AGH
Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
Wydział Fizyki i Informatyki Stosowanej

Praca inżynierska

Wiktor Hładki

kierunek studiów: **informatyka stosowana**

Interfejs graficzny do analizy wielowymiarowych widm spektroskopowych w środowisku Python

Opiekun: **dr hab. inż. Marcin Sikora**

Kraków, styczeń 2020

Oświadczenie studenta

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (tj. Dz. U. z 2018 r. poz. 1191 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta.”, oświadczam, że niniejszą pracę dyplomową wykonałem osobiście i samodzielnie i nie korzystałem ze źródeł innych niż wymienione w pracy.

Jednocześnie Uczelnia informuje, że zgodnie z art. 15a ww. ustawy o prawie autorskim i prawach pokrewnych Uczelni przysługuje pierwszeństwo w opublikowaniu pracy dyplomowej studenta. Jeżeli Uczelnia nie opublikowała pracy dyplomowej w terminie 6 miesięcy od dnia jej obrony, autor może ją opublikować, chyba że praca jest częścią utworu zbiorowego. Ponadto Uczelnia jako podmiot, o którym mowa w art. 7 ust. 1 pkt 1 ustawy z dnia 20 lipca 2018 r. --- Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.), może korzystać bez wynagrodzenia i bez konieczności uzyskania zgody autora z utworu stworzonego przez studenta w wyniku wykonywania obowiązków związanych z odbywaniem studiów, udostępniać utwór ministrowi właściwemu do spraw szkolnictwa wyższego i nauki oraz korzystać z utworów znajdujących się w prowadzonych przez niego bazach danych, w celu sprawdzania z wykorzystaniem systemu antyplagiatowego. Minister właściwy do spraw szkolnictwa wyższego i nauki może korzystać z prac dyplomowych znajdujących się w prowadzonych przez niego bazach danych w zakresie niezbędnym do zapewnienia prawidłowego utrzymania i rozwoju tych baz oraz współpracujących z nimi systemów informatycznych.

.....

(czytelny podpis)

Na kolejnych dwóch stronach proszę dołączyć kolejno recenzje pracy popełnione przez Opiekuna oraz Recenzenta (wydrukowane z systemu MISIO i podpisane przez odpowiednio Opiekuna i Recenzenta pracy). Papierową wersję pracy (zawierającą podpisane recenzje) proszę złożyć w dziekanacie celem rejestracji.

Spis treści

1 Wstęp

Rozwój nowoczesnych technologii dostępnych dzięki postępom różnych dziedzin nauki i techniki zawdzięczamy m.in. odkryciom inżynierii materiałowej. Dzięki zgłębianiu wiedzy na temat materii i wzajemnych interakcji między wykorzystywanymi substancjami możliwe jest uzyskiwanie materiałów posiadających ściśle określone cechy użytkowe, a także pozwala opracować wydajne metody ich produkcji i przetwarzania.

Jedną z dziedzin umożliwiającą badanie struktury fizykochemicznej materii jest spektroskopia rentgenowska. Zajmuje się ona analizą efektów reakcji materii na promieniowanie rentgenowskie. Dzięki technikom analitycznym spektroskopii rentgenowskiej można uzyskać informacje m.in. o wewnętrznej strukturze molekularnej analizowanej próbki.

2 Spektroskopia rentgenowska

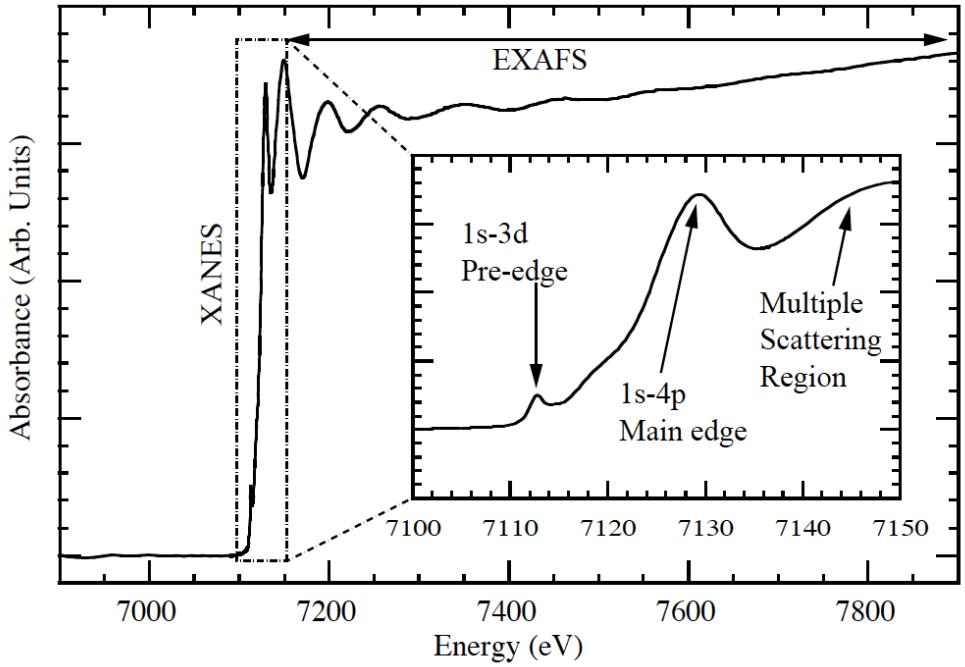
Analiza widm spektroskopowych jest istotnym narzędziem pozwalającym na zgłębienie wiedzy na temat struktury wewnętrznej materiałów. Pozwala na zdobycie informacji o zachodzących procesach fizykochemicznych i jest metodą opisywania środowiska atomów w molekułach. W tej pracy szczególna uwaga zostanie poświęcona spektroskopii rentgenowskiej (ang. *x-ray spectroscopy*).

2.1 Spektroskopia absorpcyjna

Techniki służące do pomiaru pochłaniania promieniowania w wyniku interakcji z próbką w zależności energii i częstotliwości określa się spektroskopią absorpcyjną. Zdolność absorpcyjna określona jako funkcja częstotliwości definiuje spektrum absorpcji. Istnieje wiele metod eksperymentalnych pozwalających na pomiar tego spektrum. Częstym scenariuszem jest wymierzenie kierowanej wiązki energii na próbkę i pomiar energii wiązki po jej przejściu przez obiekt. Energia zmierzona wtedy pozwala obliczyć stopień absorbcji przy danej energii.

Jednym z badanych procesów w rentgenowskiej spektroskopii absorpcyjnej (XAS – ang. *X-ray Absorption Spectroscopy*) jest zjawisko absorbcji fotonu. W procesie tym elektron rdzenia atomowego próbki reaguje z wiązką rentgenowską, potencjalnie przechodząc w wyższe dostępne stany energetyczne. Prawdopodobieństwo tego przejścia rośnie wraz ze zbliżaniem się energii fotonu do energii wiązania elektronu.

Eksperymenty kategorii XAS w obszarze K-edge prowadzone na związkach metalach przejściowych bloku 3d (m.in. tytan, chrom i miedź) są rutynową metodą analizy tych materiałów. Pomiary EXASF (ang. *Extended X-ray Absorption Fine Structure*) są powszechnie wykorzystywane celem określenia geometrii lokalnej ośrodka, a pomiary XANES (ang. *X-ray Absorption Near Edge Structure*) dostarczają danych o m.in. stanie utlenienia metalu [5, 8].

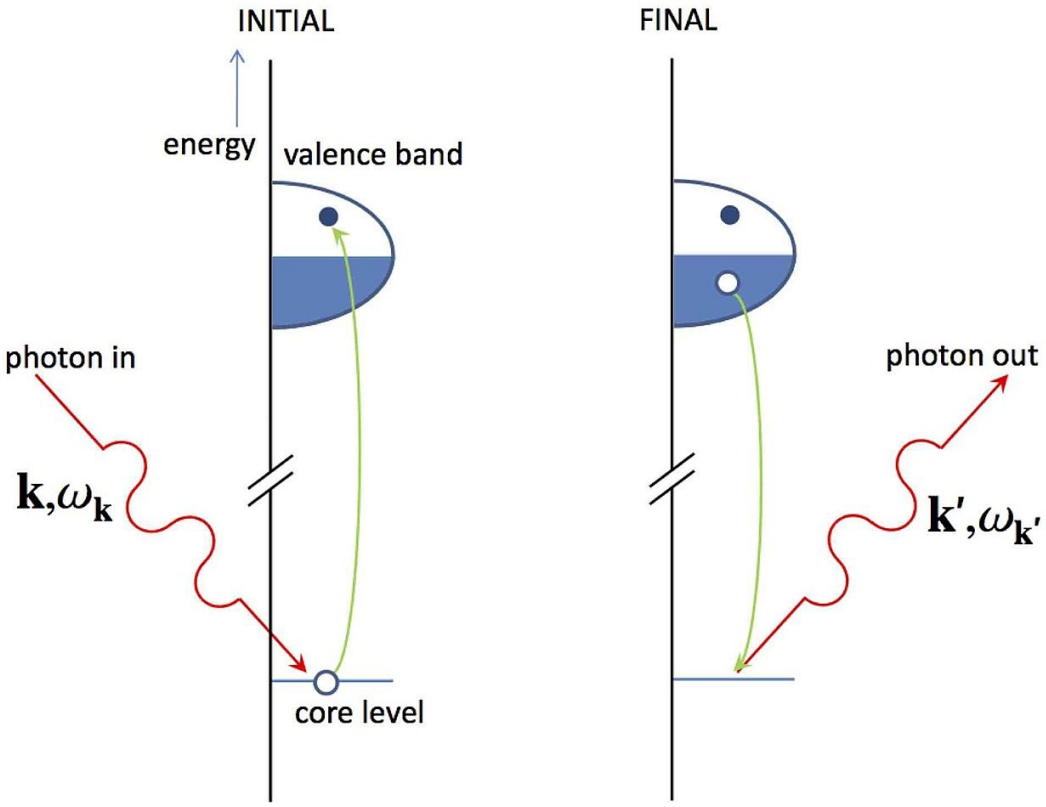


Rys. 2.1 – Wizualizacja pomiaru XAS dla żelaza w obszarze K-edge [I]

2.2 Spektroskopia emisyjna

Spektroskopią emisyjną nazywamy dziedzinę pomiarów w której badane są cząstki emitowane przez próbkę w wyniku powrotu wzbudzonego elektronu do stanu niewzbudzonego. Mogą to być fotony o różnych długościach fali zależnych od pierwiastków zawartych w badanej próbce. Analiza spektrum w rentgenowskiej spektroskopii emisyjnej (XES – ang. *X-ray Emission Spectroscopy*) pozwala na m.in. analizę składu fizykochemicznego badanej próbki.

Jedną z rozwijanych technik eksperymentalnych spektroskopii emisyjnej jest RXES (ang. *Resonant X-ray Emission Spectroscopy*). Bazuje ona na rezonansowej spektroskopii Ramana. Polega ona na wzbudzeniu elektronu rdzenia atomowego za pomocą fotonu o odpowiedniej długości fali, co powoduje powstanie dziury elektronowej. W wyniku zapełnienia jej przez inny elektron z wyższej powłoki uwolniona energia pod postacią fotonu pozwala zyskać informacje o zajętości stanów powłok elektronowych [6-7].



Rys. 2.2 – Uproszczony schemat scenariusza pomiarowego eksperymentu RXES [II]

3 Cel pracy

Celem niniejszej pracy było utworzenie aplikacji z graficznym interfejsem użytkownika (GUI – ang. *Graphical User Interface*), umożliwiającej analizę wielowymiarowych widm spektroskopowych korzystając z języka Python. Jako wzorzec obrana została istniejąca wcześniej funkcjonująca w tym celu aplikacja napisana języku programowania pakietu MATLAB przez opiekuna tej pracy. MATLAB jest oprogramowaniem licencjonowanym, co utrudnia dystrybucję i uruchamianie aplikacji opartej na tym środowisku, podczas gdy Python, jako ogólnodostępny język i środowisko uruchomieniowe (oparte na licencji zaaprobowanej przez Open Source Initiative), wraz z dużą ilością dodatkowych bibliotek oferowanych w nim, umożliwia utworzenie aplikacji prostej w dalszym rozwoju, niezależnej od systemu operacyjnego i bez zależności od licencjonowanego oprogramowania.

3.1 Założenia możliwości aplikacji

- Import jednowymiarowych i/lub dwuwymiarowych zestawów danych w określonych formatach
- Selekcja wybranych pozycji z wczytanych zestawów danych celem przygotowania wizualizacji
- Podstawowe operacje na wczytanych zestawach danych – odejmowanie i dzielenie zestawów od siebie, oraz normalizacja danych do jedności
- Wizualizacja przygotowanych i wyselekcjonowanych zestawów danych, wraz z możliwością interaktywnego dostosowania wyświetlanego obszaru (przesuwanie, przybliżanie/oddalanie)
- Eksport wyselekcjonowanych zestawów danych w określonych formatach
- Zapis wizualizacji jako pliku graficznego

4 Projekt aplikacji

4.1 Narzędzia i biblioteki wykorzystane przy tworzeniu aplikacji

4.1.1 Język Python

Python jest wysokopoziomowym językiem interpretowanym, stawiającym na czytelną i intuicyjną składnię, wspierający wiele paradygmatów programowania (m.in. obiektowy i proceduralny). Jest to język stosujący typowanie dynamiczne, w którym każda wartość jest obiektem, a zmienne je przechowujące są jedynie etykietami wskazującymi na ich położenie – typ zmiennej wynika jedynie z wartości jaką przechowuje.

W projekcie wykorzystany został Python w wersji 3.7, będącą najnowszą dostępną wersją w trakcie rozpoczęcia prac nad aplikacją. Jest to istotne ze względu na wykorzystanie funkcjonalności oferowanych dopiero od tej wersji.

4.1.2 Biblioteka Matplotlib

Matplotlib jest obszerną biblioteką umożliwiającą w łatwy sposób tworzenie różnych wykresów i wizualizacji w języku Python. Wiele elementów jej API (Application Programming Interface) oferuje podobny schemat zastosowania jak w języku pakietu MATLAB, co pozwala na łatwe odwzorowanie operacji w nim zapisanych w aplikacji Pythona.

4.1.3 Biblioteka MyPy

W języku Python fakt, iż jest to język typowany dynamicznie, nie pozwala określić typów przechowywanych przez zmienne przed uruchomieniem programu. Analiza kodu pozwala wyciągnąć pewne wnioski na ten temat, jednak począwszy od Pythona 3.5 (z dodatkowymi modyfikacjami w późniejszych wersjach) możliwe jest tworzenie adnotacji typów – komentarzy posługujących się określoną składnią, opisujących jakie wartości powinna przechowywać zmienna. Nie wpływa to w żaden sposób na naturę wykonania kodu, jednak kod pisany z adnotacjami typów pozwala na przeprowadzanie automatycznych inspekcji sprawdzających poprawne zastosowanie funkcji i zmiennych za pomocą zewnętrznych aplikacji, takich jak np. MyPy. Dzięki temu wiele błędów w trakcie rozwoju aplikacji można wychwycić nawet przed pierwszym faktycznym uruchomieniem kodu.

Projekt tworzony w ramach tej pracy wykorzystuje ten mechanizm w dużym stopniu, wraz z funkcjonalnością ich dotyczącą, która dostępna jest dopiero od Pythona w wersji 3.7.

4.1.4 Biblioteka TkInter

Biblioteka TkInter, oferowana podczas instalacji Pythona, funkcjonuje jako ogólnie przyjęty standard podczas pisania aplikacji graficznych w tym języku. Jest interfejsem do opensourcowego, wieloplatformowego zestawu narzędzi Tk, odpowiadających za faktyczną budowę silnika graficznego interfejsu użytkownika.

4.1.5 Biblioteka NumPy

Biblioteka NumPy pozwala na wydajne operacje numeryczne na macierzowych zbiorach danych. Podobnie jak biblioteka Matplotlib (będąca wraz z NumPy częścią stosu SciPy) oferuje podobny interfejs programistyczny do języku pakietu MATLAB. Operacje macierzowe w NumPy będą wykonywane szybciej i wydajniej niż w przypadku użycia standardowych list/tablic, dzięki wykonywaniu ich poprzez rozszerzenie skompilowane w języku C.

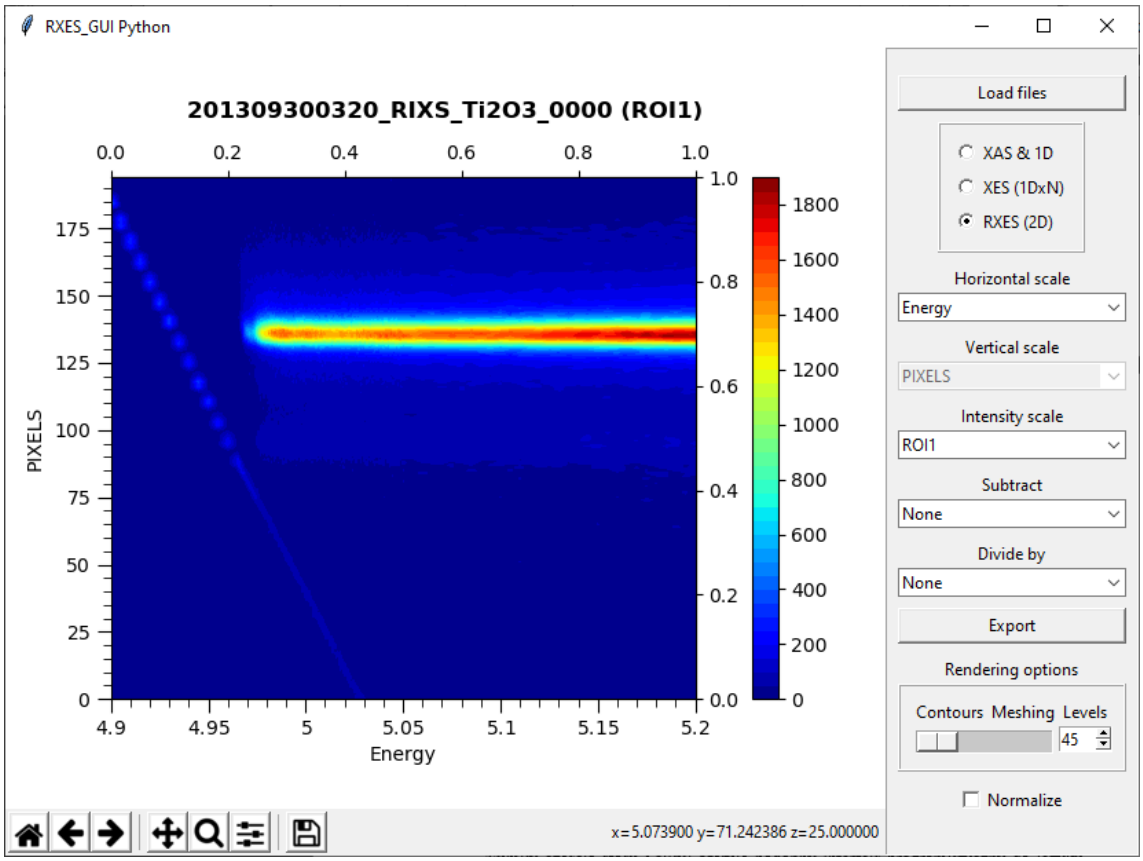
Wykorzystana zostaje również biblioteka SciPy (nie mylić ze stosem SciPy, będącym kolekcją bibliotek) celem wykorzystania funkcjonalności wczytywania plików w formacie MAT-File utworzonych za pomocą pakietu MATLAB.

4.1.6 PyCharm IDE (Integrated Development Environment)

PyCharm jest popularnym narzędziem służącym do rozwoju aplikacji w języku Python. Oferuje wygodny edytor z funkcją podświetlania składni, uzupełniania kodu i stylistycznej refaktoryzacji kodu. Poza tym pozwala na łatwą nawigację między elementami projektu, daje dostęp do łatwego w użyciu graficznego debugera oraz oferuje automatyczne inspekcje kodu usiłując znaleźć typowe potencjalne błędy w kodzie użytkownika.

Dzięki możliwości instalowania dodatkowych rozszerzeń w postaci wtyczek możliwe jest dalsze rozszerzania możliwości tego IDE. Jedna z takich wtyczek pozwala na łatwą integrację automatycznych inspekcji typowania za pośrednictwem MyPy. O ile domyślnie PyCharm zawiera system oferujący podobną funkcjonalność, nie jest ona równie konfigurowalna i dokładna, co rozwiązanie dedykowane.

4.2 Interfejs użytkownika



Rys. 4.1 – Zrzut ekranu przedstawiający interfejs użytkownika wraz z wizualizacją przykładowych danych

Interfejs podzielono na dwa obszary logiczne – obszar płótna po lewej, zajmujący się wyświetlaniem przetworzonych danych na ekranie oraz obszar panelu kontrolnego po prawej, służący do zarządzania wyświetlanymi danymi.

Na panelu płótna wyświetlana jest bieżąca wizualizacja utworzona na podstawie wybranych danych. W zależności od wybranego trybu wizualizacji (opisanych w rozdziale 4.3), przedstawiane są jednowymiarowe wykresy liniowe reprezentujące np. widma spektroskopii absorpcyjnej lub emisyjnej albo kolorowe wizualizacje dwuwymiarowe obrazujące np. pomiar eksperymentu spektroskopii rezonansowej (przykładowo RXES). Użytkownik jest w stanie manipulować wyświetlanym obszarem za pomocą myszy oraz narzędzi na dedykowanym pasku u dołu ekranu. Z poziomu tego paska możliwy jest powrót do oryginalnego widoku, jak i również eksport wizualizacji jako pliku graficznego. W prawym dolnym rogu wyświetlana jest bieżąca pozycja myszki w układzie wizualizacji oraz wartość dostępna pod kursorem (w przypadku danych 2D).

Za pomocą panelu kontrolnego użytkownik dokonuje większości kluczowych interakcji z programem dotyczących zarządzania wyświetlanymi danymi i sposobem ich wyświetlania.

W jego skład wchodzą:

- Przyciski służące do importu i eksportu danych
- Pole selekcji trybu wizualizacji (przeglądanie danych 1D lub 2D oraz wizualizacji 1DxN)
- Listy wyboru służące do selekcji zestawów danych użytych w wybranym trybie wizualizacji
- Obszar wyboru sposobu reprezentacji danych 2D (więcej w dziale 4.3.3)
- Pole aktywacji normalizacji danych (do eksportu danych i wyświetlania wartości pod wizualizacją)

Kod odpowiadający za budowę interfejsu użytkownika oraz za jego kluczowe funkcjonalności znajduje się w module *gui*.

W jego skład wchodzą pliki:

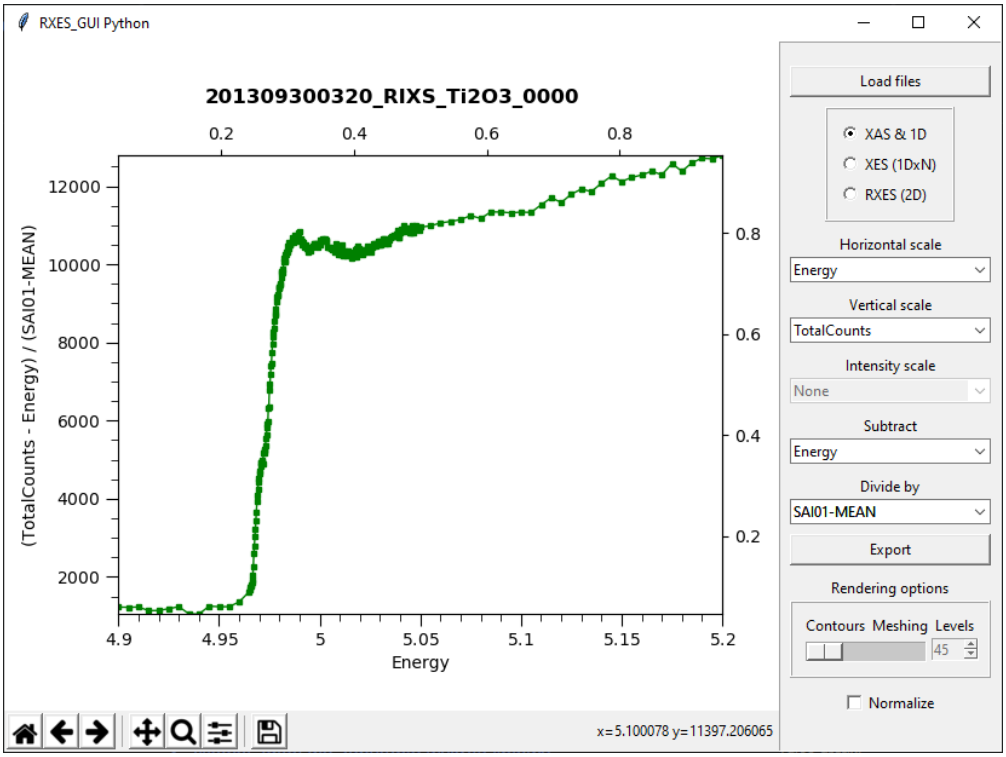
- Wydzielające struktury przechowujące interaktywne elementy panelu kontrolnego
  - buttons.py
  - comboboxes.py
  - radiobuttons.py
  - renderingoptions.py
- displaydata.py – definiujący strukturę przechowującą bieżąco wyświetlane dane i ich etykiety
- localutils.py – przechowujący definicje funkcjonalności wspólnych elementów interfejsu oraz integrujący je (np. podświetlenie elementu i/lub zmiana wyglądu kursora po najechaniu na niego)
- gui.py – plik przechowujący klasę GUI, będącą podstawą działania aplikacji

4.3 Tryby wizualizacji danych

W zależności od wczytanych zestawów danych aplikacja oferuje różne tryby wizualizacji danych. Decydują one o typie wyświetlanego wykresu, doborze dostępnych zestawów danych i selekcji danych do eksportu.

4.3.1 XAS & 1D

Jest to tryb służący do analizy danych jednowymiarowych. Za pomocą list wyboru *„Horizontal scale”* oraz *„Vertical scale”* użytkownik dobiera dane odpowiadające za wartości osi X i Y, i wyświetlany wykres liniowy z zaznaczonymi punktami odpowiadającymi parą zawartych wartości. Za pomocą list wyboru *„Subtract”* oraz *„Divide by”* można dobrać zestawy danych, którymi wartości osi Y zostaną zmodyfikowane (poprzez odjęcie lub podzielenie wartości).



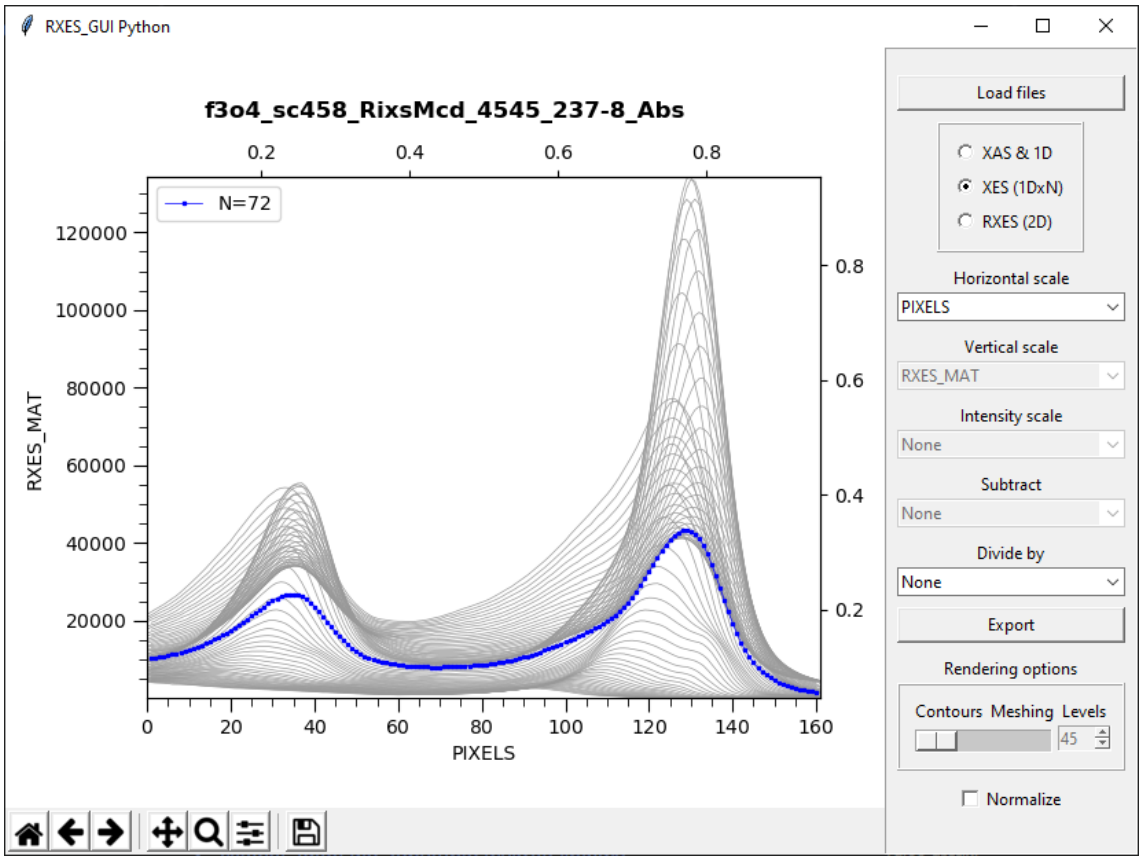
Rys. 4.2 – Przykład wykresu wyświetlanego w trybie 1D

W zależności od dobranych danych wyświetlany jest wykres reprezentujący odpowiednie dane, z odpowiadającymi etykietami danych. U góry widnieje nazwa analizowanego pliku (bez rozszerzenia).

W trakcie eksportu danych zapisywane są dwa jednowymiarowe zestawy danych – dane dotyczące osi X i odpowiednio zmodyfikowany zestaw danych dotyczący osi Y (tymczasowo zapisany w pamięci przez utworzeniem wykresu).

4.3.2 XES (1DxN)

Tryb ten służy do analizy danych dwuwymiarowych jako wielu serii jednowymiarowych. Za pomocą listy wyboru *„Vertical scale”* użytkownik wybiera analizowany dwuwymiarowy zestaw danych, którego wartości będą odpowiadać osi Y. Wartość wybrana w liście wyboru *„Horizontal scale”* decyduje o obranym kierunku trawersowania obranej macierzy dwuwymiarowej. Macierz dwuwymiarowa jest traktowana jako seria N danych jednowymiarowych, gdzie N jest rozmiarem obranego kierunku macierzy. Wizualizacja zawiera nałożone na siebie liniowe wykresy reprezentujące wartość Z zawartą w poszczególnych punktach X dla każdej serii (efektywne w wyniku podziału Y) oraz serię uśrednioną wyróżnioną na niebiesko. Legenda w rogu zawiera liczbę wydzielonych serii.



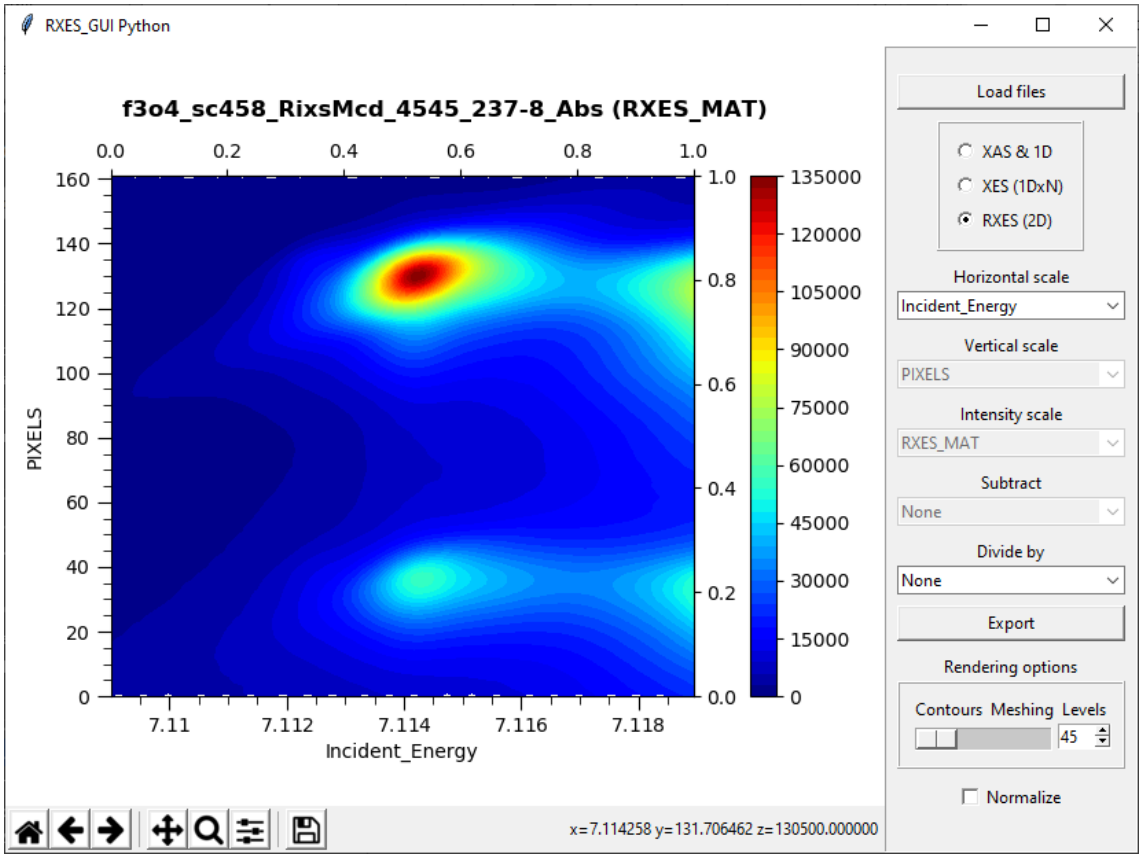
Rys. 4.3 – Przykład wykresu wyświetlanego w trybie 1DxN

Listy wyboru *„Subtract”* oraz *„Divide by”* pozwalają na odjęcie innego dwuwymiarowego zestawu danych od analizowanego zestawu, lub podzielenie wartości macierzy w danej współrzędnej X przez odpowiednią wartość z zestawu jednowymiarowego (dzielenie macierzy przez wektor).

W trakcie eksportu zapisywany jest wektor danych dotyczących osi X i N+1 wyświetlonych serii (po odpowiednich modyfikacjach). Dane te można wczytać ponownie i przeglądać w widoku 1D by uzyskać dostęp do poszczególnych serii.

4.3.3 RXES (2D)

Za pomocą tego trybu możliwe jest przeglądanie wczytanych dwuwymiarowych zestawów danych jako kolorowanych wykresów 2D. Za pomocą list wyboru *„Horizontal scale”* oraz *„Vertical scale”* dobierane są wektory opisujące odpowiednio osie poziomą i pionową, a wartość zawarta w liście wyboru *„Intensity scale”* decyduje o macierzy zawierającej reprezentowane wartości.

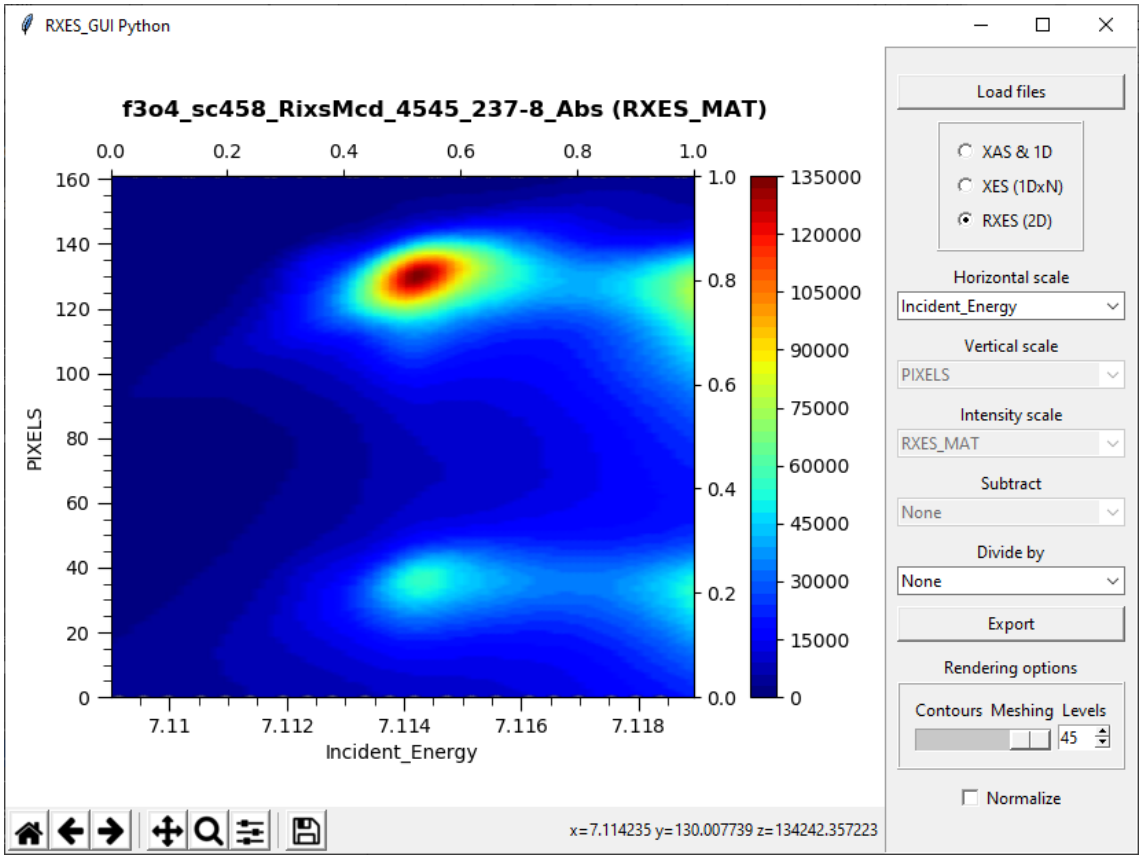


Rys. 4.4 – Przykład wykresu wyświetlanego w trybie 2D

Możliwy jest też wybór postaci graficznej reprezentacji danych.

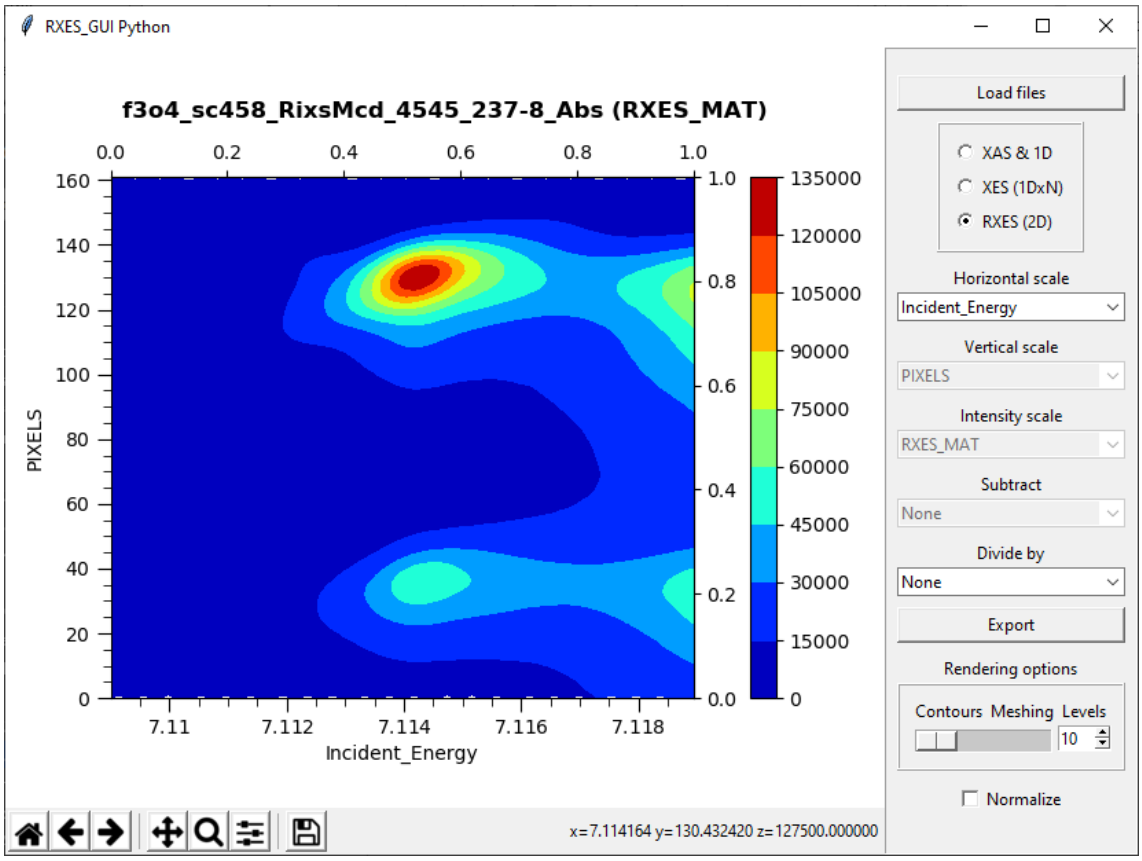
Dostępne są dwie opcje:

- Reprezentacja konturowa (opcja *„Contours”* w rubryce *„Rendering options”*) mapuje wartości macierzy do n ograniczonych wartości i określa gładkie przejścia między obszarami – jest to przystępna graficznie opcja reprezentacji danych. Niestety tracimy informację o dokładnych wartościach w poszczególnych miejscach na wykresie.
- Reprezentacja siatkowa (opcja *„Meshing”* w rubryce *„Rendering options”*) wiernie odwzorowuje wartości zawarte w macierzy i odpowiednio koloruje pola za pomocą najbliżej pasującego koloru z puli n kolorów. Ze względu na brak gładkich przejść między wartościami reprezentacja ta nie jest równie graficznie przystępna co konturowa, ale pozwala na podgląd dokładnych wartości zawartych w macierzy podczas najeżdżania myszą na poszczególne obszary.



Rys. 4.5 – Przykład z rys. 4.4 w reprezentacji siatkowej

Ilość kolorów które użyte są celem reprezentacji wartości na przestrzeni dwuwymiarowej można dobrać w polu *„Levels”* w rubryce *„Rendering options”* (możliwe wartości z zakresu 5-50).



Rys. 4.6 – Przykład z rys. 4.4 z ilością kolorów zaniżoną do 10

Modyfikacji danych 2D dokonuje się analogicznie do trybu 1DxN, za pomocą pól wyboru *„Subtract”* oraz *„Divide by”,* używając tak samo dobranych zestawów danych.

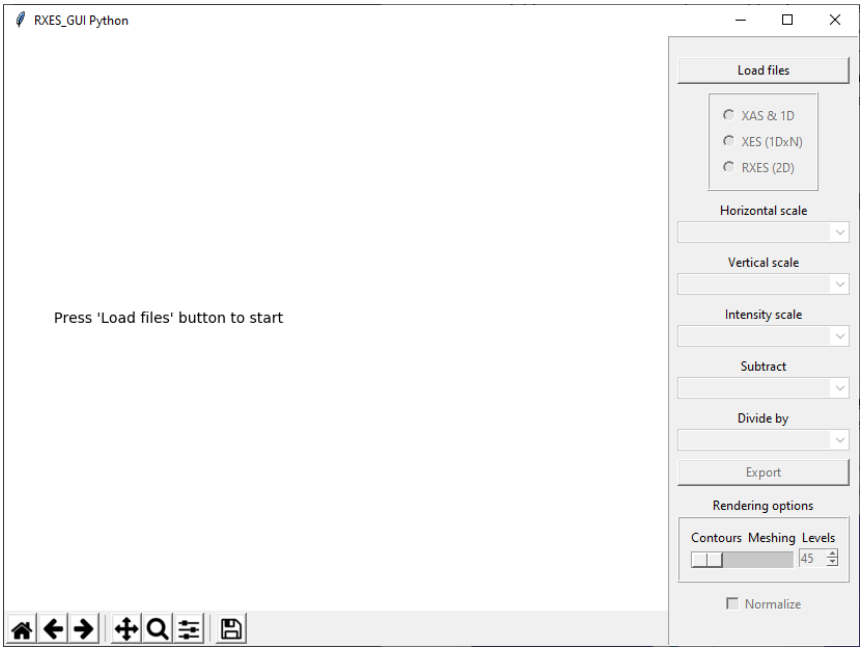
W trakcie eksportu zapisywane są wektory reprezentujące dane dotyczące osi X i Y oraz macierz danych Z. Istotną różnicą w eksporcie jest fakt, że zapisywane dane biorą pod uwagę bieżący widok panelu płótna. Jeśli użytkownik przesunął i/lub przybliżył/oddalił utworzony wykres ze względu na ciągłą reprezentację danych możliwe jest jednoznaczne wybranie jedynie wyświetlanych danych jako eksportowanych. Wykresy liniowe dla trybów 1D i 1DxN utworzone są z niejednorodnych punktów, co pozostawia niejednoznaczności gdy jeden z dwóch połączonych punktów zostanie przesunięty poza obszar rysowania.

4.4 Funkcjonowanie aplikacji

Istota działania aplikacji z interfejsem graficznym w bibliotece TkInter opiera się na relacji wydarzeń i funkcji zwrotnych. W momencie zaistnienia odpowiedniego wydarzenia, np. zmiana rozmiaru okna, wciśnięcie przycisku, wybór pola z listy lub przesunięcie myszką w określonym obszarze uruchamiana jest odpowiednia funkcja zwrotna (ang. *callback function*) oczekująca na to wydarzenie. Oznacza to, że po uruchomieniu i inicjalizacji interfejsu, dalsze fragmenty kodu będą uruchamiane zależnie od odpowiednio przewidzianych interakcji z użytkownikiem.

Podczas wczytywania poszczególnych elementów interfejsu przypisywane są odpowiednie funkcje zwrotne modyfikujące zarówno ich stan (wyświetlany tekst, wyszarzenie zależne od aktywacji elementu, podświetlenie) jak i stan aplikacji (modyfikacja zmiennych pseudoglobalnych dotyczących całego interfejsu, np. dane wczytane z pliku, wybrane wartości list wyborów, tryb renderowania).

Przed wczytaniem jakichkolwiek danych większość elementów interfejsu jest nieaktywna. Po udanym załadowaniu pliku określane są dostępne funkcjonalności, interfejs jest aktualizowany, a przetworzone dane są wyświetlane na ekran.

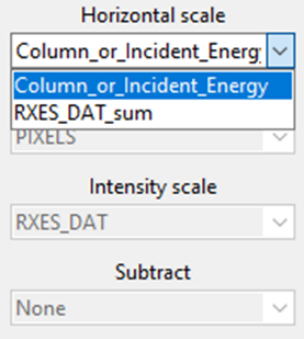


Rys. 4.7 – Ekran startowy aplikacji

Po udanym wczytaniu pliku dane są przetwarzane celem ustalenia możliwości ich reprezentacji oraz aktualizacji interfejsu. W przeciwnym razie zwracany jest komunikat o błędzie. Klasy dotyczące funkcjonalności odczytu (i zapisu) danych zostały wyodrębnione do modułu *filehandlers*.

Funkcja *determine_available_plot_types_from_data* określa jakiego typu dane zostały wczytane (sprawdzenie wymiarowość) i określa dostępne tryby przeglądania danych (1D dla danych jednowymiarowych, 1DxN i 2D dla dwuwymiarowych). Aktywuje też ona tryb aktywny po zakończeniu wczytywania danych – jeśli to możliwe, będzie to widok 2D, a w przypadku braku danych dwuwymiarowych, widok 1D.

Funkcja *configure_control_panel_from_gui_settings* odpowiada za (de)aktywację oraz populację wartości rozwijanych pól wyboru, umożliwiających wybór odpowiednich pól/kolumn z wczytanego pliku. Dobór wartości opiera się na wymiarowości danych i aktywnym trybie przeglądania. Deaktywuje również obszar trybu renderowania poza widokiem 2D. Akcja ta wykonywana jest przy każdej zmianie trybu przeglądania danych.



Rys. 4.8 – Interaktywne listy wyboru. Lista „Intensity scale” zawiera tylko jedną dostępną wartość, stąd nie oferuje możliwości wyboru. Lista „Subtract” nie zawiera dostępnych wartości, stąd również jest nieaktywna

Funkcje *generate_figure_from_gui_settings* oraz *reload_canvas_figure* są używane w każdej sytuacji mającej na celu przetworzenie ustalonych parametrów aplikacji celem utworzenia odpowiedniej reprezentacji graficznej danych (wykresów) i ich poprawnym wyświetleniem, blokując interaktywność na czas operacji. Pierwsza z nich korzystając z wartości przechowywanych w listach wyboru danych tworzy odpowiednio zmodyfikowane zestawy danych i ich etykiety, przechowuje je w pamięci tymczasowej (do eksportu) i przekazuje je do funkcji wizualizujących. Druga z nich odpowiada za integrację wizualizacji z panelem płótna. Razem tworzą uniwersalny mechanizm pozwalający na aktualizację panelu płótna odpowiednimi treściami.

Te cztery funkcje definiują kluczową funkcjonalność interaktywności aplikacji i są podstawą większości interakcji użytkownika z panelem kontrolnym, będąc zawartymi w przeważającej części funkcji zwrotnych.

4.5 Tworzenie wizualizacji

Odpowiednio dobrane i spreparowane dane muszą zostać poprawnie zwizualizowane. Niezależnie od obranego trybu wizualizacji danych do dedykowanych funkcji wysyłane są:

- Wektory i macierze danych
- Etykiety opisujące zestawy danych
- Tytuł wykresu (na podstawie nazwy wczytanego pliku)
- Informacja o normalizacji danych

Dodatkowo w przypadku trybu RXES (2D) wysyłane są informacje z rubryki *„Rendering Options”* przechowujące informacje o trybie renderowania i liczbie kolorów do wykorzystania.

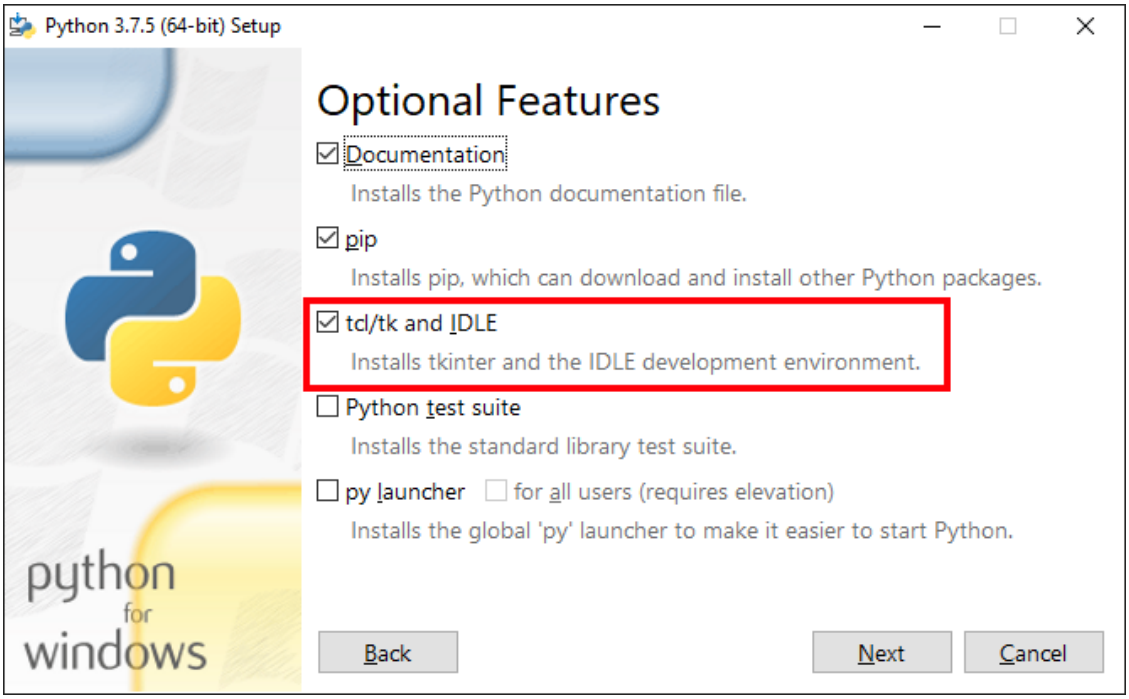
Z pomocą biblioteki Matplotlib tworzone są odpowiednie wykresy. Dodatkowo zamieszczane są etykiety i tytuł, tworzone są legendy dla widoków 1DxN i 2D, usuwane są zbędne marginesy, aktywowane są dodatkowe osie znormalizowane oraz integrowana jest funkcjonalność interakcji za pomocą myszy (domyślnie niedostępna).

Większość funkcjonalności dotyczące obsługi panelu płótna i tworzenia wizualizacji zostały umieszczone w osobnym module *plottools*. W jego skład wchodzą pliki:

- plots.py – zawierający funkcje tworzące wizualizacje
- formatters.py – przechowujące zmodyfikowane formatery współrzędnych wyświetlanych pod wykresem (liczba miejsc po przecinku, normalizacja do jedności, dostęp do wartości Z dla 2D)
- localutils.py – odpowiada za wyodrębnienie wspólnych elementów wszystkich funkcji wizualizujących

Dodatek A – przygotowanie i uruchomienie aplikacji

Do uruchomienia aplikacji wymagane jest środowisko Python w wersji przynajmniej 3.7. W przypadku systemu Windows dostępne gotowe są przystępne instalatory – należy pamiętać by w trakcie instalacji zaznaczyć opcję dodającą pakiet TkInter. Dla systemu Linux potrzebne będzie zainstalowanie odpowiednich pakietów i proces będzie zależał od dystrybucji systemu. Dla dystrybucji opartych na Debianie polecenie *apt-get install python3.7 python3.7-tk* powinno dokonać wymaganych procedur.



Rys. A.1 – Zrzut ekranu przedstawiający instalator środowiska Python dla systemu Windows

Proces pobrania i instalacja bibliotek odbywa się za pomocą polecenia *pip install -r requirements.txt* zakładając, że polecenie zostało uruchomione w folderze z aplikacją, zawierającą plik requirements.txt. Zalecane, ale nie wymagane, jest również użycie wirtualnego środowiska Pythona, temat ten jednak wykracza poza tą pracę (więcej pod https://docs.python.org/3.7/tutorial/venv.html).

Aplikację z pobranym interpreterem Pythona i zainstalowanymi bibliotekami możemy uruchomić poleceniem *python main.py*.

Bibliografia

[1] Dokumentacja biblioteki Matplotlib, *https://matplotlib.org/*

[2] An Introduction to Tkinter, *https://effbot.org/tkinterbook/*

[3] Dokumentacja biblioteki NumPy, *https://numpy.org/devdocs/*

[4] Dokumentacja języka Python w wersji 3.7, *https://docs.python.org/3.7/*

[5] Glatzel P., Bergmann U.: *High resolution 1s core hole X-ray spectroscopy in 3d transition metal complexes—electronic and structural information, Coordination Chemistry Reviews 249, 2005, 65–95*

[6] van Bokhoven J. A., Lamberti C.: *X-Ray Absorption and X-Ray Emission Spectroscopy, 2016*

[7] Diamond Instruments, *https://www.diamond.ac.uk/Instruments/Techniques/Spectroscopy*

[8] Materiały Washington CEI, *https://www.cei.washington.edu/education/science-of-solar/*

Materiały uzupełniające

[I] Carpenter M. H.: *Helium Atmosphere Chamber for Soft X-ray Spectroscopy of Biomolecules, University of California, Davis, 2010*

[II] J. P. A. Luuk, van Veenendaal M., Devereaux T. P., Hill J. P., van den Brink J.: *Resonant inelastic x-ray scattering studies of elementary excitations, Reviews of modern physics, 83, 705, 2011*