Sistema de Monitoramento e Análise Híbrida de Sinais Neurais (EEG)

Relatório Técnico - Unidade 2

Aluno: Leonardo Rosendo Faustino
Disciplina: Introdução as Técnicas de Programação

Novembro de 2025

Projeto final apresentado como requisito parcial para obtenção de nota na Unidade 2.

Sumário

1 Introdução

O presente projeto consolida a evolução do sistema desenvolvido na Unidade 1, expandindo o escopo do gerador de sinais sintéticos para um sistema híbrido e robusto. O objetivo central desta etapa foi superar as limitações de memória estática e simulação simples, permitindo ao software carregar, armazenar e processar grandes volumes de dados reais (aproximadamente 600.000 pontos de um EEG de rato), além de manter a capacidade de geração sintética.

As novas funcionalidades implementadas visam atender diretamente às sugestões de melhoria propostas na avaliação anterior, introduzindo algoritmos de suavização de sinal (Média Móvel) e uma lógica preditiva de tendências (Alta e Baixa). O projeto explora conceitos avançados da linguagem C, como alocação dinâmica de memória, manipulação de ponteiros e estruturas de repetição aninhadas.

2 Metodologia

O desenvolvimento foi realizado utilizando a linguagem C, com compilação via GCC e edição no Visual Studio Code. A abordagem metodológica focou na modularização funcional dentro de um arquivo único (`main.c`), onde o código foi estruturado em blocos lógicos distintos: carregamento de dados, geração sintética, processamento matemático e exibição.

Para o controle de versão, manteve-se o uso do Git, garantindo um histórico evolutivo das implementações. A validação dos algoritmos foi realizada através da comparação visual entre os dados brutos e os dados processados, utilizando amostras aleatórias contíguas para verificar a eficácia da suavização e da predição de tendências.

3 Análise do Código

A Unidade 2 exigiu a aplicação prática de conceitos fundamentais para lidar com a complexidade dos dados reais. A seguir, detalha-se a implementação técnica:

- **Manipulação de Arquivos e Strings:** Para integrar os dados do rato, utilizou-se a biblioteca `stdio.h` para abrir e ler o arquivo CSV. A função `fgets` foi empregada para capturar cada linha como uma string, sendo posteriormente convertida para precisão dupla (`double`) através da função `atof`. O sistema identifica dinamicamente os limites (mínimo e máximo) do sinal durante a leitura para calibração automática da análise.

- **Alocação Dinâmica e Ponteiros:** O gerenciamento de memória foi o pilar central desta unidade. Devido ao volume massivo de dados, a alocação estática na stack tornou-se inviável. Implementou-se o uso de `malloc` e `realloc` para alocar e expandir dinamicamente vetores na memória *heap*, permitindo o armazenamento flexível dos sinais. Ponteiros foram utilizados extensivamente para passagem de dados por referência entre funções, otimizando o desempenho e evitando cópias desnecessárias. A liberação de memória foi rigorosamente tratada com `free` para evitar vazamentos (*memory leaks*).

- **Estruturas de Repetição Aninhadas (Média Móvel):** Em resposta direta à sugestão do Professor para interpretar o sinal usando uma média dos últimos sinais, foi desenvolvida a função `calcularMediaMovel`. Esta funcionalidade aplica laços aninhados: um laço externo percorre o vetor de dados, enquanto um laço interno calcula a média de uma janela deslizante (5 pontos anteriores). Isso atua como um filtro de suavização, essencial para reduzir ruídos de leitura.

- **Lógica de Tendência e Previsão:** A função `analisarTendencia` introduziu a capacidade preditiva ao sistema. Calculando a derivada discreta (variação entre pontos adjacentes), o algoritmo identifica se o sinal apresenta uma subida ou descida brusca (maior que 1% da amplitude total) antes mesmo de atingir os limiares de alerta ou repouso. Isso permite notificar tendências de "Alta"ou "Baixa"em tempo real.

4 Dificuldades e Soluções

Durante a implementação, desafios técnicos que surgiram e foram superados incluíram:

- **Gerenciamento de Dependências:** Houve erros de compilação relacionados a constantes não declaradas (`DBL_MAX`). A solução envolveu a inclusão correta da biblioteca `<float.h>` e a revisão das diretivas de pré-processador.

- **Declaração Implícita de Funções:** A organização do código gerou erros onde a função `main` tentava acessar funções definidas posteriormente no arquivo. O problema foi sanado com a implementação de protótipos de função no início do código, informando as assinaturas ao compilador previamente.

- **Sincronização de Visualização:** Ao comparar o sinal original com o suavizado, observou-se inicialmente que as amostras aleatórias não coincidiam. A solução foi centralizar a lógica de sorteio do índice na função `main`, passando o mesmo índice inicial para ambas as funções de exibição, garantindo a coerência visual na análise comparativa.

5 Conclusão

O projeto da Unidade 2 representa um salto em relação à versão anterior. A transição para o uso de dados reais validou a robustez da arquitetura de software e exigiu uma nova construção mais condizente, enquanto a adoção de alocação dinâmica provou ser indispensável para aplicações de engenharia que lidam com grandes volumes de informação.

As implementações da Média Móvel e da Análise de Tendência atenderam plenamente às sugestões da U1, transformando o sistema de um simples monitor para uma ferramenta capaz de suavizar ruídos e prever comportamentos anômalos. O sistema final demonstra estabilidade, eficiência no uso de memória e clareza na apresentação dos dados, cumprindo com êxito os requisitos acadêmicos e técnicos propostos para U2.