ESCOLA POLITÉCNICA
PUCRS
Pontifícia Universidade Católica do Rio Grande do Sul
Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais

APLICAÇÃO DE CIÊNCIA DE DADOS E INTELIGÊNCIA ARTIFICIAL NA ANÁLISE DE DADOS DE CORRENTES DE DENSIDADE GERADOS POR SIMULAÇÃO COMPUTACIONAL

VLADIMIR BARCELOS DORNELES JUNIOR
COLOCAR O TÍTULO OBTIDO NA GRADUAÇÃO

ANTEPROJETO PARA A OBTENÇÃO DO TÍTULO DE MESTRE EM ENGENHARIA E TECNOLOGIA DE MATERIAIS

Porto Alegre

Novembro, 2024

Epígrafe
(Autor)

DEDICATÓRIA

AGRADECIMENTOS

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Nenhuma entrada de índice de ilustrações foi encontrada.

LISTA DE SÍMBOLOS

RESUMO

DORNELES, Vladimir. **Aplicação de Ciência de Dados e Inteligência Artificial na Análise de Dados de Correntes de Densidade Gerados por Simulação Computacional.** Porto Alegre. 2024. Dissertação. Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL.

Correntes de densidade são fenômenos que ocorrem tanto na natureza quanto provocados pelo homem. O mecanismo de geração destas correntes se baseia na diferença entre a massa específica do fluido da corrente e a massa específica do fluido ambiente. O estudo das correntes de densidade é útil para várias áreas de conhecimento: geologia, engenharia e meteorologia são algumas das áreas que se utilizam de estudos desse tipo de fenômeno. Uma das formas de estudo das correntes de densidade é através da aplicação da fluidodinâmica computacional. A *fluidodinâmica computacional (Computational Fluid Dynamics, CFD)* é uma ferramenta usada para simular numericamente o comportamento de um escoamento, e todas as leis que governam o seu estudo. Esses processos de simulação computacional são processos com um custo computacional elevado, de grande tempo de execução e geram grandes volumes de dados com alta complexidade que precisam ser capturados, processados, armazenados e analisados. Com essas características dados gerados por CFD são uma potencial matéria-prima para aplicação de técnicas de ciência de dados e inteligência artificial. O objetivo desse trabalho é implantar funcionalidades que possibilitem captura, compactação, armazenamento, processamento, consulta, visualização e predições utilizando dados gerados por simulação computacional de correntes de densidade, auxiliando pesquisadores a padronizar as operações com dados, reduzir as operações computacionais e por meio de aprendizado de máquina diminuir a complexidade das informações armazenadas em bancos de dados conservando as características necessárias para que os depósitos de sedimentos possam ser calculados, analisados, visualizados e preditos.

Palavras-Chaves: Correntes de Densidade, Fluidodinâmica Computacional, Inteligência Artificial, Aprendizado Profundo, Autoencoders.

ABSTRACT

SOBRENOME DO AUTOR (todas em maiúscula - vírgula), Nome (ponto). **Título em inglês** (negrito ou itálico - ponto)**.** Local (sem abreviação - ponto). Ano (ponto). Master/PhD Thesis. Graduation Program in Materials Engineering and Technology, PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL.

Key-words:

1. INTRODUÇÃO

A Fluidodinâmica Computacional, ou *Computational Fluid Dynamics (*CFD), utiliza métodos numéricos para simulação de fenômenos que envolvem o escoamento de fluidos com ou sem troca de calor. Com ela é possível obter soluções numéricas das equações de Navier-Stokes, a fim de obter os campos de velocidade, pressão e temperatura em qualquer região de escoamento (Costal e Ribeiro, 2010). Stancanelli et al. (2018), falam que a dinâmica das correntes de densidade é investigada por meio de simulações numéricas de CFD. O objetivo é avaliar a influência do campo de velocidade do fluido ambiente nos processos de mistura e turbulência observados. As correntes de densidade são fenômenos baseados na diferença entre a massa específica do fluido da corrente e a massa específica do fluido ambiente. Esta diferença de massa específica faz com que o fluido de maior massa específica se movimente em direção ao de menor massa específica (Simpson, 1999).

Os dados gerados por simulações numéricas para o estudo de correntes de densidade tendem a ter um crescimento elevado. No trabalho de Farenzena (2020) são apresentados dados sobre o número de processadores que executaram o código computacional, o tempo de execução e a área em disco ocupada pelos arquivos de cada simulação. De acordo com os dados apresentados é possível ter uma ideia da alta demanda por recursos computacionais que é necessária para a execução do processo de simulação computacional. As observações de Farenzena (2020) estão ilustradas na Tabela 1.1, onde são exibidas informações acerca das necessidades de recursos computacionais.

Tabela 1.1. Recursos computacionais utilizados para cada caso de simulação (Farenzena, 2020).

Caso	Número de processadores	Espaço em disco	Tempo de Cálculo
LE1e3	72	74 GB	7 horas
LE2.5e3	144	473 GB	126 horas
LE5e3	72	57 GB	6 horas
MFE1e3	168	7 GB	23 horas
MFE2.5e3	96	40 GB	890 horas
MFE5e3	96	18 GB	100 horas

Questões relacionadas a problemática de uma demanda crescente por recursos computacionais, não estão confinadas somente a área de CFD. Conforme estatísticas e previsões da Statista, a quantidade total de dados gerados globalmente crescerá a uma taxa anual de 27% de 2020 a 2025, e é esperado que atinja 2142 *zetabytes* (ZB) até 2035 (Ruan et al, 2021). Esse incremento do volume de dados demanda maior capacidade computacional, que por sua vez necessita de sistemas de refrigeração mais potentes causando um aumento da demanda energética. De acordo com Altamira et al (2019), a demanda de eletricidade dos *datacenters* na União Européia em 2015 foi de 74 TWh/ano e está projetada para ser de 160 TWh/ano em 2030. Globalmente, o aumento é ainda maior: de 203 TWh/ano em 2015 para 3.390 TWh/ano em 2030. Já os modelos de previsão apresentados por Liu et al. (2021) apontam que em 2030 datacenters na China consumirão 400 TWh/ano, representando cerca de 4% da energia do país.

Com o objetivo de buscar soluções para a redução da demanda computacional necessária para a execução de atividades relacionadas a CFD, a Ciência da Computação apresenta algumas propostas, como por exemplo a visualização *in-situ,* os dados em *streaming* e a compressão de dados utilizando técnicas de aprendizado profundo. Bauer et al. (2016) definem o processamento *in situ* como aquele onde a análise e visualização dos resultados ocorrem ao mesmo tempo em que a simulação é executada. Um exemplo apresentado no trabalho de Bauer et al. (2016) está exibido na Figura 1.1 O trabalho de Bnà et al. (2023) quantifica o impacto de uma abordagem *in situ*, baseada no software *ParaView Catalyst,* em três códigos diferentes, denominados *OpenFOAM, STREAmS* e *MIGALE,* que implementam diferentes

esquemas numéricos e operam tanto no contexto industrial, quanto em contexto acadêmico.

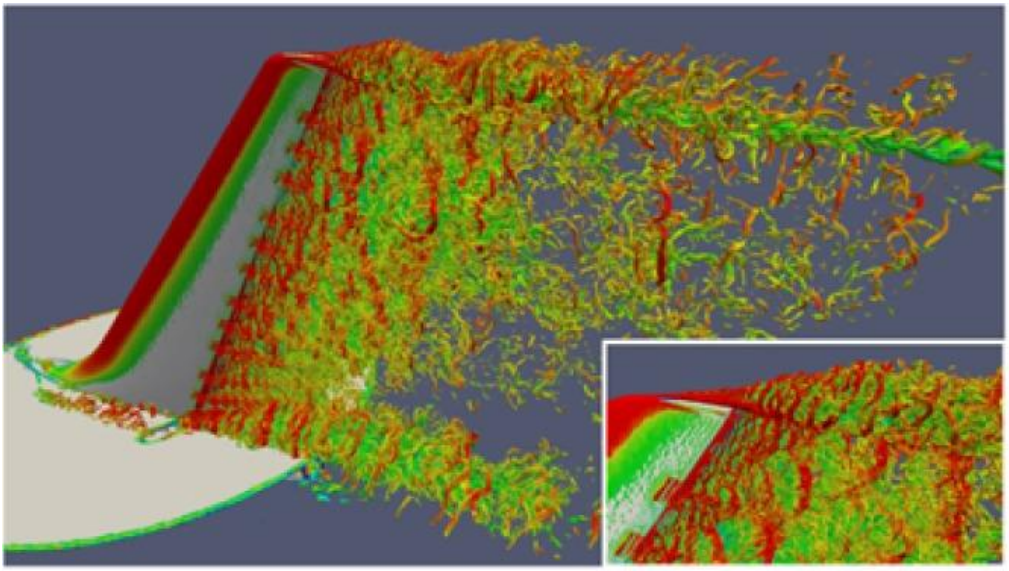


Figura 1.1. Exemplo de visualização In-Situ gerada com o uso do Paraview Catalyst (Bauer et al, 2016)

Quanto a dados de *streaming* a Amazon (2024) define como dados emitidos em alto volume de maneira contínua e incremental, com o objetivo de alcançar um processamento de baixa latência. No contexto da computação de alto desempenho (*High Performance Computing* - HPC) várias pesquisas são realizadas com o intuito de prover arquiteturas capazes de implementar esse fluxo de dados oriundos do *cluster* para outras plataformas com baixa latência. Li et al. (2020) propõe o *ElasticBroker,* que faz a ponte entre aplicações HPC e nuvem para formar um fluxo de trabalho científico *in situ*. Em vez de gravar os resultados da simulação em sistemas de arquivos paralelos, o *ElasticBroker* realiza a filtragem, a agregação e a conversão de formatos de dados para fechar a lacuna entre o ecossistema HPC e um ecossistema Cloud distinto. Matri e Ross (2021) apresentam o Neon, um projeto completamente novo de uma estrutura de processamento de dados de *streaming* para sistemas HPC que permite aos usuários criar *pipelines* de *streaming* de tamanho arbitrário. Poeschel et al. (2021) implementam um caminho de transição de operações de entrada e saída em arquivos para fluxos de trabalho baseado em *streaming* para aplicações científicas em ambientes de HPC, permitindo que fluxos de trabalho tradicionais, limitados por gargalos do sistema de arquivos, sejam superados e estendidos de forma flexível para análises *in situ*.

Por fim, a compressão de dados é outra estratégia que pode ser usada para diminuir os custos do pós-processamento. As técnicas de aprendizado profundo são utilizadas por pesquisadores para obter representações menores dos dados gerados por CFD, mantendo as características dos dados originais. Uma abordagem de aprendizado profundo para compressão de dados *in situ*, utilizando uma arquitetura de rede neural denominada *autoencoder* personalizada para fluxos turbulentos tridimensionais é demonstrada por Glaws et al. (2020). Momenifar et al. (2022) aplicam uma técnica de *Physics Informed Neural Network* (PINN), baseada em quantização vetorial, para gerar uma representação discreta e de baixa dimensionalidade dos dados de simulações de fluxos turbulentos tridimensionais. Liu et al. (2018), introduzem o aprendizado profundo na compressão de dados CFD e propõe um novo método de compressão *in situ* baseado em rede generativa adversarial (*Generative Adversarial Network* - GAN) em seu artigo. Especificamente, o método proposto amostra pequenos partes dos dados CFD e treina uma GAN que inclui duas redes neurais convolucionais: a rede discriminativa e a rede generativa. A rede discriminativa é responsável pela compressão dos dados nos nós de computação, enquanto a rede generativa é usada para reconstruir os dados nos nós de visualização. A arquitetura dessa rede aparece na Figura 1.2

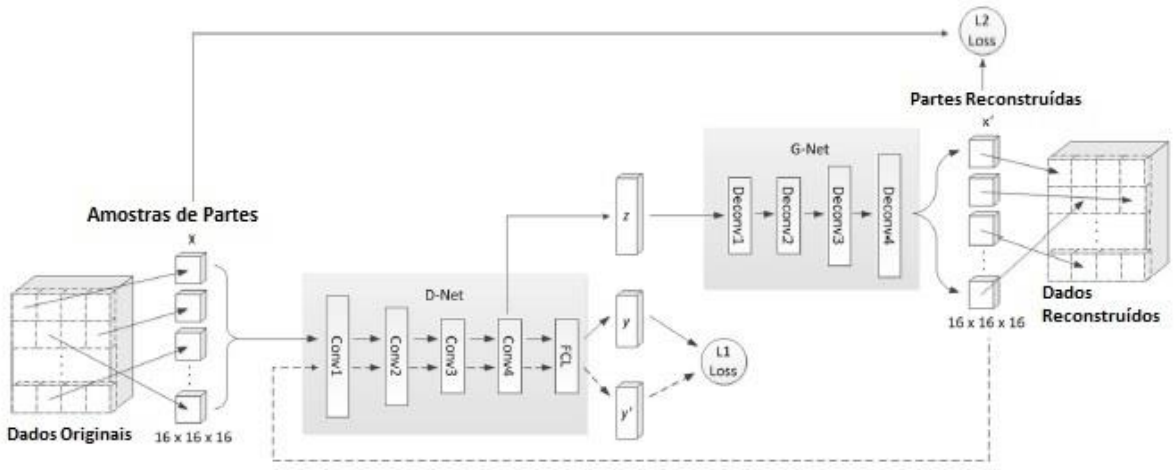


Figura 1.2. Arquitetura de método de compressão in situ proposto. A rede D-Net é utilizada para comprimir os dados, enquanto a rede G-Net é responsável pela reconstrução dos dados (Liu et al., 2018)

Outro ponto relevante a ser analisado é apresentado por Jiao et al. (2022): o compartilhamento de dados científicos tem sido incentivado e até exigido por revistas

e financiadores nos últimos anos. Seguindo essa tendência, a declaração de disponibilidade de dados tem sido cada vez mais adotada pelas comunidades acadêmicas como um meio de compartilhar dados de pesquisa em artigos científicos. Enquanto Tenopir (2020), fala que os objetivos da Ciência Aberta incluem uma maior colaboração científica interdisciplinar, acessibilidade dos dados e maior reprodutibilidade e transparência do trabalho científico. Esses objetivos dependem do aumento do compartilhamento de dados científicos e do acesso aberto aos dados.

Com base na problemática apresentada, onde por um lado existe a necessidade de armazenar e compartilhar informações e por outro lado existe a preocupação com os custos computacionais e enérgicos envolvidos em todo o trabalho de aquisição, processamento, armazenamento e análise de dados gerados por CFD. O trabalho tem por objetivo desenvolver aplicações computacionais utilizando técnicas de Ciência de Dados e Inteligência Artificial para apoiar a análise de dados de correntes de densidade gerados por simulação computacional para o LaSET (Laboratório de Simulação de Escoamentos Turbulentos. As aplicações que serão desenvolvidas são captura dos dados gerados por simulação computacional durante a sua execução, visualização *in-situ* e em tempo real dos dados gerados pela simulação computacional, compactação dos dados de uma simulação computacional utilizando técnicas de aprendizado profundo, armazenamento de dados em uma base de dados centralizada. Utilização das representações compactadas dos dados de simulação computacional para fazer cálculos de grandezas, gerar visualizações, fazer comparações entre resultados de duas simulações e prever um novo mapa com base na alteração de parâmetros sem a necessidade de rodar uma nova simulação computacional.

2. OBJETIVOS

Conforme apresentado na introdução, muitos desafios se apresentam quando os objetos de análise são dados oriundos da aplicação de técnicas de fluidodinâmica computacional. Sendo assim, as seguintes perguntas norteadoras da pesquisa são apresentadas:

- Como desenvolver um método que capture os dados de uma simulação computacional durante sua execução de forma que os dados possam ser analisados antes de serem gravados em disco?
- Como implementar um método de compressão utilizando redes neurais que reduza o tamanho, a complexidade dos dados mantendo as características do dado bruto simplificando o armazenamento, o processamento e a análise dos dados do laboratório?
- Como centralizar os dados gerados pelo laboratório em um repositório de dados único e a partir desses dados armazenados aplicar técnicas que gerem visualizações que simplifiquem a análise dos resultados obtidos?
- Com as representações resumidas dos dados, é possível realizar análises que atinjam resultados semelhantes com as análises realizadas sobre o dado em estado bruto?
- Com as representações resumidas de dados como comparar os resultados de duas simulações diferentes atribuindo um valor de similaridade entre elas?
- A partir da representação resumida, como inferir o impacto da mudança de parâmetros para uma nova simulação computacional?

Em busca de respostas a essas perguntas, o objetivo geral do presente trabalho é implantar funcionalidades que possibilitem captura, compactação, armazenamento, processamento, consulta, visualização e predições utilizando dados gerados por simulação computacional de correntes de densidade. O aplicativo deve auxiliar a equipe do laboratório a padronizar as operações com dados, reduzir as operações

computacionais e por meio de aprendizado de máquina diminuir a complexidade das informações armazenadas em bancos de dados conservando as características necessárias para que os depósitos de sedimentos possam ser calculados, analisados, visualizados e preditos.

2.1 Objetivos Específicos

Com base no objetivo geral, os objetivos específicos são:

- Desenvolver um serviço que capture os dados simulados no *cluster* durante o processamento da simulação, permitindo visualizações e análises enquanto a simulação está em execução;
- Aplicar algoritmos de aprendizado profundo para comprimir os dados gerados pela simulação computacional, mantendo características relevantes dos dados originais, simplificando o armazenamento dos dados em base de dados;
- Utilizar a representação compactada dos dados para predição de novos dados de depósito a partir de parâmetros modificados;
- Utilizar a representação compactada dos dados para calcular a similaridade entre duas aplicações;
- Desenvolver uma visualização de dados capaz de simplificar a análise dos dados gerados, trazendo detalhes relevantes e consumindo o mínimo de recursos computacional.

3. REVISÃO BIBLIOGRÁFICA

3.1. Correntes de densidade

Correntes de densidade, também conhecidas por correntes de gravidade, são fenômenos que ocorrem tanto na natureza quanto provocados pelo homem. O mecanismo de geração destas correntes se baseia na diferença entre a massa específica do fluido da corrente e a massa específica do fluido ambiente. Esta diferença de massa específica, é responsável pela geração de forças de empuxo diferentes entre o meio e a corrente, fazendo com que o fluido de maior massa específica se movimente em direção ao de menor massa específica. As diferenças de massa específica podem ser causadas por variações de temperatura, salinidade ou por material particulado em suspensão, esse último conhecido como correntes de turbidez (Simpson, 1999).

Segundo Sosa (2023) as correntes de densidade podem ser observadas em diversos ambientes. Na atmosfera pode-se encontrar como exemplo as frentes frias, que são correntes de densidade de ar denso, seco e relativamente frio, cuja propagação provoca uma queda rápida de temperatura na superfície, seguido de tempestades que se formam quando o ar quente é forçado a subir (Figura 3.1 (a)). As correntes de densidade com concentração de partículas em suspensão, chamadas de correntes de turbidez, incluem avalanches de partículas de neve ou areia transportadas pelo ar em regiões secas (Figura 3.1(b)), e avalanches de lava com gases e sólidos provenientes de fluxos piroclásticos (mistura de gás quente com matéria vulcânica, cinzas e fragmentos de rocha) (Figura 3.1(c)).

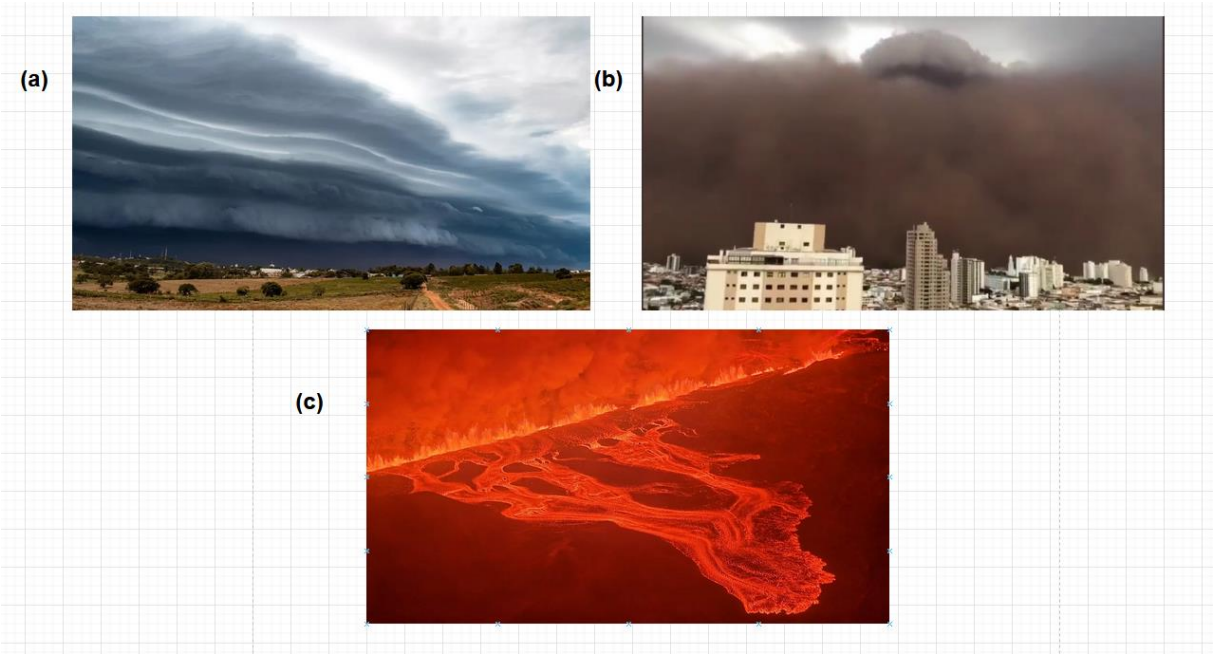


Figura 3.1. Exemplos de correntes de densidade (a) Aproximação de frente fria (METSUL, 2023). (b) Tempestade de areia ocorrida no interior de São Paulo (METSUL, 2021). (c) Imagem de erupção de vulcão na Islândia (METSUL, 2024).

Conforme Mulder e Alexander (2001), as correntes de turbidez podem ser subdivididas de acordo com a diferença de massa específica entre corrente (ρf) e um fluido de referência (ρw). Isso permite a definição de quatro tipos de fluxo exibidos na Figura 3.2, onde:

- corrente hipopicnal ocorre se ρf < ρw;
- corrente homopicnal se ρf = ρw
- corrente hiperpicnal se ρf > ρw e
- corrente mesopícnal se ρf estiver entre a massa específica de duas camadas

Dentro do contexto hídrico, as correntes hipopicnais são importantes perto das embocaduras de muitos rios, onde os sedimentos são dispersos em plumas flutuantes. Já as correntes mesopicnais são particularmente importantes em bacias marinhas fortemente estratificadas, onde os contrastes de massa específica são grandes, e em lagos estratificados termicamente. As correntes de densidade que transportam grandes volumes de sedimento para águas profundas são, provavelmente, em sua maioria corrente hiperpicnais.

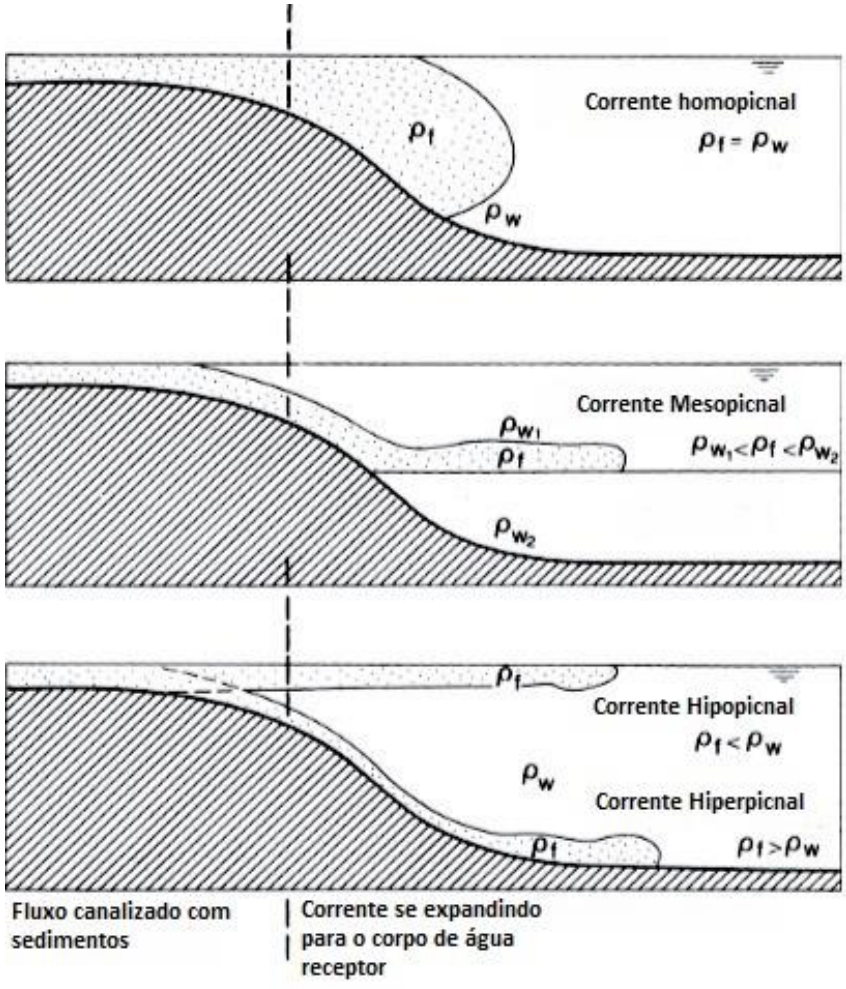


Figura 3.2. Tipos de correntes de densidade. ρf, massa específica da corrente; ρw, massa específica do fluido ambiente. Corrente hipopicnal: ρf < ρw; corrente homopicnal: ρf = ρw; corrente hiperpicnal: ρf > ρw; e corrente mesopicnal: ρw1 < ρf < ρw2. Fonte Mulder e Alexander (2001)

3.1.1. Estudo das correntes de densidade

O estudo das correntes de densidade é útil para várias áreas de conhecimento: geologia, engenharia e meteorologia são algumas das áreas que se utilizam de estudos desse tipo de fenômeno conforme ilustrado por Ruschel (2022) na Figura 3.3. As correntes de densidade, têm impactos ambientais significativos devido à sua capacidade de transportar grandes quantidades de massa a distâncias consideráveis e em altas velocidades, Ruschel (2022).

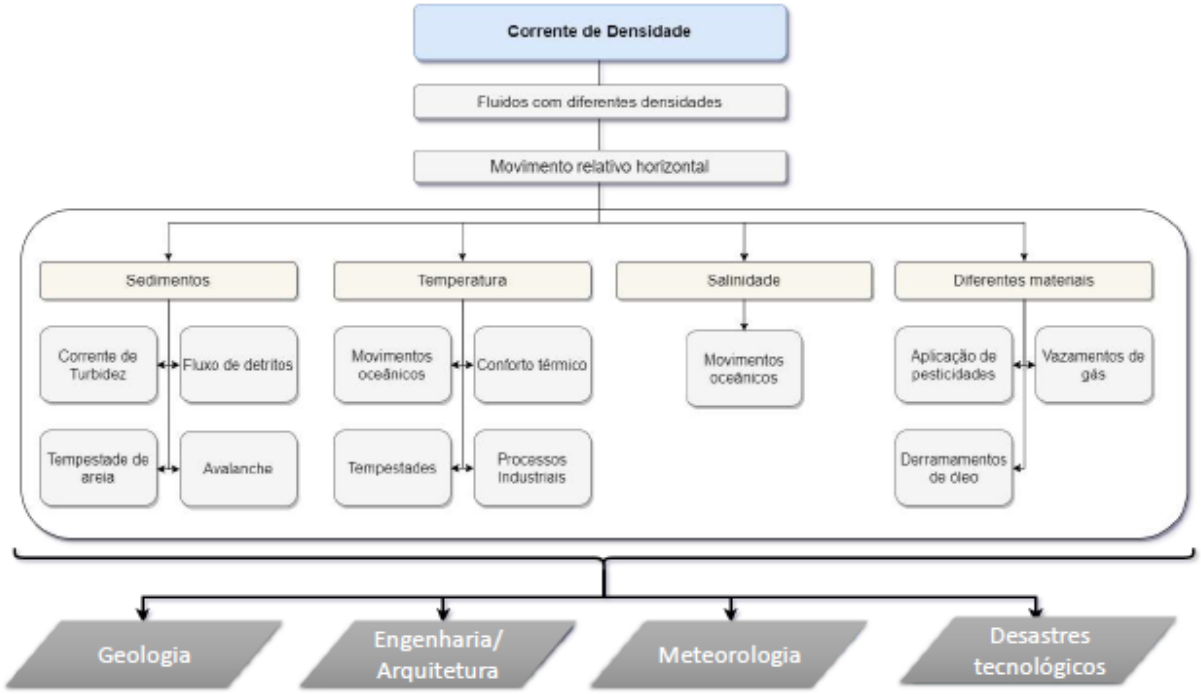


Figura 3.3. Fluxograma apresentando as áreas de conhecimento que se beneficiam do estudo de correntes de densidade (Ruschel, 2022).

Segundo Schuch (2020), com o objetivo de descrever matematicamente as correntes de densidade, utiliza-se as equações incompressíveis de Navier-Stokes sob a aproximação de Boussinesq, adicionalmente à N equações de transporte escalar, que podem representar a concentração de diferentes granulometrias de sedimentos em suspensão, concentração salina ou temperatura, por exemplo.

Dentro da Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), as correntes de densidade são objeto do Laboratório de Simulação de Escoamentos Turbulentos (LaSET), a pesquisa no LaSET é orientada ao estudo de escoamentos transicionais e turbulentos em meios geofísicos, ambientais e industriais. Por meio de simulações computacionais pode-se analisar o comportamento de escoamentos incompressíveis, internos ou externos, com ou sem efeitos gravitacionais, transferência de calor ou de massa e interação fluido-estrutura (LaSET, 2024).

Ruschel (2022), identificou quais são os parâmetros utilizados na modelagem de correntes de densidade definem a sua forma de propagação numa configuração

não confinada lateralmente. Assim, foram avaliados quais parâmetros adimensionais ou características de geometria da entrada que determinam a forma final e o seu alcance. Schuch (2020) propôs uma nova equação para a previsão da profundidade para mergulho, incluindo o papel da velocidade de sedimentação e da declividade do leito do canal. Já Farenzena (2020) formulou um modelo matemático baseado em transformação galileana do sistema de coordenadas, que permite realizar a simulação da cabeça de correntes de densidade em estado estacionário para posteriormente proporcionar uma reflexão mais detalhada a respeito da origem das estruturas de lobos e fendas.

Vianna (2020) avaliou o uso dos algoritmos de visão computacional, utilizados em detecção de cantos e no cálculo do fluxo óptico, para analisar experimentos de correntes de gravidade. Objetivando o acompanhamento automático das estruturas de lobos e fendas nestas correntes, foram desenvolvidas duas abordagens, uma combinando detecção de cantos e fluxo óptico e outra combinando detecção de mínimos locais e fluxo óptico. Sosa (2023), apresentou os resultados das simulações numéricas na configuração *lock-release* para correntes de densidade e correntes de turbidez, com números de Reynolds de ordem de grandeza $O(10^4)$ e $O(10^5)$.

3.2. Fluidodinâmica Computacional

A fluidodinâmica computacional (*Computacional Fluid Dynamic*s - CFD) é uma ferramenta usada para simular numericamente o comportamento de um escoamento, e todas as leis que governam o seu estudo, incluindo a transferência de massa e energia, reações químicas, comportamentos hidráulicos, entre outras aplicações. Essa ferramenta resolve as equações matemáticas do problema de forma específica em uma região de interesse, com condições de contorno predeterminadas nessa região (Oliveira et al, 2016). Segundo Ferziger e Peric (1999), existem 3 abordagens principais para resolver problemas de CFD: as equações médias de Reynolds (*Reynolds-averaged Navier-Stokes equations*, RANS*)*; a simulação de grandes escalas (*Large Eddy Simulation,* LES*)* e a simulação numérica direta (*Direct Numerical Simulation,* DNS). RANS modela as tensões de Reynolds (originadas pela não-linearidade das equações de Navier-Stokes) através de equações diferenciais ou algébricas em função dos valores médios do escoamento considerado (Silvestrini,

2004). LES resolve as maiores escalas de turbulência diretamente, enquanto modela as menores escalas usando modelos de subgrade (SGS). Comumente utilizado em aplicações industriais, como combustão, interações fluido-estrutura e configurações de fluxo complexas. Oferece um bom equilíbrio entre precisão e custo computacional, capturando de forma eficaz as grandes estruturas turbulentas. No entanto, requer uma seleção cuidadosa dos modelos SGS e das malhas computacionais para garantir a precisão (Malik, 2016). Uma variação da abordagem LES é conhecida por *Implicity Large Eddy Simulation* (ILES), no qual utiliza a dissipação numérica inerente a esquemas numéricos de alta ordem para modelar os efeitos das menores escalas de turbulência de forma implícita, sem modelos SGS explícitos. A abordagem ILES pode ser adequado para escoamentos com alto número de Reynolds e geometrias complexas, onde a DNS tradicional pode ser computacionalmente inviável, sendo muitas vezes chamada de DNS de grandes escalas (JIANG, 2016). Essa reduz o custo computacional ao aproveitar as propriedades do esquema numérico, tornando-o viável para simulações em grande escala. Frequentemente validado com dados experimentais e resultados de DNS para garantir a precisão, como descreve Patel (2008).

Por fim, a modelagem em DNS envolve a resolução das equações de Navier-Stokes na sua totalidade, sem o uso de modelos de turbulência, capturando todas as escalas de turbulência, desde as maiores até as menores. É utilizado para estudos detalhados do escoamento de fluidos fornecendo dados de alta fidelidade para validar outros modelos. O custo computacional é extremamente alto devido à necessidade de resolver todas as escalas de movimento, tornando-o impraticável para escoamentos com alto número de Reynolds ou grandes domínios (Baglietto, 2006). Segundo a ESSS (2024), o processo de simulação CFD pode ser dividido em 3 etapas: pré-processamento, onde são definidos os objetivos da simulação, definido o modelo e a malha computacional e configurada a simulação; o processamento, onde são realizados os cálculos da solução; e por fim o pós-processamento, onde são analisados os resultados e feitas as conclusões, o detalhamento do processo de simulação é exibido na Figura 3.4.

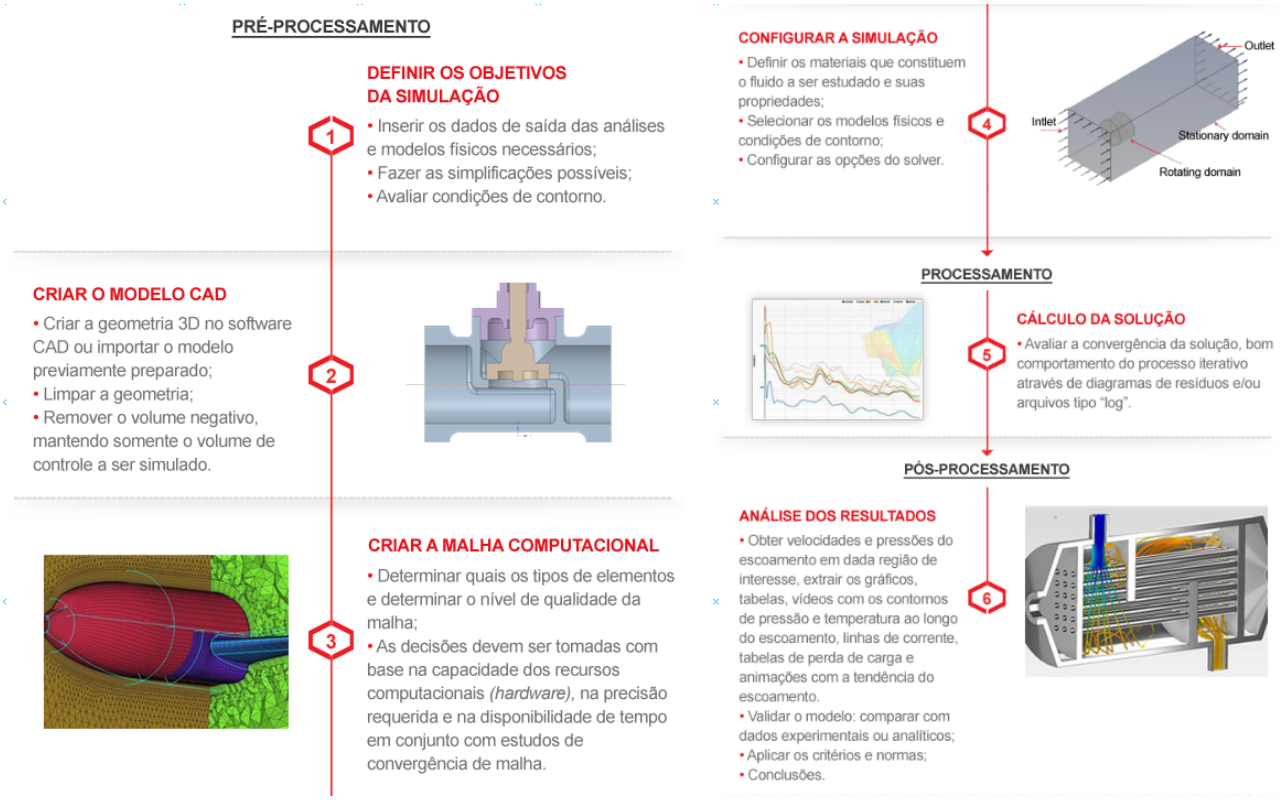


Figura 3.4. Fluxograma apresentado as etapas do processo de simulação computacional CFD (ESSS, 2024)

Por fim, para ilustrar a aplicação de CFD em estudos de correntes de densidade são apresentados na Tabela 3.1 alguns trabalhos realizados pela equipe do LaSET, indicando o autor, o ano de publicação, o título e a abordagem numérica utilizada.

Tabela 3.1 Trabalhos realizados pelo LaSET e suas respectivas abordagens numéricas.

Autor	Ano	Título	Abordagem
Francisco	2014	Modelagem Matemática e Simulação Numérica Direta de Correntes de Gravidade Poli-Dispersas	DNS
Dalpiaz	2014	Simulação Numérica de Transporte e Depósito de Sedimentos em Suspensão em Canal Inclinado	DNS
Schuch	2016	Análise de Pluma Hiperpicnal Poli-Dispersa por Simulação Numérica Direta	DNS
Francisco	2018	Modelagem Matemática e Simulação Numérica de Correntes de Gravidade em uma Configuração Canal-Bacia	DNS
Ruschel	2018	Análise de uma Corrente de Turbidez com Variação das Propriedades Físicas	DNS / ILES
Frantz	2018	Simulações Numéricas de Correntes Gravitacionais com Alto Número de Reynolds	DNS / LES
Farenzena	2020	Simulação Numérica de Correntes de Densidade Hiperpicnais Sob Referencial Móvel	DNS / LES
Schuch	2020	Mergulho de Escoamentos Hiperpicnais em Canal Inclinado por Meio de Simulações Numéricas de Grandes Escalas	LES
Dos Santos	2020	Análise de Correntes de Densidade em Configuração de Bacia com Alimentação Contínua	DNS / ILES
Vianna	2021	Feições Espaço-Temporais em Correntes de Gravidade Usando Métodos de Visão Computacional	DNS / ILES
Ruschel	2022	Simulação Numérica de Correntes de Densidade em Configuração Não-Confinada e Alimentação Contínua	ILES
Sosa	2023	Análise da Turbulência nas Correntes de Densidade em Configuração Lock-Release	DNS / ILES

3.3. Visualização In-Situ

Bauer et al. (2016) definem o processamento *in situ* como aquele onde a visualização dos dados gerados por uma simulação computacional ocorre antes da gravação em disco. Esse fato causa um grande interesse da comunidade de computação de alto desempenho (HPC) devido aos seguintes fatores:

- Economia de custo com armazenamento, pois os dados são visualizados e analisados enquanto são gerados, sem a necessidade de armazená-los primeiramente em um sistema de arquivos.
- Potencial para aumentar a precisão, pois todos os dados da simulação podem ser visualizados expondo comportamentos complexos que poderiam ser

perdidos com uma visualização de dados amostrados ao longo de passos de tempo.

- Capacidade de usar todos os recursos disponíveis, sejam unidades centrais de processamento (CPUs) ou unidades de processamento gráfico (GPUs), no cálculo dos produtos de análise.

Conforme o trabalho de Kress (2016), os *frameworks* de análise e visualização in-situ são divididos em três categorias:

- Acoplamento forte: o código de simulação e visualização é executado no mesmo processo usando os mesmos recursos da simulação. Cactus, CUMULVS, ParaView Catalyst, Strawman, VisIO e VisIt Libsim, são exemplos de *frameworks* fortemente acoplados;
- Acoplamento fraco: a simulação transfere dados pela rede para um conjunto separado de nós de visualização para processamento. EPIC, Freeprocessing, ICARUS, pV3 são *frameworks* fracamente acoplados;
- Acoplamento Híbrido: há componentes de visualização sendo executados no mesmo processo da simulação e parte dos dados é transferida pela rede para nós separados de visualização, a visualização é processada em um conjunto separado de recursos. ADIOS2, Damaris/Viz, EPSN, GLEAN, SCIRun, SENSEI são *frameworks* híbridos.

Com o objetivo de aplicar os conceitos de visualização in-situ a um ambiente de simulação computacional Bartholomew et al. (2023), abordaram o gargalo de I/O na aplicação CFD do código Xcompact3D, facilitando análises *in-situ* definidas pelo usuário. O primeiro passo da abordagem foi adicionar o framework ADIOS2 como um backend opcional à biblioteca 2DECOMP&FFT, sobre a qual o Xcompact3D é construído, proporcionando uma configuração de I/O ajustável em tempo de execução. As configurações de I/O fornecidas pelo ADIOS2 incluem redirecionar o I/O para um programa executado simultaneamente em vez de para o disco. Ao adicionar o ADIOS2 à ferramenta de pós-processamento Py4Incompact3D, os dados podem ser processados antes de serem gravados no disco, limitando o volume de dados gravados, em vez de gravar todo o campo de solução 3D para pós-processamento posterior, o que acarretaria custos adicionais de I/O para carregar os dados.

3.4. Inteligência Artificial e Machine Learning

A aplicação de Inteligência Artificial (IA) traz inúmeros benefícios para as pesquisas científicas, graças a algoritmos cada vez mais avançados de aprendizado de Máquina (Machine Learning, ML), aprendizado profundo (Deep Learning, DL) e IA Generativa (Generative IA, GenAI). Embora o significado desses termos seja distinto, muitas vezes acabam sendo usados como sinônimos. Zuhadar e Lystras (2023) apresentam as diferenças entre esses termos na Figura 3.5.

Técnicas de ML e DL podem ser aplicadas sobre os dados gerados por simulação computacional ou até mesmo buscando a substituição dos métodos tradicionais de simulação. A arquitetura de rede neural chamada de *Physics-Informed Neural Network* (PINN) proposta por Raissi et al. (2019), possibilita que sejam incorporados ao cálculo do erro de previsão da rede neural, cálculos que verificam se a solução prevista pelo modelo é válida em termos físicos.

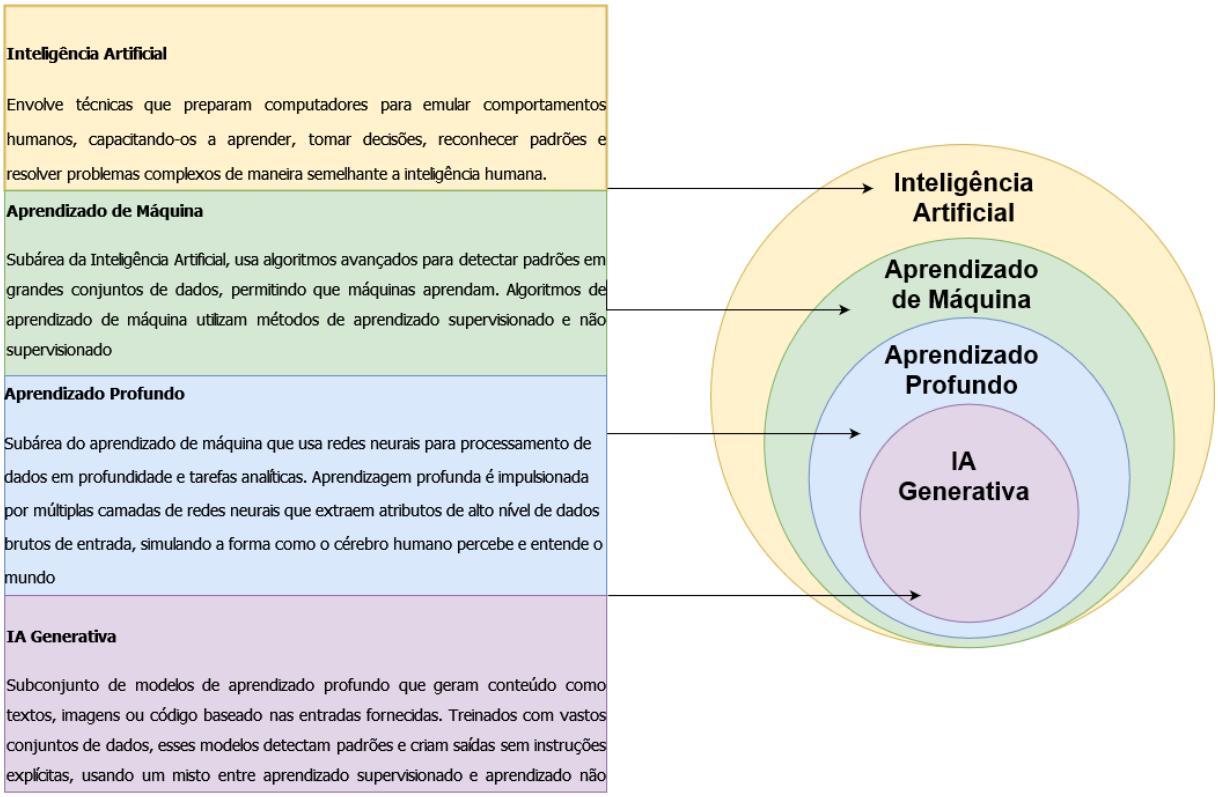


Figura 3.5. Diferenças entre AI, ML, DL e GenAI. Fonte: Zuhadar e Lystras (2023), adaptado pelo autor

O trabalho de Abuside-Armas et al. (2023) apresenta uma Rede Neural Convolucional (*Convolutional Neural Network*, CNN) que prevê as velocidades na direção do fluxo vertical, bem como os campos de pressão a jusante de um cilindro circular para uma série de instantes de tempo, gerando saídas como as exibidas na Figura 3.6.

Uma abordagem de aprendizado profundo para compressão de dados *in situ*, utilizando uma arquitetura de rede neural denominada autoencoder personalizada para fluxos turbulentos tridimensionais de dados é demonstrada por Glaws et al.(2020), essa arquitetura foi comparada ao método de Decomposição em Valores Singulares (*Singular Value Decomposition*, SVD), se mostrando mais eficiente na compressão e na representação dos dados.

Pourbagian e Ashrafizadeh (2022) trazem uma abordagem baseada em uma Rede Adversarial Renerativa (GAN) condicional, com soluções de baixa resolução e baixo custo como entradas e simulações de alta resolução como saídas. Os detalhes, incluindo a estrutura flexível, funções de perda exclusivas e estratégias de tratamento, são discutidos em profundidade, e a metodologia é demonstrada usando simulações numéricas de fluxos incompressíveis. A distinção entre soluções de baixa e alta fidelidade é feita em termos de erros de discretização e modelagem física.

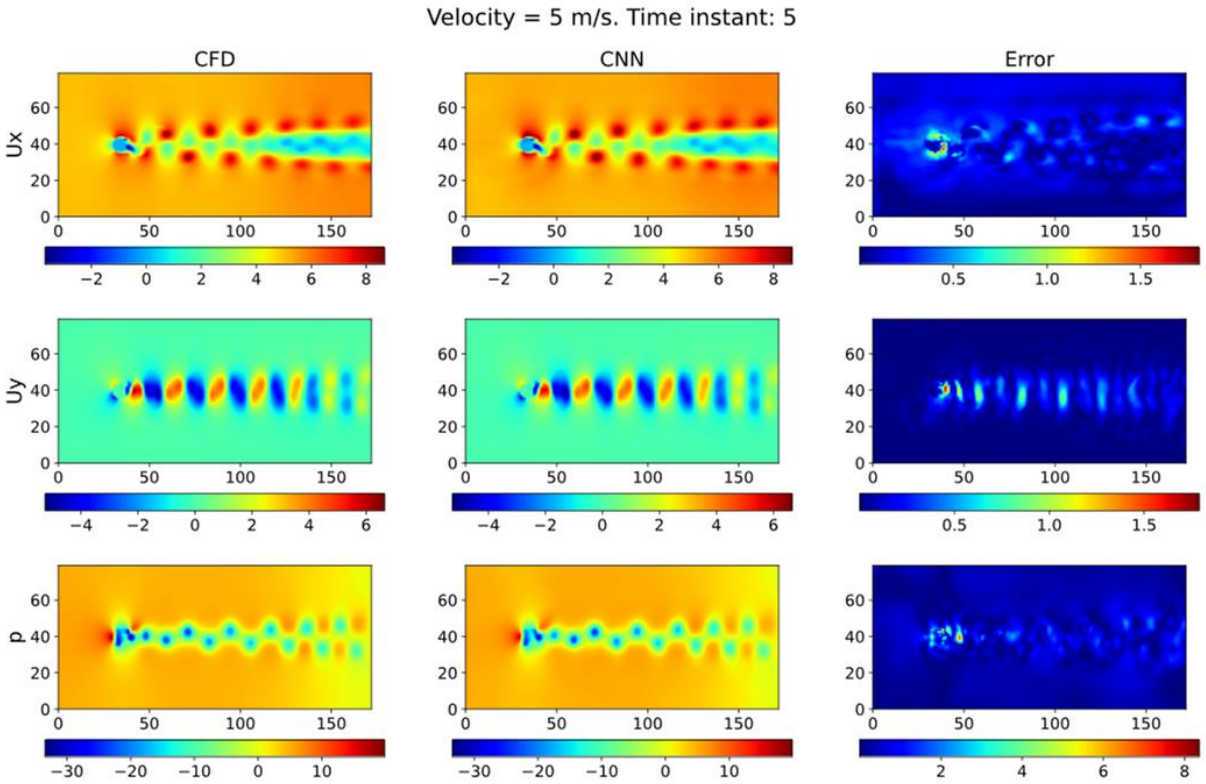


Figura 3.6. Resultados previstos pela CNN. Fonte: retirado/adaptado de Abuside-Armas (2023)

3.4.1. Autoencoders

Os *Autoencoders* (AEs), representados pela Figura 3.7 são redes neurais não supervisionadas projetadas para aprender uma representação comprimida de uma base de dados. O *encoder* é responsável por projetar os dados em um gargalo não linear de baixa dimensionalidade, também chamado de espaço latente, enquanto o *decoder* reconstrói os dados do espaço latente de volta ao espaço de referência e reduz o erro de reconstrução (Hetherington et al. 2024).

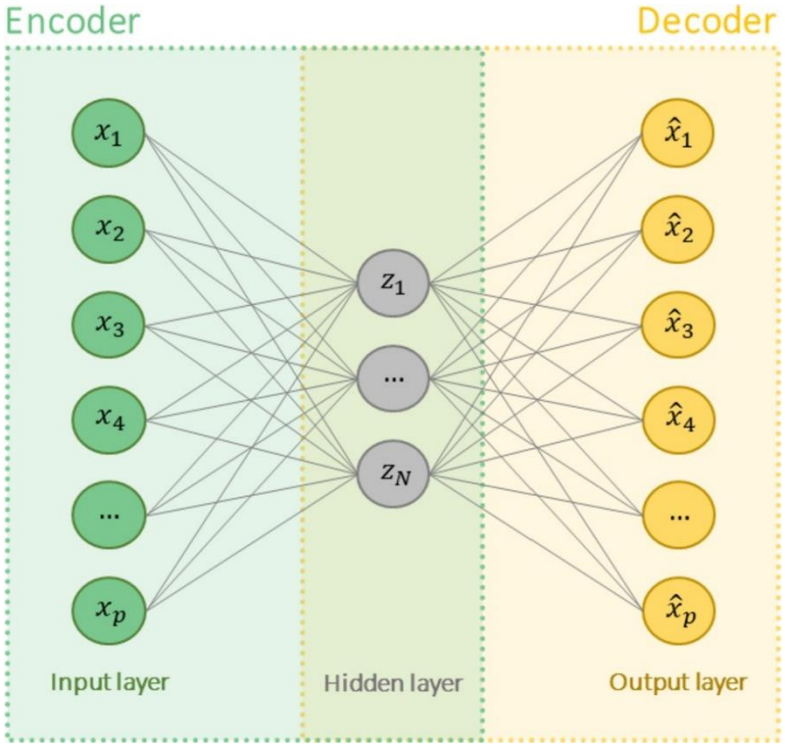


Figura 3.7 Arquitetura de um autoencoder.

3.4.2. Variational Autoencoders

Por conta de limitações que o *autoencoder* possui para reconstruir os dados compactados no espaço latente, uma nova arquitetura chamada *variational autoencoder* (VAE) foi proposta. A arquitetura VAE é uma versão modificada de um autoencoder clássico, projetada para garantir que o espaço latente tenha

propriedades adequadas para a geração de novos dados. O objetivo é que o espaço latente seja regular o suficiente para permitir a geração de novas amostras a partir dele. Assim como em um autoencoder padrão, o VAE é composto por um codificador e um decodificador. O codificador mapeia os dados de entrada para o espaço latente, e o decodificador tenta reconstruir esses dados a partir do espaço latente. A principal diferença é que, em vez de codificar os dados como um único ponto no espaço latente, no VAE eles são codificados como uma distribuição probabilística, geralmente uma distribuição normal (DeepLearningBook, 2024).

3.5. Compressão de Dados

O crescimento acelerado no volume de informações geradas tem obrigado os desenvolvedores de sistemas informatizados a procurar por técnicas de redução dos dados. Desduplicação e compactação de dados sem perdas ou com perdas estão sendo amplamente utilizadas em sistemas de armazenamento para reduzir o custo de armazenamento e I/O. A desduplicação de dados é uma técnica de redução especial de dados que elimina cópias duplicadas de dados. A compressão sem perdas, por sua vez, refere-se a uma classe de compressão de dados algoritmos nos quais os dados originais podem ser exatamente reconstruídos a partir dos dados compactados. Estes incluem os conhecidos ZIP, gzip, FPC e LZ4. As técnicas de compressão sem perdas alcançam em benchmarks gerais, uma proporção de compressão de cerca de 2x, com alguns até 4x. Enquanto desduplicação e compactação sem perdas podem reconstruir perfeitamente os dados originais sem qualquer perda de dados, porém esses algoritmos alcançam taxas de compressão relativamente baixas quando comparados a algoritmos de compactação de dados com perdas. Por outro por outro lado, esquemas de compactação de dados com perdas, às custas de perda de alguma precisão dos dados, geralmente conseguem uma compressão mais alta, proporção de 200x a 500x sendo que em alguns cenários podem até atingir até 10.000x. Em cenários onde os tamanhos dos arquivos podem ser significativamente reduzidos antes que a degradação seja percebida pelos usuários finais, como no caso de dados multimídia, a compactação com perdas geralmente é a melhor escolha (Liu et al., 2023).

3.5.1. Compressão de dados com perdas

Di et al.(2024), propõem na Figura 3.8 uma taxonomia composta por seis tipos de modelos que são o estado da arte para compressão de dados com perdas

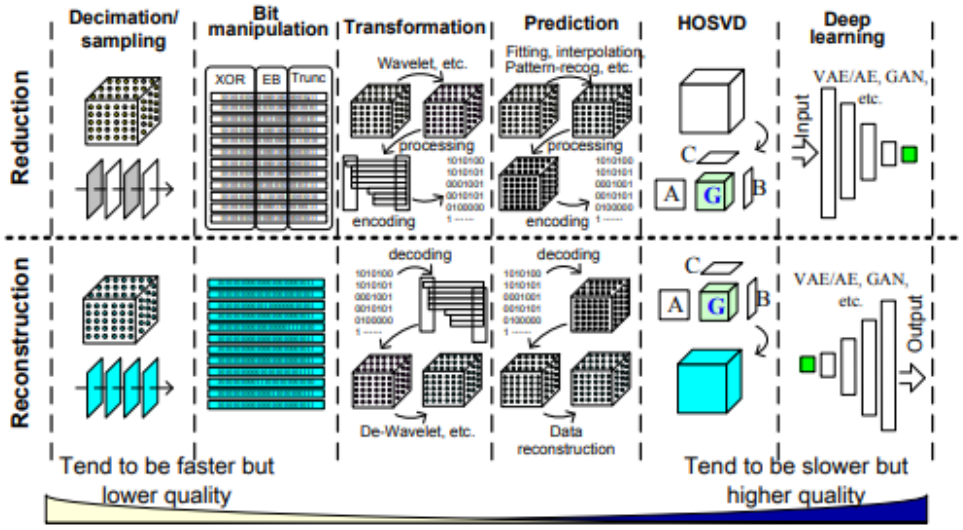


Figura 3.8. Estado da arte de modelos de compressão com perdas. Fonte: Di et al. (2024)

- Compressão baseada em decimação/filtragem: A decimação pode ser dividida em decimação espacial e decimação temporal. A primeira geralmente adota um método de amostragem durante a compressão dos dados e, em seguida, recupera os dados ausentes aplicando uma interpolação sobre os dados amostrados durante a reconstrução. A segunda amostra os instantâneos temporais a cada K passos de tempo durante a simulação ou aquisição de dados e reconstrói os instantâneos ausentes utilizando um método de interpolação.
- Compressão baseada em manipulação de bits: A manipulação de bits é comumente usada para remover bits insignificantes no conjunto de dados, o que pode reduzir o tamanho dos dados. Um exemplo típico é o *Bit Grooming*, que analisa o número de *bits* significativos com base no número especificado de dígitos em base 2 ou 10 e trunca os dados removendo partes insignificantes. Outro exemplo é o SZx, que busca uma velocidade de compressão muito alta, garantindo que cada etapa da compressão seja composta por operações leves, como adição, subtração e operações bit a bit.
- Compressão baseada em transformações: A transformação de dados (como a transformação wavelet e a transformação cosseno) tem sido

amplamente utilizada na comunidade de compressão de dados, pois pode converter efetivamente o domínio de dados original para outro domínio (geralmente chamado de domínio dos coeficientes). O domínio transformado geralmente é mais fácil de comprimir, pois os dados dos coeficientes são, em grande parte, próximos de zero e seus valores frequentemente exibem características espaciais regulares (por exemplo, valores grandes estão concentrados em um núcleo do espaço).

- Compressão baseada em predição: Um modelo de compressão baseado em predição geralmente envolve quatro etapas: predição pontual de dados, quantização, codificação de comprimento variável e codificação por dicionário. A predição de dados é a etapa mais crítica nesses compressores, pois uma maior precisão na predição pode reduzir significativamente o peso das etapas posteriores.
- Compressão baseada em valores singulares de ordem superior (HOSVD): A decomposição HOSVD, como a decomposição Tucker, pode efetivamente decompor os dados (isto é, um tensor) em um conjunto de matrizes e um pequeno tensor central, preservando bem o erro normal L2. Combinando HOSVD com outras técnicas, como codificação de plano de bits, *run-length* ou codificação aritmética, o tamanho dos dados pode ser significativamente reduzido.
- Compressão baseada em *deep learning*: Técnicas de *deep learning* têm sido usadas para melhorar a razão de compressão de dados. Redes neurais do tipo *autoencoders* (AE) *e variational autoencoders (VAE)* são exemplos de redes que permitem a compressão e a reconstrução de dados. Um autoencoder é um tipo de rede neural artificial usada para aprender codificações eficientes de dados de forma não supervisionada, com o objetivo principal de redução de dimensionalidade, o que pode ser aproveitado para reduzir o tamanho dos dados. Por outro lado, o VAE é um modelo generativo (semelhante a uma rede adversarial generativa), no qual uma abordagem variacional é usada para o aprendizado da representação latente, resultando em um estimador específico para o algoritmo de treinamento.

Cada um dos métodos de compressão com perdas apresenta vantagens e desvantagens. A tabela 3.2 apresenta os prós e os contras de cada abordagem segundo trabalho de Di et al. (2024).

Tabela 3.2 Prós e contras de cada uma das abordagens de compressão de dados com perdas. Fonte Di et al. (2024).

Abordagem	**Prós**	**Contras**
Decimação / Filtragem	Desempenho extremamente alto de compressão de dados	A reconstrução dos dados pode ser muito cara, pois é necessário recuperar pontos de dados ausentes usando métodos numéricos, como interpolação
Manipulação de Bits	Compressão de dados muito rápida devido ao uso de operações puramente bit a bit.	Razão de compressão relativamente baixa, pois não explora totalmente as características dos dados ou informações de correlação.
Transformações	Pode levar a uma alta taxa de distorção (isto é, alta compressão com alta qualidade); alto desempenho devido à multiplicação de matrizes (por exemplo, em GPU).	Difícil de controlar a margem de erro, métodos de transformação relativamente fixos.
Predição	Razão de compressão muito alta com boa qualidade; etapa de predição personalizável para se adaptar a diferentes conjuntos de dados; controle de erros fácil e eficaz.	Desempenho inferior (velocidade) devido à codificação de comprimento variável e ao uso de dicionário; difícil de acelerar em GPUs
HOSVD	Razão de compressão extremamente alta, já que aproveita a correlação de longo alcance no conjunto de dados em diferentes dimensões (como tempo e diferentes campos)	Muito caro devido aos seus passos iterativos intrínsecos no controle de erros
Deep Learning	Técnica emergente com grande potencial para obter uma razão de compressão muito alta.	Qualidade de reconstrução de dados inferior, treinamento muito caro, codificação e decodificação relativamente caras

3.5.2 Compressão de dados baseada em deep learning

Técnicas de aprendizado profundo, baseadas em modelos AE podem aplicadas para compressão de dados científicos, afirmam Liu et al. (2023), que realizam um estudo abrangente sobre o uso de AE para comprimir dados científicos do mundo real e ilustra várias descobertas importantes sobre essa técnica para a redução de dados científicos. O estudo mostra que a implementação padrão precisa ser ajustada para alcançar altas taxas de compressão e limites de erro satisfatórios e que os resultados de avaliação indicam que, para a maioria dos conjuntos de dados testados, o autoencoder ajustado supera o algoritmo de compressão SZ em até 4 vezes e o algoritmo de compressão ZFP em até 50 vezes nas taxas de compressão, respectivamente.

Le e Tao (2024), apresentam uma rede neural que não apenas comprime significativamente dados científicos em larga escala, mas também mantém alta a qualidade de reconstrução. O modelo proposto foi testado com dados científicos de referência disponíveis publicamente e aplicado a um conjunto de dados de modelagem climática de alta resolução em grande escala. O modelo atinge uma taxa de compressão de 140 em vários conjuntos de dados de referência, sem comprometer a qualidade da reconstrução. Dados de simulação 2D do Modelo Comunitário de Sistema Terrestre de Alta Resolução (CESM) versão 1.3, também foram comprimidos com uma taxa de compressão de 200, enquanto o erro de reconstrução é insignificante para a análise científica.

3.6 Visualização de dados

Segundo a empresa Tableau (2024), a visualização de dados é a representação gráfica de informações. Usando elementos visuais como gráficos, mapas e diagramas, as ferramentas de visualização de dados oferecem uma maneira acessível de ver e entender tendências, valores atípicos e padrões nos dados. Liu et al (2021) fala que a visualização é um recurso indispensável para explorar informações em dados científicos e é necessária em quase todas as áreas de pesquisa. Em particular, para a pesquisa em CFD, a visualização desempenha um papel crucial na análise dos comportamentos do fluxo e na extração de características e princípios fundamentais dos fluidos. Na Figura 3.9 é exibida uma imagem de um depósito de sedimentos gerada a partir de dados pós-processados de uma simulação

computacional. Enquanto na Figura 3.10 é exibida uma visualização de um perfil de depósito de sedimentos conforme metodologia proposta por Schuch (2020).

Liu et al (2021) desenvolveram um software de código aberto em Python para visualização científica de conjunto de dados de dinâmica de fluidos. O software utiliza renderização do *Visualization Toolkit* (VTK) e possui uma interface interativa construída com PyQT5.

Solmaz e Van Gerven (2023) propõem o Acrossim, um kit de ferramentas versátil projetado para integrar dados de simulação CFD em motores de jogos de forma prática. Este kit modular e baseado em componentes incorpora dados de *solvers* de CFD amplamente utilizados em engenharia. O objetivo do kit de ferramentas é incentivar o desenvolvimento de aplicações multiplataforma, como realidade virtual, utilizando simulação de CFD. Essa abordagem permite o uso fácil de dados de CFD em ambientes digitais amigáveis, oferecendo uma alternativa atraente aos softwares tradicionais de simulação em engenharia.

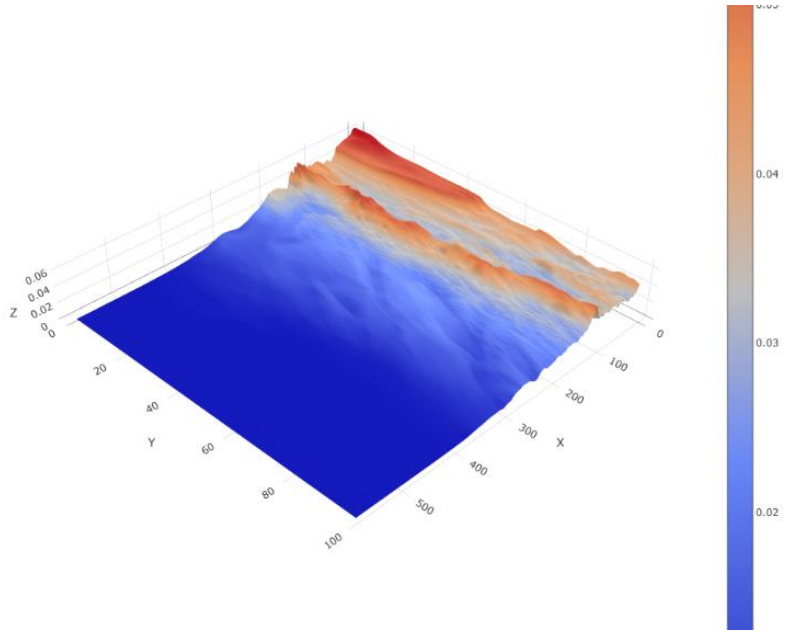


Figura 3.9. Imagem tridimensional de um deposito de sedimentos – Sistema Grande Ideia

Salminen (2023) criou uma aplicação web para visualizar resultados de CFD na web. Essa aplicação permite que o usuário faça o upload dos resultados de

simulação para a nuvem, que podem então ser facilmente acessados por outras pessoas.

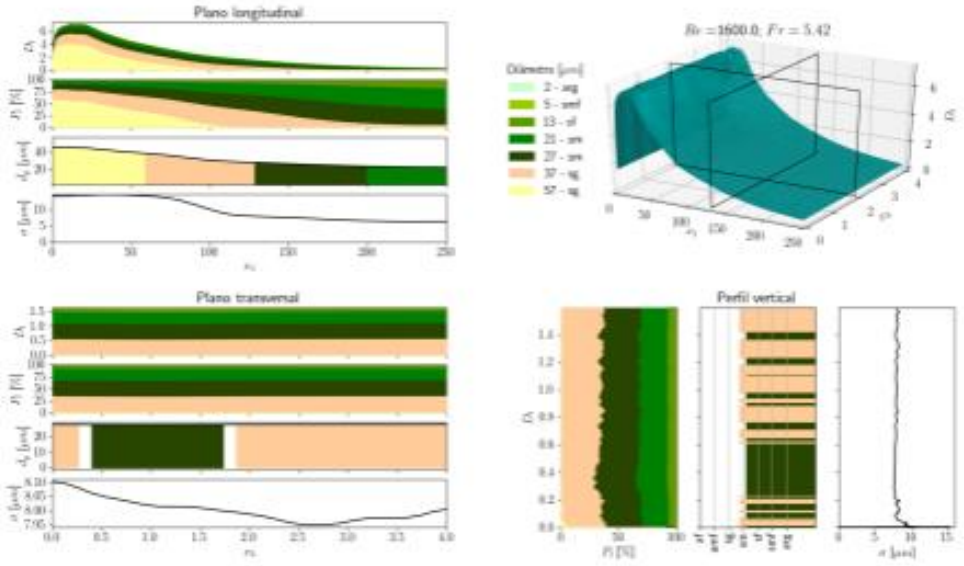


Figura 3.10. Perfil de depósito de sedimentos segundo metodologia desenvolvida por Schuch (2020)

4. MATERIAIS E MÉTODOS

Esta seção apresenta as propostas para a resolução dos objetivos propostos na seção de objetivos do trabalho. É uma breve explanação sobre as ferramentas que serão utilizadas no processo de desenvolvimento das funcionalidades do aplicativo, sobre os conjuntos de dados utilizados. Por fim são apresentadas métricas para avaliação dos resultados obtidos.

4.1 Ferramentas computacionais

A linguagem de programação utilizada para escrita dos códigos-fontes das funcionalidades será a linguagem Python, por ser de simples utilização e apresentar uma ampla variedade de bibliotecas computacionais voltadas a desenvolvimento de aplicações de inteligência artificial e de ciência de dados. As bibliotecas que se destacam para essa finalidade: Numpy (2024), a qual fornece um objeto de array multidimensional, vários objetos derivados (como arrays mascarados e matrizes), além de uma variedade de rotinas para operações rápidas em arrays, incluindo operações matemáticas, lógicas, manipulação de forma, ordenação, seleção, entrada/saída, transformadas de Fourier discretas, álgebra linear básica, operações estatísticas básicas, simulação aleatória e dentre outras funções; o Scipy (2024), é uma coleção de algoritmos matemáticos e funções convenientes construídas como a extensão biblioteca NumPy do Python. O Scipy adiciona um poder significativo à sessão interativa do Python, fornecendo ao usuário comandos e classes de alto nível para manipulação e visualização de dados; O Matplotlib (2024), é uma biblioteca abrangente para criar visualizações estáticas, animadas e interativas em Python. Ainda falando em bibliotecas para desenvolvimento o Plotly e Dash que permitem a visualização de dados e a criação de dashboards em um navegador de internet e a biblioteca Pyvista que permite a plotagem 3D e análise de malhas por meio de uma interface simplificada para o Visualization Toolkit (VTK)

O Pytorch (2024), é uma biblioteca de tensores otimizada para aprendizado profundo utilizando GPUs e CPUs. A MPI4Py (2024), é uma biblioteca que permite que aplicações Python aproveitem múltiplos processadores em estações de trabalho, clusters e supercomputadores.

O código computacional da simulação é gerado através do framework Incompact3d, um conjunto de solucionadores de fluidos de alta ordem por diferenças finitas, dedicados ao estudo de fluxos turbulentos. Incompact3d é o solucionador principal do Xcompact3d e foi projetado para estudar escoamentos incompressíveis. Para o pós-processamento, além das bibliotecas Python mencionadas, serão utilizados dos pacotes python focados nesse tipo de tarefa, o Xcompact3d-toolbox, projetado para lidar com o pré e pós-processamento do solucionador de alta ordem de Navier-Stokes XCompact3d. Ele visa ajudar usuários e desenvolvedores de código a construir soluções específicas para casos, oferecendo um conjunto de ferramentas e processos automatizados (Schuch, 2024).

Py4Incompact3D é uma biblioteca para pós-processamento de dados produzidos por simulações do Xcompact3D. O objetivo deste projeto é facilitar o pós-processamento automatizado de simulações do Xcompact3D, fornecendo, inicialmente duas classes: a classe mesh a qual armazena os dados do domínio da simulação e a classe case a qual armazena as informações do caso, condições de contorno, campos e outros. Com esses blocos de construção, outras ferramentas de pós-processamento complexas podem ser desenvolvidas, como calculadoras de derivadas para calcular a vorticidade e o critério Q a partir do campo de velocidade.

Para implantação de visualizações *in-situ* as ferramentas analisadas serão: ADIOS2 e Paraview Catalyst. ADIOS2 é a segunda geração do Sistema Adaptável de Entrada e Saída (Adaptable Input Output System), inicialmente introduzido por Lofstead et al. A adaptabilidade é atribuída à capacidade das bibliotecas ADIOS de usar diferentes técnicas de entrada e saída (*input/output,* I/O), formatos de arquivo e até mesmo de transportes, que são conFigurados em tempo de execução usando um arquivo de conFiguração XML, mantendo-se otimizadas para operação em escalas que vão de laptops a supercomputadores. Embora ADIOS2 seja baseado em C++, ele oferece suporte a outras linguagens, como C, Fortran, Python e Matlab. O ADIOS2 é mais do que uma simples biblioteca de I/O baseada em arquivos, como o Parallel

NetCDF e o HDF5, pois permite análise *in-situ,* capacidades de transporte em redes de longa distância (*Wide Area Network*), bem como operações de dados em linha (por exemplo, compressão com perdas/sem perdas e transformações de dados), (Laufer e Fredj, 2022).

O ADIOS2 permite o uso de linguagem Python e suas principais ferramentas de análise, como Scipy, Matplotlib e Jupyter Notebook através de sua API. Arquivos BP do ADIOS podem agora ser integrados diretamente ao Visualization Toolkit (VTK) e Paraview (ADIOS, 2024).

As principais características consideradas na escolha da biblioteca são:

- Suporte a fluxos de trabalho in-situ.
- Integração com uma ampla seleção de softwares de HPC.
- Funciona em uma grande variedade de tipos de rede.
- Integração com Paraview Catalyst

Paraview Catalyst (Paraview, 2024) é um framework *in situ* criado como parte do ParaView há alguns anos. Também conhecido como coprocessamento ou análise simultânea, o fluxo de trabalho in situ permite realizar análises e visualizações durante a simulação. Como apenas o resultado da análise é gravado no disco, as operações de I/O são consideravelmente reduzidas em comparação a:

- gravar a saída completa da simulação
- ler os dados completos em uma ferramenta de análise
- gravar a saída da análise.

4.2 Arquitetura da aplicação

A aplicação será dividida em módulos que procuram atender cada uma das necessidades das equipes envolvidas no projeto, a Figura 4.1 ilustra o a arquitetura da aplicação sendo que as setas indicam o fluxo de dados / requisições dentro da aplicação. O usuário poderá executar as seguintes atividades:

- Importação / pós-processamento dos dados de uma simulação concluída;

- Compactação e armazenamento dos dados pós-processados em banco de dados;
- Consulta de similaridade entre duas simulações;
- Visualização de mapa e detalhamento do depósito de sedimentos;
- Visualização em tempo real dos dados uma simulação em execução;
- Pré-visualização dos dados uma simulação, essa atividade tem por objetivo permitir que o pesquisador estime o impacto que mudanças de parâmetros causam em uma simulação sem precisar rodar todo o processo de simulação.

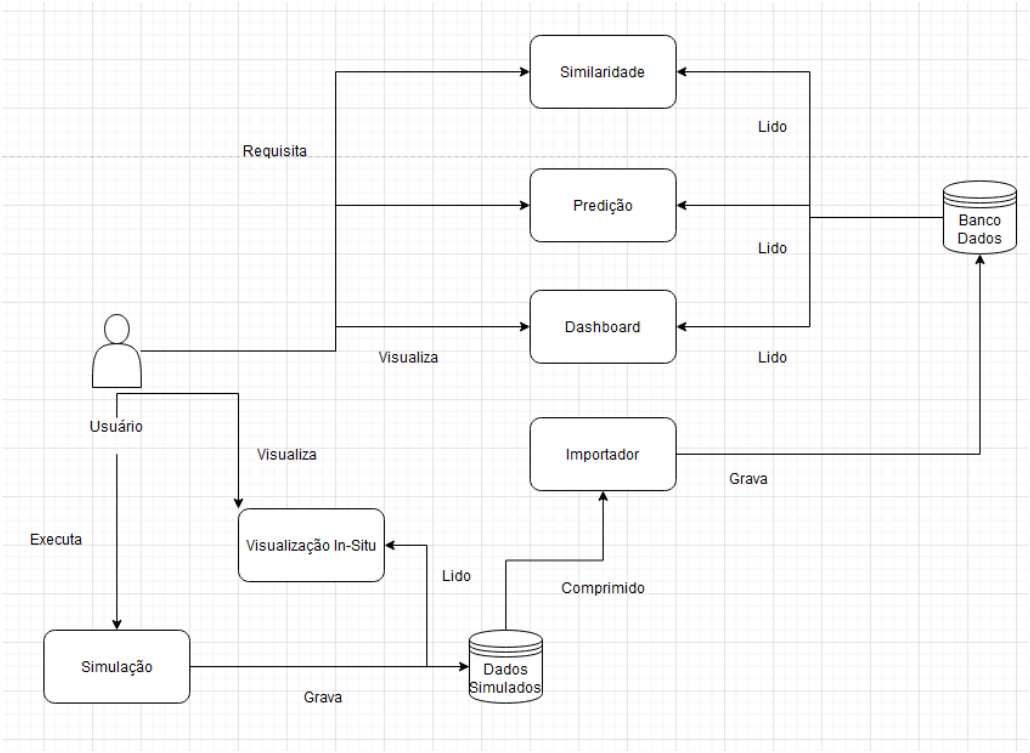


Figura 0.1. Arquitetura proposta.

4.2.1 Importador de dados da simulação

Esse módulo irá executar os códigos de pós-processamento dentro do cluster de processamento da simulação, gerando uma representação resumida dos dados resultante do processo de simulação computacional de correntes de densidade e gravar os dados pós-processados e os parâmetros de simulação dentro de um banco

de dados *PostgreSQL*. Para tal tarefa serão utilizados os dados de uma simulação computacional, ou seja, os parâmetros de execução e o arquivo de saída contendo os dados da corrente de densidade ao longo do tempo de simulação A execução dentro do cluster de simulação busca a redução de dados antes de trafegá-los pela rede de computadores, dessa forma haverá somente a necessidade de mover pela rede os dados pós-processados para serem armazenados no banco de dados, reduzindo o tempo e custo envolvido na movimentação de dados. Os códigos de pós-processamento serão desenvolvidos em linguagem *Python* que é uma linguagem de programação amplamente difundida na comunidade de desenvolvimento de sistemas, sendo famosa pela simplicidade de uso e pelo grande número de bibliotecas de análise de dados. O importador irá trabalhar com dados gerados pelo software de simulação *Xcompact3d* utilizado pelo time de pesquisadores do laboratório. O importador será capaz de ler os arquivos com os parâmetros de configuração da simulação e executar os cálculos de pós-processamento informados pelo usuário, padronizando a etapa de pós-processamento de dados. Outro fator importante é que o importador por armazenar os dados em um banco de dados irá garantir a centralização dos dados do laboratório em um repositório único, facilitando o compartilhamento dos dados científicos gerados pelos pesquisadores.

4.2.2 Visualização de dados pós-processados

O módulo de visualização de dados irá permitir ao usuário a inspeção e a análise visual dos resultados do pós-processamento dos dados. Para esse fim serão avaliadas as bibliotecas Plotly e Dash que permitem a visualização de dados em um navegador de internet e biblioteca Pyvista que permite a plotagem 3D e análise de malhas por meio de uma interface simplificada para o *Visualization Toolkit* (VTK). Essas bibliotecas permitem visualizações com bastante detalhamento.

O foco inicial das visualizações será nos mapas de depósito de sedimentos e caracterização de espessura e distribuição dos grãos ao longo do depósito. O objetivo é criar uma visualização interativa onde o usuário consiga escolher pontos específicos do mapa de depósito exibido em 3 dimensões e exibir informações referentes a espessura e a distribuição do depósito no ponto selecionado em visualizações auxiliares.

Esse tipo de visualização procura resolver algumas deficiências encontradas

nas visualizações existentes. Atualmente para exibir as informações de pontos específicos do depósito de sedimentos é necessário uma rotina que gere uma imagem *Picture Network Graphics* (png), para cada coordenada do depósito, o que em uma simulação computacional de correntes de densidade obriga a criação de milhares de imagens, pois temos milhares de coordenadas dentro do depósito. Essa necessidade de processamento de imagens, acarreta um grande custo computacional de processamento e armazenamento das imagens, outro fator é que como as imagens são armazenadas em disco existe um tempo de espera entre o usuário escolher o ponto a ser visualizado e o gráfico ser exibido na tela, o que prejudica a experiência do usuário. Com o uso do dashboard será possível a integração entre a visualização em 3 dimensões do depósito e a visualização de testemunhos que hoje estão separadas em telas diferentes do sistema.

4.2.3 Acesso aos dados de simulação em tempo real

O módulo de acesso em aos dados de simulação em tempo real, permitirá que os pesquisadores façam análises e visualizações preliminares enquanto a simulação ainda está em execução, sem a necessidade de mover dados para outras máquinas para processamento, este tipo de visualização é chamado *in-situ*. Para permitir a captura de dados da simulação em tempo real gerados pelo framework de simulação Xcompact3d será utilizada a biblioteca ADIOS2, conforme trabalho de Bartolomew et al (2023), a biblioteca ADIOS2 permite que um arquivo possa ter acessos de leitura e escrita de forma simultânea. Sendo assim o código computacional da simulação deverá gravar a saída dos cálculos nesse arquivo gerenciado pela biblioteca ADIOS2. Enquanto isso o software Paraview o usuário irá criar uma conexão com o ADIOS2 para poder capturar os dados e gerar as visualizações dos dados capturados em tempo real Figura 4.2.

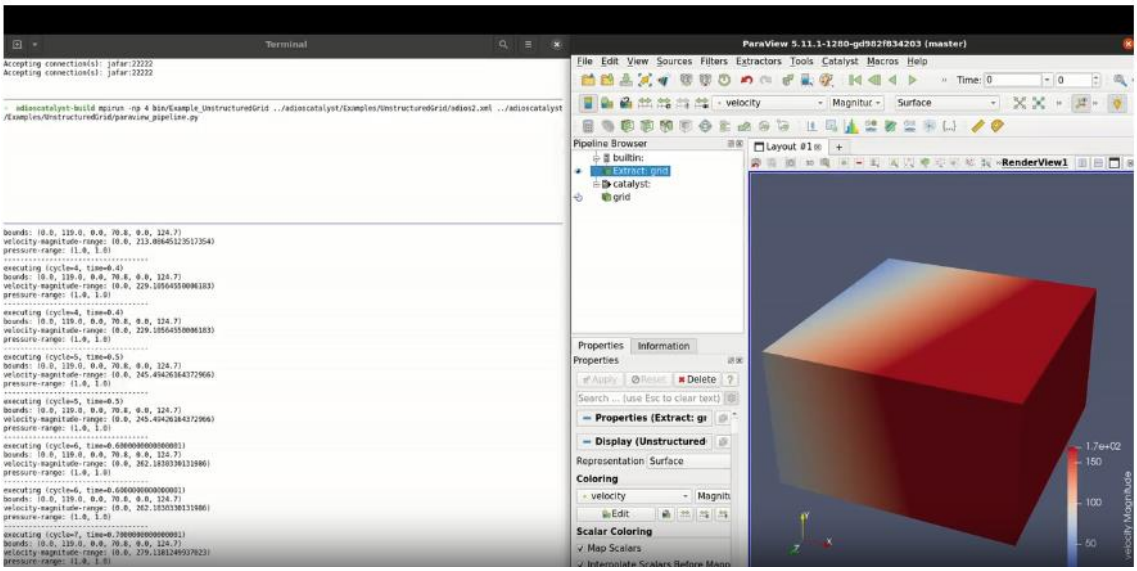


Figura 0.2. Visualização de dados in-situ. A esquerda, dados gerados pelo código computacional e a direita visualização dos dados capturados visualizados no software Paraview. Fonte Givord et al. (2024)

4.2.4 Compressão dos dados gerados pela simulação

A funcionalidade de compressão de arquivos irá possibilitar uma maior redução dos dados gerados pelas rotinas de processamento da simulação, possibilitando o armazenamento mais eficiente dos dados. O algoritmo de compressão de dados irá reduzir a representação atual dos dados que estão dispostos em matrizes multidimensionais para vetores procurando preservar as características originais dos dados antes da compressão. Para realizar a tarefa de compressão será aplicado um tipo de redes neurais profunda chamada Autoencoder, mais especificamente sua variante conhecida com Autoencoder Variacional. Essa arquitetura que tem seu funcionamento baseado em um codificador e um decodificador, o codificador realiza a compressão do dado com perda e o decodificador realiza a reconstrução dos dados em formato muito próximo ao dado bruto original. Para avaliação dos resultados da compressão, será feita uma comparação com algoritmo de compressão com perdas SZ, conforme trabalho de Liu et al (2023), nesse caso será verificado o percentual de compressão obtido em relação ao tamanho dos dados originais. Para avaliação da qualidade da reconstrução dos dados será executada a rotina de pós-processamento sobre o dado reconstruído após compactação e verificado o quão similar é o resultado deste pós-processamento em relação ao pós-processamento realizado diretamente

sobre o dado bruto antes da compactação.

4.2.5 Predição de resultados de uma nova simulação

Essa funcionalidade tem por objetivo reconstruir os dados de uma simulação computacional executada anteriormente, essa reconstrução tem por base parâmetros de uma segunda simulação computacional que também foi executada anteriormente, como essa reconstrução é possível avaliar as modificações que uma alteração de parâmetros causa sobre os dados de uma simulação de correntes de densidade. O modulo servirá como uma pré-visualização do resultado de uma nova simulação, pois irá estimar o resultado sem que seja necessário executar a uma nova simulação no cluster, economizando tempo do pesquisador e recursos computacionais. Com essa pré-visualização o usuário poderá testar diferentes combinações de parâmetros e executar a simulação apenas daquelas combinações de parâmetros que se mostrarem mais promissoras.

A rede neural chamada por Disentangled Variational Autoencoder (DisVAE) no trabalho de Wang et al (2023), foi escolhida pois com seu uso é possível separar diferentes fatores de variação latentes em um conjunto de dados, permitindo que características específicas sejam isoladas de outras. No trabalho de Wang et al (2023) o autoencoder proposto foi utilizado para separar as características do rosto de uma pessoa das expressões faciais do mesmo rosto. A rede é composta por dois codificadores e um decodificador. O primeiro codificador chamado IdEncoder é responsável por detectar características específicas do rosto de uma pessoa enquanto o segundo codificador chamado ExpEncoder detecta a expressão facial em um rosto, seja ela de alegria, tristeza, espanto, entre outras. O decodificador chamado Decoder é responsável por realizar a fusão entre o rosto de uma pessoa que foi codificado no primeiro codificador com a expressão facial codificada pelo segundo codificador, gerando ao final uma imagem com o rosto da pessoa cuja imagem foi passada pelo primeiro codificador com a expressão facial detectada pelo segundo codificador Na Figura 4.3 é demonstrado a arquitetura da rede.

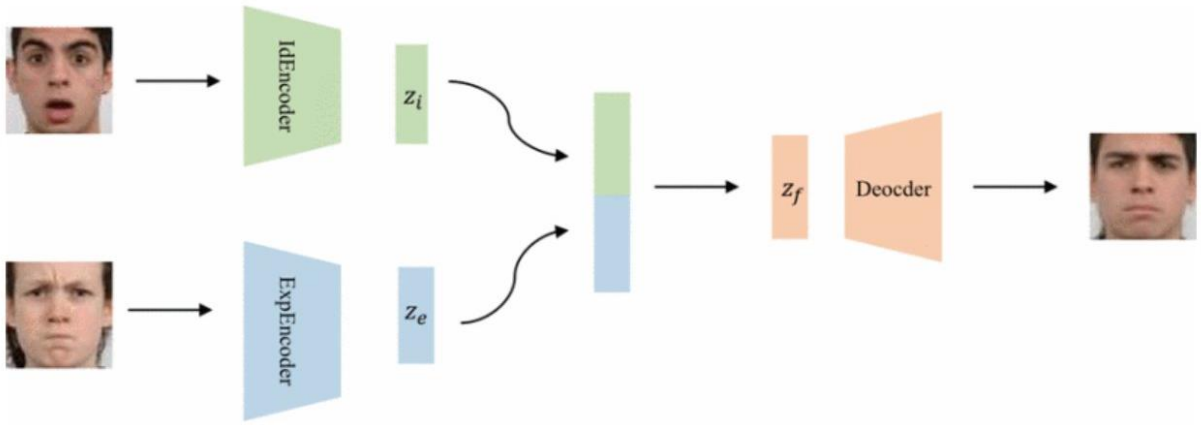


Figura 0.3. DisVAE, proposto por Wang et al. (2023).

Para o treinamento da rede neural proposta por Wang et al (2023), foram usados conjuntos de dados de imagens do rosto de pessoas, para cada uma das pessoas existentes no conjunto de dados existem imagens com expressões faciais diferentes. O IdEncoder foi treinado calculando o erro de reconstrução da imagem alvo não contra a imagem alvo, mas uma imagem da mesma pessoa com uma expressão facial diferente, assim esse codificador procura reconstruir somente as características comuns do rosto da pessoa. Enquanto isso, ExpEncoder foi treinado a partir dos pesos congelados do IdEncoder e o erro de reconstrução de ExpEncoder é calculado comparando a imagem original com a imagem reconstruída somado ao erro de uma segunda rede neural treinada para classificar qual classe de expressão facial está retratada na imagem. Por fim o erro do Decoder o erro de da imagem reconstruída comparada a imagem original, o erro do IdEncoder, o erro de ExpEncoder e um erro calculado entre a imagem reconstruída pelo Decoder, comparada a essa imagem reconstruída de pelo Decoder passada novamente por IdEncoder e por ExpEncoder.

No contexto de dados gerados por simulações de correntes de densidade o autoencoder que será treinado, terá dois codificadores. O primeiro codificador irá receber dados de um passo de tempo específico de uma simulação computacional, enquanto o segundo receberá um arquivo de deposição de uma segunda simulação de um mesmo passo de tempo que o primeiro codificador. O decodificador terá o trabalho de aplicar as características detectadas pelo segundo encoder aos dados da simulação que foi recebida pelo primeiro codificador. Aplicando dessa forma os efeitos dos parâmetros de uma simulação em uma outra simulação.

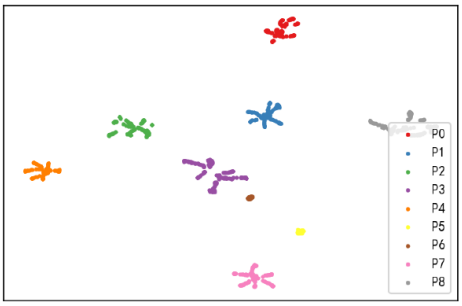
Para o treinamento da rede serão necessários arquivos de deposição

resultantes de várias simulações computacionais, essas simulações deverão ter arquivos gerados para os mesmos passos de tempo. O primeiro codificador será treinado comparando o erro de reconstrução entre os dados reconstruídos de uma simulação em uma posição de tempo comparada aos dados originais de uma mesma simulação, porém em um passo de tempo diferente sorteado de forma aleatória. O segundo codificador irá calcular o erro de reconstrução utilizando o dado reconstruído de deposição em um passo de tempo, comparado ao dado original da mesma simulação em um mesmo passo de tempo, esse erro é somado ao erro de classificação, nesse caso o classificador irá classificar de qual passo tempo é determinado dado de deposição. O decodificador segue a mesma lógica do treinamento aplicado por Wang et al (2023) em DisVAE.

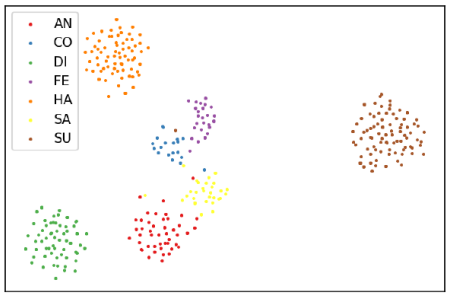
4.2.6 Verificação de similaridade entre duas visualizações

Esse módulo irá permitir que o usuário calcule a similaridade entre dois depósitos de sedimentos, para poder demonstrar o quão parecidos são os resultados de duas simulações. Essa funcionalidade é útil pois com a verificação de similaridades, o pesquisador poderá verificar se dois depósitos são semelhantes mesmo que a simulação tenha sido realizada com bibliotecas / softwares de simulação diferentes.

Para desenvolver a tarefa será utilizada rede neural as representações resumidas dos arquivos das simulações que serão geradas pela funcionalidade do item 4.2.4. Uma característica muito interessante do espaço latente do autoencoder demonstrada por Wang et al (2023), na Figura 4.4 é que elementos de mesma classe costumam ficar localizadas de forma próxima dentro do autoencoder.



(a)



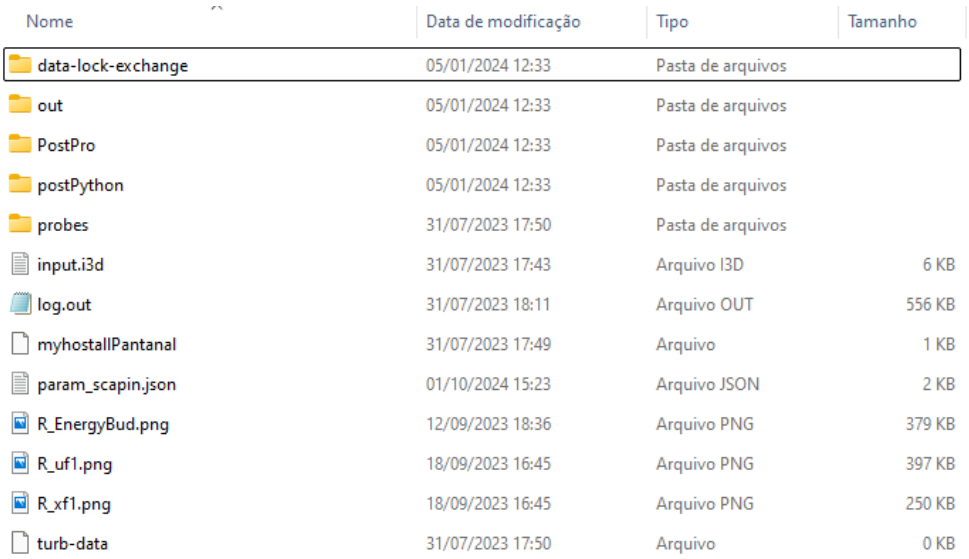
(b)

Figura 4.4. Visualização de características no conjunto de imagens. (a) mostra a visualização das características de identidade. (b) mostra a visualização das características de expressão (Wang et al., 2023)

Como é possível visualizar na Figura 4.4 as representações comprimidas das imagens de uma determinada pessoa ficam agrupadas em localizações próximas dentro do espaço latente esse mesmo fenômeno ocorre quando a análise é feita dividindo as imagens por expressões faciais. No contexto dos resultados de simulações de correntes de densidade será verificado se os dados de deposição para cada uma das posições de tempo da simulação estão agrupados de forma próxima dentro do espaço latente, ou seja, se os dados referentes a uma única simulação serão encontrados próximos dentro do espaço latente. Outra verificação que será feita é quais das simulações se encontram próximas dentro do espaço latente procurando determinar dessa forma simulações com resultados semelhantes, mesmo que seus parâmetros de entrada sejam completamente diferentes.

4.3 Dados Preliminares

Para os estudos iniciais sobre as funcionalidades de ciência de dados e inteligência artificial a serem para análise de dados de correntes de densidade gerados por simulação computacional, foi disponibilizado pelo LaSET um conjunto de dados de 25 simulações com uma camada de sedimentos, totalizando 510 *gigabytes* (GB) de dados. Os dados de cada uma das simulações estão armazenados um diretório separado, a estrutura de diretórios de uma simulação é apresentada na Figura 4.5.



Nome	Data de modificação	Tipo	Tamanho
data-lock-exchange	05/01/2024 12:33	Pasta de arquivos	
out	05/01/2024 12:33	Pasta de arquivos	
PostPro	05/01/2024 12:33	Pasta de arquivos	
postPython	05/01/2024 12:33	Pasta de arquivos	
probes	31/07/2023 17:50	Pasta de arquivos	
input.i3d	31/07/2023 17:43	Arquivo I3D	6 KB
log.out	31/07/2023 18:11	Arquivo OUT	556 KB
myhostallPantanal	31/07/2023 17:49	Arquivo	1 KB
param_scapin.json	01/10/2024 15:23	Arquivo JSON	2 KB
R_EnergyBud.png	12/09/2023 18:36	Arquivo PNG	379 KB
R_uf1.png	18/09/2023 16:45	Arquivo PNG	397 KB
R_xf1.png	18/09/2023 16:45	Arquivo PNG	250 KB
turb-data	31/07/2023 17:50	Arquivo	0 KB

Figura 0.5. Estrutura de arquivos de uma simulação computacional de correntes de densidade.

Dentro dessa estrutura se destacam o diretório data-lock-exchange, dentro desse diretório estão os arquivos com extensão dep com a posição do depósito de sedimentos coletada em cada um dos passos de tempo definidos na simulação. Os arquivos dep são arquivos binários que podem ser lidos através de funções da biblioteca *Numpy* da linguagem *Python,* esses arquivos serão processados pelo importador na etapa de pós-processamento e essa versão processada será armazenada no banco de dados centralizado do laboratório. Além desses usos a funcionalidade de compressão de dados será aplicada sobre esses arquivos dep, de forma que seja possível criar uma estrutura resumida que possa ser guardada no banco de dados. As funcionalidades de verificação de similaridade entre duas simulações e previsão de resultados de uma simulação também usarão os arquivos de extensão dep.

O arquivo input.i3d é o arquivo de parâmetros de uma simulação, as informações desse arquivo serão importadas para o banco de dados, dessa forma todos os parâmetros da simulação estarão armazenados dentro da base de dados do laboratório. Os parâmetros da simulação são importantes pois são usados nos cálculos de pós-processamento que serão realizados pelo importador.

Por fim o arquivo log.out, é o arquivo de saída da simulação, nesse arquivo temos a informação da precisão numérica dos valores que foi utilizada na simulação.

Essa precisão pode ser Single ou tipo de dado float do python ou Double equivalente ao tipo de dado double da linguagem Python. Essa informação é necessária para que o processamento realizado pelo importador não apresente erro. Para os modelos de aprendizado empregados nesse trabalho todos os dados estarão no formato double.

RESULTADOS PRÉVIOS

Nessa seção serão descritos os resultados prévios obtidos durante os primeiros testes realizados com as aplicações de Ciência de Dados e Inteligência Artificial aplicadas a análise de dados de correntes de densidade gerados por simulação computacional. Cada uma das aplicações será descrita em uma seção a seguir.

5.1 Importador de dados da simulação

Os primeiros testes com o importador foram realizados utilizando a seguinte sequência de ações:

- Importador recebe os parâmetros da simulação a partir da leitura arquivo input.i3d;
- Importador determina a precisão numérica utilizada na simulação lendo o arquivo log.out, foi necessário realizar esse tratamento para evitar erros referentes a incompatibilidade de tipos de dados durante execução do importador;
- Importador utiliza os arquivos de extensão dep do diretório data-lock-exchange calculando a altura e a proporção para cada uma das posições do fundo do depósito.

Ao final da execução são gerados dois arquivos, o arquivo proporcao.npy e o arquivo sedimentacao.npy que possuem os valores de proporção e altura do depósito respectivamente. Esses arquivos são gravados em disco e um ponteiro para esses arquivos é criado em uma tabela do banco dados, nesse momento a solução foi desenhada dessa forma devido ao fato de os dados de proporção e sedimentação apresentarem alta dimensionalidade, ou seja, são matrizes com muitas linhas e dimensionalidade variável, ou seja, cada simulação gera matrizes de tamanhos diferentes. Esse comportamento torna o armazenamento dos dados diretamente em

tabelas do banco de dados Postgresql muito complexo e caro de ser realizado. Por fim os parâmetros da simulação são gravados em uma tabela de parâmetros de simulação, dessa forma não é necessário que o usuário precise informar manualmente as informações de simulação, dessa forma pe garantido que os dados das simulações estarão centralizados em uma base de dados única.

Outro tratamento que o simulador realiza é que se a simulação já foi importada uma vez os arquivos proprocao.npy e sedimentacao.npy não serão gerados novamente, o importador buscará os arquivos que foram salvos anteriormente e ususará esses dados no cálculo das grandezas.

O tamanho dos arquivos com os dados calculados de todas as simulações disponibilizadas é de aproximadamente 6GB, aproximadamente 1% do tamanho original dos dados que é de 510GB.

Mesmo com esse resultado será realizada a avaliação de uma estratégia de compressão complementar, objetivando diminuir a complexidade do dado, de forma a possibilitar o seu armazenamento em tabelas do banco de dados. Outra avaliação que será realizada é a execução do código Python diretamente no cluster de simulação de forma a evitar a necessidade de trafegar os arquivos da simulação pela rede, dessa forma todo o pós-processamento é executado no cluster e somente o dado reduzido é trafegado pela rede para ser guardado no banco de dados.

5.2 Visualização do depósito de sedimentos

Foi desenvolvido um protótipo da visualização utilizando a biblioteca *Python* chamada *Dash,* combinada com uma visualização em 3 dimensões gerada com a biblioteca *Plotly* dessa forma o usuário pode clicar em algum ponto da visualização em 3 dimensões e exibir o corte nos eixos atualizando os gráficos que estão no dashboard. O protótipo foi implantado em uma interface Web, sendo necessário para sua execução apenas um navegador de Internet, o carregamento da simulação leva poucos segundos e a atualização dashboard a partir da interação do usuário com o gráfico se mostrou eficiente. Uma imagem do protótipo implantado é exibida na Figura 5.1.

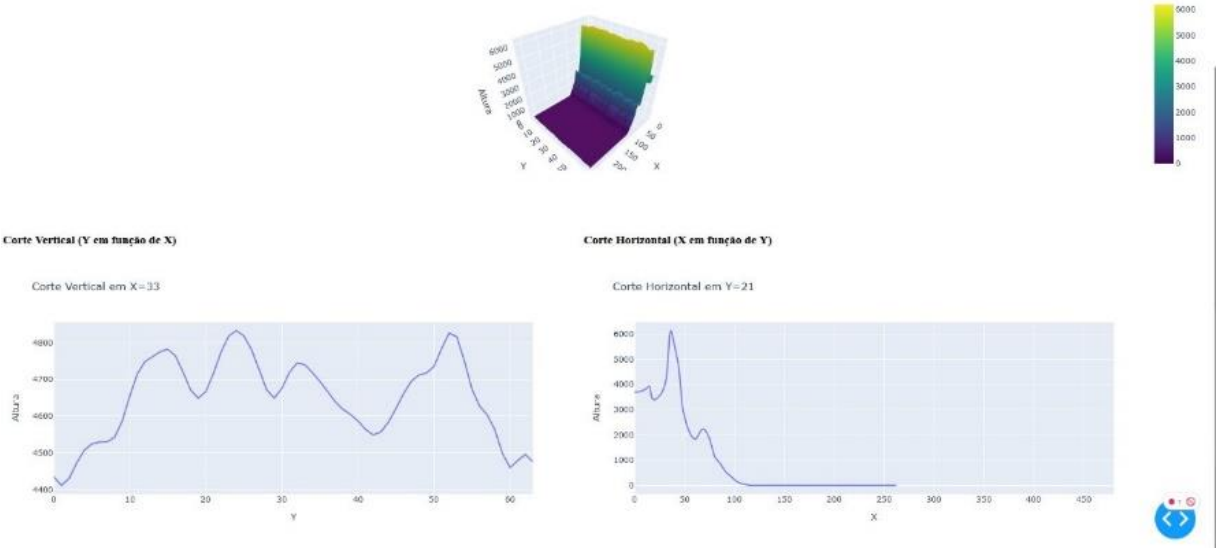


Figura 5.1. Protótipo da visualização de dados de depósito.

Um segundo protótipo foi desenvolvido com o objetivo de o usuário ver com mais nível de detalhamento a topografia superior do depósito de sedimentos, nela usuário interage com a visualização cuja câmera está colocada de forma a dar uma visão aérea do depósito, a partir dessa visão aérea é possível ampliar algum ponto específico de forma a visualizar os contornos da topologia de uma forma mais detalhada. Uma imagem do protótipo é exibida na Figura 5.2.

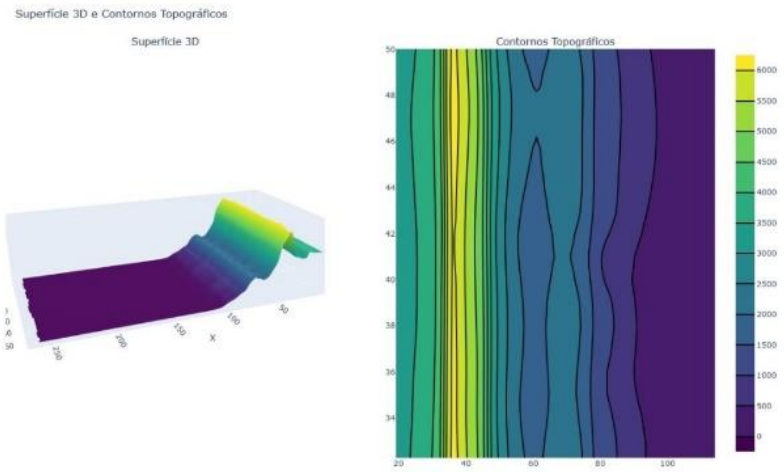


Figura 5.2. Protótipo da visualização de dados de depósito.

Uma melhoria que está em desenvolvimento é a substituição da biblioteca do *Plotly*, pela biblioteca Pyvista que é baseada no VTK de forma que a visualização em 3 dimensões mais parecida com mapas geológicos, um exemplo de visualização em três dimensões com Pyvista é exibida na Figura 5.3:

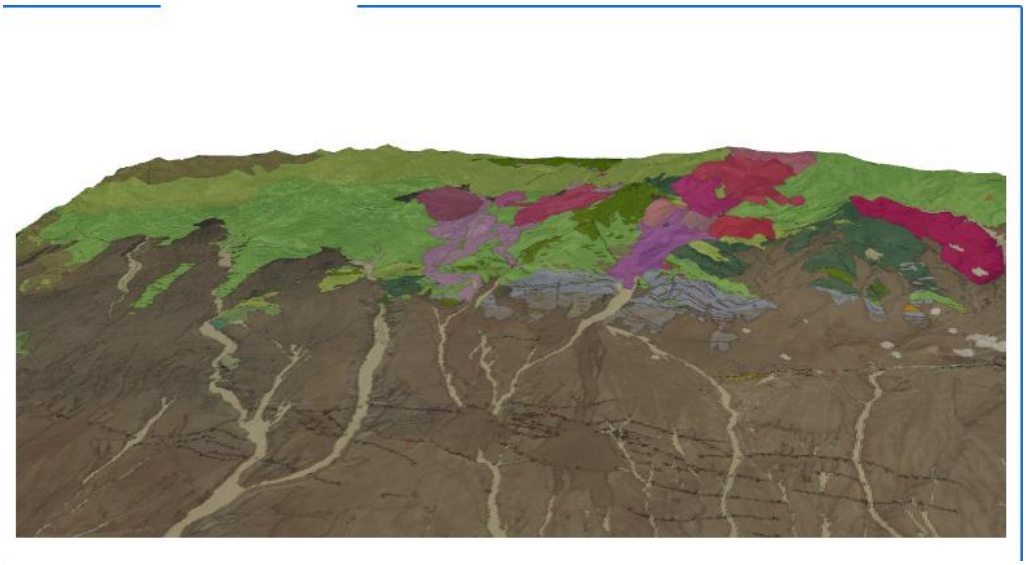


Figura 5.3. Mapa geológico construído com a biblioteca Pyvista. Fonte Pyvista

5.3 Visualização de dados em tempo real

Buscando iniciar os testes de visualização *In-Situ,* foi instalada a biblioteca ADIOS2 em máquina virtual Linux e executados os tutoriais disponibilizados na documentação do ADIOS2. Os códigos de simulação testados são baseados na linguagem Fortran, a mesma linguagem de programação do XCompact3d. Nos testes foi possível criar um processo de gravação os dados em um arquivo e ou segundo processo de leitura no mesmo arquivo. Ambos os processos executaram simultaneamente e um laço de repetição totalizando 1 milhão de execuções de escrita e 1 milhão de execuções de leitura. Os próximos passos que serão testados são:

- Conexão do software Paraview ao arquivo onde a simulação está sendo gravada;
- Criação de uma visualização, para teste da visualização de dados em tempo real;
- Avaliação das alterações necessárias no código da simulação gerada no XCompact3d para que os dados sejam passados pelo arquivo gerenciado pelo ADIOS2;
- Avaliação das conFigurações necessárias para a implantação do ADIOS2 em ambiente de cluster.

5.4 Compressão dos dados gerados pela simulação

A arquitetura de rede neural selecionada para os primeiros testes de treinamento do modelo de compressão de dados da simulação foi a arquitetura autoencoder e sua variante chamada autoencoder variacional, essa arquitetura precisa de camadas que irão realizar a compressão do dado passado como entrada da rede, conhecidas como *encoder*, e de camadas que irão realizar o processo de reconstrução do dado original conhecidas como *decoder*, a entrada do *decoder* é o dado que foi comprimido pelo *encoder*. Para implementação do *encoder* e do *decoder* foram testadas camadas completamente conectadas. O cálculo da perda na reconstrução do dado da simulação foi realizado através do uso das *funções erro médio quadrático (mean squared error, MSE)* e da função de entropia *binária cruzada (binary cross entropy).*

Os dados utilizados no treinamento dos modelos de compressão são os dados finais de deposição gerados com o importador dos dados de uma simulação, essa medida foi adotada para reduzir a amostra dos dados e utilizar menos recursos computacionais nessa etapa inicial do trabalho. Os tratamentos realizados na etapa de preparação de dados foram:

- substituição de dados faltantes pelo menor valor encontrado na respectiva matriz
- por estarem em diferentes escalas de grandeza os dados foram reescalados para ficarem entre valores 0 e 1, mantendo as suas proporções originais.

- como as matrizes geradas em cada uma das diferentes simulações possuem dimensões variadas, as matrizes foram ajustadas para ficarem todas com a maior dimensão no eixo x e a maior dimensão no eixo z, os dados faltantes nesse caso foram completados com 0.

A saída do encoder é dada por um vetor de tamanho 10. Foi avaliada a possibilidade de compressão da matriz inteira e de dividir a matriz em partes menores, realizando a compressão individual de cada uma das partes da matriz, essa divisão tem como objetivo diminuir o tamanho do modelo gerado reduzindo a demanda computacional para o processamento do modelo. No modelo treinado a rede implementada possui 4 camadas de tamanho fixo, sendo que a entrada é determinada pelo tamanho das dimensões da matriz e saída conforme dito anteriormente é um vetor de tamanho 10.

A reconstrução dos dados realizada com o autoencoder variacional se mostrou de melhor qualidade conforme imagens apresentadas nas Figuras 5.4 e 5.5

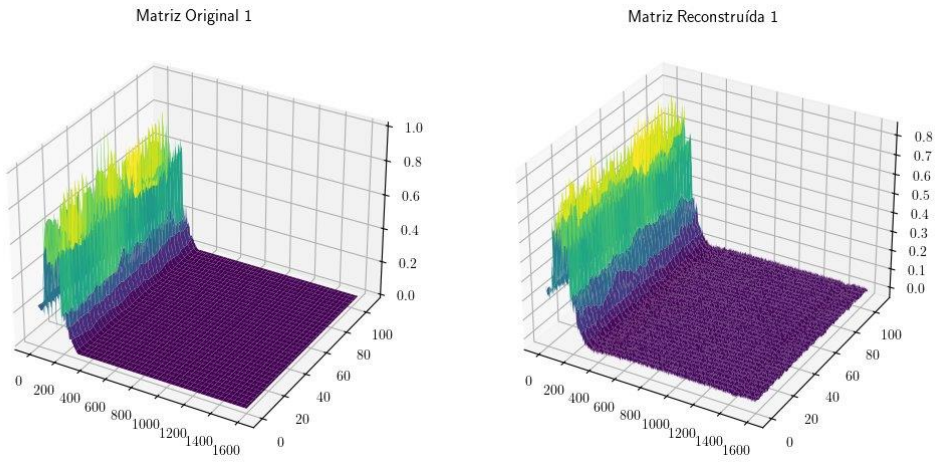


Figura 0.4. Imagem do mapa original a esquerda e imagem reconstruída com o autoencoder a direita.

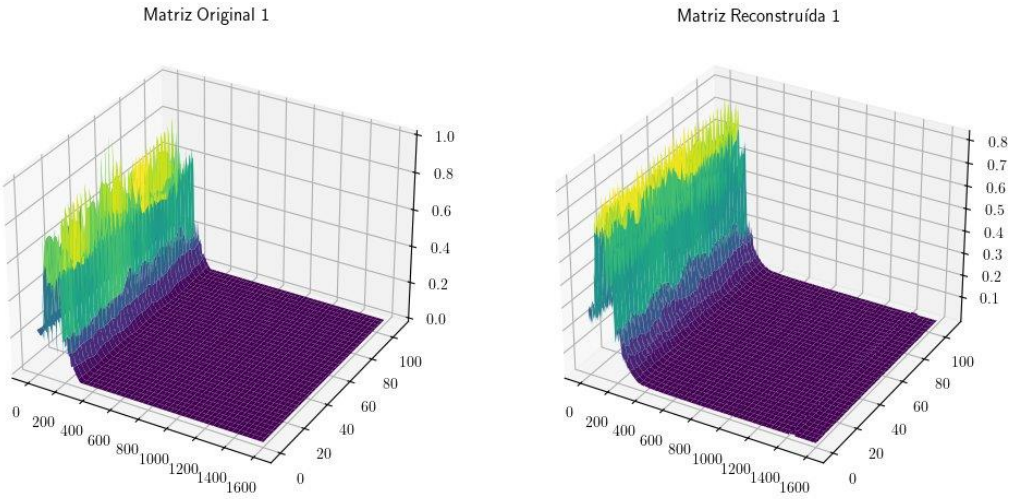


Figura 5.5. Imagem do mapa original a esquerda e imagem reconstruída com o autoencoder variacional a direita.

Nessa abordagem foram encontradas algumas limitações:

- Tamanho da matriz padronizada ficou limitado a 1601x108, acima disso houve falta de memória para o treinamento do modelo, porém a maior matriz padronizada existente no conjunto de dados de simulações é de 2881 x 512.
- Tamanho do modelo treinado é de 1354.70 megabytes com 177562742 de parâmetros treináveis.

Para contornar esses limitadores foi implementada a abordagem de divisão da matriz em pedaços, nesse caso as matrizes foram todas padronizadas com a dimensão 2881 x 512 e cada pedaço possui a dimensão de 1 x 512. Para permitir a reconstrução, cada pedaço é identificado com a identificação da matriz a qual pertence e a sua posição dentro da matriz. Os erros em cada época de treinamento estão mostrados Figura 5.6 e a matriz original e a matriz reconstruída aparecem na Figura 5.7.

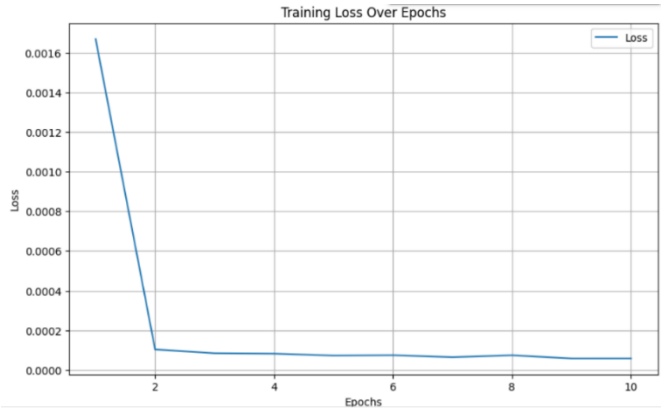


Figura 5.6. Cálculo da perda do treinamento do autoencoder treinado com pedaços das matrizes

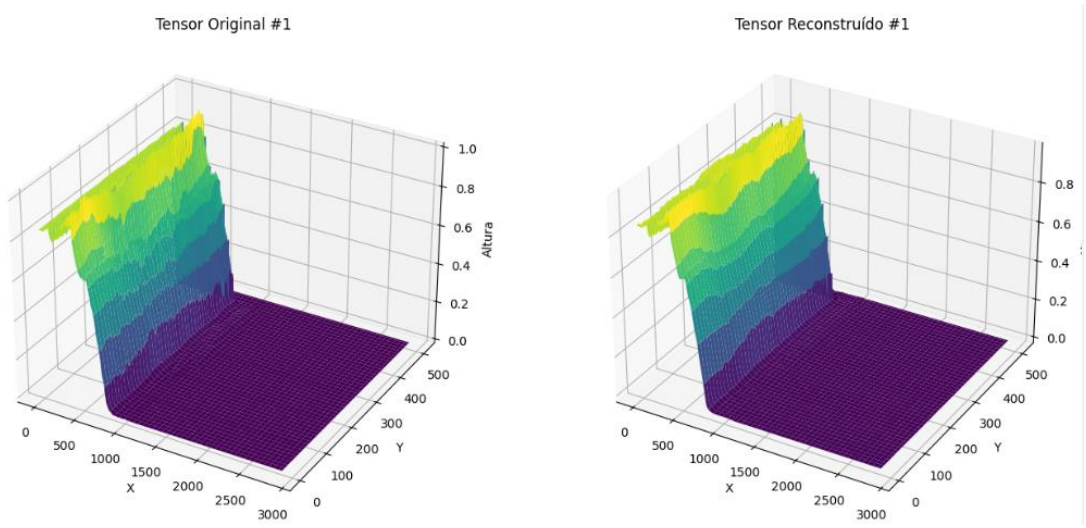


Figura 5.7. Imagem do mapa original a esquerda e imagem reconstruída com o autoencoder a direita

A abordagem de treinamento utilizando partes se mostra bastante eficiente na reconstrução e gerou um modelo muito menor em relação ao modelo treinado para compactar a matriz inteira para esse caso o modelo treinado possui 856842 parâmetros treináveis e ocupa 6.56 megabytes na memória, mesmo treinando matrizes menores. A desvantagem dessa abordagem é que perdemos capacidade de compressão, pois no primeiro modelo treinado havia a redução da matriz inteira a um vetor de 10 posições agora cada coluna da matriz é reduzida a um vetor de tamanho 10.

Os próximos resultados a serem buscados nessa funcionalidade são redução da saída do encoder a um vetor de duas posições buscando uma maior compactação das partes das matrizes. Aumento dos dados usando todos os passos de tempo de uma simulação, dessa forma podendo implementar etapas de treino e teste do modelo. Avaliar o tamanho dos dados comprimidos com o autoencoder comparando com o algoritmo de compressão com perdas SZ. Avaliar a reconstrução do autoencoder aplicando os cálculos de pós-processamento dos dados reconstruídos e comparando os resultados alcançados com os resultados obtidos com o pós-processamento realizado sobre os dados originais. Uso do conceito de Physics Informed Neural Network (PINN), para melhorar os resultados de reconstrução do autoencoder.

5.5 Verificação de similaridade entre duas visualizações

A similaridade entre simulações foi verificada da seguinte forma: foram selecionadas 3 simulações, 2 com os mesmos parâmetros e 1 simulação com parametrização diferente conforme Figura 5.8.

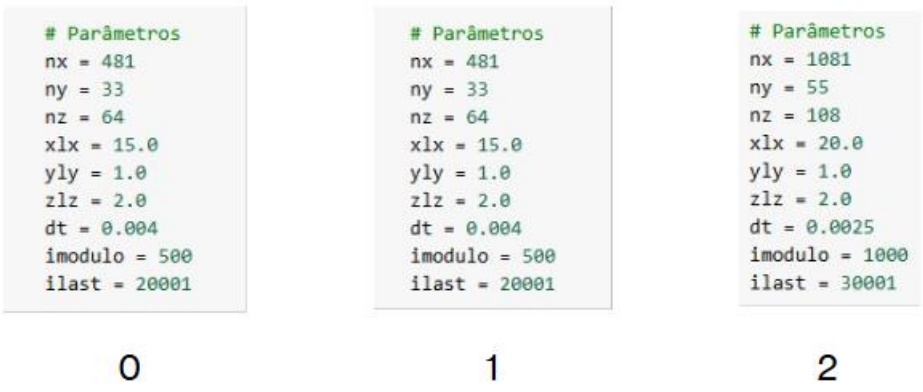


Figura 5.8. Parâmetros das simulações utilizadas no teste.

Para essa verificação foram usados todos os passos de tempo da simulação. Sendo que todos os passos de tempo foram divididos em pedaços, os pedaços foram

identificados de forma que é possível saber a simulação, o passo de tempo e a sua posição dentro da matriz. Esses pedaços foram passados pelo encoder para que fosse obtida a representação resumida de cada pedaço, nesse caso cada parte foi resumida a um vetor de tamanho 2 para possibilitar a plotagem de um gráfico bidimensional que represente a distribuição dos vetores no espaço latente conforme mostrado na Figura 5.9.

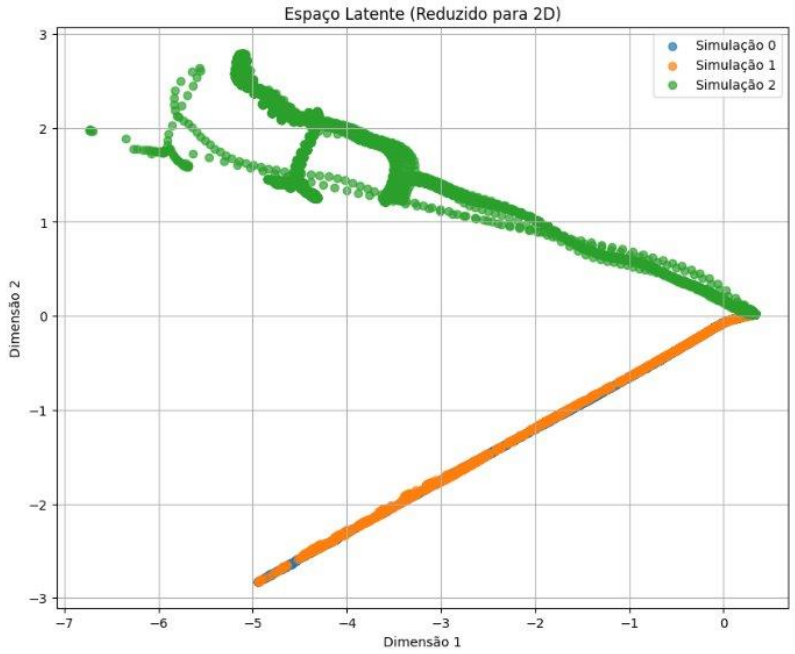


Figura 5.9. Distribuição dos vetores no espaço latente.

Com base no gráfico é possível observar que os vetores gerados pelas simulações 0 e 1 que possuem parametrização semelhante ocupam posições próximas, muitas vezes sobrepostas no espaço latente, enquanto a simulação 2 que possui outros parâmetros fica em posições completamente separadas dentro do espaço latente. A única exceção ocorre no ponto 0 do espaço, esse comportamento ocorre por conta da rotina que realiza a padronização do tamanho das matrizes, esse tratamento de dado completa os espaços com dados faltantes com 0. Como próximos passos serão adicionadas mais simulações para ver se o comportamento se repete. Será testada uma métrica como similaridade do cosseno para buscar quantificar o quão semelhante são duas simulações. O mesmo teste será feito com o depósito pós-processado de forma que seja possível verificar se a similaridade pode ser identificada com menos arquivos, buscando utilizar menos recurso computacional no cálculo.

Algumas outras possibilidades interessantes que a análise do espaço latente proporciona é verificar se é possível encontrar posições específicas no espaço latente que são ocupadas de acordo com características específicas da simulação, por exemplo será possível observar que vetores oriundos de simulações com o mesmo número de Reynolds ocupam posições próximas dentro do espaço latente? Outra questão que pode ser interessante é buscar por arquivos de simulações gerados outra ferramenta de simulação, fazer a sua codificação usando o encoder treinado e tentar verificar se essa simulação tem seus vetores ocupando posições semelhantes aos vetores gerados por simulações feitas no XCompact3d de forma a buscar estabelecer uma similaridade entre simulações geradas com bibliotecas e até mesmo parametrizações distintas.

5.6 Predição de resultados de uma nova simulação

Nessa etapa do trabalho apenas foi desenvolvido o código de treino conforme o artigo de Wang et al. (2023), para teste da implementação foi utilizado o conjunto de dados MNIST, que é um conjunto de dados de dígitos escritos a mão (Lecun et al., 2024). Na Figura 5.10 é exibido o resultado da inferencia feita pela rede, no caso do exemplo a imagem do dígito 8 é passada pelo idEncoder e a imagem do dígito 9 é passada pelo expEncoder, como resultado da rede é gerado um dígito modificado para um dígito 9. Como próximos passos o modelo será treinado com os dados de simulação para verificação da viabilidade de aplicação desse modelo na predição do resultado de uma simulação.

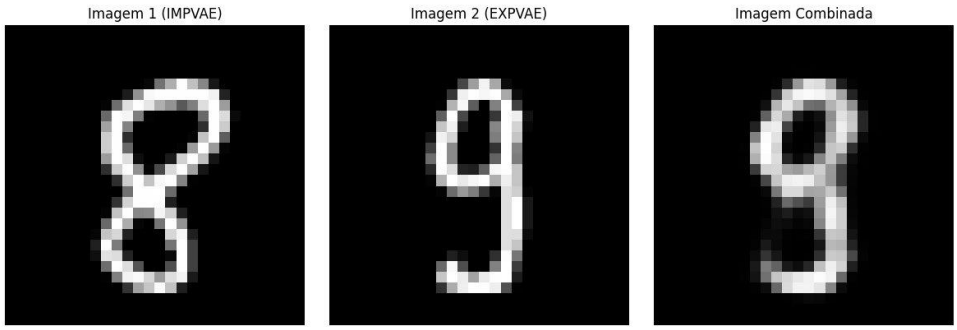


Figura 5.10. Testes iniciais com o DisVAE.

CRONOGRAMA

As atividades serão desenvolvidas no período março de 2023 a dezembro de 2025, e resumem-se nas seguintes tarefas:

1. Disciplinas mestrado;
2. Revisão bibliográfica;
3. Escrita do anteprojeto;
4. Desenvolvimento inicial das funcionalidades do aplicativo;
5. Avaliação dos resultados iniciais;
6. Qualificação do projeto de pesquisa;
7. Implantação aplicativo;
8. Escrita da dissertação;
9. Defesa dissertação.

O cronograma das atividades pode ser visto a seguir.

Quadro 0.1. Cronograma de Atividades.

	Semestres					
Atividades	**1**	**2**	**3**	**4**	**5**	**6**
1	**X**	**X**	**X**	**X**		
2			**X**	**X**	**X**	**X**
3				**X**		
4				**X**		
5				**X**	**X**	
6					**X**	
7					**X**	**X**
8					**X**	**X**
9						**X**

REFERÊNCIAS BIBLIOGRÁFICAS

[X] ADIOS. ADIOS 2: The Adaptable Input/Output System version 2. Disponível em: https://adios2.readthedocs.io/en/v2.10.1/. Acesso em 25 de setembro de 2024.

[X] ALTAMIRA, L. M.; VIEGAND, J.; POLVERINI D.; HUANG B.; FLUCKER S. The role of data centres in reducing energy consumption through policy measures. **ECEEE Summer Study on Energy Efficiency: Is Efficient Sufficient?** 2019, Presqu'ile de Giens. Proceedings... Presqu'ile de Giens, 2019, p. 1581-1591

[X] AMAZON. O que são dados em streaming? Disponível em: https://aws.amazon.com/pt/what-is/streaming-data/. Acesso em 18 de setembro de 2024.

[X] BAGLIETO, E.; NINOKATA, H.; MISAWA, M. CFD and DNS methodologies development for fuel bundle simulations. **Nuclear Engineering and Design**, v. 236, p. 1503-1510, 2006

[X] BARTOLHOMEW, P.; MOULINEC, C.; WEILAND, M.; LAIZET, S. eCSE-0302 Final Report Disponível em: < https://www.archer2.ac.uk/ecse/reports/ARCHER2-eCSE03-02_final_report.pdf >. Acesso em: 25 setembro 2024.

[X] BAUER, A.; ABBASI, H.; AHRENS, J.; CHILDS, H.; GEVECI, B.; KLASKY, S; MORELAND, K.; O'LEARY, P.; VISHWATH, V., WHITLOCK, B; BETHEL, E. *In Situ* Methods, Infrastructures, and Applications on High Performance Computing Platforms. **Computer Graphics Forum**, v. 35, p. 577-597, 2016.

[X] BNÀ, S.; COLOMBO, A.; CRIVELLINI A.; MEMMOLO, A.; SALVADORE, F.; BERNARDINI, M.; GHIDONI, A.; NOVENTA, G. In situ visualization for high-fidelity CFD—Case studies. **Computers & Fluids**, v. 267, 2023.

[X] CANTERO M. I.; LEE J.; BALANCHADAR S.; GARCIA M. H. On the front velocity of gravity currents, **Journal of Fluid Mechanics**, 2007b, vol. 586, p. 1

[X] COSTAL L.; RIBEIRO M.; Propriedades Dinâmicas de Fluidos por Simulação Computacional: métodos híbridos atomístico-contínuo, **Química Nova**, 2010, vol. 33

[X] DEEPLEARNINGBOOK. Variational Autoencoders (VAEs) – Definição, Redução de Dimensionalidade, Espaço Latente e Regularização. Disponível em:

https://www.deeplearningbook.com.br/variational-autoencoders-vaes-definicao-reducao-de-dimensionalidade-espaco-latente-e-regularizacao/. Acesso em 25 de setembro de 2024.

[X] DI et al. **A Survey on Error-Bounded Lossy Compression for Scientific Datasets.** arXiv preprint arXiv:2404.02840, 2024

[X] ELISON T.; TURNER J. Turbulent entrainment in stratified flows, **J. Fluid Mech.**, 1959, vol. 6, p. 423

[X] ESSS. Processo de Simulação Fluidodinâmica. Disponível em: https://www.esss.com/blog/processo-de-simulacao-fluidodinamica-cfd/. Acesso em 11 de novembro de 2024.

[X] FARENZENA, B. **Simulação numérica de correntes de densidade hiperpicnais sob referencial móvel.** Porto Alegre. 2020. 104p. Tese (Doutorado). Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, Pontificia Universidade Católica do Rio Grande do Sul, Brasil.

[X] FARENZENA, B.;. SILVESTRINI J. Density currents front velocity uncertainty, **Computers and Fluids**, 2022, vol. 232

[X] FERZIGER, Joel.; PERIC Milovan **Computational Methods for Fluid Dynamics**. 2ª ed. Berlim: Springer, 1999. 389p.

[X] GLAWS, A.; KING, R.; SPRAGUE M. Deep learning for in situ data compression of large turbulent flow simulations. **Phys. Rev. Fluids**, v. 5 2020.

[X] GIVORD, L.; MAZEN, F.; GUEUNET, C. Catalyst-ADIOS2: a new Catalyst2 implementation for in-transit analysis. Disponível em: https://www.kitware.com/catalyst-adios2-a-new-catalyst2-implementation-for-in-transit-analysis/. Acesso em: 20 novembro 2024.

[X] JIAO, C.; LI, K.; FANG, Z. Data sharing practices across knowledge domains: A dynamic examination of data availability statements in PLOS ONE publications. **Journal of Information Science**, v. *50*, p. 673-689, 2024.

[X] JIANG, Xi; LAI, Choi-Hong. **Numerical techniques for direct and large-eddy simulations**. CRC press, 2016.

[X] LECUN, Y; CORTES, C; BURGES, C. The MNIST Database of handwritten digits. Disponível em: http://yann.lecun.com/exdb/mnist/. Acesso em 28 de novembro de 2024.

[X] KRESS, J. In Situ Visualization Techniques for High Performance Computing Disponível em: < https://www.cs.uoregon.edu/Reports/AREA-201703-Kress.pdf >. Acesso em: 23 setembro 2024.

[X] HETHERINGTHON, A.; CORROCHANO, A.; HEREDIA, R.; ENEKO, L.; MUNHÕZ, E.; DÍAZ, P.; EGOITZ, M.; MARTIN, M.; CLAINCHE, S. ModelFLOWs-app: Data-driven post-processing and reduced order modelling tools. **Computer Physics Communications**, v. 301 2024.

[X] LAUFER, M.; FREDJ, E. **High Performance Parallel I/O and In-Situ Analysis in the WRF Model with ADIOS2.** arXiv preprint arXiv:2201.08228, 2022

[X] LE, M.; TAO, E. Hierarchical Autoencoder-based Lossy Compression for Large-scale High-resolution Scientific Data**.** arXiv preprint arXiv:2307.04216v2, 2024

[X] LI, F.; WANG, D.; YAN, F.; Song, F. ElasticBroker: Combining HPC with cloud to provide realtime insights into simulations. arXiv preprint arXiv:2010.04828, 2020

[X] LIU, T.; WANG, J.; LIU, Q.; ALIBHAI, S.; TAU, L.; HE, S. High-Ratio Lossy Compression: Exploring the Autoencoder to Compress Scientific Data. I**EEE TRANSACTIONS ON BIG DATA**, 2023

[X] LIU, W.; TAN, W.; MAO, H.; DING, Z.; CAO, Y.; WANG, P. Long-term Energy Consumption Forecasting for Data Center Industry in China. **5th IEEE Conference on Energy Internet and Energy System Integration: Energy Internet for Carbon Neutrality**, 2021, Taiyuan. Procedings... Taiyuan, EI2 2021, p. 4392 – 4397

[X] LIU, Y.; WANG, Y.; DENG, L. A novel in situ compression method for CFD data based on generative adversarial network. **J Vis** v. 22, p. 95–108, 2018.

[X] LIU, Q.; QIAO Z.; LV, Y. PyVT: A python-based open-source software for visualization and graphic analysis of fluid dynamics datasets. **Aerospace Science and Technology** v. 117, 2021.

[X] MALIK, S.; LÉVÊQUE, E.; BOUAIFI, M.; GAMET L.; FLOTTES, E.; SIMOËNS, S.; EL-HAJEM M. Shear improved Smagorinsky model for large eddy simulation of flow in a stirred tank with a Rushton disk turbine. **Chemical Engineering Research and Design**, 108, p. 69 – 80, 2016.

[X] MATPLOTLIB. Matplotlib: Visualization with Python Disponível em: https://matplotlib.org/. Acesso em 25 de setembro de 2024.

[X] MATRI, P.; Ross, R. Neon: Low-Latency Streaming Pipelines for HPC. **IEEE 14th International Conference on Cloud Computing (CLOUD)**, Chicago, IL, USA, 2021, p. 698-707

[X] METSUL. Fotos: As incríveis imagens da nova erupção do vulcão na Islândia. Disponível em: https://metsul.com/fotos-as-incriveis-imagens-da-nova-erupcao-do-vulcao-na-islandia/. Acesso em 20 de novembro de 2024.

[X] METSUL. METSUL alerta para intensos temporais isolados neste fim de semana. Disponível em: https://metsul.com/metsul-alerta-para-intensos-temporais-isolados-neste-fim-de-semana/. Acesso em 20 de novembro de 2024.

[X] METSUL. Tempestade de areia enorme assusta e faz o dia virar noite em São Paulo. Disponível em: https://metsul.com/tempestade-de-areia-enorme-assusta-e-faz-o-dia-virar-noite-em-sao-paulo/. Acesso em 20 de novembro de 2024.

[X] MOMENIFAR M.; Diao E.; TAROKH V.; BRAGG A., A Physics-Informed Vector Quantized Autoencoder for Data Compression of Turbulent Flow. **Data Compression Conference (DCC),** Snowbird, UT, USA, 2022, p. 1-10

[X] MPI4py. MPI for Python Disponível em: https://mpi4py.readthedocs.io/en/stable/. Acesso em 25 de setembro de 2024.

[X] MOLLER, S.; SILVESTRINI, J. Turbulência: Fundamentos. In: **Turbulência**. Associação Brasileira de Engenharia e Ciências Mecânicas. Rio de Janeiro, 2004, v. 4, p. 1-31.

[X] MULDER T.; ALEXANDER J. The physical character of subaqueous sedimentary density flows and their deposits., Sedimentology, 2001, vol. 48, p. 269

[X] NUMPY. Numpy Documentation Disponível em: https://numpy.org/doc/stable/. Acesso em 25 de setembro de 2024.

[X] OLIVEIRA, W.; PIRES, M.; CANNO, L.; RIBEIRO, L. Flow study in channel with the use computational fluid dynamics (CFD). Journal of Physics: Conference Series, v. 738/1, 2016.

[X] PATEL, S.; DRIKAKIS, D. Large eddy simulations of transitional and turbulent flows in synthetic jet actuators. **Solid Mechanics and its Applications**, 7, pp. 141 - 144, 2008.

[X] PARAVIEW. Paraview Catalyst Disponível em: https://www.paraview.org/Wiki/ParaView/Catalyst/Overview. Acesso em 25 de setembro de 2024.

[X] POESCHEL, F. *et al*. Transitioning from File-Based HPC Workflows to Streaming Data Pipelines with openPMD and ADIOS2. **Driving Scientific and Engineering Discoveries Through the Integration of Experiment, Big Data, and Modeling and Simulation.** Springer, p. 99-118, 2022.

[X] POURBAGIAN M.; ASHRAFIZADEH A. Super-resolution of low-fidelity flow solutions via generative adversarial networks. **Simulation: Transactions of the Society for Modeling and Simulation International**, v. 98, pp. 645–663, 2022.

[X] PYTORCH. Pytorch Documentation. Disponível em: https://pytorch.org/docs/stable/index.html. Acesso em 25 de setembro de 2024.

[X] RAISSI, M.; PERDIKARIS P.; KARNIADAKIS G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, **Journal of Computational Physics**, v. 378, p. 686-707, 2019.

[X] RUAN, H.; HU Q.; ZHAO M. Research Progress of Ultra-High Density Optical Storage. **Zhongguo Jiguang/Chinese Journal of Lasers**, v. 51/11, 2024.

[X] RUSCHEL, K. **Simulação numérica de correntes de densidade em configuração não-confinada com alimentação contínua**. Porto Alegre. 2022. 125p. Tese (Doutorado). Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, Pontifícia Universidade Católica do Rio Grande do Sul, Brasil.

[X] SALMINEM, R. Development of an online/web-based 3D visualization tool for CFD results. Disponivel em: https://aaltodoc.aalto.fi/server/api/core/bitstreams/8d3c8e98-7581-41b5-a943-5c865ca92730/content. Acesso em 20 de novembro de 2024.

[X] SCHUCH, F. Xcompact3d-toolbox. Disponível em: https://github.com/fschuch/xcompact3d_toolbox/. Acesso em 25 de setembro de 2024.

[X] SCHUCH, F. **Análise do mergulho de escoamentos hiperpicnais em canal inclinado por meio de simulação numérica de grandes escalas**. Porto Alegre. 2020. 150p. Tese (Doutorado). Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, Pontifícia Universidade Católica do Rio Grande do Sul, Brasil.

[X] SCIPY. Scipy User Guide. Disponível em:

https://docs.scipy.org/doc/scipy/tutorial/index.html#user-guide. Acesso em 25 de setembro de 2024.

[X] SIMPSOM, J. **Gravity Currents: In the Environment and the Laboratory.** Cambridge University Press, 1999.

[X] SOLMAZ, S.; VAN GERVEN, T. Acrossim: A toolkit for cross-platform integration of CFD simulation data in computer graphics. **SoftwareX**, v. 24, 2023

[X] SOSA, S. **Análise da turbulência nas correntes de densidade em configuração lock-release**. Porto Alegre. 2023. 122p. Tese (Doutorado). Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, Pontifícia Universidade Católica do Rio Grande do Sul, Brasil.

[X] STANCANELLI, L.; MUSUMECI, R.; FOTI, E. Computational Fluid Dynamics for Modeling Gravity Currents in the Presence of Oscillatory Ambient Flow. **Water**, v. 10 p. 635, 2018

[X] TABLEAU. What Is Data Visualization? Definition, Examples, And Learning Resources. Disponível em: https://www.tableau.com/visualization/what-is-data-visualization#:~:text=Data%20visualization%20is%20the%20graphical,outliers%2C%20and%20patterns%20in%20data. Acesso em 20 de novembro de 2024.

[X] TENOPIR, C.; RICE, N.; ALLARD, S; BAIRD, L.; BORYCZ, J.; CHRISTIAN L.; et al. Data sharing, management, use, and reuse: Practices and perceptions of scientists worldwide. **PLoS ONE**, v. 15, 2020

[X] VIANNA, F. **Feições espaço-temporais em correntes de gravidade usando métodos de visão computacional**. Porto Alegre. 2021. 106p. Tese (Doutorado). Programa de Pós-Graduação em Engenharia e Tecnologia de Materiais, Pontifícia Universidade Católica do Rio Grande do Sul, Brasil.

[X] WANG, T; Zhang, M; Shang, L. DisVAE: Disentangled Variational Autoencoder for High-Quality Facial Expression Features. **IEEE 17th International Conference on Automatic Face and Gesture Recognition (FG)**, p. 1 – 8, 2023.

[X] WANG, R. Postprocessing. **Computational Fluid Dynamics: Applications in Water, Wastewater and Stormwater Treatment**, p. 57 – 62, 2019.

[X] XCOMPACT3D. PY4Incompact3d. Disponível em: https://github.com/xcompact3d/Py4Incompact3D/. Acesso em 25 de setembro de 2024.

[X] ZHUHADAR, L.; LYSTRAS, M. The application of AutoML Techniques in Diabetes Diagnosis: Current approaches, performance, and future directions. **Sustainability**, 15, 2023.

ANEXOS

Inserir informações que achar necessário, e que não merecem mérito de estarem inseridas no corpo do trabalho.

APÊNDICES

Inserir informações que achar necessário, e que não merecem mérito de estarem inseridas no corpo do trabalho.