Do Dado Bruto ao Insight: Automatizando o Fluxo de Análise de Dados de Fluorescência de Raios X

Victor Guilherme Rosa Pimenta*, Matheus Dias Melo†, Fábio Luiz Melquiades‡, Leonardo Carmezini Marques§
Instituto Federal do Paraná , Londrina, Paraná* †§
Universidade Estadual de Londrina, Londrina, Paraná ‡
victor.pimeenta@gmail.com * matheus.firc@gmail.com † fmelquiades@uel.br ‡ leonardo.carmezini@ifpr.edu.br §

***Abstract*—The analysis of materials using Energy Dispersive X-Ray Fluorescence (ED-XRF) is a fast and effective technique, but its potential is often limited by the output of data in unstructured text formats. This operational bottleneck requires analysts to perform a time-consuming and error-prone manual process of data extraction and formatting before any analysis can begin. This work-in-progress paper presents the development of an open-source software, built in Python, to fully automate this workflow, specifically targeting files generated by the Shimadzu EDX-7000 spectrometer. The methodology employs libraries such as Pandas and NumPy for data parsing and structuring, and PyQt5 with Matplotlib for the creation of an interactive graphical interface. As preliminary results, the functional prototype is already capable of loading raw CSV files, processing the information automatically, and displaying the elemental composition results in an organized table, as well as plotting the XRF spectrum directly within the interface. Initial validation confirms that the tool optimizes analysis time while significantly reducing the risk of manual transcription errors. Future work includes expanding compatibility to other equipment models and implementing advanced analytical tools.**

***Keywords*—Data Automation, ED-XRF, Spectral Analysis, Scientific Software, Scientific Workflow.**

***Resumo*—A análise de materiais por Fluorescência de Raios X por Dispersão de Energia (ED-XRF) é uma técnica rápida e eficaz, mas seu potencial é frequentemente limitado pela exportação de dados em formatos textuais não estruturados. A análise de dados é retardada por um gargalo operacional crítico: a extração e formatação manual dos dados, um processo demorado e com alta propensão a erros. Este trabalho (*work in progress*) apresenta o desenvolvimento de um software de código aberto, concebido em Python, para automatizar completamente este fluxo, focando especificamente em arquivos gerados pelo espectrômetro EDX-7000 da Shimadzu. A metodologia emprega bibliotecas como Pandas e NumPy para o *parsing* e estruturação de dados, e PyQt5 com Matplotlib para a criação de uma interface gráfica interativa. Como resultados preliminares, o protótipo funcional já é capaz de carregar arquivos brutos, processar as informações de forma automática e exibir os resultados de composição elementar em uma tabela organizada, além de plotar o espectro XRF diretamente na interface. A validação inicial confirma que a ferramenta otimiza o tempo de análise, reduzindo significativamente o risco de erros de transcrição manual. Os próximos passos incluem a expansão da compatibilidade para outros modelos de equipamentos e a implementação de ferramentas analíticas avançadas.**

***Palavras-chave*—Automação de Dados, ED-XRF, Análise Espectral, Software Científico, Fluxo de Trabalho Científico.**

I. Introdução

A descoberta dos raios X por Wilhelm Conrad Röntgen e a posterior lei de Henry Moseley, que estabeleceu a relação fundamental entre o número atômico de um elemento e seu espectro de emissão, foram os pilares para a análise moderna de materiais. Destas bases, emergiu a Fluorescência de Raios X (XRF), uma técnica analítica consolidada e versátil [2]. Sua variante, a Fluorescência de Raios X por Dispersão de Energia (ED-XRF), destaca-se pela rapidez e custo-benefício, tornando-se indispensável no controle de qualidade e pesquisa [3].

Apesar da eficácia da técnica, a complexidade e a falta de padronização dos dados tornam sua interpretação um desafio, exigindo um conhecimento prévio e específico sobre o funcionamento de cada equipamento. Pesquisadores apontam que o crescente volume de dados espectrométricos representa um desafio que exige soluções de software mais eficientes para a sua organização e abstração [1]. Sem elas, os analistas são forçados a um processo manual de extração e formatação. Este fluxo de trabalho demanda um esforço cognitivo focado primariamente nos níveis mais básicos da Taxonomia de Bloom: recordar a localização dos dados e compreender sua estrutura.

Esse processo manual não apenas retarda a análise, mas é criticamente suscetível a erros, pois força o analista a lidar com um grande volume de informações e a realizar operações de curadoria que, ao menor descuido, podem invalidar todo o valor analítico dos dados. Diante disso, este trabalho busca superar os desafios da despadronização para simplificar os processos complexos originados, especificamente, pelo espectrômetro EDX-7000 da Shimadzu.

O objetivo principal deste trabalho é, portanto, ofertar uma aplicação de interface intuitiva que automatiza e abstrai etapas de baixo valor de pesquisa. Pela sua facilidade de uso, a ferramenta visa oferecer a possibilidade de que pessoas não especializadas possam extrair o potencial total do equipamento EDX-7000, encurtando ao máximo o caminho entre o usuário e seu objetivo de pesquisa.

II. Trabalhos Correlatos

Para contextualizar a contribuição deste projeto, é fundamental analisar as soluções de software existentes para o processamento e visualização de dados. A análise se divide em duas categorias principais: ferramentas de *Business Intelligence* (BI) de propósito geral e pacotes de software científico especializados.

A. Ferramentas de Business Intelligence (BI)

Plataformas como *Microsoft Power BI Desktop* e *Tableau* oferecem interfaces intuitivas para a criação de *dashboards* interativos. Sua principal vantagem reside na acessibilidade para criação de gráficos interativos e diversos sem a necessidade de programação. Contudo, sua aplicação no contexto de EDXRF é limitada, é necessário que o usuário escreva a lógica de processamento uma vez que não possuem mecanismos nativos para realizar o *parsing* de arquivos semiestruturados como os gerados por equipamentos científicos, requerindo soluções de contorno obtusas e ineficientes.

B. Ferramentas Científicas Especializadas

No outro extremo, existem pacotes de código aberto altamente especializados, como o *PyXRF*. Ferramentas como esta são eficazes, oferecendo funcionalidades avançadas para a análise espectral detalhada e ajuste de picos. Sua principal limitação, no entanto, reside na acentuada curva de aprendizado, que pressupõe conhecimento prévio em programação e nos fundamentos da espectroscopia.

Com a análise das aplicações acima, se revela uma lacuna clara. De um lado, as ferramentas de BI oferecem visualização de dados de alto nível, mas necessitam que o próprio usuário escreva rotinas de processamento dos dados científicos em questão para tornar o processo viável. Do outro, as ferramentas de análise científicas foram criadas sob medida para oferecer poder analítico, mas podem não ser intuitivas para o usuário médio, também, sem dinamismo. Este projeto se insere nesta lacuna, propondo uma solução que seja acessível para o usuário iniciante, sem sacrificar a robustez necessária para o profissional.

III. Método

Este capítulo detalha a abordagem metodológica utilizada para a concepção e implementação do software. A pesquisa é caracterizada como aplicada, com foco central em desenvolvimento tecnológico. O percurso metodológico foi estruturado para progredir de uma prova de conceito para a entrega de um protótipo funcional seguindo um ciclo de desenvolvimento iterativo com base na metodologia *Scrum*. A seção a seguir está dividida em subtópicos que descrevem a arquitetura, os requisitos de negócio e a lógica interna do sistema.

A. Desenvolvimento do Projeto

O processo foi dividido em fases, desde a concepção à implementação. A ideia de seu desenvolvimento partiu de uma necessidade prática identificada em uma iniciação científica. Constatou-se que os dados exportados de um equipamento EDXRF local eram volumosos e não estruturados exigindo muito tempo gasto em processos transformativos para de fato iniciar-se a análise, o que motivou a automação dos mesmos. Diante da ideia e definição de requisitos, uma metodologia semelhante à metodologia *Scrum* foi seguida, na qual reuniões semanais são realizadas para validação da lógica implementada e definição de novas regras de negócio e requisitos. Essa iteratividade permitiu identificar problemas de forma rápida e aflorar novas ideias para serem implantadas, o que acelerou significativamente a geração de valor na aplicação.

B. Arquitetura de Software e Tecnologias

A arquitetura do software é baseada no *Model-View-Controller* (MVC) para garantir uma clara separação de responsabilidades e a escalabilidade ao projeto. As responsabilidades de cada camada são definidas da seguinte forma:

1) Model (Modelo): O Modelo é a camada lógica que descreve: o estado do objeto, o que compõe o objeto e como se deve interagir com ele ou outros objetos;

2) View (Visão): A Visão é a camada de apresentação, guardando a lógica de definição das interfaces a serem usadas no projeto. Ela não contém lógica de negócio e se mantém passiva, tendo a única funcionalidade de mostrar o estado atual dos modelos;

3) Controller (Controlador): O Controlador atua como o intermediário entre o Modelo e a Visão. Ele recebe e interpreta as entradas do usuário como cliques ou comandos de teclado, processa essas ações e aciona as atualizações necessárias no Modelo. Em seguida, atualiza ou seleciona a Visão de apropriada para responder à interação do usuário.

A *stack* usada para o desenvolvimento inclui: o editor de código *Visual Studio Code*, PyQt5 Designer para o desenho de interfaces, e Python para a escrita da lógica e funcionalidades citadas no artigo. A escolha destas ferramentas se justifica pela sua gratuidade e suporte de uma comunidade ativa com funcionalidades bem documentadas, Python se destaca pelo seu robusto ecossistema de bibliotecas para ciência de dados, alinhando-se às práticas modernas de computação científica, em que o ecossistema SciPy—incluindo NumPy, Pandas e Matplotlib—estabeleceu-se como padrão [4]; também, as bibliotecas Matplotlib e PyQt5 se mostram extremamente úteis e versáteis na criação de gráficos e interfaces, respectivamente. As bibliotecas usadas são detalhadas no Quadro 1.

Quadro I
BIBLIOTECAS PYTHON UTILIZADAS NO DESENVOLVIMENTO DO SOFTWARE

Nome do Pacote	Papel no Projeto	Descrição da Função Específica
Pandas	Processamento de Dados	Ferramenta central para a extração de dados brutos e sua conversão em estruturas de dados tabulares *(DataFrames)*, viabilizando a organização, limpeza e análise.
NumPy	Processamento Numérico	Garante a performance em cálculos e manipulações matemáticas aplicadas aos dados espectrais, operando sobre arrays e matrizes multidimensionais.
PyQt5	Interface Gráfica (GUI)	*Framework* responsável pela criação da interface gráfica de desktop, incluindo todos os elementos interativos como menus, botões e áreas de visualização.
Matplotlib	Visualização de Dados	Utilizada para a geração das visualizações dos espectros de XRF, permitindo a inspeção visual dos picos característicos diretamente na interface da aplicação.

FONTE: Adaptado de [7], [6], [8] e [5]

C. Requisitos Funcionais e Fluxo de Operação

O escopo do software foi definido a partir de um conjunto de requisitos funcionais elaborados especificamente para atender às necessidades de um analista profissional do campo e de pessoas iniciantes que se propunham utilizar equipamentos EDX-7000. O sistema foi projetado para permitir a importação de dados brutos em formato CSV e para automaticamente extrair resultados quantitativos e/ou espectrais através de opções de processamento customizáveis. Os processamentos customizados incluem a avaliação cruzada dos dados com certificados de materiais de referência. Na camada de apresentação, os requisitos incluem a visualização interativa dos dados tanto em formato tabular quanto gráfico. Por fim, o sistema deve ser capaz de gerar relatórios customizáveis em formato Excel. Este fluxo de operação garante que o usuário tenha controle total sobre o processo para se dedicar à atividade de maior valor: a análise e a geração de *insights*.

D. Lógica de Processamento e Extração de Dados

O núcleo do software reside em rotinas calibradas para operar sobre arquivos de origem gerados pelo espectrômetro EDX-7000, executando processos de curadoria de dados. Para navegar na natureza semiestruturada desses arquivos, o algoritmo de *parsing* opera com base em um sistema de *tokens*. Estes *tokens* são marcadores lógicos — como cabeçalhos de seção, *flags* específicas, ou palavras-chave — que servem para localizar e isolar blocos de informação com precisão.

O fluxo de pré-processamento para uma população de amostras é uma orquestração de múltiplas etapas. Primeiramente, a rotina de contextualização extrai os metadados essenciais de cada amostra, como nome, data e condições instrumentais, utilizando *tokens* para localizar essas informações. Dada a primeira limpeza, depende ao usuário decidir quais informações extrair, resultados quantitativos ou espectros.

1) Espectros: O bloco de dados correspondente ao espectro de 2048 pontos é localizado e reestruturado em um formato tabular, automaticamente indentificando os canais presentes na amostra e os separando respectivamente.

2) Resultados quantitativos: De forma análoga, a seção dos resultados quantitativos é identificada para que os dados de concentração elementar, intensidade e desvio padrão sejam extraídos e organizados em formato tabular.

Por fim, após a extração individual, o software disponibiliza ao usuário, por meio de um menu de contexto, funcionalidades que possibilitam o isolamento de elementos de interesse via filtros, o cálculo de métricas estatísticas e a aplicação de fatores de correção vindos de certificados de calibração. Um pilar da arquitetura é a generalização, que garante robustez e escalabilidade. Em vez de depender de uma lista fixa de canais ou elementos, o software prega pelo processamento genérico. Para os espectros, é mapeado todos os canais de elementos disponíveis em uma amostra de referência e utiliza este mapa dinâmico para processar as demais, adaptando-se a (N) número de canais. Para os resultados quantitativos, a rotina localiza o *token* da seção e extrai todas as linhas de dados que se seguem, lidando nativamente com amostras que contenham poucos ou muitos elementos. Isso torna o software flexível para operar sobre arquivos com diferentes configurações sem necessidade de modificação no código.

IV. RESULTADOS PRELIMINARES E DISCUSSÃO

O protótipo funcional atual possui o fluxo descrito já em funcionamento junto com as funcionalidades de exportação em Excel, o usuário pode selecionar um ou varios arquivos CSV's de um EDX-7000 da empresa Shimadzu de forma simultânea, que é/são lido(s) e processado(s) automaticamente, e também, selecionar uma ou mais populações para a exportação em Excel, porém, somente um tipo por vez, espectros ou resultados quantitativos. Como resultado imediato, a aplicação exibe todos os arquivos carregados em formato hierárquico em árvore, contendo as informação de cada amostra em formato tabular após um clique na mesma. A Figura 1 e 2 ilustra a interface da aplicação em funcionamento.

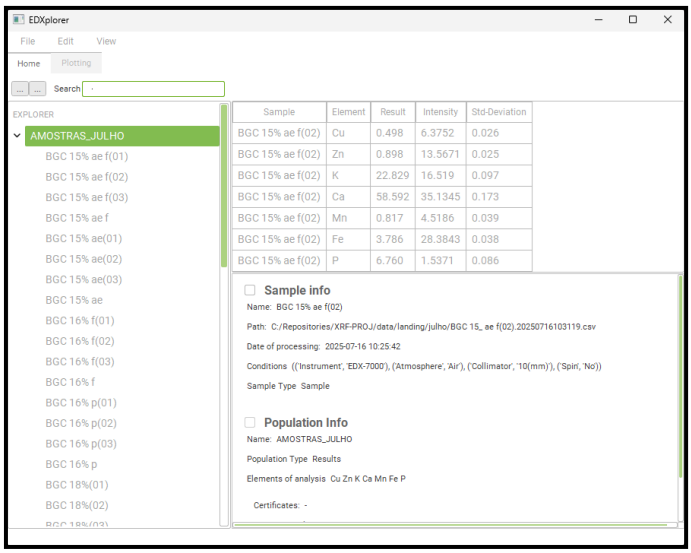

Fig. 1. Interface principal do protótipo em desenvolvimento.
FONTE: Elaborado pelos autores (2025).

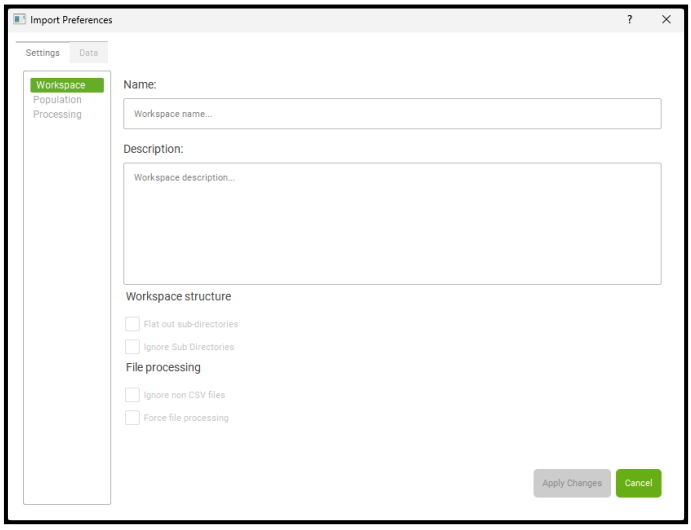

Fig. 2. Interface para importação de arquivos.
FONTE: Elaborado pelos autores (2025).

V. Conclusão final e Trabalhos Futuros

Este trabalho apresenta resultados do desenvolvimento de um protótipo funcional capaz de automatizar o processo de extração e organização de dados provenientes de equipamentos EDX-7000, posicionando-se como uma solução intermediária entre ferramentas correlatas citadas no item II. A arquitetura modular e a lógica de processamento genérica garantiram a flexibilidade e a escalabilidade da aplicação, que permite a implementação de ideias futuras com maior facilidade. O software em seu corrente estado já suporta a importação de múltiplos arquivos e a extração de seus resultados quantitativos e espectrais e a visualização desses dados formatados de forma tabular. Também é suportado a exportação dos mesmos para planilhas Excel para futuro re-uso. Porém, a aplicação como um todo ainda não está totalmente realizada. Os passos seguintes incluem a adição da plotagem de gráficos interativos diretamente na aplicação, além disso, é planejado o desenvolvimento e implementação de ferramentas para a manipulação de populações de amostras, permitindo análises comparativas e estatísticas mais complexas, consolidando o software como uma ferramenta viável e principalmente acessível para a comunidade científica, profissional e leiga. Futuramente, é pretendido refinar as interfaces gráficas tomando inspiração de softwares já existentes de modo a aprimorar a experiência do usuário, também, para seguir o ponto focal do projeto, almeja-se o desenvolvimento de novas rotinas que suportem dados de diferentes equipamentos ED-XRF possibilitando o uso da aplicação não só para equipamentos da Shimadzu, mas como para todo o ecossistema.

Agradecimentos

Agradecemos, primeiramente, a Deus. Estendemos nossa gratidão aos nossos familiares, pelo incentivo e apoio incondicional durante todas as etapas deste projeto. Aos(Às) professores Isaías Venâncio da Luz Filho, pela orientação adicional, paciência e ensinamentos que auxiliaram no desenvolvimento do trabalha, e a professora Talita Canônico Silva Zavilenski, pela fundamental revisão e pelas sugestões precisas que aprimoraram a qualidade deste texto.

Referências

[1] R. M. Jarvis and R. Goodacre, "Genetic algorithm optimization for pre-processing and variable selection of spectroscopic data," *Bioinformatics*, vol. 21, pp. 860–868, 2004.
[2] B. Beckhoff *et al.*, *Handbook of Practical X-Ray Fluorescence Analysis*. Springer, 2006.
[3] M. Haschke, J. Flock, and M. Haller, *X-ray Fluorescence Spectroscopy for Laboratory Applications*. Wiley-VCH, 2021.
[4] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.
[5] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.
[6] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
[7] Riverbank Computing Limited, "PyQt5 Documentation," 2025. [Online]. Available: https://www.riverbankcomputing.com/static/Docs/PyQt5. [Accessed: Aug. 21, 2025].
[8] T. pandas development team, "pandas-dev/pandas: Pandas," Zenodo, 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134.