JOÃO LUIS RIBEIRO OKIMOTO

MODELAGEM COMPUTACIONAL LAGRANGIANA OFFLINE DE TRANSPORTE DE RESÍDUOS PLÁSTICOS NO OCEANO

(versão pré-defesa, compilada em 26 de fevereiro de 2023)

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Eduardo Todt.

CURITIBA PR

2023

RESUMO

A poluição de plástico em rios e oceanos tornou-se um problema de escala global, afetando uma diversidade de ecossistemas e causando impactos ambientais. Devido a imensidão dos oceanos, e a um conjuto de fatores externos como afundamento, fragmentação e dispersão, tornou-se impossível mensurar e mapear a poluição de resíduos plásticos de maneira precisa.

Neste contexto, a modelagem computacional de sistemas ocêanicos torna-se fundamental para compreender a escala do problema de poluição de plástico, buscando visualizar o fluxo de partículas e mapear as regiões mais afetadas. No entanto, existem grandes desafios numéricos, computacionais e de modelagem de fluxo que precisam ser superados para que uma modelagem fiel possa ser feita. Além disso, é necessário que profissionais de diversos campos como cientistas da computação, oceanográfos, engenheiros e físicos cooperem para compreender o fenômeno de maneira concreta.

Este trabalho busca detalhar como simulações que modelam o fluxo de resíduos plásticos nos oceanos funcionam de maneira fundamental, revisando métodos numéricos e algoritmos utilizados para resolver o problema de rastreamento lagrangeano de partículas. Procuramos compreender qual o papel da ciência da computação neste problema ecológico, revisando o papel do processamento de alto desempenho e aspectos práticos da modelagem lagrangeana.

Por fim, busca-se aplicar os fundamentos através de uma simulação de um cenário de poluição no litoral Paranaense com o framework Parcels.

Palavras-chave: Poluição de plásticos. Modelagem Computacional. Análise Lagrangiana

ABSTRACT

Plastic pollution in rivers and oceans has become a global problem, affecting a diversity of ecosystems and causing economic and social impacts. Due to the immensity of the oceans, its extensive connectivity, turbulence and a set of external factors such as sinking and fragmentation, it has become impossible to fully measure and map the pollution of plastic waste.

In this context, computational modeling of oceanic systems becomes essential to understand the scale of the plastic pollution problem, seeking to visualize the flow of particles and map the most affected regions. However, there are major numerical, computational and flow modeling challenges that need to be overcome so that faithful and accurate modeling can be done. In addition, professionals from diverse fields such as computer scientists, oceanographers, engineers, and physicists need to come together to understand the phenomenon in a concrete way.

This work seeks to understand how simulations that model the flow of plastics in the oceans work in a fundamental way, reviewing numerical methods and algorithms used to solve it computationally. We seek to understand the role of computer science in this ecological problem, reviewing the role of high-performance computing and practical aspects of Lagrangian modeling. In addition, we seek to apply the fundamentals through a simulation on the coast of Paraná with the Parcels framework.

Keywords: Plastic Pollution. Computational Modelling. Lagrangian Analysis.

LISTA DE FIGURAS

LISTA DE ACRÔNIMOS

DINF	Departamento de Informática
PPGINF	Programa de Pós-Graduação em Informática
UFPR	Universidade Federal do Paraná
PET	Politereftalato de Etilo
HPC	High Performance Computing
PARCELS	Probably a Really Efficient Lagragian Simulator
OGCM	Ocean General Circulation Model
GPS	Global Positioning System
EDO	Equação Diferencial Ordinária
NetCDF	Network Common Data Form
MPI	Message Passing Interface
MITGcm	Massachussets Institute of Technology General Circulation Model
HYCOM	Hybrid Coordinate Ocean Model
NEMO	Nucleus for European Modelling of the Ocean
ROMS	Regional Ocean MOdelling System
MPAS-O	Massive Parallel Ocean Model
JIT	Just in Time

SUMÁRIO

1 INTRODUÇÃO

O termo “plástico” refere-se a um conjunto de materiais constituídos por polímeros utilizados em diversas aplicações, principalmente para a produção de embalagens. Um exemplo de plástico muito utilizado para este fim é o Politereftalato de Etilo (PET), que constitui produtos como garrafas de refrigerante, embalagens de garra e fios de poliéster. Plásticos possuem uma grande importância econômica em uma gama de outros setores como o de construção, transporte e eletrônicos, permitindo que novos materiais sejam manufaturados de maneira viável, alimentos sejam armazenados por períodos mais extensos, entre outros. (Hale e Song, 2020)

Apesar das grandes vantagens que plásticos proporcionam, estes materiais não são biodegradáveis, e o único meio de tratamento deste tipo de resíduo constitui-se pela incineração ou reciclagem Ritchie e Roser (2018). No entanto, a grande parte dos resíduos plásticos são tratados de maneira imprópria, sendo majoritariamente descartados nos oceanos e ambientes marinhos. Uma estimativa com dados em campo foi feita em 2015 por Eriksen et al. (2014), e pelo menos 5.25 trilhões de párticulas de plástico pesando 268.940 toneladas estariam flutuando sobre todos os oceanos do planeta. Outras projeções como a de Lebreton et al. (2019) sugerem um número ainda maior de mais de 500.000 toneladas.

A figura 1.1 apresenta um gráfico com perspectivas futuras de emissões de plástico.

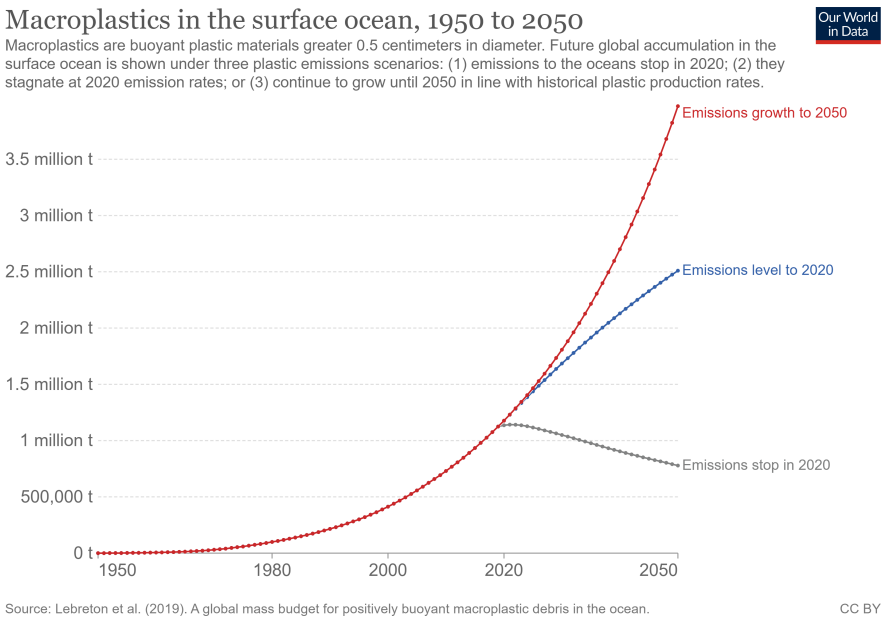

Figura 1.1: Projeções de macroplásticos nas superficíes do oceano, retirado de (Ritchie e Roser, 2018)

Uma grande variedade de plásticos circulam diariamente sobre todas as porções do oceano, diferenciando-se em composição, tamanho e densidade. Apesar de não existir um padrão entre diferentes tipos de resíduos, existem quatro grandes grupos de plásticos considerados em

análises de poluição, sendo eles microplásticos, mesoplásticos, macroplásticos e megaplásticos, tendo como distinção seu comprimento. A figura 1.2 mostra a distinção entre estas categorias, representando (a) macroplásticos (b) mesoplásticos e (c-f) microplásticos.

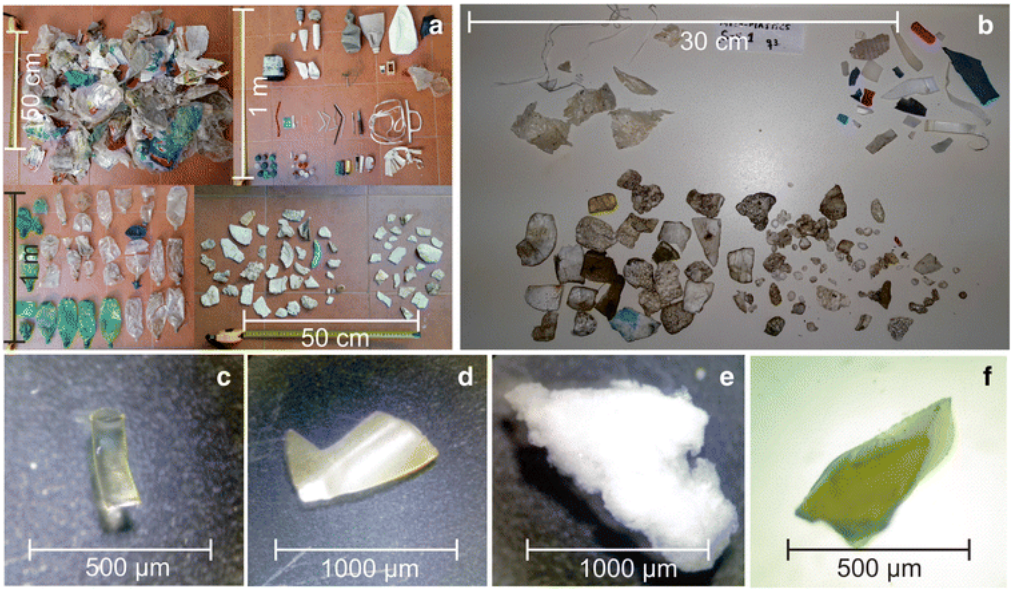

Figura 1.2: Exemplo da distribuição de tipos de plástico, retirado de (Blettler et al., 2017)

A incidência de raios solares, impactos físicos e a composição química e biológica do oceano fazem com que grandes unidades de plástico sejam gradativamente fragmentadas ao longo do tempo, gerando microplásticos que muitas vezes não são captáveis ao olho nú. Em um estudo recente feito pelo Centro de Estudos do Mar da UFPR (França, 2022), foi identificado a presença de microplásticos em 16 das 19 praias estudadas do Paraná, indicando a prevalência desta substância no ecosistema.

Mesmo sabendo da magnitude do problema, especialistas ainda não compreendem completamente os efeitos que o descarte de plásticos pode ter. Sabe-se que macroplásticos podem ser danosos a eco-sistemas marinhos, que de acordo com uma análise feita por Rochman et al. (2016) animais como tartarugas, pinguins, leões-marinho, baleias, passáros marinhos, peixes, carangueijos, tubarões,focas e até mesmo micro-organismos, são prejudicados diretamente ou indiretamente pela presença de resíduos plásticos no oceano. No caso de focas, a ingestão de linhas de pesca pode causar constrições, o consumo de sacolas plásticas por tartarugas pode causar a morte, e um estudo até mesmo apontou a diminuição do tamanho de orgãos em passáros devido ao consumo de microplásticos.

1.1 MOTIVAÇÃO

De acordo com (Geyer et al., 2017) cerca de metade de todos os plásticos produzidos devem flutuar sobre o oceano, por serem menos densos que a água salgada, e assim, tais fragmentos acabam entrando na circulação marinha e deslocando-se para outras partes do globo, enquanto a outra metade acaba afundando para regiões mais profundas. Essa dispersão se tornou

tão problemática que é possível encontrar plástico até mesmo no gelo do mar ártico, e sacolas de supermercado na região mais profunda do oceano pacífico, na fossa das marianas (Gibbens, 2019). Apesar de ser um problema pertinente e atual, van Sebille et al. (2020) afirma que a distribuição da concentração de resíduos plásticos no oceano ainda não é muito bem mapeada, pelo fato de que um grande número de fenômenos físicos, químicos e biológicos estão associados ao seu transporte, decomposição e destino destas partículas.

Quando uma unidade de plástico como uma garrafa PET é descartada no oceano, a ação dos ventos, circulação maritima, diferenças de densidade e composição química podem influenciar no seu destino e transporte. Sua composição muda ao longo do tempo pelo crescimento de colônias de algas e micro-organismos, causando o seu afundamento. Além disso, processos caóticos como o afundamento temporário resultado pela ação de ondas e eventos de mudança de maré podem temporariamente deslocar verticalmente o poluente, como representado pela figura 1.3. Determinar o destino desta quantidade massiva de resíduos é praticamente impossível utilizando apenas observações em campo, em razão da magnitude do oceano e do grande número de fatores envolvidos. Por se tratar de um ciclo de extrema complexidade, o fluxo de plásticos no oceano ainda não é muito bem compreendido pela comunidade oceanográfica (van Sebille et al., 2020) e torna-se necessário empregar o uso de modelos computacionais que forneçam uma visão mais rigorosa do problema, através de métodos que simulem o fluxo do plástico em rios e oceanos.

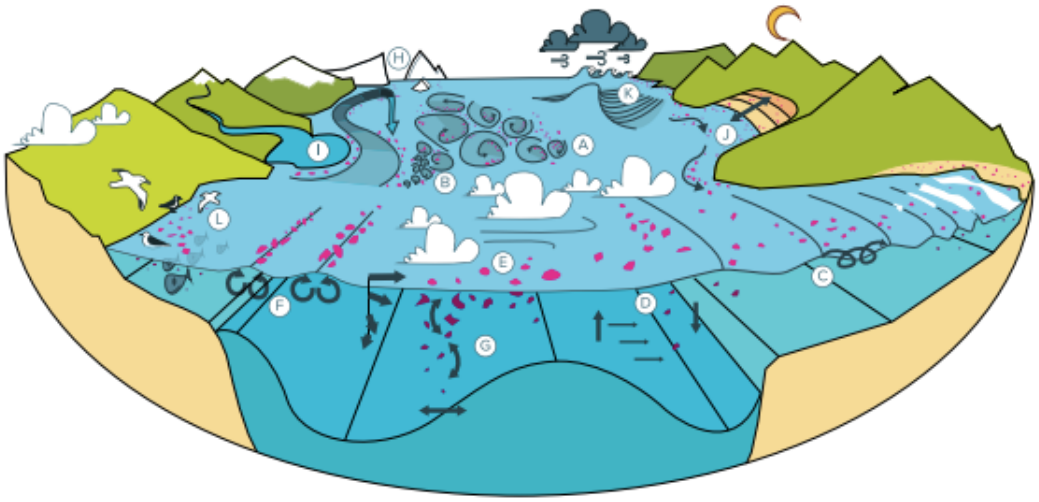

Figura 1.3: Representação gráfica de processos que causam a dispersão de resíduos na circulação marinha. Retirado de (van Sebille et al., 2020)

Para que seja possível realizar uma análise de fluxo de poluentes que seja escalável e fiel, modelos computacionais de advecção de partículas são utilizados. Tais modelos buscam recriar trajetórias de partículas virtuais baseadas em campos vetoriais, através de uma série de passos de advecção, no qual cada passo avança uma partícula no espaço por uma curta distância ao resolver uma equação diferencial. Esta trajetória é utilizada para visualizar o fluxo de resíduos

em diferentes cenários, permitindo que especialistas e organizações mapeiem a poluição nos oceanos de maneira menos custosa, exemplificado na figura 1.4.

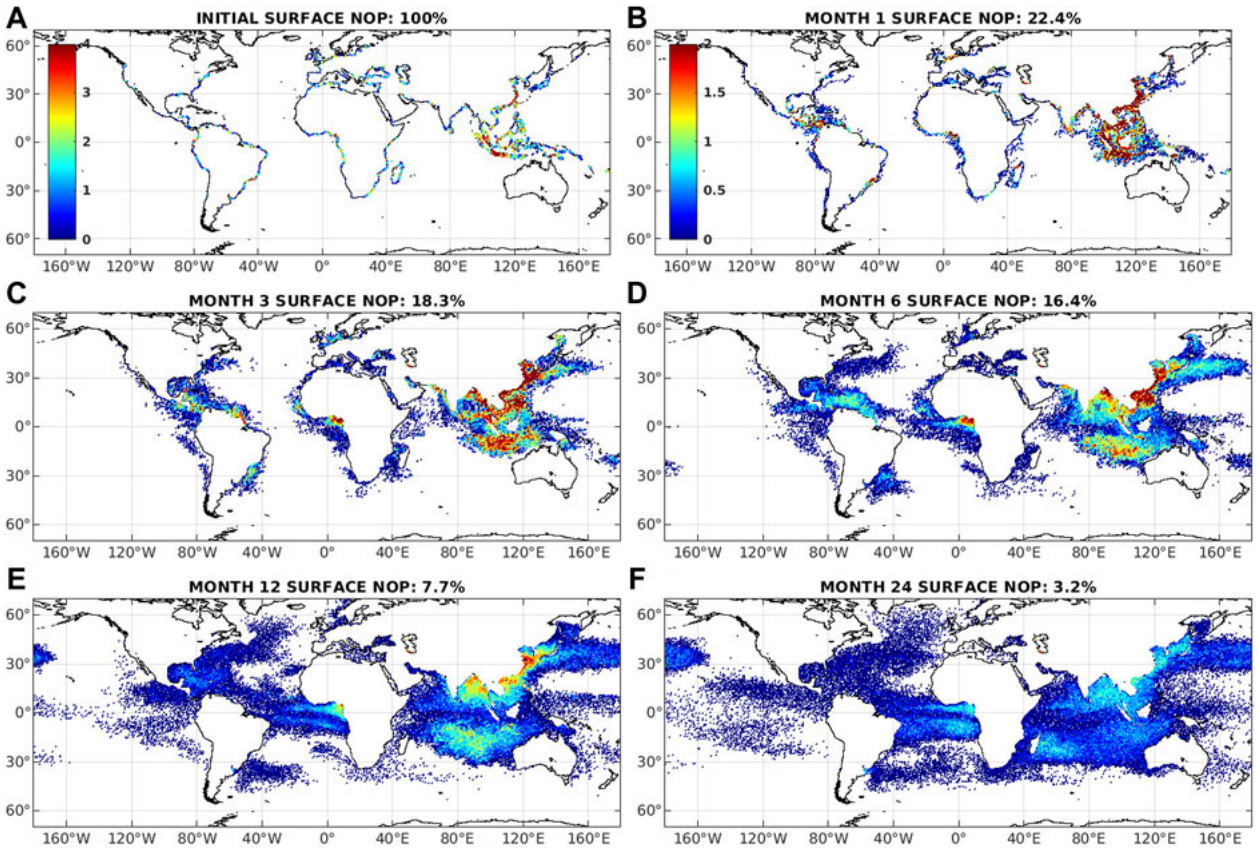

Figura 1.4: Exemplo de simulação de dispersão global de partículas por 24 meses, retirado de (Huck et al., 2022)

1.2 DESAFIOS

A fundamentação de modelos computacionais para a simulação de estudos climáticos baseia-se num esforço cooperativo entre o campo da oceanografia, física, métodos numéricos, geoprocessamento e computação, desta forma, é necessário que um consenso seja estabelecido entre estes diversos campos para que uma modelagem fiel e precisa seja realizada.

Simulações do oceano geram grandes volumes de dados e requerem alto processamento, o alto número de partículas e a granularidade fina do domínio da aplicação geram um elevado custo computacional, que mostram-se como grandes desafios de performance (Yenpure et al., 2022). Por tratar-se de um ciclo de escala planetária, é necessário que um grande número de partículas seja empregado na simulação para uma análise adequada. O domínio espacial geralmente extende-se por uma grande área, englobando bacias hidrográficas, mares e oceanos. Por fim, a escala temporal tende a extender-se por anos, décadas e até mesmo séculos, gerando simulações longas que descrevem comportamentos do oceano ao longo da história. Apesar disso, oceanográfos e cientistas da comunidade carecem de modelos robustos e rápidos, com baixos

custos e acessíveis para que experimentos sejam constantemente repetidos e novas hipóteses e idéias sejam testadas.

O processamento de alto desempenho (HPC tem um papel fundamental em aprimorar modelos e garantir a eficiência de modelos hidrodinâmicos, já que tradicionalmente os maiores supercomputadores sempre foram usados para estudos climáticos e previsão numérica do tempo. Neste contexto, é fundamental encontrar estratégias de otimização para o algoritmo de advecção de partículas, que forma a base da modelagem de fluxos no oceano.

1.3 OBJETIVOS

O objetivo deste trabalho resume-se a compreender como modelos hidrodinâmicos funcionam e como eles encaixam-se no contexto da análise de poluição de plástico nos oceanos. Também busca-se entender o papel do processamento de alto desempenho e do paralelismo através do reconhecimento de quais estratégias podem ser utilizadas para melhorar o processo. Por fim, buscamos modelar um possível cenário de poluição de plástico no litoral paranaense, procurando entender como esta modelagem aplica-se no contexto local.

1.4 PROPOSTA

Neste trabalho uma revisão sistemática dos métodos numéricos, algoritmos e aspectos computacionais utilizados para o problema de modelagem computacional da poluição de plástico é feita. Busca-se também criar uma ponte entre o processamento de alto desempenho e o geoprocessamento, criando base para a comunicação entre ambos os campos através do framework Parcels. Buscamos utilizar como experimento o lançamento de um número variado de partículas no litoral paranaense por longos períodos de tempo, como forma de benchmark e também como forma de conscientização do destino do plástico produzido pelo estado.

1.5 ESTRUTURA DO DOCUMENTO

Os capítulos deste documento estão organizados da seguinte forma. O capítulo 2 introduz conceitos fundamentais da modelagem hidrodinâmica, geoprocesamento, o algoritmo de advecção de partículas e definições práticas na modelagem de fluxo.

O capítulo 3 revisa o estado da arte e diferentes pacotes e bibliotecas disponíveis com funcionalidades de dispersão de partículas no oceano, além de introduzir o pacote de maior interesse Parcels (Probably a very efficient lagrangian simulator).

O capítulo 4 descreve o procedimento, configuração e parâmetros dos experimentos realizados com base no framework Parcels.

O capítulo 5 apresenta os resultados adquiridos ao longo do trabalho, indicando qual o impacto da análise no contexto do litoral paranaense e a utilização de recursos computacionais.

O capítulo 6 discute limitações e possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conceitos fundamentais da análise lagrangiana do oceano, introduzindo conceitos da hidrodinâmica, o algoritmo de advecção, aspectos de geoprocessamento e metodologia computacional.

2.1 MODELAGEM LAGRANGIANA NO OCEANO

Kreuzer e Sichermann (2006) classifica o oceano e muitos dos fenômenos climáticos como sistemas não lineares, caóticos e altamente dinâmicos. Tais modelos descrevem o comportamento de sistemas físicos, biológicos ou sociais ao longo do tempo. Ao contrário dos sistemas lineares, os sistemas não lineares exibem um comportamento complexo e às vezes caótico que não pode ser descrito usando métodos matemáticos lineares.

Em sistemas dinâmicos não lineares, a relação entre as entradas e saídas não é proporcional. Isso significa que pequenas mudanças nas entradas podem resultar em mudanças grandes e não previsíveis nas saídas. Os sistemas dinâmicos não lineares são frequentemente caracterizados por múltiplas soluções, múltiplos pontos de equilíbrio e a presença de *feedback loops*, que podem levar a um comportamento complexo e dinâmico.

Exemplos de sistemas dinâmicos não lineares incluem padrões climáticos, propagação de doenças, crescimento populacional, sistemas econômicos e dinâmica de fluidos. O comportamento desses sistemas geralmente é difícil de prever, mas modelos matemáticos podem ser usados para estudar os padrões e melhor compreendê-los.

Para analisar sistemas dinâmicos não lineares, diversos métodos matemáticos e computacionais são usados, incluindo equações diferenciais, teoria do caos, teoria da bifurcação e atratores. Tais métodos permitem a análise e simulação de sistemas não lineares e podem ser usados para obter insights sobre os padrões e relacionamentos subjacentes.

No contexto de análise da poluição, a abordagem principal para o problema de dispersão de partículas de plástico sustenta-se na modelagem hidrodinâmica lagrangiana, que permite simular cenários plausíveis e gerar dados utéis para análises.

Um pipeline de processos que envolvem toda a modelagem computacional é mostrado na figura 2.1, no qual cada etapa é descrita por um diagrama de fluxo.

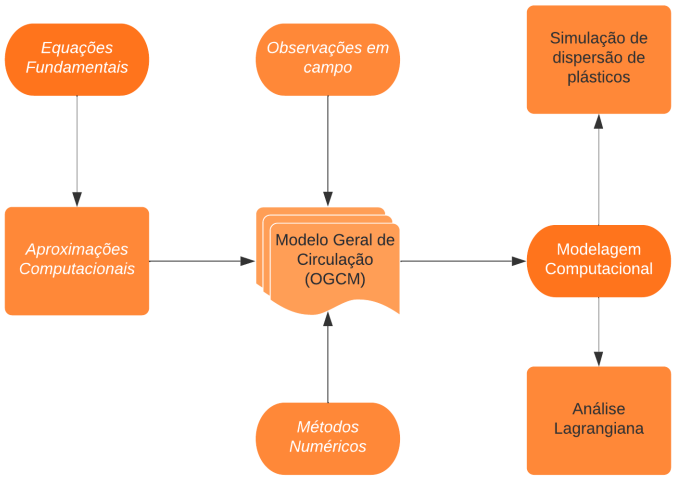

Figura 2.1: Diagrama do pipeline de modelagem da hidrodinâmica do plástico
fonte: autoria própria

Inicialmente é necessário descrever o comportamento físico do oceano por meios de equações primitivas, e aplicar aproximações numéricas adequadas para simplificar o modelo. Em seguida, um modelo de circulação (Ocean General Circulation Model - OGCM) é gerado a partir da resolução de equações diferenciais e sistemas dinâmicos com base em aproximações numéricas, ou a partir de observações de campo experimentais. Este modelo é utilizado para executar simulações e analisar possíveis cenários de dispersão de partículas virtuais, que é utilizado por oceanográfos, biólogos e cientistas que desejam compreender diferentes fenômenos da circulação global (Yoon e Ma, 2012).

2.1.1 Modelos Físicos Hidrodinâmicos

O principal processo físico de disseminação de plástico no oceano caracteriza-se pelo fluxo das correntes marítimas e processos oceânicos de larga escala, cuja base teórica se sustenta sob o campo da dinâmica dos fluídos geofísicos.

Um Modelo de Circulação Geral do Oceano (OGCM) é um modelo matemático que simula a circulação em larga escala do oceano e suas interações com a atmosfera, gelo marinho e terra. O modelo é construído resolvendo as equações que governam os processos oceânicos, como correntes oceânicas, temperatura, salinidade e nível do mar.

O ponto de partida para a criação de um OGCM são as equações primitivas do oceano, que são baseadas nas leis da física e descrevem a dinâmica do oceano em termos de velocidade, pressão e densidade. Essas equações são então discretizadas e resolvidas numericamente usando métodos computacionais, como diferenças finitas ou métodos de volumes finitos.

Para executar um OGCM, um conjunto de condições iniciais, incluindo temperatura, salinidade e velocidade, é especificado no início da simulação. O modelo então integra essas equações ao longo do tempo para simular a evolução da circulação oceânica. As condições de

contorno, como a troca de calor, água doce e momento com a atmosfera, também são especificadas no modelo.

Um exemplo de OGCM é demonstrado na figura 2.2, com temperaturas da superfície de uma região do oceano pacífico.

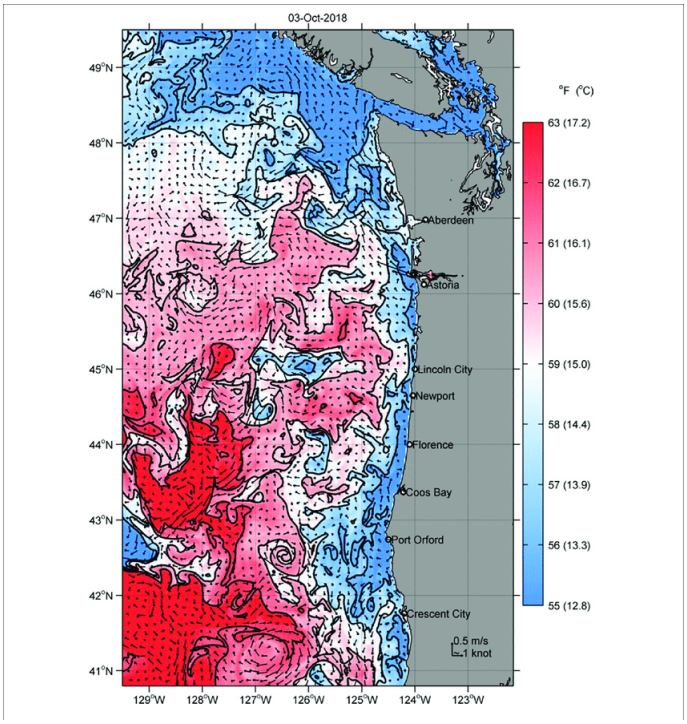

Figura 2.2: Exemplificação de um modelo de temperatura da superfície do noroeste do oceano pacífico retirado de (Barth et al., 2019)

Os OGCMs normalmente incluem parametrizações ou simplificações de processos físicos complexos que ocorrem no oceano, como mistura e turbulência. A precisão de um OGCM depende do realismo dessas parametrizações e da resolução da grade numérica utilizada no modelo.

Os OGCMs são usados para uma variedade de propósitos, incluindo estudar a resposta do oceano às mudanças climáticas, simular o transporte de calor e carbono no oceano e entender o papel do oceano no sistema climático da Terra.

2.1.2 Modelos lagrangianos

De acordo com (Versteeg e Malalasekera, 1995) no contexto de dinâmica dos fluidos, existem duas representações possíveis que possibilitam compreender e analisar um fluido qualquer. É possível avaliar diversas funções como pressão, temperatura, velocidade e aceleração de um elemento do fluído em uma região específica do espaço, denominada de abordagem Euleriana. Por outro lado, é possível também acompanhar um conjunto de partículas individualmente, calculando sua trajetória e outras variáveis em função do tempo, que é conhecida como a abordagem Lagrangiana.

Modelos hidrodinâmicos que seguem uma abordagem Lagrangiana são extremamente utéis para acompanhar a distribuição de partículas em uma região do espaço(van Sebille et al., 2018), e são muito utilizados em diversos contextos, como o rastreamento da migração de espécies de peixes e larvas (North Elizabeth W., 2009), icebergs, plankton e plástico. Tais modelos tem como objetivo determinar as rotas tomadas por objetos que encontram-se na circulação oceânica, através da caracterização do seu transporte por meio de equações que descrevem a posição de uma partícula em função do tempo. Esta abordagem procura acompanhar um conjunto de partículas virtuais cujas velocidades são determinadas por um campo vetorial de velocidade, extraído a partir de um OGCM. A posição destas partículas é atualizada em intervalos de tempo discretos, através da interpolação dos vetores de velocidade em determinado ponto com a posição atual.

A análise lagrangiana é um método usado para estudar o transporte e a mistura de marcadores passivos, como poluentes ou temperatura, no oceano ou na atmosfera. Modelos online e offline são dois tipos diferentes de modelos lagrangianos.

Os modelos online são integrados em um modelo numérico maior, como um Ocean General Circulation Model (OGCM) ou um modelo atmosférico, e são resolvidos simultaneamente com as outras equações do modelo. Nesse tipo de modelo, a equação do traçador lagrangiano é resolvida juntamente com as equações das correntes oceânicas, temperatura e salinidade. Isso permite uma representação mais abrangente do oceano ou do sistema atmosférico, pois os traçadores lagrangianos são influenciados pelos mesmos processos físicos que o restante do sistema.

Os modelos off-line, por outro lado, são separados do modelo numérico maior e são usados para pós-processar a saída de uma simulação. Neste tipo de modelo, os campos de velocidade oceânica ou atmosférica da simulação são usados como entrada, e a equação do traçador lagrangiano é resolvida para simular o transporte do traçador. A vantagem dos modelos off-line é que eles são menos dispendiosos computacionalmente do que os modelos on-line e podem ser executados com maior resolução espacial e temporal.

Em resumo, a principal diferença entre os modelos lagrangianos online e offline é a forma como eles são integrados ao modelo numérico maior. Os modelos online são integrados ao modelo maior e resolvidos simultaneamente, enquanto os modelos offline são usados para pós-processar a saída de uma simulação.

A execução online claramente possui vantagens de resolução em relação ao offline, por permitir uma granularidade mais fina e de maior precisão do comportamento das correntes, no entanto, requer uma computação muito maior a cada iteração do modelo. Apesar disso, executar modelos de maneira offline permite uma maior flexibilidade na simulação, possibilitando que partículas sejam analisadas em diferentes períodos de tempo e posicionadas em configurações iniciais variadas de maneira menos custosa. Poderiamos classificar a execução online como CPU-Bound por ser limitada pela capacidade de processamento da hidrodinâmica do modelo, e a execução offline como Memory-Bound por restringir-se à capacidade de leitura dos campos de velocidade.

Modelos offline são amplamente utilizados pela comunidade oceanográfica para a visualização de fluxo das correntezas marítimas, e apesar da necessidade do armazenamento de grandes volumes de dados, ainda é o padrão para cientistas que buscam executar experimentos com diferentes parâmetros e configurações variadas. Desta forma, neste trabalho serão abordados somente aspectos da modelagem *offline*, por se tratar de uma técnica mais simples e acessível para cientistas.

Existem também modelos gerados por dados retirados em campo, utilizando marcadores que são rastreados por GPS e armazenam informações da sua trajetória ao longo do tempo. Além disso, existem modelos baseados em observações por satélite que identificam a altura da superfície do oceano. De acordo com (van Sebille et al., 2018), tais métodos possuem a limitação de não representarem o transporte do volume das correntes, identificando apenas trajetórias horizontais na superfície de maneira bi-dimensional.

2.2 MÉTODOS NUMÉRICOS DA VISUALIZAÇÃO DO FLUXO

A visualização de fluxos é uma técnica amplamente utilizada em outras disciplinas, como engenharia aeroespacial e climatologia, pois permite ilustrar o comportamento de partículas sob campos vetoriais, e assim, a extração de características para análise científica.

Existem diversas abordagens para a extração de fluxos, que podem ser encontradas em (McLoughlin et al., 2009), no entanto, este trabalho procura explorar técnicas de *visualização geométrica*, que são mais adequadas para o problema em questão. A abordagem geométrica procura definir um conjunto de pontos iniciais para cada partícula, e computar uma trajetória afim de construir objetos geométricos.

Esta abordagem aplica-se na modelagem de fenômenos da atmosfera e do oceano através do algoritmo de advecção, por meio do espalhamento de partículas sem massa através de um campo vetorial, que busca resolver de maneira iterativa um conjunto de equações diferenciais em um período de tempo especificado.

Inicialmente, uma *seed* é inicializada com a posição inicial do conjunto de partículas, que são iterativamente deslocadas por meio do passo de advecção, até que uma condição de parada seja alcançada. Uma trajetória é armazenada para cada elemento, e a saída do algoritmo constitui um formato renderizável do fluxo analisado.

2.2.1 Algoritmo de Advecção

(Stewart, 2012) define campos vetoriais tridimensionais a partir de funções que associam a cada ponto do plano (x, y, z) um vetor $F(x, y, z)$, exemplificado na figura 2.3. Para determinar o fluxo de correntes marítimas, é associado a cada ponto do plano um vetor $v(\vec{t})$, que determina a direção e magnitude da velocidade de uma corrente no ponto (x, y, z) no tempo t. Também é possível associar campos de pressão, temperatura, densidade e outras características físicas a estes pontos, dependendo da aplicação de interesse.

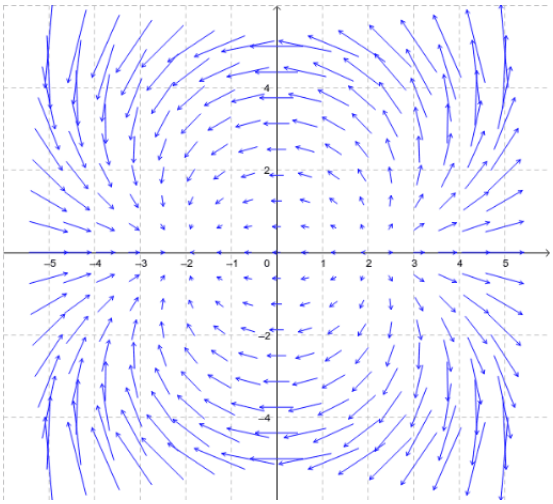

Figura 2.3: Exemplo de um campo vetorial bidimensional, retirado de: `https://www.geogebra.org/m/QPE4PaDZ`

Para determinar uma trajetória de uma partícula, integra-se o campo de velocidade em relação ao tempo e a posição inicial da partícula. Podemos definir a função de posição S em relação ao tempo t como $S(t)$, e dado uma posição inicial $S(t_0)$:

$$P(t) = P(t_0) + \int_{t_0}^{t} v(\vec{t}, p)\, dt \tag{2.1}$$

De acordo com (Lange e van Sebille, 2017) também é introduzido na equação (1) o termo $\Delta S_b(t)$ que engloba outros comportamentos na posição $S(t)$, como o afundamento de partículas em relação a diferença de densidade. Portanto, determinar o deslocamento de todas as partículas de interesse resume-se a resolver a equação (2):

$$P^i(t) = P^i(t_0) + \int_{t_0}^{t} v(\vec{t}, p)\, dt + \Delta P_b(t) \tag{2.2}$$

Apesar da simplicidade, as equações (1) e (2) assumem um domínio de espaço e tempo contínuos, tornando-se necessário a discretizá-los para aproximá-las de maneira numérica.

Procura-se re-escrever a equação (1) em forma derivativa

$$v(t, p) = \frac{dP}{dt} \tag{2.3}$$

Assim, a aproximação numérica da trajetória resume-se a um problema de valor inicial de uma equação diferencial ordinária (EDO) de primeiro grau, sujeito a condições inicias $t = t_0$ e $p = p_0$

Uma EDO é uma equação que descreve a taxa de variação de uma variável dependente em relação a uma variável independente, e tem como intuito descrever sistemas dinâmicos que variam muito em relação ao tempo (Schroers, 2011).

Na prática, existem dois métodos comumente utilizados em aplicações de advecção para resolver numericamente a EDO 2.6, o método de Euler e Runge Kutta de quarta ordem.

O método progressivo de Euler é uma técnica numérica simples e direta que aproxima a solução de uma EDO em intervalos de tempo discretos. Ele usa a tangente no passo de tempo atual para estimar a solução no próximo passo de tempo. A equação usada para estimar a solução é:

$$P(t + \Delta t) = P(t) + \Delta t \cdot v(t, P(t)) \tag{2.4}$$

No qual Δt é o tamanho do passo.

O método progressivo de Euler é fácil de implementar e computacionalmente eficiente, mas pode ser impreciso para certas EDOs e pode produzir soluções que não são bem comportadas, particularmente para passos de tempo longos ou para EDOs com fortes não linearidades.

O método de quarta ordem de Runge-Kutta é uma técnica numérica mais precisa para resolver EDOs. Ele usa uma média ponderada de quatro estimativas da solução em cada passo de tempo para estimar a solução no próximo passo de tempo.

$$k_1 = \Delta t \cdot v(t, P(t)) \tag{2.5}$$

$$k_2 = \Delta t \cdot v(t + \frac{\Delta t}{2}, P(t) + \frac{k_1}{2}) \tag{2.6}$$

$$k_3 = \Delta t \cdot v(t + \frac{\Delta t}{2}, P(t) + \frac{k_2}{2}) \tag{2.7}$$

$$k_4 = \Delta t \cdot v(t + \Delta t, P(t) + k3) \tag{2.8}$$

A equação usada para estimar a solução é definida como:

$$P(t + \Delta t) = P(t) + \frac{(k1 + 2k2 + 2k3 + k4)}{6} \tag{2.9}$$

O método de quarta ordem Runge-Kutta é mais preciso do que o método progressivo de Euler, mas também é mais caro computacionalmente. O método é particularmente útil para EDOs com fortes não linearidades e para passos de tempo longos.

Em resumo, o método progressivo de Euler é um método simples e computacionalmente eficiente para resolver EDOs, enquanto o método de quarta ordem de Runge-Kutta é um método mais preciso, mas computacionalmente mais caro para resolver EDOs. A escolha do método depende dos requisitos de precisão, recursos computacionais e da EDO específica que está sendo resolvida.

A escolha do tamanho do passo de tempo, Δt, em simulações numéricas é importante para garantir a precisão e estabilidade da simulação. Um passo de tempo muito grande pode levar a instabilidade numérica, enquanto um passo de tempo muito pequeno pode levar a um custo computacional excessivo.

A escolha de Δt depende de vários fatores, incluindo a estabilidade do método numérico utilizado para resolver o problema determina o tamanho máximo do intervalo de tempo permitido. Em algumas aplicações, o método progressivo de Euler é estável apenas para pequenos intervalos de tempo. Também é necessário levar em conta a precisão desejada da simulação, que determinará o tamanho do intervalo de tempo mínimo permitido. O erro na simulação dependerá do tamanho do passo de tempo e a magnitude da solução muda ao longo do tempo.

Em geral, a escolha do parâmetro é um trade-off entre precisão e custo computacional, e o tamanho ideal do passo de tempo dependerá do problema específico a ser resolvido. Uma abordagem comum é iniciar com um tamanho de passo relativamente pequeno e então aumentá-lo gradativamente até que a simulação esteja estável e a precisão desejada seja alcançada. Alternativamente, pode-se usar um algoritmo de passo de tempo adaptativo, que ajusta o tamanho do passo de tempo em tempo real com base no comportamento da solução.

Em aplicações de advecção, $P \in \mathbb{R}^2$ representa modelos de fluxos na superfície, desconsiderando o transporte de volume. Enquanto modelagens com $S \in \mathbb{R}^3$ possuem uma dimensão vertical adicional, o que eleva ainda mais o custo de armazenamento. Neste trabalho, o foco principal concentra-se em modelos de

Podemos então formular um pseudo-algoritmo para o processo de advecção, que percorre todo o conjunto de partículas e atualiza-as de acordo com o passo estabelecido.

Algoritmo 1 Algoritmo de Advecção

1: Inicialize a posição inicial do conjunto de partículas $P = p_0, \ldots p_{n_i}$
2: **for all** $t = 0 \ldots n_t$ **do**
3: **for all** $i = 0 \ldots n_i$ **do**
4: Resolva a EDO $v(t, p) = \frac{dP}{dt}$
5: Atualize a posição da i-ésima partícula
6: Armazene $P^i(t)$ no conjunto de saída Out
7: **end for**
8: **end for**
9: **return** Out

A complexidade computacional do algoritmo de advecção é analisada por (Yenpure et al., 2022), no qual um modelo de custo é apresentado através da equação 2.6

$$Cost = \sum_{i=1}^{P} \sum_{j=0}^{N_i} (solve_i + \sum_{k=0}^{k=K} (locate_{i,j,k} + interp_{i,j,k}) + analyze_{i,j} + term_{i,j}) \tag{2.10}$$

$solve_i$ é o termo de custo da resolução da equação diferencial para cada partícula i, $locate_{i,j,k}$ é o custo de localizar e acessar a posição da partícula na posição k, $interp_{i,j.k}$ ao custo de interpolação espacial e temporal e por fim, $analyze_{i,j}$ e $term_{i,j}$ referem-se ao custo de analisar a saída e verificar o critério de parada do algoritmo.

Em termos de complexidade de tempo, o algoritmo de advecção depende da quantidade de passos necessários para atingir o critério de terminação e do número de partículas, sendo portanto $O(nt)$. Em relação a complexidade espacial, é necessário considerar número de dimensões do problema, além do componente temporal para armazenamento de diferentes campos, pode-se concluir que dado um domínio bi-dimensional, o algoritmo assume complexidade espacial $O(n^2t)$.

2.2.2 Discretização Espacial

A discretização temporal e espacial são essenciais em modelos de advecção de partículas porque permitem que o movimento contínuo de partículas em um fluido seja representado por uma série de etapas discretas. O tempo entre cada passo é referido como o passo de tempo, e o espaço entre cada posição é referido como o espaçamento da grade. A discretização é necessária pois na prática, modelos offline utilizam dados hidrodinâmicos que são armazenados de maneira discreta, isto é, campos vetoriais armazenados em períodos diferentes separados por espaçamentos temporais e espaciais.

A discretização temporal e espacial também afeta a precisão e a estabilidade das simulações de advecção de partículas, e uma consideração cuidadosa deve ser dada ao tamanho do intervalo de tempo e ao espaçamento da grade. Um intervalo de tempo muito grande pode resultar em erros significativos nas posições das partículas, enquanto um espaçamento de grade muito pequeno pode resultar em tempos de simulação muito longos. A escolha do intervalo de tempo e espaçamento da grade deve ser um equilíbrio entre precisão e eficiência computacional.

Inicialmente, discretiza-se o domínio espacial através de um conjunto de vetores de velocidade distribuídos pela região de interesse, atribuindo direção e magnitude a cada um, como visto na figura 2.4.

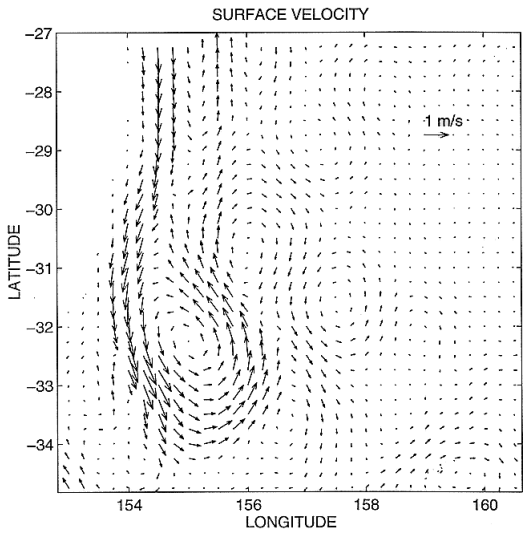

Figura 2.4: Exemplo de um campo vetorial discretizado, retirado de: `https://journals.ametsoc.org/view/journals/phoc/28/6/1520-0485_1998_028_1271_oarfco_2.0.co_2.xml`

Estes vetores são distribuídos através de um gradeamento (*gridding*), no qual características físicas contínuas são representadas no oceano por meio de células finitas. De acordo com (N et al., 2013), existem diferentes tipos de gradeamento comumente usados em aplicações oceanográficas e meterológicas, diferenciadas principalmente pela sua composição **estruturada** ou **desestruturada**. Grades estruturadas fornecem um domínio retangular e regular, fornecendo uma maneira conveniente de trabalhar com a malha de grandezas. Um exemplo pode ser visto na figura 2.5.

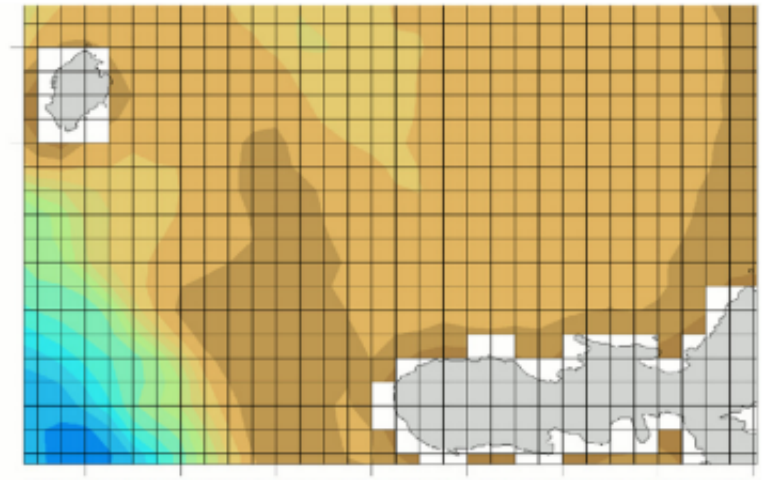

Figura 2.5: Exemplo de uma malha estruturada, retirado de (N et al., 2013)

Diferentes grades são comumente sub-categorizadas por meio da localização das grandezas físicas em cada célula. As mais utilizadas são as grades Arakawa (ARAKAWA e LAMB, 1977) de tipos A, B e C, ilustradas na figura (8). O tipo de grade influencia no método de interpolação, e aplicações específicas beneficiam-se de diferentes escolhas de grade, representado na figura 2.6.

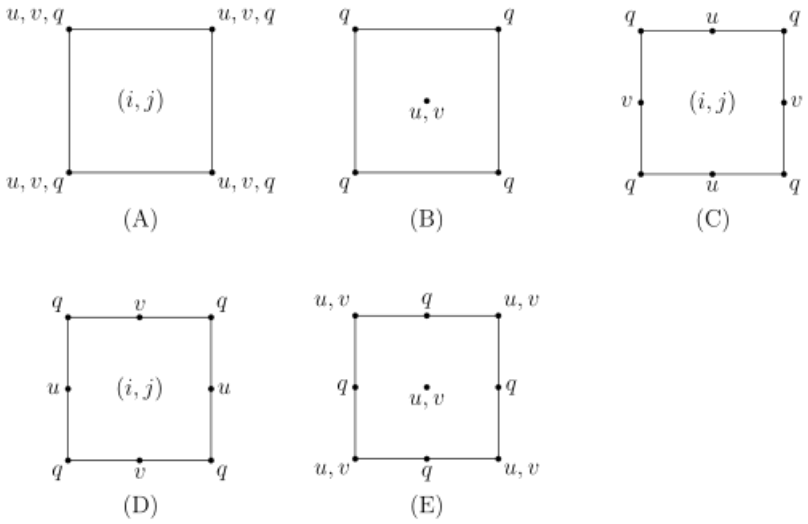

Figura 2.6: Grades Arakawa A, B, C, D e E.

Outro tipo de gradeamento utilizado é o não-estruturado, que procura subdividir o domínio em regiões não regulares, geralmente por meio de triângulos, ilustrado na figura 2.7. Este método ajuda na integração de costas através de uma granularidade maior da malha, no entanto, exige um custo de armazenamento maior e esquemas mais complexos de interpolação.

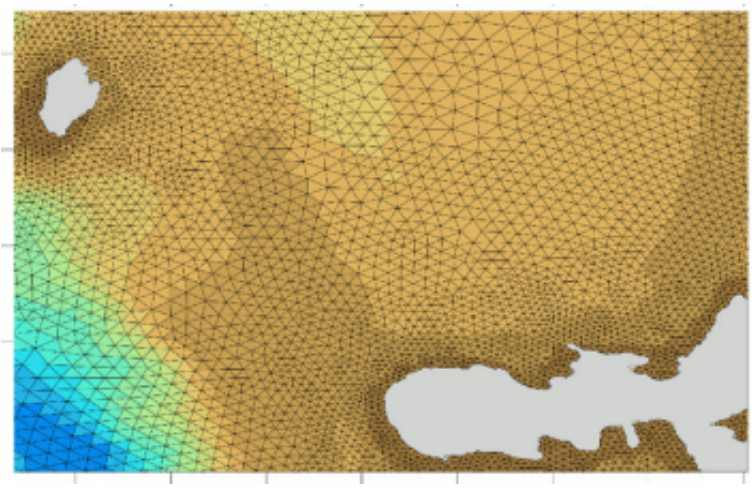

Figura 2.7: Exemplo de uma malha não estruturada, retirado de (N et al., 2013)

2.2.3 Discretização Temporal

O passo do algoritmo de advecção é um parâmetro importante na simulação, podemos observar um conjunto de partículas deslocar-se temporalmente por diferentes campos vetoriais. No entanto, dados vetoriais precisam ser armazenados em intervalos discretos, como dia ou semanas, no caso do famoso dataset Copernicus de análise e previsão física do oceano global 1/12°, cada campo vetorial é armazenado e registrado a cada hora, dia ou semana.

No contexto da análise lagrangiana off-line, a interpolação de tempo é um método usado para estimar a posição das partículas em tempos intermediários entre duas etapas de tempo em uma simulação numérica. O objetivo da interpolação de tempo é aumentar a resolução temporal da simulação, e capturar o transporte da partícula com maior precisão.

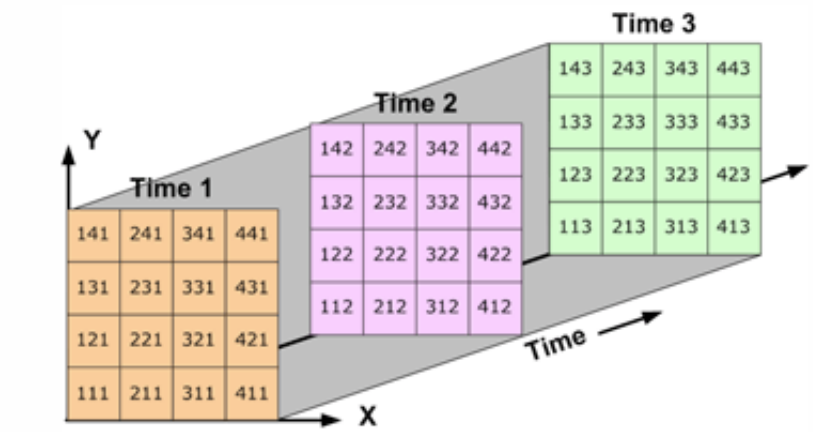

Figura 2.8: Exemplo de campo vetorial discretizado em três intervalos de tempo, retirado de (Documentation, 2018)

Existem diferentes métodos para interpolação de tempo na análise lagrangiana, incluindo interpolação linear, interpolação spline cúbica e interpolação polinomial. A escolha do método de interpolação depende da precisão desejada e do custo computacional da simulação.

A interpolação linear é o método mais simples e rápido, no qual a posição a posição da partícula é estimada como uma função linear do tempo entre dois passos de tempo. A interpolação spline cúbica é um método mais complexo que usa polinômios cúbicos para interpolar a posição da partícula. Este método é mais preciso do que a interpolação linear, mas também é mais caro computacionalmente.

A interpolação polinomial é outro método que usa uma função polinomial para interpolar a posição, que pode ser mais preciso do que a interpolação linear, mas também pode ser mais caro computacionalmente, dependendo do grau do polinômio usado.

Em geral, a precisão da interpolação de tempo na análise Lagrangiana off-line é limitada pela precisão da simulação numérica e pela resolução espacial e temporal do campo de velocidade usado na simulação.

2.3 GEOPROCESSAMENTO

O georreferenciamento e o geoprocessamento desempenham um papel importante na análise lagrangiana, que é um tipo de análise que envolve o rastreamento do movimento de partículas ou recursos ao longo do tempo (Hastings e Hill, 2009).

O georreferenciamento refere-se ao processo de anexar informações de referência espacial aos dados, como coordenadas ou um sistema de coordenadas geográficas. Esta informação é usada para colocar os dados em um local específico na superfície da Terra. No contexto da análise lagrangiana, o georreferenciamento é usado para anexar informações geográficas às partículas ou traços de recursos que estão sendo analisados. Isso permite que a análise leve em consideração as relações espaciais entre as partículas e o ambiente, como o efeito de ventos ou correntes no movimento das partículas.

Geoprocessamento refere-se a um conjunto de técnicas e processos usados para analisar e manipular dados geográficos. No contexto da análise lagrangiana, técnicas de geoprocessamento são usadas para analisar as partículas ou rastros de recursos e extrair informações significativas dos dados. Por exemplo, técnicas de geoprocessamento podem ser usadas para calcular a velocidade, direção e trajetória das partículas ao longo do tempo, ou para identificar áreas de alta concentração ou acumulação.

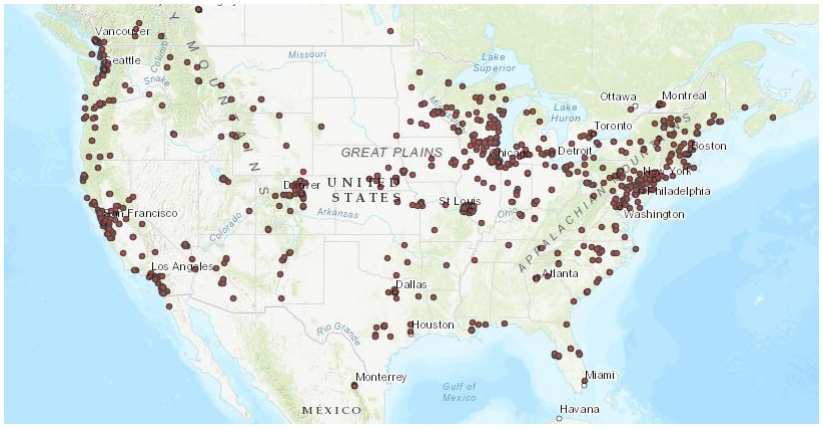

Figura 2.9: Exemplo de distribuição de cervejarias nos Estados Unidos. Retirado de `https://www.outalivepodcast.com/the-blog/gis-tool-hot-spot-analysis`

Em resumo, o georreferenciamento e o geoprocessamento desempenham um papel crucial na análise lagrangiana, fornecendo as informações de referência espacial necessárias e ferramentas de análise para rastrear e analisar o movimento de partículas ou recursos ao longo do tempo.

A base do georrefenciamento de pontos no globo sustenta-se sobre sistemas de coordenadas projetadas, que consiste em um sistema de coordenadas cartesianas bidimensionais usado para representar a superfície da Terra em um plano plano. É definido por uma transformação matemática da forma esférica ou elipsoidal da Terra em um plano bidimensional e fornece uma representação mais precisa de distâncias, formas e direções para uma região ou área de interesse específica.

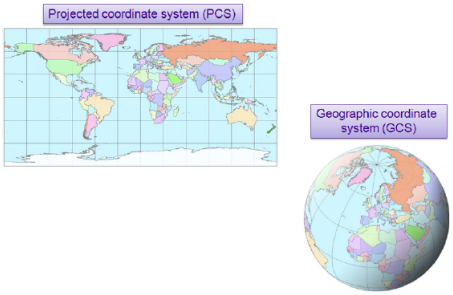

Figura 2.10: Exemplo de projeção de coordenadas esféricas para o sistema mercator, retirado de (Gross, 2004)

Os sistemas de coordenadas projetadas são usados em muitos campos diferentes, incluindo geografia, cartografia e GIS, pois permitem medições e análises exatas e precisas de dados geográficos. Ao projetar a superfície da Terra em um plano bidimensional, a superfície curva da Terra é transformada em um plano plano e as coordenadas resultantes podem ser usadas para realizar cálculos e analisar dados com mais facilidade e eficácia.

Existem muitos tipos diferentes de sistemas de coordenadas projetadas, cada um com suas próprias vantagens e desvantagens. A escolha do sistema de coordenadas projetado dependerá dos requisitos específicos da análise e da região de estudo. Alguns tipos comuns de

sistemas de coordenadas projetadas incluem a projeção Transversal de Mercator muito usada para mapear regiões com grande extensão norte-sul, como Reino Unido ou Japão e a projeção Lambert Conformal, usada para mapear regiões com extensão norte-sul moderada, como Estados Unidos ou Canadá.

Um sistema de coordenadas projetadas é um sistema de coordenadas cartesianas bidimensionais que fornece uma representação mais precisa de distâncias, formas e direções para uma região ou área de interesse específica. Existem muitos tipos diferentes de sistemas de coordenadas projetadas, cada um com suas próprias vantagens e desvantagens, e a escolha do sistema de coordenadas projetadas dependerá dos requisitos específicos da análise e da região que está sendo estudada.

É necessário incorporar informações de latitude e longitude nos modelos de advecção, para que a posição de cada partícula seja atualizada de maneira correta, além de converter unidades métricas como metros e kilômetros no contexto georreferenciado e na projeção escolhida. A conversão de pares de latitude e longitude para o sistema métrico (por exemplo, metros ou quilômetros) é um processo de duas etapas que envolve primeiro converter as coordenadas geográficas em um sistema de coordenadas projetadas e, em seguida, converter as coordenadas projetadas nas unidades métricas desejadas.

É importante observar que a precisão da conversão dependerá da projeção específica usada e da precisão dos dados geográficos. O processo de conversão pode apresentar erros e distorções, especialmente para grandes regiões ou para regiões com formas complexas, como litorais. Também é importante usar uma projeção apropriada que seja adequada para os requisitos específicos da análise e da região que está sendo estudada.

2.3.1 Formatos de Entrada e Saída

Para representar modelos hidrodinâmicos, o formato NetCDF (Network Common Data Form) é amplamente utilizado pela comunidade oceanográfica, que de acordo com a documentação oficial (documentation, 2021), é um conjunto de bibliotecas e formatos de dados que são independentes e autodescritivos, permitindo o compartilhamento, criação e acesso de dados científicos vetoriais.

Esses arquivos podem ser usados como entradas para modelos de advecção de partículas, que simulam o movimento de partículas em um fluido. Os modelos de advecção de partículas podem rastrear o movimento de poluentes, algas ou outras substâncias em oceanos, lagos ou fluxos atmosféricos. O formato NETCDF fornece uma maneira padronizada de armazenar e acessar grandes quantidades de dados gerados por simulações de advecção de partículas, facilitando o compartilhamento de dados e a colaboração em pesquisas. Além disso, os arquivos NETCDF podem ser lidos e processados por uma variedade de pacotes de software, tornando-o um formato flexível e interoperável para modelagem de advecção de partículas.

Um arquivo em formato NetCDF possui a extensão .nc, e contém todas as informações necessárias para descrever os dados que estão contidos nele através de um header, facilitando o

compartilhamento de informações entre diferentes pesquisadores. Modelos de circulação offline utilizam dados **.nc** como entrada para determinar o campo vetorial do algoritmo de advecção, que podem chegar a escalas de terabytes em muitas aplicações oceanográficas.

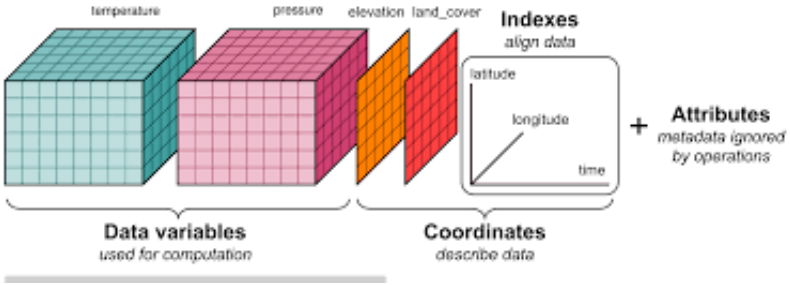

Figura 2.11: Representação do formato de dados de entrada de aplicações que utilizam NetCDF, retirado de (documentation, 2021)

Dados hidrodinâmicos de entrada para modelos são representados por dimensões e variáveis, de acordo com a convenção descrita em (cf conventions, 2022). Tempo, latitude e longitude são as dimensões mais comuns para este tipo de aplicação, também podendo contar com profundidade em casos tridimensionais. No caso das variáveis, velocidade, pressão e temperatura compõem o conjunto de vetores utilizados.

O padrão adotado pela comunidade científica para modelos de circulação é especificado em (Gross, 2004), e contém as especificações para a representação de dados vetoriais que incluem dados como a latitude e longitude, assim como variáveis de temperatura, pressão e velocidade, além de tipos suportados, nomeclaturas e terminologias.

O resultado da análise de circulação não possui uma convenção formal, e é geralmente representado por uma coleção de vetores de posições, associadas a cada partícula em um determinado instante de tempo da simulação. Apesar disso, muitas aplicações utilizam o formato Zarr para armazenar informações de maneira eficiente.

O formato .zarr é um formato de armazenamento de array hierárquico otimizado para arrays multidimensionais. Ele foi projetado para armazenamento, recuperação e manipulação rápidos e eficientes de arrays grandes e complexos, especialmente para arrays muito grandes para caber na memória. Neste formato uma matriz é dividida em partes menores e cada parte é armazenada como um arquivo separado. Isso permite o carregamento parcial de arrays e o processamento paralelo de pedaços de array. A estrutura hierárquica do formato também permite compactação e codificação eficientes de dados de matriz, o que pode resultar em tamanhos de arquivo menores e leitura e gravação mais rápidas de dados de matriz.

O formato .zarr também fornece uma API flexível e extensível para trabalhar com vetores e é suportado por várias bibliotecas populares de computação científica, incluindo xarray e dask. Isso facilita o uso de dados vetoriais em uma variedade de aplicações e fluxos de trabalho, além da intercomunicação com outros formatos de dados e sistemas de armazenamento.

2.4 PARALELIZAÇÃO E DESAFIOS COMPUTACIONAIS

Algoritmos de aplicações cientifícas que envolvem o cálculo de comportamento de um grande número de partículas podem ser facilmente paralelizados, através da distribuição da computação de um grupo de partículas em diferentes processadores. Para aplicações de modelagem ambientais isso traz uma grande vantagem em termos de eficiência, pois permite reduzir o custo computacional significativamente,

Aplicações oceanográficas e atmosféricas são amplamente beneficiadas pelo paralelismo, por se tratarem de modelagens numéricas com alta carga computacional a longos períodos de tempo. A motivação para a implementação de algoritmos paralelos é naturalmente a incapacidade de aumentar indefinidamente a frequência e manter um consumo de energia razoável de um único processador (Trobec et al., 2018), além da crescente tendência de sistemas distribúidos e computação em GPU. Existem um grande número de estratégias para paralelizar o algoritmo de advecção em modelos offline, cada uma com características diferentes que são apropriadas para problemas diferentes. Nesta seção, estratégias de paralelismo de memória compartilhada são explorados.

O paralelismo de memória compartilhada é um tipo de computação paralela em que vários núcleos de processamento compartilham o acesso a um pool comum de memória. Isso permite que cada núcleo acesse e modifique os dados armazenados na memória, levando a cálculos mais eficientes e rápidos. A principal vantagem do paralelismo de memória compartilhada é que ele elimina a necessidade de comunicação explícita de dados entre os núcleos, facilitando a programação e a depuração.

2.4.1 Paralelismo de Memória Compartilhada

MPI (Message Passing Interface) é uma biblioteca que desempenha um papel crucial no paralelismo de memória compartilhada. O MPI fornece uma interface padronizada para comunicação entre núcleos de processamento em um sistema de memória compartilhada. O MPI permite a distribuição de dados e tarefas computacionais em diversos núcleos, permitindo o processamento paralelo de simulações em larga escala. Em um sistema paralelo de memória compartilhada que utiliza MPI, cada núcleo pode executar sua própria instância de um programa, e a biblioteca fornece uma maneira de os núcleos se comunicarem e trocarem dados. O MPI é amplamente utilizado em muitas aplicações científicas e de engenharia, e sua versatilidade e portabilidade o tornam uma escolha popular para paralelismo de memória compartilhada.

Yenpure et al. (2022) afirma que no caso do cálculo de trajetórias sobre um modelo de circulação, poderiamos assinalar a cada partícula i uma posição $X^i(t)$, e assinalar um processador para uma região do domínio vetorial, cada processador portanto estaria responsável por computar a trajetória de um grupo de párticulas em uma porção do espaço. Isso é feito para que modelos offline sejam mais eficientes em relação ao acesso a memória, facilitado pelo acesso contínuo baseado em “chunks” do campo vetorial. Esta abordagem é denominada de *paralelismo sobre os*

dados, e elimina a necessidade da constante leitura de novos chunks do domínio. No entanto, é necessário que um número considerável de processadores esteja disponível para que o domínio seja dividido de maneira a não ocupar espaços significativos em cada um deles, já que campos hidrodinâmicos tendem a ocupar grandes espaços. Além disso, é possível que uma região receba mais partículas que outras, causando um desbalanceamento de carga entre os processos.

Outra abordagem é o *paralelismo sobre partículas*, no qual assinala-se a um grupo de partículas um determinado processo baseado em algum critério, e cada processo atualiza o passo de advecção do seu grupo. Esta abordagem é dinâmica e carrega chunks do domínio conforme são necessários, a medida em que a computação ocorre. O problema desta abordagem é a possibilidade de um espalhamento muito grande de um conjunto de partículas sobre o domínio, fazendo com que processos carreguem grandes quantidades de memória para realizar o passo de advecção.

O grande problema em aplicações oceanográficas é a necessidade de constante sincronia do conjunto global de partículas, devido a possíveis interações entre partículas ou pelo campo vetorial definidas em tempo de compilação. Não é possível portanto que partículas sejam individualmente dispersadas em um intervalo diferente do global, pois prejudicaria a legitimidade da simulação.

2.4.2 Balanceamento de Carga

Idealmente, cada processador acessaria apenas um conjunto de chunks do campo para atualizar a posição do grupo de partículas, minimizando seu acesso a regiões fora do seu alcance inicial. No entanto, o problema de circulação de partículas é extremamente imprevisível e caótico, por se tratar de um sistema altamente não linear. Além disso, comportamentos adicionais podem ser inclusos para aplicações específicas, acrescentando mais uma camada de complexidade sobre o modelo, como interação entre partícula-partícula e partícula-campo.

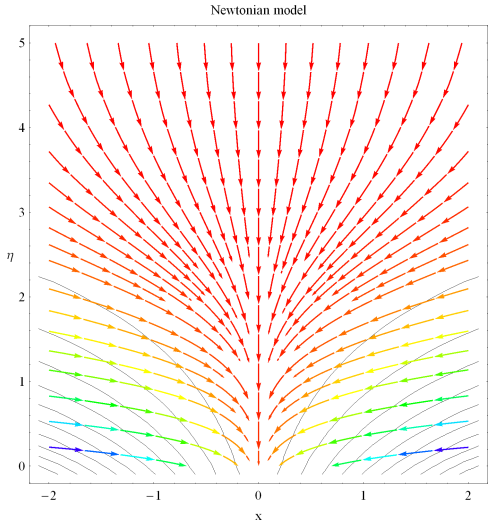

Figura 2.12: Representação gráfica de um vórtice que atrai partículas, retirado de (Versteeg e Malalasekera, 1995)

Campos vetoriais também apresentam regiões de atração e dispersão local, conhecidos como *sinks* e *sources*, são regiões que atraem ou dispersam partículas e acabam formando conjuntos densamente ou esparsamente povoados. O comportamento de cada campo em particulas pode afetar o desempenho de operações de leitura, fazendo com que existam regiões de maior concentração e amplas regiões esparsas. Naturalmente, não é possível prever quais regiões do dominio cada processador deverá acessar em tempo de execução, e é possível que processadores distintos com o mesmo número de partículas carreguem em memória uma porção do domínio muito diferente.

Um desbalanceamento de carga do algoritmo de advecção ocorre como resultado deste comportamento imprevisível. Este fenômeno interfere na performance e como consequência, na escalabilidade do código em relação a entrada, fazendo com que seja necessário construir um esquema de redistribuição de carga computacional para um melhor desempenho.

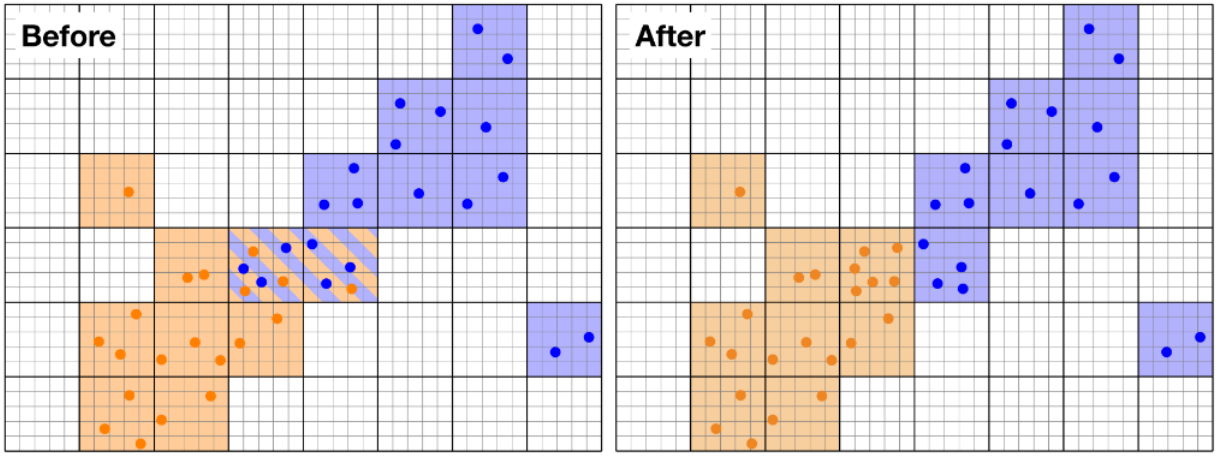

Figura 2.13: Ilustração de um possível desbalanceamento de carga, partículas em laranja e azul são representadas por processadores diferentes na esquerda, enquanto um possível balanceamento é representado na direita. Retirado de: `https://nbviewer.org/github/OceanParcels/parcels/blob/master/parcels/examples/documentation_MPI.ipynb`

Este desbalanceamento de carga deve ser resolvido a medida em que a computação ocorre, diferentemente de técnicas que utilizam-se de implementações prévias para balancear a carga computacional, que são denominadas 'estáticas' e são utilizadas como etapa de pré-processamento da modelagem. O balanceamento de carga dinâmico é portanto muito mais delicado, pois deve-se otimizar o equilibrio entre a carga total assinalada para cada processador sem incluir um custo computacional elevado, afim de não criar um novo gargalo na performance. Deve-se portanto manter em mente todos estes fatores para que um esquema eficiente de balanceamento seja estabelecido, minimizando a comunicação inter-processos e o cache-miss de cada um deles.

Uma série de estratégias podem ser empregadas para o balanceamento dinâmico em modelagem computacional de sistemas físicos, (Hendrickson e Devine, 2000) indica que existem duas grandes classes de algoritmos projetados para resolver este problema. A primeira procura explorar as interações entre os diferentes elementos do modelo, descrevendo um grafo de conectividade ao longo do tempo, que é denominada de abordagem *Topológica*. A segunda e

mais adequada para o problema, é caracterizada pela divisão do dominio e o aproveitamento da localização dos objetos na simulação, denominada abordagem *Geométrica*.

2.5 K-MEANS E CLUSTERIZAÇÃO ESPACIAL

O algoritmo k-means é uma técnica popular de agrupamento usada em aprendizado de máquina e mineração de dados. Ele particiona um conjunto de n pontos de dados em k clusters, onde k é um parâmetro especificado pelo usuário. O algoritmo funciona atribuindo iterativamente cada ponto de dados ao centróide do cluster mais próximo e, em seguida, atualizando o centróide do cluster para a média dos pontos atribuídos a ele. O processo continua até que os centróides do cluster não mudem mais ou um número máximo de iterações seja atingido. Os clusters resultantes representam regiões compactas e bem separadas no espaço de dados.

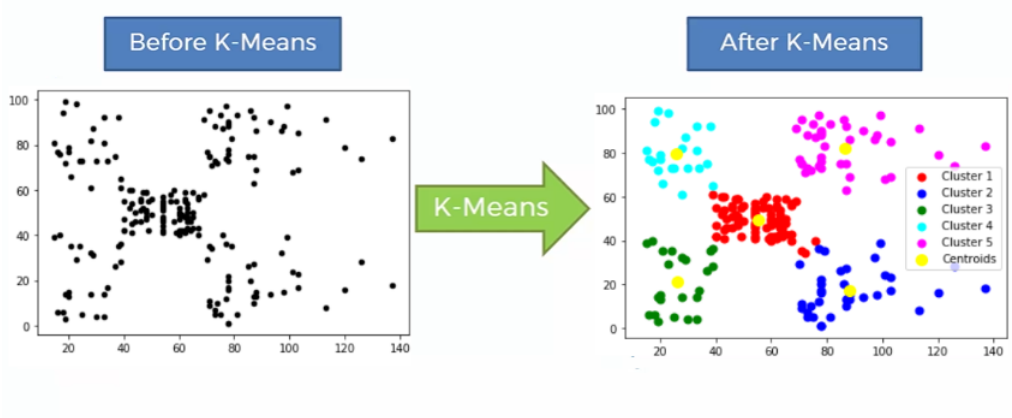

Figura 2.14: Representação visual da clusterização via k-means, retirado de (McLoughlin et al., 2009)

Em termos de complexidade de tempo, o algoritmo k-means tem uma complexidade de tempo média de O(knId), onde I é o número de iterações, d é o número de dimensões e n é o número de pontos de dados. A complexidade espacial do algoritmo k-means é O(nd), pois requer armazenamento para os pontos de dados e os centróides do cluster. O algoritmo é altamente dependente da escolha inicial dos centróides e às vezes pode convergir para soluções subótimas, portanto, várias inicializações aleatórias são frequentemente realizadas para reduzir a chance de obter uma solução ruim. Apesar de suas limitações, o algoritmo k-means é um método amplamente utilizado e bem compreendido para agrupamento e análise de dados.

K-means é comumente usado em agrupamento de dados espaciais, que envolve o particionamento de dados geográficos em grupos de recursos semelhantes. Nesse contexto, os pontos de dados podem ser representados por suas coordenadas espaciais, como latitude e longitude, e o algoritmo k-means pode ser usado para identificar agrupamentos de pontos espacialmente próximos uns dos outros. O algoritmo atualiza iterativamente os centróides dos clusters, levando em consideração tanto a localização espacial dos pontos de dados quanto sua similaridade com relação a outras variáveis, como características demográficas ou padrões de uso da terra. O resultado do algoritmo k-means é uma partição dos dados em k clusters, cada

um representado por seu centróide, que pode ser usado para identificar padrões espaciais e fazer inferências sobre a geografia subjacente.

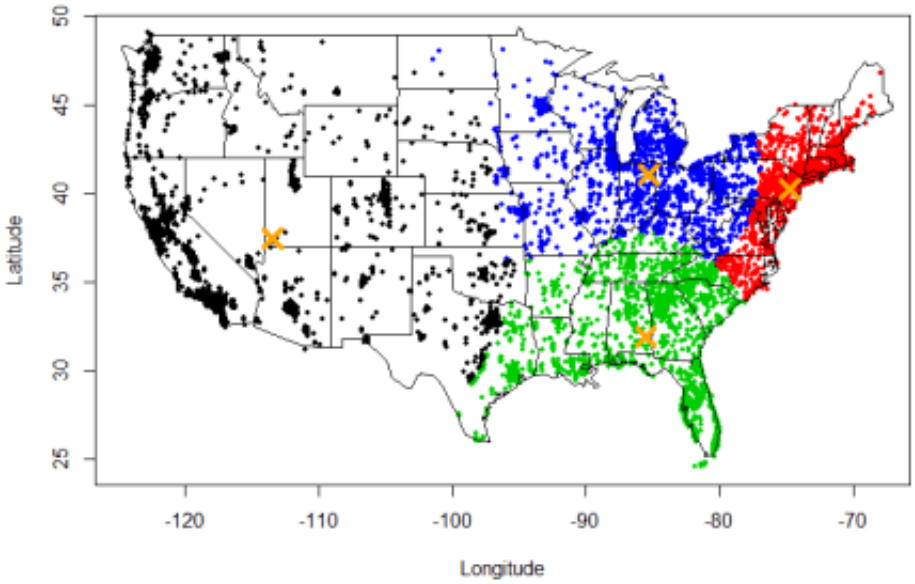

Figura 2.15: Clusterização K-means aplicada a um problema georeferenciado, retirado de (Versteeg e Malalasekera, 1995)

No agrupamento de dados espaciais, o algoritmo k-means pode ser estendido para incorporar informações espaciais de várias maneiras, como ponderar a distância entre pontos de dados com base em sua proximidade espacial ou usar estruturas de dados espaciais para acelerar o processo de agrupamento. A escolha da extensão dependerá do problema específico de agrupamento de dados espaciais e das questões de pesquisa abordadas. Apesar de sua simplicidade, o algoritmo k-means continua sendo uma ferramenta amplamente utilizada para explorar e entender padrões espaciais em dados geográficos.

2.6 CONCLUSÃO

Neste capítulo uma introdução ao campo da modelagem lagrangiana foi apresentado. Uma introdução a fundamentação teórica do algoritmo de advecção, discretizações e padrões de entrada e saída foram expostos na seção 2.1, 2.2 e 2.3. Conceitos de geoprocessamento, clusterização e a introdução do paralelismo baseado em proximidade foram descritos para estabelecer uma conexão entre o processamento de alto desempenho e maneiras de identificar partículas de plástico no oceano nas seções 2.4, 2.5 e 2.6.

3 ESTADO DA ARTE E TRABALHOS RELACIONADOS

Uma quantidade de bibliotecas e modelos de partículas virtuais foram desenvolvidas com o intuito de modelar fenômenos de transporte no oceano e na atmosfera, atribuindo formatos de saída e entrada específicos de acordo com a necessidade dos desenvolvedores. O conjunto de pacotes de software para análises lagrangianas de circulação pode ser divido em dois grandes grupos.

Este capítulo detalha e identifica os dois grandes grupos mais utilizados pela comunidade científica, exibindo pontos positivos e negativos de cada um deles. Também introduzimos o framework de maior interesse e o utilizado para a experimentação de poluição de plásticos, Parcels do projeto OceanParcels é relativamente recente e conta com diversas características positivas para o problema de dispersão de plásticos.

3.1 MODELOS ONLINE

O conjunto de modelos online disponíveis é atrelado a modelos específicos com formatos únicos, no entanto, permitem a computação online do modelo, e portanto acabam gerando simulações com uma maior precisão.

MITgcm, HYCOM, NEMO, ROMS e MPAS-O são todos pacotes ou bibliotecas usadas na modelagem oceânica.

O MITgcm (Massachusetts Institute of Technology General Circulation Model) é um modelo numérico que simula a circulação oceânica, incluindo sua dinâmica e termodinâmica em escala global e regional. Foi desenvolvido pelo MIT como uma das primeiras grandes bibliotecas destinadas para este tipo de aplicação (Marotzke et al., 1999).

HYCOM (Hybrid Coordinate Ocean Model) é um modelo híbrido que combina um esquema numérico semi-implícito com um sistema de coordenadas de profundidade multicamadas, resultando em melhor resolução vertical e precisão. Foi desenvolvido por um esforço multi-institucional patrocinado pelo programa nacional de parceria oceânical (National Ocean Partnership Program), como parte do programa experimental de assimilação de dados ocêanicos globais dos Estados Unidos da América (Program, 2022).

O NEMO (Nucleus for European Modeling of the Ocean) constitui um conjunto de pacotes que busca modelar de maneira flexível e orientado para a comunidade que suporta uma ampla gama de configurações e aplicações, incluindo escalas regionais e globais e escalas de tempo multidecadais a sazonais. O pacote conta com três componentes principais, um modelo padrão de termodinâmica e de resolução de equações primitivas NEMO-OCE (Group, 2018), um modelo de termodinâmica exclusivamente dedicado a interações de geleiras e oceanos NEMO-ICE (Gurvan et al., 2019) e um modelo offline de fenômenos biogeoquímicos NEMO-

TOP-PISCES (Gurvan et al., 2022). Ele foi desenvolvido por um consórcio europeu com 5 intitutos de preservação e estudo de fenômenos marinhos.

O ROMS (Regional Ocean Modeling System) é um framework oceânico regional que simula a circulação, as ondas e a qualidade da água em águas costeiras e de plataforma. Foi desenvolvido pelo grupo DMCS de modelagem oceânica, (Shchepetkin e McWilliams, 2005) como uma maneira de incorporar elementos de alto desempenho para simulações.

O MPAS-O (Massive-Parallel Ocean Model) é um modelo oceânico de malha não estruturada baseado em componentes que suporta simulações paralelas em larga escala em plataformas de supercomputação modernas. O pacote busca melhorar a escalabilidade do problema, com esquemas de paralelização que permitem simulações mais rápidas e estáveis (Kang et al., 2021).

3.2 MODELOS OFFLINE

O primeiro conjunto constitui pacotes que possuem comunidades amplas open-source e que oferecem a possibilidade de executar modelos offline, tendo como entrada um arquivo NETCDF ou algum outro padrão. Neste conjunto existem quatro grandes pacotes construídos ao longo dos anos, Ariane (d'Océanographie Physique et Spatiale , LOPS), CMS(Connectivity Modelling System), TRACMASS e Parcels. Este conjunto é independente do modelo e possui uma maior flexibilidade, possibilitando um maior controle sobre a análise.

Ariane (Bruno Blanke, 2012), TRACMASS (Döös et al., 2013) e Connectivity Modeling System (CMS) são ferramentas de software usadas para modelar a advecção de partículas em fluxos de fluidos. No entanto, cada ferramenta tem suas próprias características e pontos fortes que a diferenciam das outras. O Ariane é conhecido por sua interface amigável e flexibilidade, tornando-o uma escolha popular para simulações de advecção de partículas. O TRACMASS foi projetado especificamente para o estudo das correntes oceânicas e é otimizado para simulações em larga escala, com foco em computação eficiente e processamento de dados. O CMS, por outro lado, é uma estrutura modular que fornece um conjunto abrangente de ferramentas para simular e analisar o movimento de partículas em sistemas complexos do mundo real, com ênfase particular na conectividade entre diferentes regiões e habitats. Cada uma dessas ferramentas atende a uma necessidade diferente, e a escolha do software por um cientista dependerá de seus requisitos e objetivos de pesquisa específicos.

3.3 PARCELS

O framework de maior interesse recente é o software Parcels (Lange e van Sebille, 2017) (Delandmeter e Sebille, 2019), que constitui um pacote Python com diferentes módulos para calcular trajetórias de partículas lagrangianas. A característica principal do pacote é a flexibilidade oferecida, que permite um grande controle sobre o comportamento do conjunto de partículas de maneira simples e declarativa. Além disso, sua interface mais amigável e

conjunto restrito de dependências facilita o desenvolvimento de novos modelos, permitindo acoplar bibliotecas utéis para análise dos resultados, como **matplotlib** e **sklearn**, sendo como único requisito o interpretador python.

Parcels ("Probably A Really Computationally Efficient Lagrangian Simulator") foi desenvolvido desde o início com escalabilidade e desempenho em mente pelo projeto OceanParcels, e é altamente modular com alta capacidade de configuração de experimentos. Além disso, existe um grande esforço do projeto em possibilitar a customização de kernels e possibilidades de diferentes comportamentos de partículas, como a fragmentação e o afundamento de partículas definidas pelo usuário, algo relativamente novo em relação aos frameworks e bibliotecas previamente mencionados.

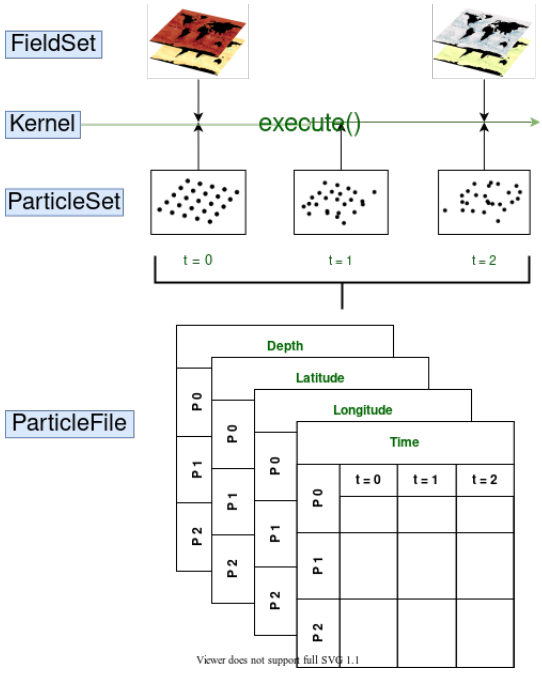

Figura 3.1: Diagrama de uso do framework Parcels, retirado de `https://oceanparcels.org/`

A figura 3.1 indica a arquitetura do framework, que consiste de um **FieldSet** com dados vetoriais e restrições locais e temporais, como tamanho do passo de advecção e região do domínio. O fieldset geralmente lê um arquivo NetCDF e armazena os componentes de velocidade horizontais e verticais, além da possibilidade de outros campos como profundidade, compondo a estrutura de dados de acesso dinâmica do framework. É nesta estrutura que são definidos o padrão de acesso as partículas sobre o campo, através de callbacks que deverão ser instanciadas em determinados estados pré-definidos de párticulas, como no caso de partículas estarem fora de alcance do domínio e outros erros em tempo de execução.

O conjunto de partículas é definido por um **ParticleSet**, com posições iniciais georeferenciadas, além dos esquemas de interpolação que serão utilizados na execução e a qual Fieldset este conjunto está atrelado. O conjunto contém todas as informações de cada partícula, como posição e estado em cada iteração do algoritmo.

Por fim, uma estrutura **Kernel** é definida como o conjunto de operações que deverá ser conduzido sobre o conjunto de partículas e sua interação com o campo, podendo incluir não somente deslocamento, mas também interações com outros campos como temperatura e pressão, interação entre partículas e partículas-campo. Existem dois modos de execução disponiveis, por padrão o modo Scipy e Um modo Python puro que utiliza objetos interpoladores fornecidos pelo pacote Scientific Python (SciPy) para executar a interpolação de dados de campo. Este modo destina-se principalmente como um opção de depuração devido à penalidade de desempenho da execução kernels no próprio interpretador Python

Modo JIT A geração de código em tempo de execução e a compilação just-in-time (JIT) são utilizadas para gerar código C de baixo nível que realiza a atualização do estado da partícula e a interpolação dos dados de campo. O mecanismo de geração de código traduz principalmente um subconjunto restrito da linguagem Python em código C equivalente, enquanto um conjunto de módulos utilitários fornece funcionalidade auxiliar, como número aleatório geração ou utilitários matemáticos (math.h).

A compilação Just-in-Time (JIT) é uma técnica usada na programação de computadores para compilar dinamicamente o código do programa durante o tempo de execução, em vez de antes do tempo (AOT). O principal objetivo do JIT é melhorar o desempenho de linguagens dinâmicas, como Python, Ruby e JavaScript, que são tradicionalmente mais lentas do que linguagens estaticamente tipadas, como C e Java.

O JIT funciona convertendo o bytecode, que é uma representação intermediária do código do programa, em código de máquina logo antes de o programa ser executado. Isso permite que o compilador JIT leve em consideração as informações sobre o ambiente de tempo de execução e otimize o código gerado para o hardware específico e o sistema operacional em que está sendo executado. O compilador JIT também pode executar otimizações mais agressivas, como funções inlining, remoção de código morto e desenrolar loops, que não são possíveis em compiladores AOT.

A compilação JIT pode melhorar significativamente o desempenho de programas de linguagem dinâmica. Em alguns casos, o aumento de velocidade pode ser da ordem de várias vezes em comparação com a execução interpretada. No entanto, a compilação JIT também pode aumentar o tempo de inicialização de um programa, pois o compilador precisa de tempo para analisar o código e gerar o código de máquina otimizado. Além disso, os compiladores JIT podem consumir quantidades significativas de memória e tempo de CPU, o que pode levar a pausas mais longas durante a execução do programa.

A possibilidade de customização é extremamente atraente para aplicações que buscam modelar comportamentos de afundamento e dispersão de micro-plásticos, por permitirem um conjunto de ferramentas que alteram parâmetros e definem comportamentos complexos como o afundamento por ondas, fragmentação por raios UV e biodegradação do material.

O resultado final de toda modelagem é armazenado em uma estrutura **ParticleFile** que contém toda informação histórica individual de cada partícula ao longo da simulação. O

framework também conta com ferramentas de análise e processamento de saídas, que permitem visualizar o fluxo em formato gráfico como **.gif**, e analisar a trajetória georeferenciada de cada partícula.

O projeto Parcels também conta com uma implementação MPI, que é feita através de intercomunicação de diferentes processos, no qual cada um conta com um ParticleSet distinto, dividindo o trabalho total da simulação. Inicialmente, cada conjunto é divido pelo algoritmo K-means descrito na seção 2.5, com o intuito de minimizar a área de cada cluster por número de partículas. Esta estratégia crê que partículas que iniciam próximas uma das outras acabam se deslocando para regiões relativamente próximas conforme a simulação anda, e portanto, padrões de acesso de memória são otimizados.

3.4 CONCLUSÃO

Neste capítulo foram descritos alguns dos grandes pacotes, bibliotecas e frameworks utilizados pela comunidade científica para a modelagem de sistemas oceânicos. Na seção 3.1 modelos online baseados em computação simultânea da hidrodinâmica foram descritos. Modelos offline mais comumente utilizados foram apresentados na seção 3.2, e finalmente, o framework de maior interesse recente Parcels foi descrito em detalhe.

4 PROPOSTA

Neste capítulo, introduzimos os experimentos realizados com o framework Parcels, a partir de uma configuração definida através de potencial cenário de poluição no litoral paranaense. O objetivo destes experimentos foi colocar em prática a fundamentação teórica e entender como simulações são feitas.

4.1 EXPERIMENTO

Com o framework Parcels, uma simulação foi construída com cerca de 1 milhão de partículas quye foram dispersadas no oceano atlântico por cerca de quatro anos e meio, totalizando 2000 dias com profundidade constante. Todos os experimentos foram executados de maneira paralela através da configuração MPI, com a chamada `mpirun` em cada instância para avaliar a escalabilidade e eficiência da implementação.

Os dados hidrodinâmicos foram retirados do projeto Copernicus, que promove datasets relacionados a climatologia e oceanografia. O dataset "Global Ocean 1/12° Physics Analysis and Forecast updated Daily" inclui arquivos médios diários e mensais de temperatura, salinidade, correntes e nível do mar. Também inclui campos de superfície média horária para altura do nível do mar, temperatura e correntes. Os arquivos de saída do oceano global são exibidos com uma resolução horizontal de 1/12 graus com projeção equiretangular de longitude/latitude regular. Através de uma API de requisição, os dados atualizados a cada hora foram baixados no período de um mês, entre os dias 01/01/2022 e 01/02/2022 de velocidades longitudinais e latitudinais, com profundidade constante. O dataset conta com grids espaçadas de 0.083° × 0.083° e é atualizado desde 29/11/2020, tem limites latitudinais de -90° a 90°, e longitudinais de -180° a 180°, cobrindo o globo terrestre inteiro.

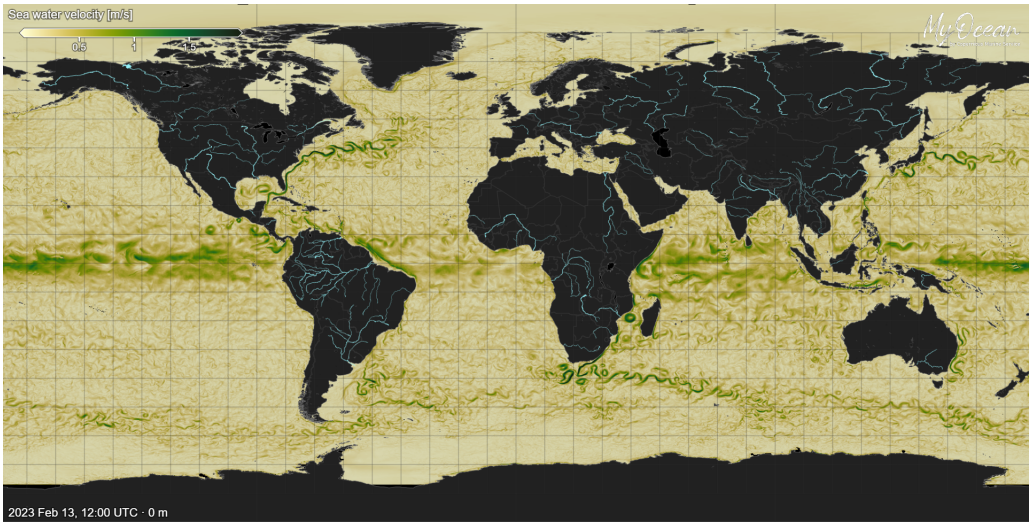

Figura 4.1: Representação visual do dataset "Copernicus Global Ocean Physics Analysis and Forecast updated Daily"', retirado de `https://data.marine.copernicus.eu/product/GLOBAL_ANALYSISFORECAST_PHY_001_024/description`

Apesar da faixa de tempo extraída não ser completamente fiel ao contexto global, tornou-se inviável a requisição de dados com períodos de tempo mais longos, por tratarem-se de arquivos NetCDF com Terabytes de dados. Desta forma, optou-se por adquirir um conjunto de dados curto e ciclicamente repetí-lo a medida em que os limites da dimensão temporal são ultrapassados.

Mesmo com uma granularidade maior, o dataset é global e não restringe a dispersão por uma única região, permitindo visualizar a conectividade dos oceanos atlântico com o pacífico e índico. É importante destacar que áreas não identificadas como oceano são classificadas com valores inválidos no arquivo NetCDF, que restringe a advecção somente a regiões marinhas. A escolha por um dataset maior foi motivada pelos esquemas únicos de interpolação do Parcels, que auxilia em uma representação mais fiel com menos informações.

4.1.1 Configuração Parcels

O dataset foi carregado para um FieldSet através do método de leitura de arquivos NetCDF , explicitando quais dimensões e variáveis serão incorporadas na execução do modelo, neste caso apenas as variáveis de velocidade vertical `U` e horizontal `V`. O chunkSize de leitura máximo é definido, além da flag `time_periodic` que permite ultrapassar o limite de 30 dias e realizar a simulação pelo tempo desejado.

```
1 def set_field(file, cs):
2   variables = {'U': 'uo', 'V': 'vo'}
3   dimensions = {'lon': 'longitude', 'lat': 'latitude', 'time': 'time'}
4   cs = {'time': ('time',1), 'lat': ('latitude',cs), 'lon': ('longitude',cs)}
5   return FieldSet.from_netcdf(file,
6                               variables,
7                               dimensions,
```

```
                        time_periodic=delta.time(days=30),
                        chunksize=cs)
```

Um ParticleSet foi inicializado com o número desejado de partículas, e o tipo de partícula da simulação. Em seguida, uma região de interesse foi demarcada através de coordenadas próximas ao litoral paranense, e populacionada com uma lista de coordenadas georeferenciadas criada de maneira aleatória, buscando distribuir o domínio de interesse.

```
latRange = (-26.2027, -25.4721)
lonRange = (-47.5966, -46.742)
lat, lon = get_rand_seed(latRange, lonRange, N_PARTICLES)
fieldset = set_field(FILEPATH, CHUNKSIZE)

pset = ParticleSet.from_list(fieldset=fieldset,
                        pclass=JITParticle,
                        lat=lat,
                        lon=lon)
```

O kernel de advecção é definido como a resolução da EDO por Runge-Kutta de quarta ordem, além de incorporar o kernel de `recovery`, que define como erros de execução devem ser tratados pelo programa.

```
    pset.execute(AdvectionRK4,
                runtime=delta(days=2000),
                dt=delta(days=1),
                output_file=output,
                recovery={ErrorCode.ErrorOutOfBounds: DeleteParticle})
```

Com a chamada `execute` da classe, a simulação teve ínicio e os resultados foram armazenados, procurando acompanhar a posição das partículas a medida do tempo, o tempo decorrido e a quantidade de memória gasta por cada processo MPI.

5 EXPERIMENTOS E VALIDAÇÃO

5.1 CONFIGURAÇÃO

Os experimentos foram realizados em um desktop com sistema operacional Ubuntu 20.04.2 LTS, com versão de kernel do Linux 5.4.0-137-generic. A arquitetura da configuração conta com uma CPU Intel i5-8400 com arquitetura x86_64, com 6 cores e 1 thread por core, com clock base de 800Mhz e máximo de 4.0Ghz. A disposição da hierarquia de memória conta com Cache L1d de 192Kib, cache L1i de 192Kib, Cache L2 de 1.5Mib, Cache L3 de 9Mib, RAM de 16Gb com frequência de 2444MHz DDR4 e um disco rígido HD.

Com a simulação em execução, procuramos também compreender o papel do paralelismo e quais potenciais gargalos existem no esquema de balanceamento de carga, desta forma, também foram registrados o uso total de memória e tempo decorrido para cada número de processos. Neste caso, um número variado de partículas foi adveccionado para testes de escalabilidade.

5.2 RESULTADOS

O ínicio da simulação é marcada pela presença das partículas na região de interesse, no atlântico sul próximo ao litoral paranaense e paulista.

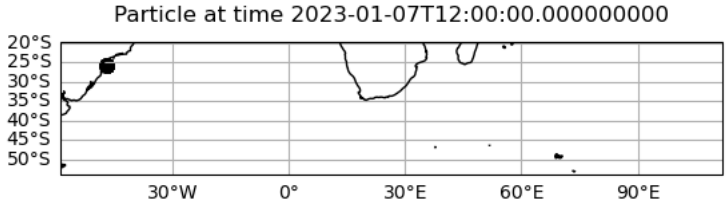

Figura 5.1: Inicio da simulação
fonte: Autoria própria

Após apenas 6 meses, é possível observar um grande deslocamento em direção próxima da costa sul argentina.

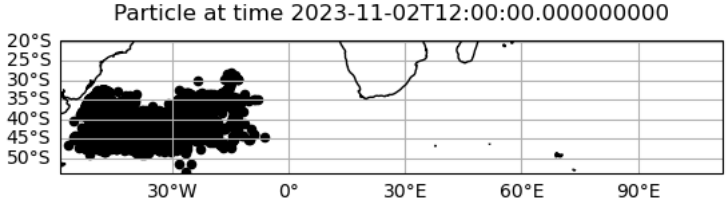

Figura 5.2: Simulação após 6 meses
fonte: Autoria própria

Em uma escala de 1 a 3 anos, o grupo se desloca até o oceano índico, aproximando-se da áfrica do sul e madagascar.

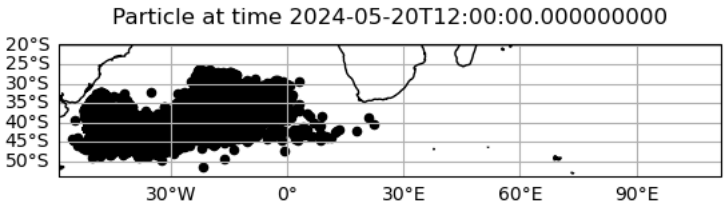

Figura 5.3: Simulação após 17 meses
fonte: Autoria própria

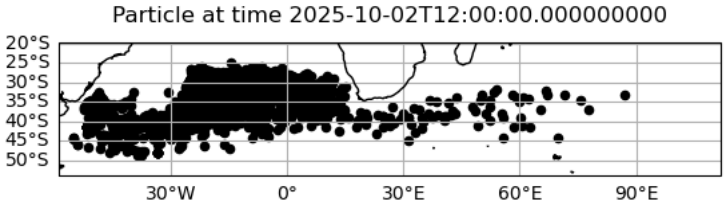

Figura 5.4: Simulação após 30 meses
fonte: Autoria própria

A simulação acaba com um grande aglomerado entre a porção sul da áfrica e da américa do sul, com um número alto de partículas flutuando sobre o oceano atlântico. No entanto, um número considerável deslocou-se até porções bastante distantes do oceano índico, marcando presença em regiões costeiras de madagacasr e mozambique. É provável que com uma previsão ainda mais distante, este conjunto passe a alcançar regiões como indonésia e índia.

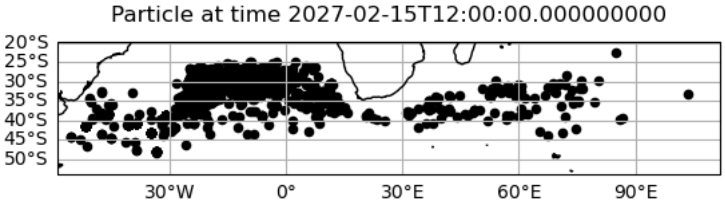

Figura 5.5: Simulação após 4 anos
fonte: Autoria própria

Um fato bastante curioso é a ausência de partículas que permaneceram na região no qual foram soltas, indicando que quase nenhuma teve como destino praias do litoral paranaense. Isso pode ter sido decorrido do fato de que a posição inicial escolhida fica a alguns kilômetros de distância do litoral. Outra possibilidade é a parametrização escolhida para o experimento, que pode afetar como regiões costeiras interagem com a malha de partículas.

O benchmark de performance foi feito com um número variado de partículas na mesma configuração descrita anteriormente. Buscou-se verificar como a separação de diferentes ParticleSets em processos MPI distintos poderia melhorar a performance do algoritmo.

Foi apresentado um speed-up relativamente pequeno, que pode ser atribuído a uma eficiência baixa de cada processo.

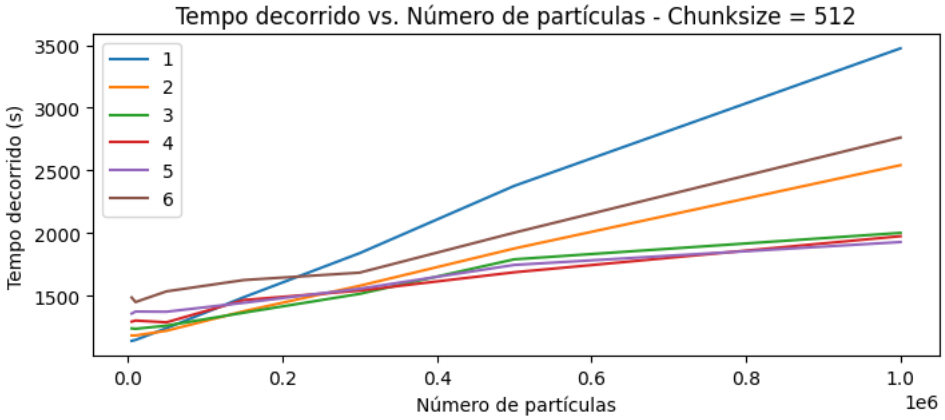

Figura 5.6: Relação do número de partículas com tempo decorrido
fonte: Autoria própria

A baixa eficiência tem como causa o aumento da quantidade de memória total necessária para completar cada iteração de advecção. Conforme a simulação avança, a intersecção de cada área atribuída a diferentes processos cresce, e portanto, a quantidade de memória acessada também.

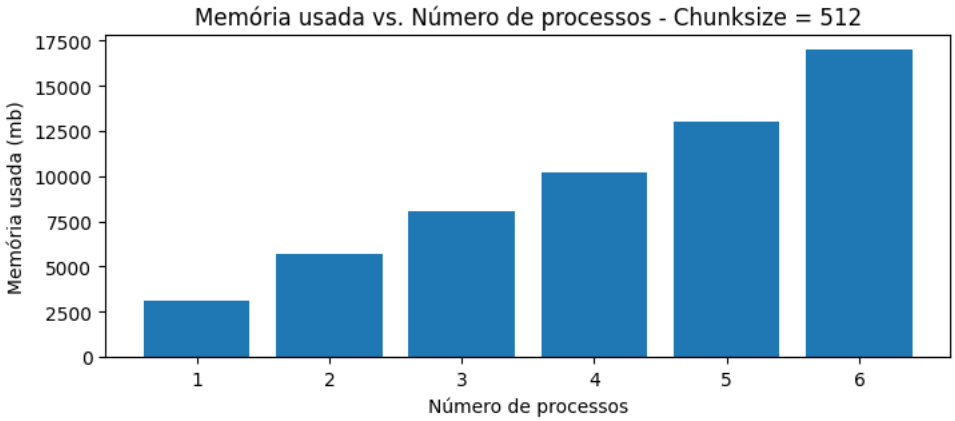

Figura 5.7: Relação memória usada e número de processos
fonte: Autoria própria

6 CONCLUSÃO

Uma sólida fundamentação teórica foi realizada no campo de análise lagrangiana computacional e técnicas de visualização de fluxo, procurando compreender o papel da ciência da computação em problemas ecológicos e criando pontes entre os diferentes campos necessários para a uma modelagem do panorama geral do problema da poluição de plástico.

A grande contribuição deste trabalho é a abertura de portas para estudos futuros e novas análises, em todos os aspectos da modelagem.

O framework Parcels continua tendo grande influência e é constantemente atualizado com novos recursos e funcionalides, e existe um grande esforço para construir um esquema de balanceamento dinâmico que atenda as necessidades específicas da modelagem com alto grau de interatividade. Novas estratégias e algoritmos podem ser avaliados e testados na etapa de resolução de equações diferenciais, com parametrizações e otimizações que podem a levar a speed-ups consideráveis. Tudo isso faz com que existam vastas oportunidades para o campo de processamento de alto desempenho, que pode auxiliar muito a performance de modelos maiores, com melhor escalabilidade e adaptação a novas arquiteturas.

O campo de aspectos computacionais também pode ser de grande interesse a comunidade oceanográfica, que procura compreender fenômenos de transporte, não apenas de resíduos plásticos, mas assim como outros tipos de poluentes. É através da experimentação que validações e novas hipóteses são formuladas, e portanto, novos materiais como tutoriais e documentações detalhadas podem ser criadas, para auxiliar e unir experimentalistas e cientistas mais focados em aspectos teóricos.

REFERÊNCIAS

ARAKAWA, A. e LAMB, V. R. (1977). Computational design of the basic dynamical processes of the ucla general circulation model. Em CHANG, J., editor, *General Circulation Models of the Atmosphere*, volume 17 de *Methods in Computational Physics: Advances in Research and Applications*, páginas 173–265. Elsevier.

Barth, J., Allen, S., Dever, E., Dewey, R., Evans, W., Feely, R., Fisher, J., Fram, J., Hales, B., Ianson, D., Jackson, J., Juniper, S., Kawka, O., Kelley, D., Klymak, J., Konovsky, J., Kosro, P., Kurapov, A., Mayorga, E. e Wingard, C. (2019). Better regional ocean observing through cross-national cooperation: A case study from the northeast pacific. *Frontiers in Marine Science*, 6.

Blettler, M., Ulla, M., Rabuffetti, A. e Garello, N. (2017). Plastic pollution in freshwater ecosystems: macro-, meso-, and microplastic debris in a floodplain lake. *Environmental Monitoring and Assessment*, 189:581.

Bruno Blanke, N. G. (2012). Ariane.

cf conventions (2022). Netcdf climate and forecast (cf) metadata conventions.

Delandmeter, P. e Sebille, E. (2019). The parcels v2.0 lagrangian framework: new field interpolation schemes. *Geoscientific Model Development*, 12:3571–3584.

Documentation, A. (2018). Fundamentals of netcdf data storage. *ArcGIS*. https://pro.arcgis.com/en/pro-app/2.7/help/data/multidimensional/fundamentals-of-netcdf-data-storage.htm.

documentation, N. (2021). Network common data form (netcdf). *unidata*. https://www.unidata.ucar.edu/software/netcdf/.

d'Océanographie Physique et Spatiale (LOPS), L. (2021). Ariane.

Döös, K., Kjellsson, J. e Jönsson, B. (2013). *TRACMASS-A Lagrangian trajectory model*.

Eriksen, M., Lebreton, L. C. M., Carson, H. S., Thiel, M., Moore, C. J., Borerro, J. C., Galgani, F., Ryan, P. G. e Reisser, J. (2014). Plastic pollution in the world's oceans: More than 5 trillion plastic pieces weighing over 250,000 tons afloat at sea. *PLOS ONE*, 9(12):1–15.

França, A. F. (2022). Pesquisadores detectam microplástico em 16 praias do litoral do paraná. *Portal Ciência UFPR*. https://ciencia.ufpr.br/portal/microplastico-presente-em-16-praias-do-complexo-estuarino-de-paranagua-litoral-do-parana/.

Geyer, R., Jambeck, J. R. e Law, K. L. (2017). Production, use, and fate of all plastics ever made. *Science Advances*, 3(7):e1700782.

Gibbens, S. (2019). Plastic bag found at the bottom of world's deepest ocean trench.

Gross, T. (2004). Netcdf standards for data interchange between common hydrodynamic models. *Community Modelers NetCDF Page*.

Group, N. T. W. (2018). TOP – Tracers in Ocean Paradigm – The NEMO Tracers engine.

Gurvan, M., Bourdallé-Badie, R., Chanut, J., Clementi, E., Coward, A., Ethé, C., Iovino, D., Lea, D., Lévy, C., Lovato, T., Martin, N., Masson, S., Mocavero, S., Rousset, C., Storkey, D., Müeller, S., Nurser, G., Bell, M., Samson, G., Mathiot, P., Mele, F. e Moulin, A. (2022). Nemo ocean engine.

Gurvan, M., Bourdallé-Badie, R., Chanut, J., Clementi, E., Coward, A., Ethé, C., Iovino, D., Lea, D., Lévy, C., Lovato, T., Martin, N., Masson, S., Mocavero, S., Rousset, C., Storkey, D., Vancoppenolle, M., Müeller, S., Nurser, G., Bell, M. e Samson, G. (2019). Nemo ocean engine. Add SI3 and TOP reference manuals.

Hale, R. C. e Song, B. (2020). Single-use plastics and covid-19: Scientific evidence and environmental regulations. *Environmental Science & Technology*, 54(12):7034–7036. PMID: 32510208.

Hastings, J. T. e Hill, L. L. (2009). *Georeferencing*, páginas 1246–1249. Springer US, Boston, MA.

Hendrickson, B. e Devine, K. (2000). Dynamic load balancing in computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 184(2):485–500.

Huck, T., Bajon, R., Grima, N., Portela, E., Molines, J.-M. e Penduff, T. (2022). Three-dimensional dispersion of neutral "plastic" particles in a global ocean model. *Frontiers in Analytical Science*, 2.

Kang, H.-G., Evans, K. J., Petersen, M. R., Jones, P. W. e Bishnu, S. (2021). A Scalable Semi Implicit Barotropic Mode Solver for the MPAS Ocean. *Journal of Advances in Modeling Earth Systems*, 13(4):e2020MS002238.

Kreuzer, E. e Sichermann, W. (2006). *Nonlinear Dynamics in Ocean Engineering*, páginas 173–185.

Lange, M. e van Sebille, E. (2017). Parcels v0.9: prototyping a lagrangian ocean analysis framework for the petascale age. *Geoscientific Model Development*, 10(11):4175–4186.

Lebreton, L., Egger, M. e Slat, B. (2019). A global mass budget for positively buoyant macroplastic debris in the ocean. *Scientific Reports*, 9.

Marotzke, J., Giering, R., Zhang, K. Q., Stammer, D., Hill, C. e Lee, T. (1999). Construction of the adjoint mit ocean general circulation model and application to atlantic heat transport sensitivity. *Journal of Geophysical Research: Oceans*, 104(C12):29529–29547.

McLoughlin, T., Laramee, R. S., Peikert, R., Post, F. H. e Chen, M. (2009). Over Two Decades of Integration-Based, Geometric Flow Visualization. Em Pauly, M. e Greiner, G., editores, *Eurographics 2009 - State of the Art Reports*, páginas 73–92. The Eurographics Association.

N, S., S, R., Ray, P., Chen, K., Lassman, A. e Brownlee, J. (2013). *Grids in Numerical Weather and Climate Models*.

North Elizabeth W., Gallego Alejandro, P. P. (2009). Manual of recommended practices for modelling physical biological interactions during fish early life.

Program, N. O. P. (2022). Hybrid coordinate ocean model.

Ritchie, H. e Roser, M. (2018). Plastic pollution. *Our World in Data*. https://ourworldindata.org/plastic-pollution.

Rochman, C. M., Browne, M. A., Underwood, A. J., van Franeker, J. A., Thompson, R. C. e Amaral-Zettler, L. A. (2016). The ecological impacts of marine debris: unraveling the demonstrated evidence from what is perceived. *Ecology*, 97(2):302–312.

Schroers, B. J. (2011). *References*, página 116–116. AIMS Library of Mathematical Sciences. Cambridge University Press.

Shchepetkin, A. F. e McWilliams, J. C. (2005). The regional oceanic modeling system (roms): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, 9:347–404.

Stewart, J. (2012). *Calculus : early transcendentals*. Brooks/Cole, Cengage Learning, Belmont, Cal.

Trobec, R., Slivnik, B., Bulic, P. e Robic, B. (2018). *Introduction to Parallel Computing - From Algorithms to Programming on State-of-the-Art Platforms*. Undergraduate Topics in Computer Science. Springer.

van Sebille, E., Aliani, S., Law, K. L., Maximenko, N., Alsina, J. M., Bagaev, A., Bergmann, M., Chapron, B., Chubarenko, I., Cózar, A., Delandmeter, P., Egger, M., Fox-Kemper, B., Garaba, S. P., Goddijn-Murphy, L., Hardesty, B. D., Hoffman, M. J., Isobe, A., Jongedijk, C. E., Kaandorp, M. L. A., Khatmullina, L., Koelmans, A. A., Kukulka, T., Laufkötter, C., Lebreton, L., Lobelle, D., Maes, C., Martinez-Vicente, V., Maqueda, M. A. M., Poulain-Zarcos, M.,

Rodríguez, E., Ryan, P. G., Shanks, A. L., Shim, W. J., Suaria, G., Thiel, M., van den Bremer, T. S. e Wichmann, D. (2020). The physical oceanography of the transport of floating marine debris. *Environmental Research Letters*, 15(2):023003.

van Sebille, E., Griffies, S. M., Abernathey, R., Adams, T. P., Berloff, P., Biastoch, A., Blanke, B., Chassignet, E. P., Cheng, Y., Cotter, C. J., Deleersnijder, E., Döös, K., Drake, H. F., Drijfhout, S., Gary, S. F., Heemink, A. W., Kjellsson, J., Koszalka, I. M., Lange, M., Lique, C., MacGilchrist, G. A., Marsh, R., Mayorga Adame, C. G., McAdam, R., Nencioli, F., Paris, C. B., Piggott, M. D., Polton, J. A., Rühs, S., Shah, S. H., Thomas, M. D., Wang, J., Wolfram, P. J., Zanna, L. e Zika, J. D. (2018). Lagrangian ocean analysis: Fundamentals and practices. *Ocean Modelling*, 121:49–75.

Versteeg, H. K. e Malalasekera, W. (1995). *An introduction to computational fluid dynamics - the finite volume method.* Addison-Wesley-Longman.

Yenpure, A., Sane, S., Binyahib, R., Pugmire, D., Garth, C. e Childs, H. (2022). A guide to particle advection performance. *CoRR*, abs/2201.08440.

Yoon, J.-H. e Ma, P.-L. (2012). *Oceanic General Circulation Models ocean/oceanic general circulation models (OGCM)*, páginas 7365–7381.