Avaliação de desempenho da equação do calor (Python vs Julia)

Enzo Lisbôa Peixoto
Nathan Mattes
Pedro Scholz Soares
Lucas Mello Schnorr (orientador)

Resumo

Este trabalho tem como objetivo comparar o desempenho das linguagens de programação Python e Julia na resolução de equações do calor em 1, 2 e 3 dimensões. Se atentando ao tempo percorrido na execução dos programas, assim como a memória utilizada pelos mesmos, a fim de determinar qual das duas é a mais eficiente para esta aplicação específica. Este trabalho é feito com o intuito de ser um estudo em cálculo computacional e análise de desempenho.

1. DESCRIÇÃO

O experimento é executado em um Docker que carrega as dependências das linguagens Julia e Python. Neste Docker é executado um script que gera uma ordem aleatória para a execução dos testes, os quais são combinações de uma linguagem (Julia ou Python), uma dimensão (1 dimensão e 2 dimensões) e uma carga de trabalho (low, mid e high). Cada combinação será executada 10 vezes.
Com o plano de experimento pronto, outro script é utilizado para executar os códigos em Julia e Python com as determinadas configurações na ordem gerada aleatóriamente, salvando os resultados e coletando dados de uso de memória e tempo de execução.
Com os dados em mãos, são executados scripts em R para a geração dos gráficos que serão utilizados na análise.

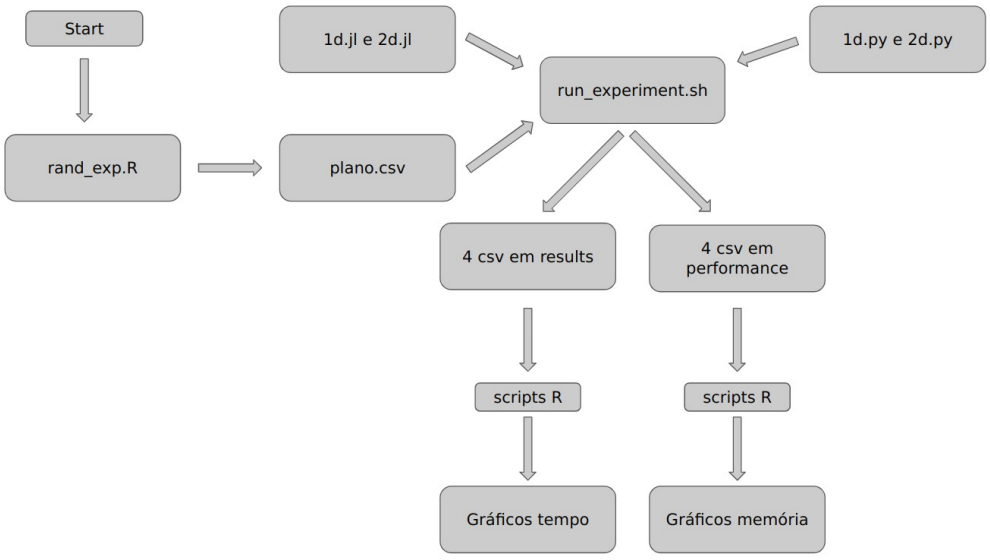

Figura 1: Organograma dos Experimentos

2. MÉTODO DE COLETA DE DADOS

Os dados coletados são o uso de memória e o tempo de execução dos algoritmos. O primeiro é coletado através do Docker no qual os experimentos são executados, já que a ferramenta proporciona fácil acesso a esse dado. O

segundo, por sua vez, é calculado pelos próprios algoritmos durante a execução. Estes dados são distruibuídos em duas pastas para serem tratados e analisados.

3. RESULTADOS PRELIMINARES

Com os dados obtidos agora conseguimos gerar os gráficos dos tempos de execução. Todos os dados foram obtidos com um computador com a seguinte configuração: 12th Gen Intel(R) Core(TM) i3-1215U e 8GB de ram.
Na figura 2 podemos ver os tempos do Julia, que, apesar de ter alguns pontos fora da curva, mantém uma média consistente. Com exceção do caso de duas dimensões com carga alta que teve uma variação maior tendendo para um tempo de execução mais alto.

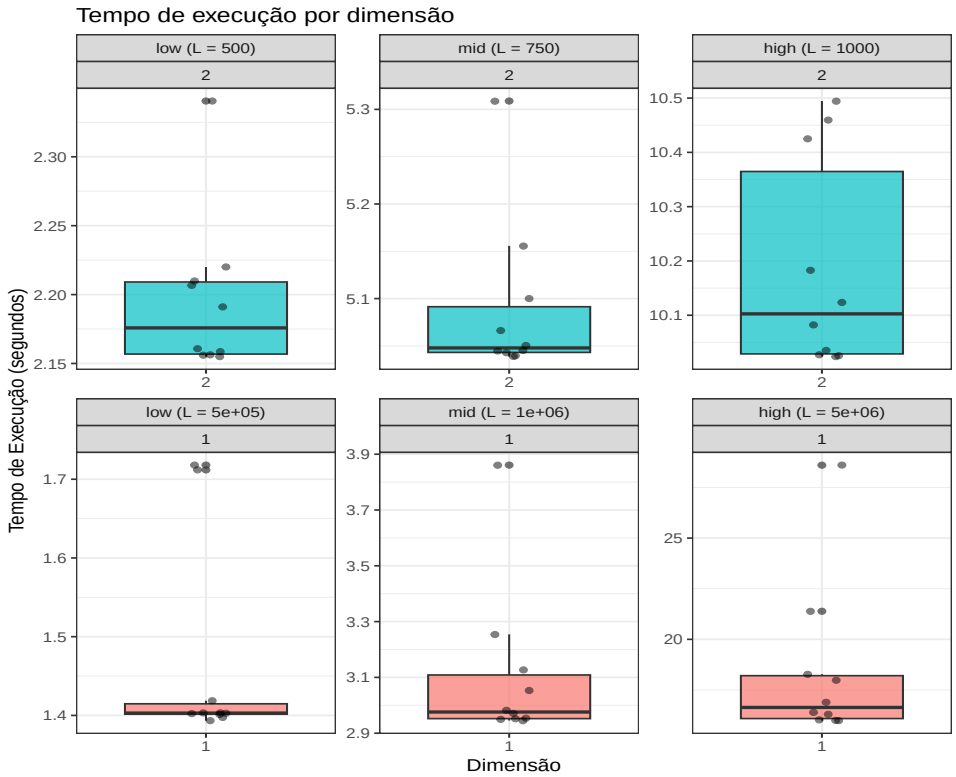

Figura 2: Tempo de execução do Julia para todos os casos

Em seguida, na figura 3, temos os tempos de execução do Python, que se mostra mais consistente que o Julia em suas médias de tempo, apesar de eles serem muito maiores.

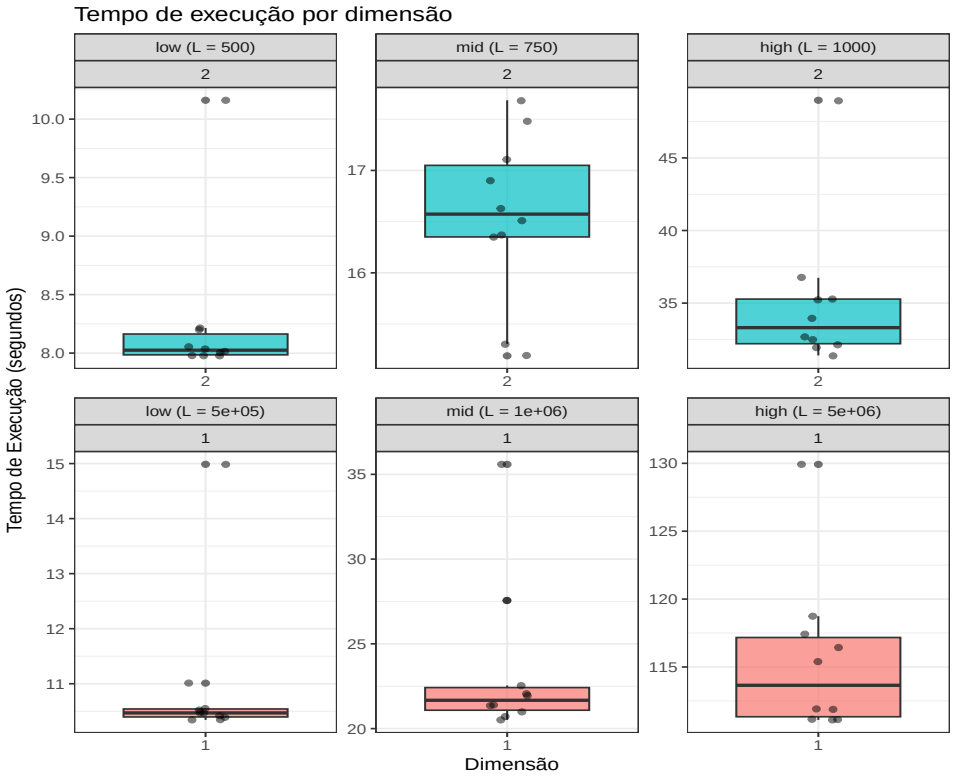

Figura 3: Tempo de execução do Python para todos os casos

Colocando todos os dados em um mesmo gráfico temos uma ideia melhor de o quão mais rápida é a execução em Julia em relação ao Python.

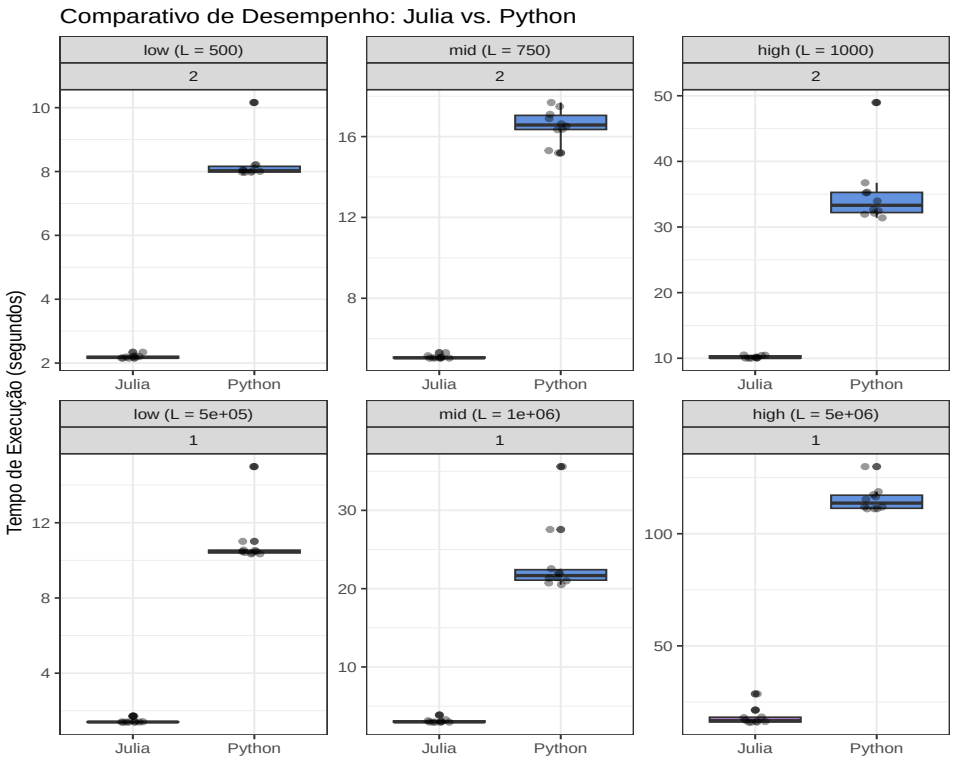

Figura 4: Comparação do tempo de execução entre Python e Julia para todos os casos

Em comparação, podemos ver nas figuras 5 e 6 que o Julia consome, proporcionalmente, uma quantidade muito maior de memória que o Python. Principalmente com um maior número de dimensões.

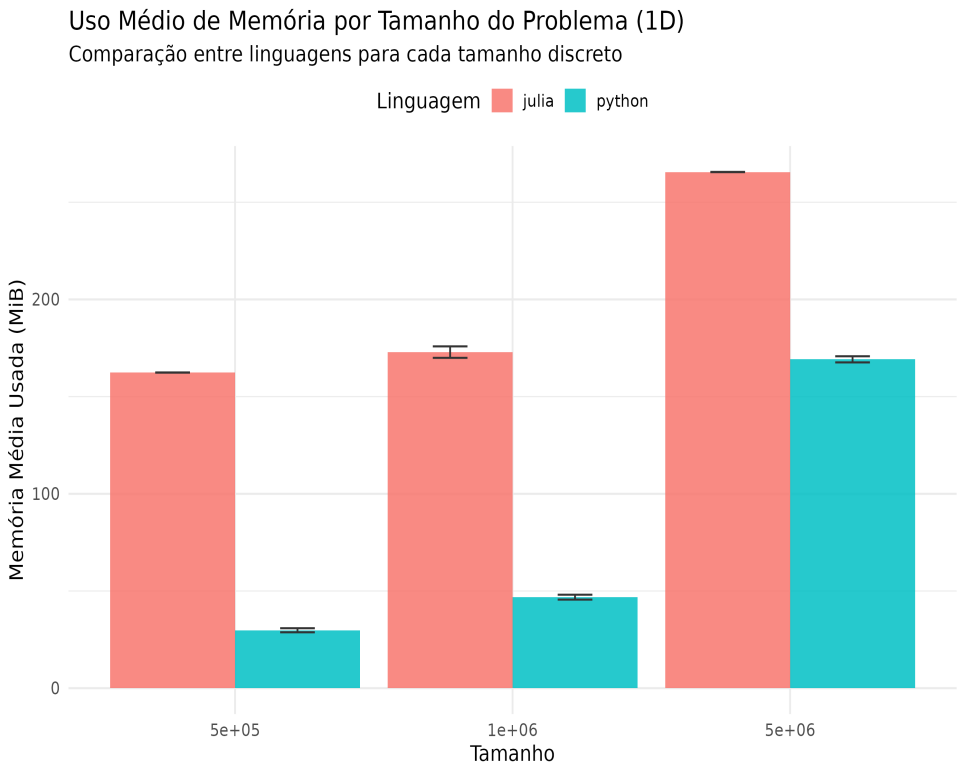

Figura 5: Uso de memória para 1 dimensão

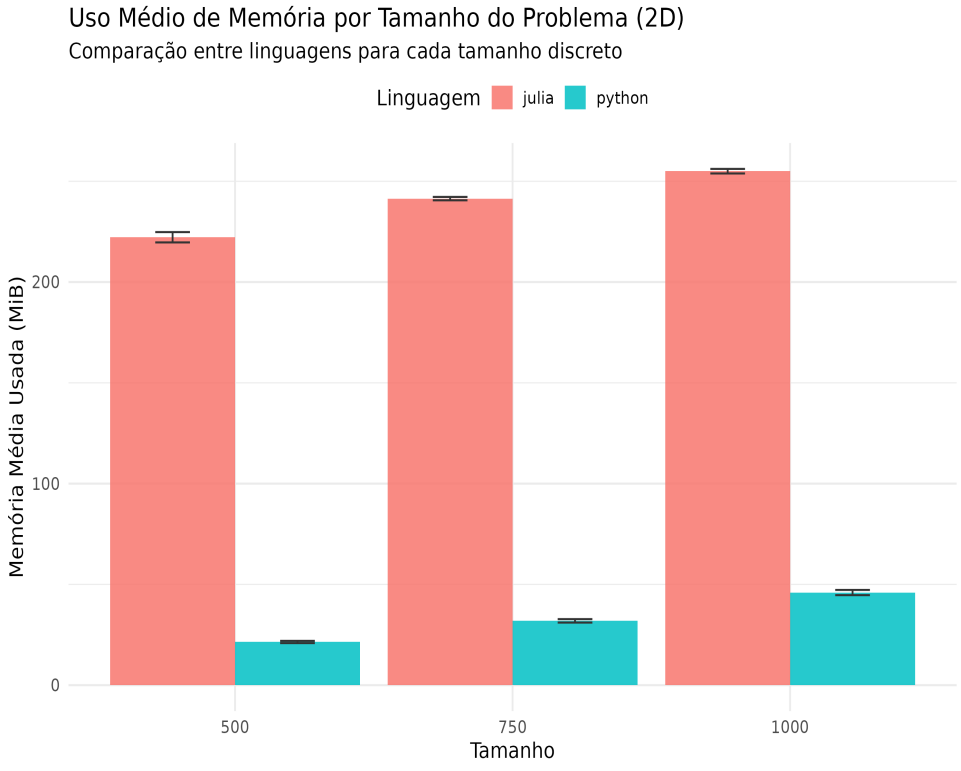

Figura 6: Uso de memória para 2 dimensões

4. DIFICULDADES E SOLUÇÕES

Dentre as dificuldades encontradas estão: como formatar os dados de uma maneira que facilite o entendimento pelo grupo; Criar scripts nas linguagens Bash e R; Criar os códigos que executem os cálculos das equações; A configuração das ferramentas utilizadas.
Que foram solucionadas: separando os dados em dois casos, dados de performance e dados de resultados. Os primeiros tem uma linha extraída a cada intervalo de tempo, enquanto o segundo tem uma linha extraída a cada experimento. Cada linha possui 5 variáveis preditoras e as restantes vem do experimento, mantendo assim os dados tidy; a criação dos scripts e códigos foi feita à base de pesquisa, tentativa e erro até que se chegasse no resultado aqui apresentado; e por fim, as ferramentas que utilizamos por vezes não funcionavam corretamente para todos os membros do grupo, que precisavam com pesquisas e tentativas corrigir o comportamento delas.

5. PLANO PARA FINALIZAÇÃO E CRONOGRAMA ATUALIZADO

Partindo desta etapa parcial, pretende-se: organizar os scripts auxiliares que estão e serão utilizados pelo experimento; ampliar a análise ao incluir a 3° dimensão que ainda não foi analisada e usando comparações que forneçam maior esclarecimento sobre os dados; uma análise qualitativa dos resultados; estabelecer um limite claro para o hardware no qual os testes serão realizados, não apenas utilizar o que nos é disponível; a coleta total dos dados; a análise final dos dados; e, por fim, a escrita do relatório final.

Atividade	Prazo Máximo
Organizar os scripts auxiliares	11/10
Ampliar a análise com novas comparações	18/10
Análise qualitativa dos resultados	25/10
Expecificação do hardware para o experimento	01/11
Coleta total dos dados	15/11
Análise final dos dados	22/11
Escrita do relatório final	30/11