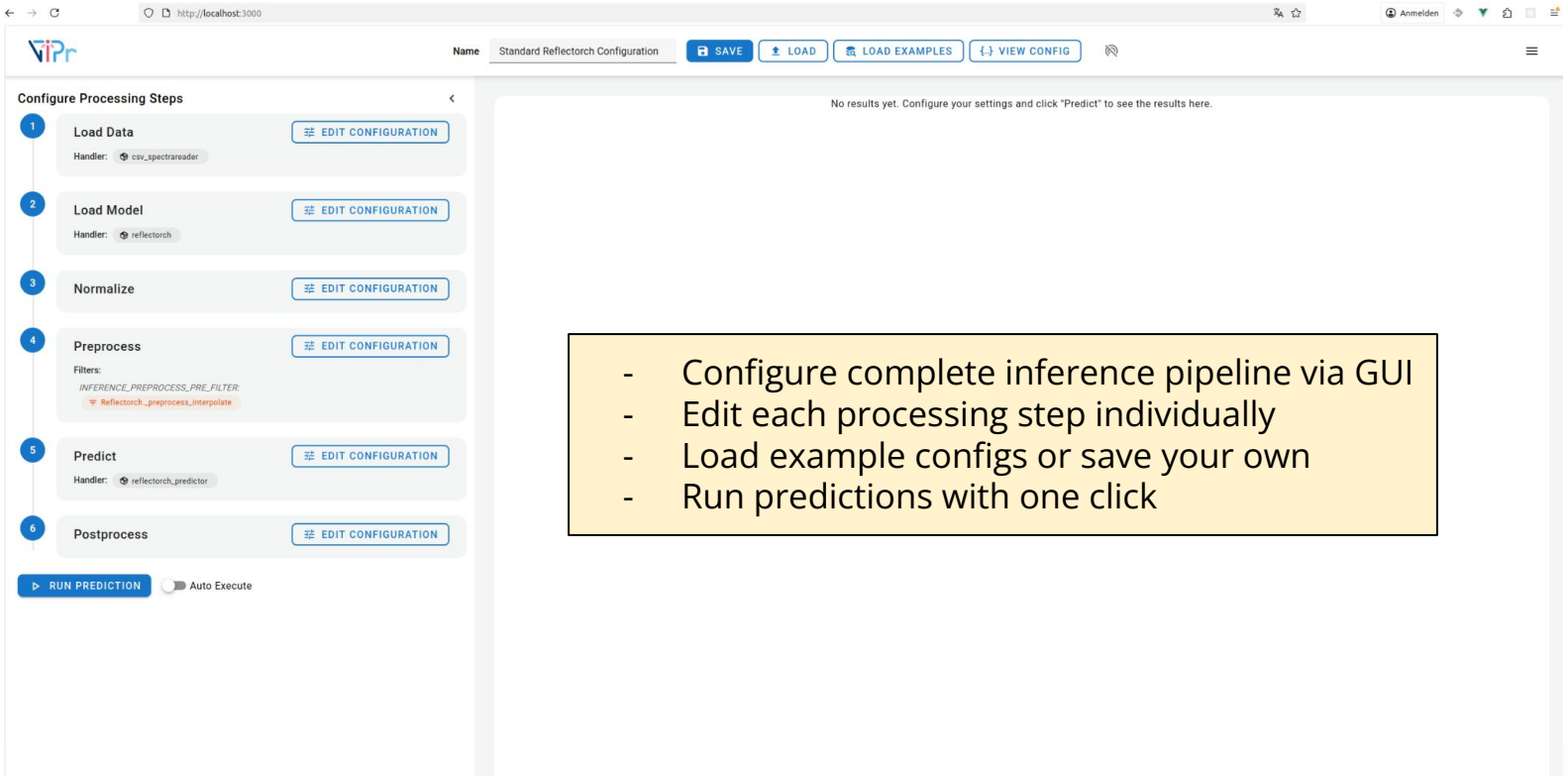
Guide:
https://codebase.helmholtz.cloud/vipr/vipr-framework/-/blob/init/VIPR_web_app.md

Requirements:
- Access to registry.hzdr.de via Personal Access Token
- Docker

Start WebApp (VIPR api + VIPR frontend): `docker compose up`

VIPR frontend: http://localhost:3000
VIPR api: http://localhost:8000/docs

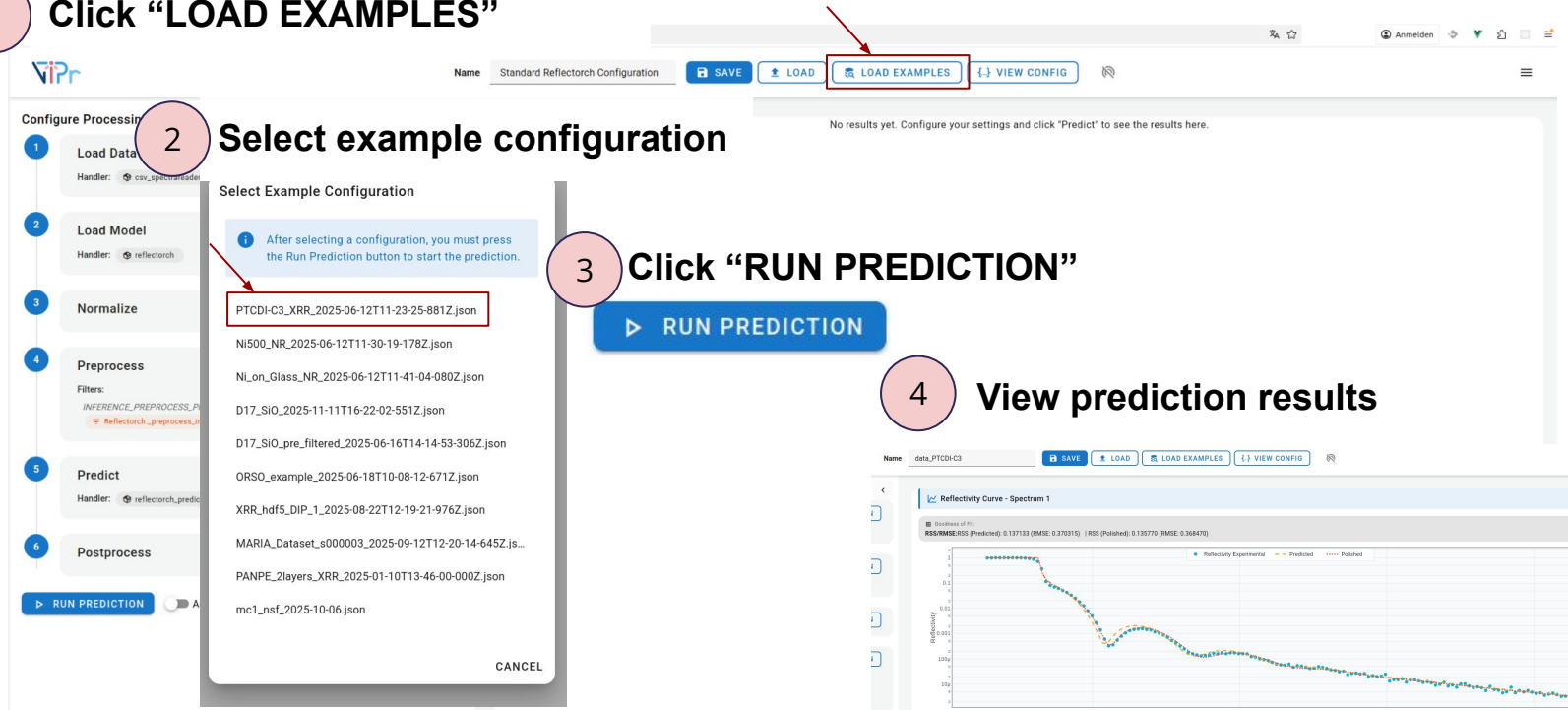
VIPR frontend (accessible via http://localhost:3000)



- Configure complete inference pipeline via GUI
- Edit each processing step individually
- Load example configs or save your own
- Run predictions with one click

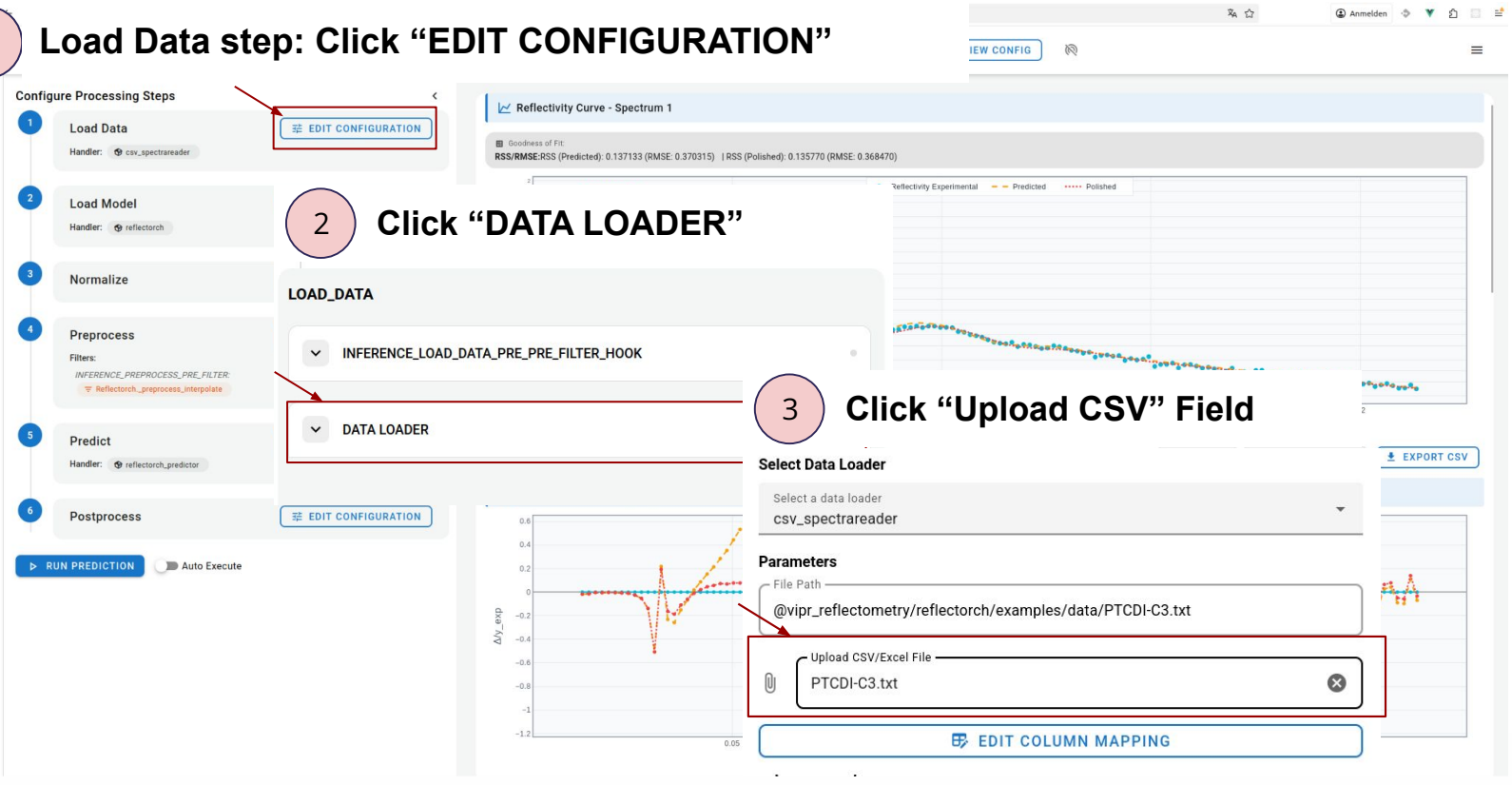
Prediction for predefined examples

1 **Click “LOAD EXAMPLES”**



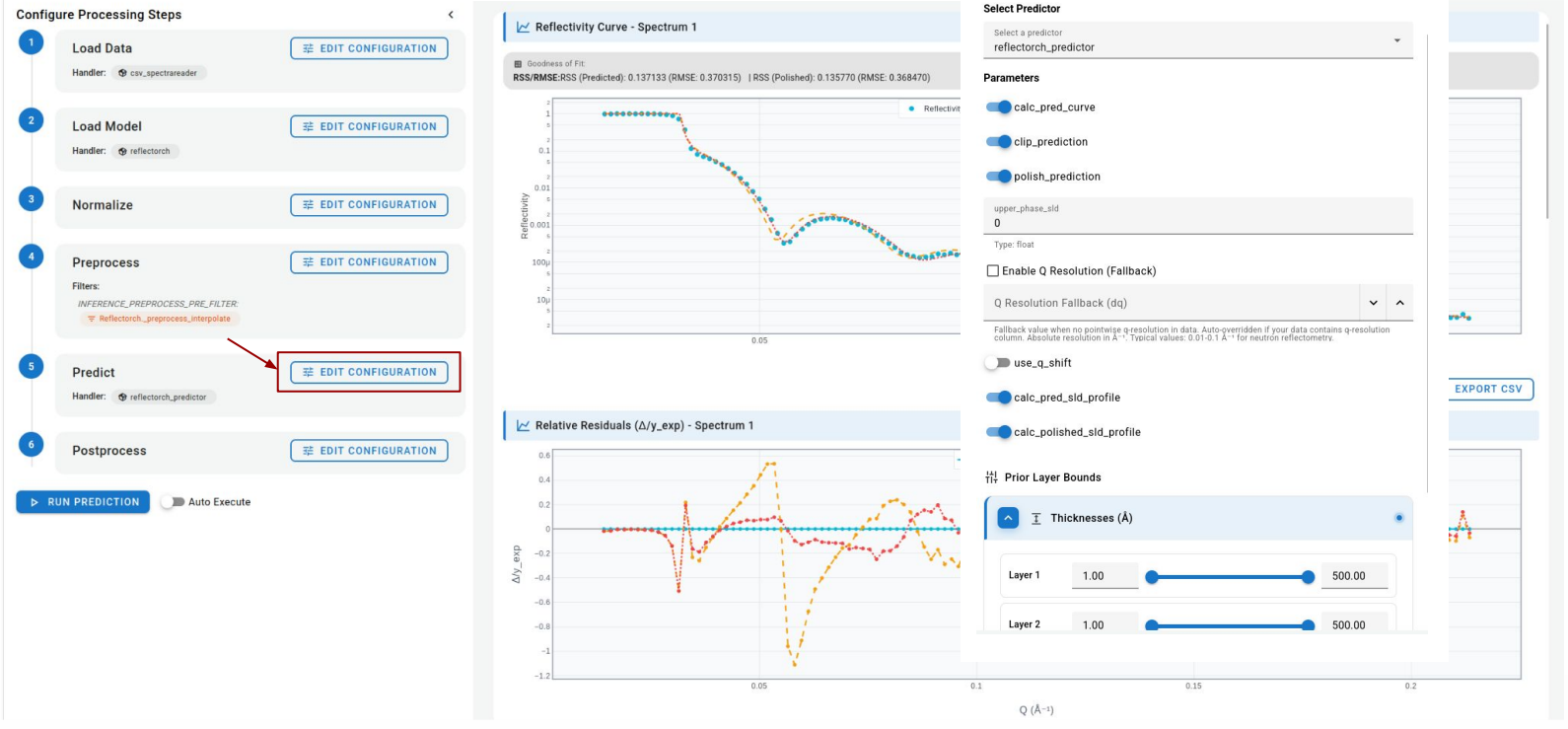
Use your own experimental data set

1 **Load Data step: Click “EDIT CONFIGURATION”**

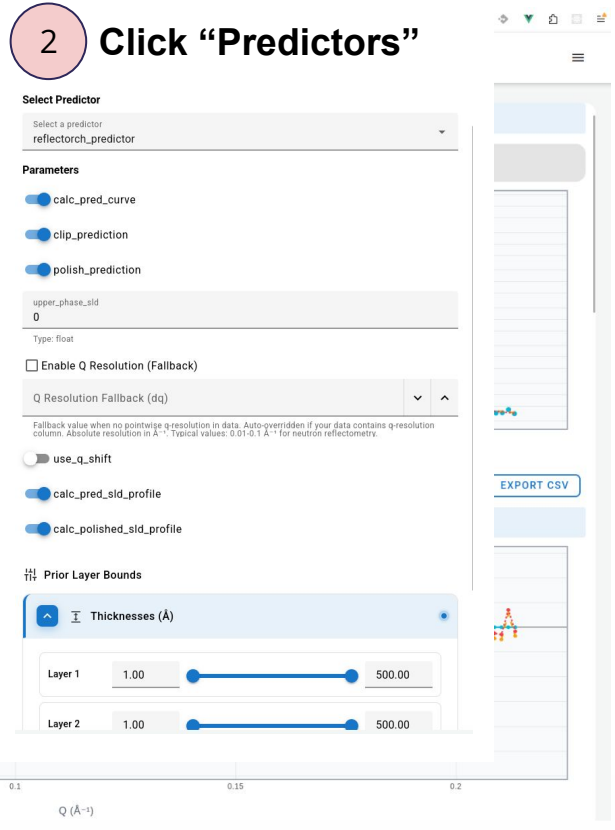


Adjust the prediction parameters

1 Predict step: Click “EDIT CONFIGURATION”



2 Click “Predictors”



Summary

- Modular ML framework for inverse problems
- CLI + Web-App for different user needs
- Plugin ecosystem
- availability of a Reflectometry plugin
- YAML-based configuration

Guide VIPR CLI: https://codebase.helmholtz.cloud/vipr/vipr-framework/-/blob/init/VIPR_CLI.md

Guide VIPR web-app: https://codebase.helmholtz.cloud/vipr/vipr-framework/-/blob/init/VIPR_web_app.md