



Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciências

Instituto Politécnico

Ralph Alves Bini da Silva Almeida

**Solução numérica do escoamento não-isotérmico em
reservatórios de óleo pesado empregando computação paralela**

**Nova Friburgo
2021**

Ralph Alves Bini da Silva Almeida

**Solução numérica do escoamento não-isotérmico em reservatórios de óleo
pesado empregando computação paralela**

Dissertação apresentada, como requisito parcial para a obtenção do título de Mestre, ao Programa de Pós-graduação em Modelagem Computacional do Instituto Politécnico, da Universidade do Estado do Rio de Janeiro.

Orientador: Prof. Dr. Helio Pedro Amaral Souto
Prof. Dr. Grazione de Souza

Nova Friburgo

2021

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/BIBLIOTECA CTC/E

A447 Almeida, Ralph Alves Bini da Silva.
Solução numérica do escoamento não-isotérmico em
reservatórios de óleo pesado empregando computação paralela /
Ralph Alves Bini da Silva Almeida. – 2021.
64 f. : il.

Orientadores: Helio Pedro Amaral Souto e Grazione de Souza.
Dissertação (mestrado) - Universidade do Estado do Rio de
Janeiro, Instituto Politécnico.

1. Escoamento em meios porosos – Métodos de simulação -
Teses. 2. Simulação numérica - Teses. 3. Reservatórios de petróleo –
Métodos de simulação. 5. Método dos volumes finitos – Soluções
numéricas – Teses. I. Souto, Helio Pedro Amaral. II. Souza, Grazione
de. III. Universidade do Estado do Rio de Janeiro. Instituto
Politécnico. IV. Título.

CDU 531.3:519.87

Bibliotecária Cleide Sancho CRB7/5843

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta
dissertação, desde que citada a fonte.


Assinatura

04/03/2021
Data

Ralph Alves Bini da Silva Almeida

**Solução Numérica do Escoamento Não-isotérmico em Reservatórios de Óleo
Pesado Empregando Computação Paralela**

Dissertação apresentada como requisito parcial para
obtenção do título de Mestre, ao Programa de Pós-
Graduação em Modelagem Computacional do Instituto
Politécnico, da Universidade do Estado do Rio de
Janeiro.

Aprovado em 24 de fevereiro de 2021.

Banca examinadora:



Prof. Helio P. A. Souto, Docteur de L'INPL – Orientador
Instituto Politécnico – UERJ



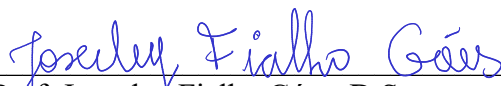
Prof. Grazione de Souza Boy, D.Sc. - Orientador
Instituto Politécnico – UERJ



Prof. Joaquim Teixeira de Assis, D.Sc.
Instituto Politécnico – UERJ



Prof. Alexandre Santos Francisco, D.Sc.
Universidade Federal Fluminense – UFF



Prof. Josecley Fialho Góes, D.Sc.
Universidade Federal do Oeste do Pará – UFOPA

Nova Friburgo

2021

DEDICATÓRIA

Este trabalho é dedicado à minha mãe Eliane Bini da Silva, ao meu pai Ralph Alves Pereira de Almeida, ao meu irmão Richard Bini Almeida, à minha madrinha Danielle Breder Salarini, à minha namorada Jéssica Braga da Costa, aos meus queridos tios e a todos os meus amados primos que sempre estiveram ao meu lado me apoiando e acreditando em meu potencial. Dedico este trabalho também, *in memoriam*, a minha avó Palmyra Bini da Silva, ao meu avô Ezequiel Duarte da Silva e ao meu tio Eliandro Bini da Silva, que foram fundamentais na minha formação pessoal.

Uma pessoa que nunca cometeu um erro nunca tentou algo novo.

Albert Einstein

Não é o conhecimento, mas o ato de aprender, não a posse, mas o ato de chegar lá,
que concede a maior satisfação.

Carl Friedrich Gauss

O que sabemos é uma gota; o que ignoramos é um oceano.

Isaac Newton

RESUMO

ALMEIDA, Ralph Alves Bini da Silva. *Solução numérica do escoamento não-isotérmico em reservatórios de óleo pesado empregando computação paralela*. 2021. 64 f. Dissertação (Mestrado em Modelagem Computacional) - Instituto Politécnico, Universidade do Estado do Rio de Janeiro, Nova Friburgo, 2021.

Neste trabalho foi implementada uma versão paralelizada de um simulador numérico para o escoamento não-isotérmico em um reservatório de óleo pesado. Utilizou-se a API OpenACC na paralelização de partes específicas do código original, o que permitiu o uso simultâneo de diferentes núcleos da placa de vídeo NVIDIA GTX 750 Ti para executar tarefas em uma arquitetura de memória compartilhada. Estudou-se o problema de produção utilizando poço vertical junto ao aquecimento da jazida pela aplicação de aquecedores estáticos, em um domínio bidimensional em geometria cartesiana. O Método das Diferenças Finitas foi utilizado na discretização das equações governantes do escoamento e o Método dos Gradientes Conjugados foi adotado para se determinar as variáveis pressão e temperatura. No estudo do desempenho computacional foram consideradas variações na malha computacional e em parâmetros do modelo de escoamento, tais como a taxa de aquecimento, tendo sido obtido redução nos custos computacionais da versão paralela quando comparado à versão serial.

Palavras-chave: Simulação numérica de reservatórios. Óleo pesado. Computação de alto desempenho. OpenACC. Recuperação terciária.

ABSTRACT

ALMEIDA, Ralph Alves Bini da Silva. *Numerical solution of non-isothermal flow in heavy oil reservoirs applying parallel computing*. 2021. 64 f. Dissertação (Mestrado em Modelagem Computacional) - Instituto Politécnico, Universidade do Estado do Rio de Janeiro, Nova Friburgo, 2021.

In this work a parallel version of a numerical simulator has been implemented for non-isothermal flow in a heavy oil reservoir. The OpenACC API was applied to parallelize specific parts of the original code, which allowed the simultaneous use of different cores of the NVIDIA GTX 750 Ti graphics card to perform tasks in a shared memory architecture. It was studied the oil production from a vertical well applying static heaters in a two-dimensional domain in Cartesian geometry. The Finite Differences Method was used in the discretization of the flow governing equations and the Conjugate Gradients Method was adopted to determine the unknowns pressure and temperature. In the computational performance study, variations in the computational grid and in flow model parameters, such as heating rate, were considered, and a reduction in computational costs was obtained when compared to the serial version.

Keywords: Numerical reservoir simulation. Heavy oil. High performance computing. OpenACC. Tertiary recovery.

LISTA DE FIGURAS

Figura 1 – Domínio bidimensional discreto	24
Figura 2 – Fluxograma para um passo de tempo	31
Figura 3 – Modelo de paralelização usando o OpenACC	34
Figura 4 – Reservatório com poço produtor e aquecedores	43
Figura 5 – Pressão do poço produtor: seis diferentes malhas	44
Figura 6 – Pressão do poço produtor: serial e em paralelo	45
Figura 7 – <i>Speedup</i> : seis diferentes malhas	46
Figura 8 – Uso da GPU por unidade de tempo	47
Figura 9 – Imagem do NVIDIA X server	48
Figura 10 – Pressão do poço produtor: 1.000 dias de produção	48
Figura 11 – Pressão do reservatório: ao longo do eixo x	49
Figura 12 – Pressão do reservatório: 100 dias	50
Figura 13 – Pressão do reservatório: 500 dias	50
Figura 14 – Pressão do reservatório: 1.000 dias	51
Figura 15 – Pressão do reservatório: diferentes Q_H e $y=6.400$ m	52
Figura 16 – Pressão do reservatório: diferentes Q_H e $y=3.200$ m	53
Figura 17 – Pressão do reservatório: diferentes permeabilidades	53
Figura 18 – Pressão do reservatório: diferentes viscosidades	54
Figura 19 – Campos de pressão: diferentes vazões	55
Figura 20 – Temperatura do reservatório: 100, 500 e 1.000 dias	56
Figura 21 – Campos de temperatura: diferentes vazões	57

LISTA DE TABELAS

Tabela 1 – Tabela de Cláusulas	38
Tabela 2 – Malhas computacionais	42
Tabela 3 – Caso padrão de simulação	42
Tabela 4 – Tempos de execução: seis diferentes malhas	45
Tabela 5 – <i>Speedup</i> : 100, 500 e 1.000 dias de produção	51
Tabela 6 – <i>Speedup</i> : 10, 20 e 40 kW	52

SUMÁRIO

	INTRODUÇÃO	9
1	ESCOAMENTO NÃO-ISOTÉRMICO DE ÓLEO EM RESERVATÓRIOS	15
1.1	O meio poroso e as hipóteses adotadas	15
1.2	Temperatura média	17
1.3	Equação para a pressão	18
1.4	Equação para a temperatura	20
2	RESOLUÇÃO NUMÉRICA	23
2.1	Discretização da equação do <i>momentum</i>	23
2.2	Discretização da equação da energia	26
2.3	Decomposição de operadores	27
2.4	O método dos Gradientes Conjugados	29
3	PARALELIZAÇÃO COM O OPENACC	33
3.1	Processamento paralelo usando o OpenACC	33
3.2	Diretivas, cláusulas, <i>flags</i> e PGI <i>Compiler</i>	36
3.2.1	<u>Diretivas e cláusulas</u>	36
3.2.2	<u>PGI <i>Compiler</i></u>	37
3.3	Speedup, Eficiência e Escalabilidade	39
3.3.1	<u>Speedup</u>	39
3.3.2	<u>Eficiência</u>	40
3.3.3	<u>Escalabilidade</u>	40
4	RESULTADOS NUMÉRICOS	41
4.1	Verificação numérica	43
4.2	Desempenho computacional	45
4.3	Análise de sensibilidade	47
4.3.1	<u>Pressão no poço produtor</u>	47
4.3.2	<u>Pressão no reservatório</u>	49
4.3.3	<u>Temperatura no reservatório</u>	56
	CONCLUSÃO	58
	REFERÊNCIAS	60

INTRODUÇÃO

Na indústria do petróleo, há um cenário de incertezas associado às atividades de exploração (busca por reservas) e a exploração (produção). Os reservatórios, meios porosos rochosos nos quais os hidrocarbonetos estão armazenados, encontram-se em profundidades que podem variar de dezenas a milhares de metros na plataforma continental (produção *onshore*) ou abaixo do leito submarino (produção *offshore*). No que diz respeito às propriedades de rocha, elas tendem a apresentar variações ao longo do reservatório que, em geral, tem extensão de quilômetros em seus comprimento e largura (direções x e y) e dezenas de metros de altura (direção z). Lembrando que a distribuição dos fluidos e as suas propriedades também variam em função da posição no interior do reservatório.

As simulações numéricas do escoamento em reservatórios são, frequentemente, utilizadas na avaliação dos cenários de operação, auxiliando na escolha da melhor estratégia de produção. Neste sentido, a indústria de petróleo e gás tem feito uso de ferramentas de computação de alto desempenho que permitem a obtenção de resultados com uma consequente redução do custo computacional.

Recuperação de óleo pesado

Os fenômenos de escoamento de fluidos em meios porosos são estudados em diferentes áreas, tais como os processos de filtração na indústria, a dinâmica dos solos, a modelagem de processos de contaminação no subsolo, os aquíferos, o sequestro de carbono (CO_2) e na recuperação de hidrocarbonetos. O escoamento de fluidos em meios porosos é aquele no qual há uma ou mais fases fluidas se deslocando em um meio formado, predominantemente, por uma estrutura sólida e por espaços, os chamados poros, que se encontram preenchidos pelos fluidos (MINDEN, 2012). Segundo Class (2007), os meios porosos podem ser classificados em naturais ou artificiais. Os primeiros são aqueles criados a partir de fenômenos naturais, onde se encontram grande parte dos casos de interesse prático associados ao subsolo, tais como os reservatórios de petróleo e os aquíferos. No caso dos meios porosos artificiais, consideram-se as estruturas geradas a partir da atuação humana, que deseja que um determinado fluido percole através de um meio poroso, com características específicas, em função da aplicação desejada. Como alguns exemplos tem-se os filtros,

em atividades de separação na indústria, e certos modelos de células de combustível (CLASS, 2007), com aplicações na indústria automobilística.

Assim, a estimativa da estrutura e das propriedades do meio poroso são de grande importância para diferentes áreas das engenharias, sendo que os fatores mais investigados são a estrutura, a heterogeneidade do meio e a sua molhabilidade (CLASS, 2007). A heterogeneidade em meios porosos naturais decorre do fato de que as propriedades de rocha e a sua geometria variam no espaço, impactando nas condições de escoamento. Já a molhabilidade determina qual das fases do sistema multifásico é a molhante, a qual preferencialmente cobrirá a fase sólida. Normalmente, em sub-superfícies, a fase molhante é a água (DANDEKAR, 2013), o que significa dizer que é mais favorável que ela mantenha contato com a superfície do sólido do que as fases óleo e gás, por exemplo.

Os tipos de rocha e de fluido influenciam sobremaneira na escolha da forma de como os hidrocarbonetos serão recuperados, i.e., retirados do reservatório e transportados ao poço produtor. Segundo Dake (2001), a recuperação de hidrocarbonetos pode ser classificada em primária, secundária ou terciária. Na recuperação primária, o óleo é retirado do reservatório utilizando-se apenas a energia que já existia no meio poroso antes de a produção começar. Nesse caso, a produção ocorre pela descompressão dos fluidos e/ou da rocha. Na recuperação secundária, uma quantidade adicional de óleo é obtida, com relação à obtida na recuperação primária, por meio de uma suplementação energética, artificialmente transferida para o reservatório. A injeção de água ou de gás, para um escoamento imiscível, são técnicas denominadas como de recuperação secundária. Por último, na recuperação terciária são aplicados métodos miscíveis, químicos, de deslocamento miscível, biológicos ou térmicos (LAKE, 1989).

Especificamente, em se tratando dos métodos térmicos, é possível citar técnicas tais como a combustão *in situ* (CHEN et al., 2019), a injeção de vapor aquecido (MOHAMMADI; AMELI, 2019) ou o aquecimento do reservatório mediante o uso de equipamentos estáticos (RANGEL-GERMAN et al., 2004; AOUIZERATE; DURLLOFSKY; SAMIER, 2015). Nessa última, considerada neste trabalho, um equipamento é introduzido no meio poroso e um processo de aquecimento ocorre sem a injeção de qualquer fluido, baseado, por exemplo, no uso do eletromagnetismo (BERA; BABADAGLI, 2015). O aquecimento de uma jazida favorece o escoamento ao reduzir a viscosidade do óleo, como no caso de reservatórios de óleo pesado (AMIRIAN; DEJAM; CHEN, 2018; ADO; GREAVES; RIGBY, 2019; MARQUEZ et al., 2020). Esses óleos possuem uma viscosidade que varia na faixa de 20×10^{-3} Pa.s a 400×10^{-3} Pa.s.

Uma das abordagens interessantes seria o escoamento monofásico de óleo pe-

sado em um reservatório aquecido via a utilização de aquecedores estáticos estrategicamente posicionados. A modelagem desse tipo de escoamento é feita empregando-se as equações diferenciais parciais (EDPs) não lineares que governam o escoamento. Elas possuem como variáveis dependentes a pressão e a temperatura do óleo. Para tais problemas, soluções analíticas só são possíveis de serem determinadas para casos particulares, contendo hipóteses simplificadoras. Assim sendo, considera-se a aplicação de metodologias numéricas na obtenção de soluções aproximadas, no contexto da simulação de reservatórios de petróleo.

Simulação numérica de reservatórios

Os modelos físico-matemáticos que descrevem o escoamento em reservatórios de óleo e gás, via de regra, são compostos por EDPs ou sistemas de EDPs não lineares, sendo que as suas soluções analíticas costumam estar disponíveis somente para casos específicos simplificados. Como a realização de experimentos práticos, utilizando amostras de fluido e de rocha em laboratório, e outros tipos de avaliações experimentais são limitadas (podem ocorrer, por exemplo, quando da aplicação de testes de pressão em poços (BOURDET, 2002)), as simulações numéricas do escoamento em reservatórios (AZIZ; SETTARI, 1979; ERTEKIN; ABOU-KASSEM; KING, 2001; ABOU-KASSEM; ALI; ISLAM, 2006) são normalmente utilizadas como ferramenta pela indústria petrolífera desde a década de 1960. Por meio dos resultados de simulações é possível prever a variação da quantidade de hidrocarbonetos produzida, testando-se vários cenários de recuperação, a fim de se prever a produção de reservatórios ao longo de décadas de extração.

Um simulador computacional de reservatórios de petróleo é, normalmente, desenvolvido devido ao desejo de engenheiros e profissionais, ligados à área de petróleo, de descrever acuradamente o escoamento dos fluidos presentes nos reservatórios, de modo a saber em quanto tempo seria possível extraí-los e qual seria o volume total de hidrocarbonetos a ser recuperado. As etapas de construção e utilização de um simulador numérico incluem o entendimento do problema de engenharia a ser tratado; a formulação físico-matemática do escoamento através do emprego de EDPs; o emprego de correlações para o cálculo das propriedades de fluido e de rocha; a discretização das equações diferenciais e do domínio físico; o acoplamento ou não do poço com o reservatório; a obtenção da solução propriamente dita das equações governantes; o pós-processamento; e a análise dos resultados obtidos tendo em vista

a aplicação em questão (ERTEKIN; ABOU-KASSEM; KING, 2001).

No caso de reservatórios de óleo pesado, estudos têm sido realizados com o uso da simulação numérica (TAMIM; ABOU-KASSEM; Farouq Ali, 2000; ADO; GREAVES; RIGBY, 2019; CREMON; GERRITSEN, 2021). Por exemplo, no caso da aplicação de aquecedores estáticos, Rousset (2010) considerou o emprego de um conjunto de poços verticais e um conjunto de aquecedores distribuídos no reservatório, ao redor dos poços produtores, para favorecer o aumento da produção. O esforço computacional, nesse tipo de simulação aumenta quando comparado ao caso isotérmico, devido ao fato do maior número de equações resolvidas e às não linearidades envolvidas. Portanto, a área de simulação de reservatórios só tem a se beneficiar com a aplicação de técnicas da computação de alto desempenho (MA; CHEN, 2004; REDONDO, 2017).

Computação Paralela

A Computação Paralela consiste, de modo geral, na utilização de *hardware* e de técnicas de programação específicas que possam viabilizar a redução do tempo necessário para a execução de programas computacionais, quando comparado ao respectivo tempo de execução utilizando os códigos seriais (sem paralelização) (LEE et al., 2019; AMARAL et al., 2020; LESHCHINSKIY et al., 2020). As técnicas de paralelização (CHAPMAN; JOST; PAS, 2008) incluem, por exemplo, o uso da programação aplicando o *Message Passing Interface* (MPI), o *Open Multi-Processing* (OpenMP), o *Open Accelerators* (OpenACC), *Compute Unified Device Architecture* (CUDA) ou híbridos destes (LOSADA et al., 2016). À medida que o *hardware* dos computadores evolui, mais problemas complexos de engenharia são passíveis de serem resolvidos, embora o esforço computacional despendido na resolução destes problemas tendam a ser muito elevados. Em contrapartida, melhorias no desempenho computacional podem ser alcançadas através da paralelização de todo o código numérico (programa de computador), ou de parte dele, implicando, assim, na utilização de arquiteturas possibilitando a execução distribuída e/ou em paralelo (JAQUIE, 1999).

A programação usando o MPI (CARPEN-AMARIE; HUNOLD; TRÄFF, 2017) foi criada para padronizar a troca de informações em ambientes de memória distribuída. Além disso, ele também visa a otimizar a comunicação e a aumentar o desempenho de aplicações executadas em paralelo ou de forma distribuída. O padrão MPI para a comunicação de dados, em computação paralela, possibilita a troca de informação

entre diferentes processadores de máquinas distintas. Através de comandos específicos, diferentes tarefas dos programas podem se dividir, entre os processadores, executando os comandos em paralelo, de modo que o desempenho da execução do programa paralelizado seja superior ao do código serial. Costuma-se denominar *speedup* à razão entre o tempo necessário para a execução dos códigos em modo serial e paralelo.

O uso da API (*Application Programming Interface*) OpenMP ou da API OpenACC na paralelização de um determinado código computacional implica, assim como no caso do MPI, na inserção de comandos específicos, que visam a instruir o programa a dividir entre as *threads* (linhas de processamento), de uma única máquina contendo múltiplos núcleos de processamento de uma *Central Processing Unit* (CPU) no caso do OpenMP) ou de uma *Graphics Processing Unit* (GPU) em se tratando do OpenACC, partes do código numérico que serão executadas. Assim como no caso do MPI, o objetivo final é o de se executar o programa de forma mais rápida em comparação à execução em modo serial. Quando utiliza-se o OpenMP e o OpenACC o acesso à memória é compartilhado. Então, pode-se dizer que, em geral, o principal objetivo da paralelização via o MPI, o OpenMP, o OpenACC e a CUDA é o de melhorar o desempenho computacional, buscando atingir os maiores *speedups* possíveis (LOSADA et al., 2016; YAKUBOV et al., 2013; WU; TAYLOR, 2013). A API OpenACC foi a escolhida para ser utilizada na redução do tempo de simulação, quando da resolução numérica do escoamento não-isotérmico em reservatórios de óleo pesado na presença de poços aquecedores.

Objetivo

Dentro do contexto já apresentado, este trabalho tem como objetivo principal a implementação de uma versão computacionalmente eficiente de um simulador desenvolvido pelos pesquisadores do Laboratório de Modelagem Multiescala e Transporte de Partículas (LABTRAN) do Instituto Politécnico da Universidade do Estado do Rio de Janeiro. Para tanto, é utilizada a API OpenACC na paralelização do método numérico (Gradientes Conjugados) utilizado na obtenção da solução do sistema de equações algébricas linearizadas, oriundas do processo de discretização, utilizado na obtenção da pressão e da temperatura no reservatório durante o processo de recuperação de óleo pesado em uma jazida aquecida por elemento(s) estático(s).

Organização do trabalho

Nesta parte introdutória, abordou-se resumidamente o escoamento não-isotérmico de óleo em reservatórios de petróleo aquecidos estaticamente, a importância da simulação numérica de reservatórios, além de ter sido traçado o objetivo principal a ser alcançado, ou seja, o uso da API OpenACC com o intuito de se paralelizar o método dos Gradientes Conjugados.

O Capítulo 1 apresenta a parte teórica da modelagem do escoamento não-isotérmico em reservatórios de petróleo, nele é feita uma breve revisão das propriedades de fluido e de rocha e são introduzidas as equações diferenciais parciais que governam o escoamento.

A seguir, no Capítulo 2, trata-se da metodologia numérica e computacional empregada neste trabalho, englobando a discretização do domínio e das EDPs usando o Método das Diferenças Finitas e a determinação da solução por meio de uma Decomposição de Operadores.

A estratégia de resolução das equações discretizadas, sistema algébrico de equações linearizado, a partir do Método dos Gradientes Conjugados e uma resumida iniciação à programação paralelizada utilizando o OpenACC são examinadas no Capítulo 3.

No Capítulo 4, expõem-se os resultados numéricos obtidos. A apresentação é dividida em casos destinados à verificação da versão adaptada (paralelizada) do simulador, mediante a confrontação entre os resultados oriundos das versões serial e paralelizada, e os testes voltados para a avaliação do desempenho computacional, considerando as do escoamento não-isotérmico de óleo pesado em reservatórios aquecidos artificialmente.

Finalmente, no último capítulo, são relatadas as conclusões depreendidas assim como são fornecidas algumas perspectivas para possíveis trabalhos futuros.

1 ESCOAMENTO NÃO-ISOTÉRMICO DE ÓLEO EM RESERVATÓRIOS

Quando da modelagem do escoamento monofásico, em reservatórios de petróleo, é necessário conhecer as propriedades do fluido e da rocha. Tais propriedades estão presentes nas equações de balanço de massa, da quantidade de movimento e da energia utilizadas na modelagem do escoamento de óleo. A partir destas equações, EDPs não lineares, obtém-se as equações governantes escritas em termos das variáveis dependentes pressão e temperatura do óleo. Na Seção 1.1 são apresentadas as hipóteses utilizadas na construção do modelo de escoamento, além de serem introduzidas diferentes propriedades de fluido e de rocha. A seguir, na Seção 1.3, as equações de conservação de massa e quantidade de movimento, que descrevem o escoamento não-isotérmico de óleo em meios porosos, são apresentadas e mostra-se como é obtida a EDP que fornece a pressão do óleo. Por fim, a Seção 1.4 é dedicada à obtenção da forma final da EDP proveniente do balanço da energia, cuja variável dependente é a temperatura do óleo.

1.1 O meio poroso e as hipóteses adotadas

Uma das principais propriedades quando se estuda o escoamento em um meio poroso é a porosidade. Trata-se da relação entre os volumes poroso (poros) e total do material (sólido mais poros), a qual mede a capacidade de armazenamento de fluidos no meio poroso. Quanto mais poroso for um reservatório, maior é o volume acondicionado de hidrocarbonetos (DANDEKAR, 2013). Em uma rocha reservatório podem ser encontrados dois tipos de porosidade, a efetiva e a total. A porosidade total inclui os poros isolados e os interconectados. Já a porosidade efetiva considera exclusivamente os poros interconectados (TIAB; DONALDSON, 2004). Na Engenharia de Reservatórios, a porosidade efetiva é a de maior relevância pois apenas os poros interconectados permitem que haja escoamento. Assim sendo, daqui em diante utiliza-se o termo porosidade para designar a porosidade efetiva ϕ , salvo menção em contrário.

No caso geral, a porosidade varia em função da posição na formação rochosa e o reservatório é dito ser heterogêneo em relação a tal propriedade. Contudo, se a porosidade for independente da posição e constante, a rocha reservatório é chamada de homogênea (TIAB; DONALDSON, 2004). Como os reservatórios homogêneos são exceções, esse conceito é geralmente utilizado para uma formação rochosa idealizada,

útil, por exemplo, na obtenção de uma solução analítica para um caso simplificado de escoamento.

Uma porosidade elevada não é suficiente para caracterizar a viabilidade técnico-econômica de um reservatório, pois os fluidos contidos nos poros das rochas têm que se deslocar para que cheguem à superfície. A facilidade (ou não) de escoamento do fluido depende de uma outra propriedade do meio poroso, a permeabilidade, denotada em geral pelo *tensor* k (DANDEKAR, 2013). A medida da capacidade de um meio poroso permitir a passagem de fluidos através de seus poros interconectados é chamada de permeabilidade absoluta, ou simplesmente permeabilidade se o meio poroso encontra-se 100% saturado por uma única fase líquida. A permeabilidade efetiva é a medida da capacidade de o reservatório permitir o movimento de qualquer fase, levando em consideração a presença das demais fases presentes no meio poroso.

A permeabilidade também pode variar em um meio poroso em função da posição e da direção. Em muitas aplicações é razoável considerar que a permeabilidade pode ser representada, segundo as três direções principais x , y e z , pelas componentes k_x , k_y e k_z do tensor de permeabilidade, respectivamente. O meio poroso é classificado como sendo isotrópico se $k_x=k_y=k_z$. Por outro lado, se a permeabilidade varia segundo as direções espaciais diz-se que o meio é anisotrópico em relação à permeabilidade.

Do ponto vista da transferência de calor em um meio poroso, observa-se que a transferência de energia devida ao movimento do fluido (advecção) será favorecida pelos maiores valores de porosidade (maior capacidade de armazenamento) e permeabilidade (maior facilidade de escoamento). No entanto, cabe ressaltar que a transferência de energia via a condução de calor influencia a temperatura no reservatório mesmo nas regiões com permeabilidade muito baixa ou que encontram-se mesmo isoladas.

Além das propriedades do meio poroso, sem sombra de dúvidas, as propriedades dos fluidos também afetam diretamente o escoamento, conforme será visto ao longo deste capítulo.

Quando da proposição das equações que formam o modelo físico-matemático utilizado na descrição do escoamento, as seguintes hipóteses foram assumidas:

1. a permeabilidade é homogênea e isotrópica;
2. a compressibilidade da rocha é pequena e constante;
3. não ocorrem reações químicas entre o fluido e a rocha;

4. o escoamento é bidimensional, no plano xy , e ocorre a baixas velocidades;
5. o escoamento é monofásico e não-isotérmico;
6. os efeitos gravitacionais são desprezados;
7. a transferência de calor por radiação é desprezível;
8. as condutividades térmicas da rocha e do fluido são constantes;
9. despreza-se os efeitos da tortuosidade e da dispersão hidrodinâmica na transferência de calor;
10. não assume-se o equilíbrio térmico local;
11. o poço de produção é vertical e penetra totalmente a formação;
12. ausência de estocagem no poço¹.

1.2 Temperatura média

Tendo em vista que não é admitida a hipótese do equilíbrio térmico local, ou seja, as temperaturas do fluido e da rocha não estão localmente em equilíbrio termodinâmico (não são iguais), introduz-se uma temperatura média, levando em consideração as variações das temperaturas do fluido e da rocha, de modo a se trabalhar com um modelo a uma única equação para descrever a transferência de calor no reservatório (MOYNE et al., 2000). Então, a temperatura média é obtida como uma média ponderada das temperaturas do fluido e da rocha (MOYNE et al., 2000)

$$(\rho c_p)T = \phi \rho_o c_{p_o} T_o + (1 - \phi) \rho_r c_{p_r} T_r$$

sendo T_o a temperatura do óleo, T_r a temperatura da rocha, T a temperatura média representativa do meio poroso e

$$(\rho c_p) = \phi \rho_o c_{p_o} + (1 - \phi) \rho_r c_{p_r}$$

¹ Quando um poço é colocado em produção, os fluidos inicialmente produzidos originam-se do volume do próprio poço, pois decorre um tempo até que os fluidos do reservatório alcancem a superfície. Este efeito é denominado de estocagem.

representa a capacidade térmica do meio poroso, ρ a massa específica e c_p é o calor específico a pressão constante, sendo que os subscritos o e r indicam o óleo e a rocha, respectivamente.

1.3 Equação para a pressão

Ao se introduzir o fator-volume-formação B (DAKE, 2001),

$$B = \frac{\rho_{osc}}{\rho},$$

onde sc indica as condições padrão de produção, permite-se que a equação de conservação da massa pode ser reescrita na forma alternativa

$$\frac{\partial}{\partial t} \left(\frac{\phi}{B} \right) + \nabla \cdot \left(\frac{\mathbf{v}_o}{B} \right) - \frac{q_{mo}}{V_b \rho_{osc}} = 0, \quad (1)$$

onde $\mathbf{v}_o$ é a velocidade superficial do óleo, q_{mo} é o termo fonte que fornece uma vazão mássica, V_b é o volume total do volume de controle (rocha mais poros) e ρ_{osc} representa a massa específica do óleo em condições padrão de pressão e temperatura: p_{sc} e T_{sc} , respectivamente.

Para escoamentos a baixas velocidades, a equação usada para descrever a conservação da quantidade de movimento no escoamento de fluidos em meios porosos é a lei de Darcy (DARCY, 1856)

$$\mathbf{v}_o = -\frac{\mathbf{k}}{\mu_o} (\nabla p - \rho_o g \nabla D), \quad (2)$$

onde μ_o é a viscosidade do óleo, ρ_o é a massa específica do óleo, g é a magnitude da aceleração da gravidade e D é a profundidade (DAKE, 2001). Substituindo-se a Eq. (2) na Eq. (1), tem-se que

$$\frac{\partial}{\partial t} \left(\frac{\phi}{B} \right) - \nabla \cdot \left[\frac{\mathbf{k}}{B \mu_o} (\nabla p - \rho_o g \nabla D) \right] - \frac{q_{mo}}{V_b \rho_{osc}} = 0.$$

De acordo com as hipóteses adotadas, os efeitos gravitacionais são desconsiderado e introduz-se a relação $q_{mo} = q_{osc} \rho_{osc}$ de modo que

$$\frac{\partial}{\partial t} \left(\frac{\phi}{B} \right) - \nabla \cdot \left(\frac{\mathbf{k}}{B \mu_o} \nabla p \right) - \frac{q_{osc}}{V_b} = 0. \quad (3)$$

O termo $\partial(\phi/B)/\partial t$ da Eq. (3) pode ser expandido levando-se em consideração que as propriedades do fluido e da rocha variam em função da pressão e da temperatura (DYRDAHL, 2014). Portanto, alcança-se o resultado esperado utilizando-se a regra da cadeia,

$$\frac{\partial}{\partial t} \left(\frac{\phi}{B} \right) = \frac{1}{B} \frac{\partial \phi}{\partial t} + \phi \frac{\partial}{\partial t} \left(\frac{1}{B} \right)$$

$$\begin{aligned}
&= \frac{1}{B} \frac{\partial \phi}{\partial p} \frac{\partial p}{\partial t} + \phi \frac{\partial}{\partial p} \left(\frac{1}{B} \right) \frac{\partial p}{\partial t} + \frac{1}{B} \frac{\partial \phi}{\partial T} \frac{\partial T}{\partial t} + \phi \frac{\partial}{\partial T} \left(\frac{1}{B} \right) \frac{\partial T}{\partial t} \\
&= \left[\frac{1}{B} \frac{\partial \phi}{\partial p} + \phi \frac{\partial}{\partial p} \left(\frac{1}{B} \right) \right] \frac{\partial p}{\partial t} + \left[\frac{1}{B} \frac{\partial \phi}{\partial T} + \phi \frac{\partial}{\partial T} \left(\frac{1}{B} \right) \right] \frac{\partial T}{\partial t}.
\end{aligned} \tag{4}$$

Neste trabalho, considera-se que na determinação da variação de B e ϕ em função da pressão e da temperatura, as seguintes expressões podem ser utilizadas:

$$B = \frac{B^0}{1 + c_o(p - p^0) - c_{oT}(T - T^0)}, \tag{5}$$

e

$$\phi = \phi^0 [1 + c_\phi(p - p^0) - c_{\phi T}(T - T^0)] \tag{6}$$

onde B^0 e ϕ^0 são, respectivamente, o fator-volume-formação e a porosidade calculados nas condições de referência para a pressão (p^0) e a temperatura (T^0). Os termos c_o e c_{oT} representam, nessa ordem, a compressibilidade e o coeficiente de expansão térmica do óleo, enquanto que c_ϕ e $c_{\phi T}$ são a compressibilidade e o coeficiente de expansão térmica da rocha, respectivamente.

Usando-se as Eqs. (5) e (6), a Eq. (4) pode ser reescrita como

$$\frac{\partial}{\partial t} \left(\frac{\phi}{B} \right) = \left(\frac{\phi c_o}{B^0} + \frac{\phi^0 c_\phi}{B} \right) \frac{\partial p}{\partial t} - \left(\frac{\phi c_{oT}}{B^0} + \frac{\phi^0 c_{\phi T}}{B} \right) \frac{\partial T}{\partial t}$$

e, então, é possível reescrever a Eq. (3) na forma

$$\Gamma_p \frac{\partial p}{\partial t} - \Gamma_T \frac{\partial T}{\partial t} - V_b \nabla \cdot \left(\frac{\mathbf{k}}{B\mu} \nabla p \right) - q_{osc} = 0, \tag{7}$$

onde

$$\Gamma_p = V_b \left(\frac{\phi c_o}{B^0} + \frac{\phi^0 c_\phi}{B} \right) \tag{8}$$

e

$$\Gamma_T = V_b \left(\frac{\phi c_{oT}}{B^0} + \frac{\phi^0 c_{\phi T}}{B} \right). \tag{9}$$

Para finalizar, resta a determinação da viscosidade como uma função da temperatura. Considera-se, que, a correlação utilizada em [Rousset \(2010\)](#) para óleos pesados,

$$\mu = a \exp \left(\frac{b}{T - T_{ref,\mu}} \right), \tag{10}$$

onde a e b são coeficientes específicos para o óleo considerado e $T_{ref,\mu}$ é uma temperatura de referência para o cálculo da viscosidade.

A Eq. (7) é uma equação diferencial parcial não-linear utilizada na determinação da pressão do óleo. Para resolver-se a Eq. (7), além de se considerar o seu acoplamento com a equação do balanço da energia, deve-se fornecer as condições inicial e de contorno apropriadas a fim de que o problema seja bem-posto.

Como condição inicial no instante inicial utiliza-se

$$p(x, y, t = 0) = p_{ini}(x, y) = p_{inic},$$

onde a pressão inicial antes do reservatório ser perturbado pela produção/injeção é representada por p_{inic} .

No que diz respeito às condições de contorno externas, considera-se que não há escoamento através das fronteiras externas do reservatório

$$\left(\frac{\partial p}{\partial x}\right)_{x=0,L_x} = \left(\frac{\partial p}{\partial y}\right)_{y=0,L_y} = 0,$$

onde L_x e L_y são os respectivos comprimentos do reservatório nas direções x e y .

Se o termo fonte q_{sc} for utilizado para a representação da vazão no poço, aplica-se uma condição de contorno interna no acoplamento poço-reservatório através da relação

$$q_{sc} = -J_w (p - p_{wf}), \quad (11)$$

onde J_w é o índice de produtividade e p_{wf} é a pressão no poço. A determinação do índice de produtividade no contexto da simulação numérica de reservatórios será discutida no Capítulo 2. Por hora, ressalta-se que a Eq. (11) permite o cálculo da pressão no poço se sua vazão é prescrita e vice-versa. Optou-se por utilizar uma condição de vazão de produção prescrita.

1.4 Equação para a temperatura

Utilizando-se a hipótese do não equilíbrio térmico local (MOYNE et al., 2000), para a definição da temperatura média, em conjunto com uma modelagem incluindo a presença de termos fonte para representar o aquecimento da jazida (ROUSSET, 2010), a conservação da energia pode ser expressa por

$$\frac{\partial}{\partial t} [(\rho c_p)T] + \nabla \cdot (\rho_o c_{p_o} \mathbf{v}_o T) - \nabla \cdot (\kappa \nabla T) - \frac{q_H}{V_b} - \frac{\rho_o c_{p_o} T q_{osc}}{V_b} = 0 \quad (12)$$

onde q_H é um termo de fonte térmico e κ é o tensor de condutividade térmica efetivo.

Na obtenção da Eq.(12) as seguintes decomposições espaciais foram introduzidas (MOYNE et al., 2000)

$$T_o = T + \tilde{T}_o$$

e

$$T_r = T + \tilde{T}_r$$

com $\tilde{T}_o$ e $\tilde{T}_r$ representando as flutuações espaciais de temperatura em torno da temperatura média.

Desprezando-se os efeitos devidos à tortuosidade do meio poroso e à dispersão hidrodinâmica (MOYNE et al., 2000) tem-se que

$$\kappa = [\phi \kappa_o + (1 - \phi) \kappa_r] \mathbf{I}$$

onde κ_o e κ_r são, respectivamente, as condutividades térmicas do fluido e da rocha e $\mathbf{I}$ é o tensor identidade.

A Eq. (12) é uma equação diferencial parcial não-linear que será utilizada para a determinação da variável independente temperatura. Para se resolver a Eq. (12), além de se considerar a dependência com a pressão, p (por meio da velocidade de Darcy), são necessárias condições inicial e de contorno apropriadas. Neste trabalho, a condição inicial é dada por

$$T(x, y, t = 0) = T_{ini}(x, y) = T_{inic},$$

onde T_{inic} é a temperatura inicial antes do reservatório ser perturbado pela produção/injeção.

As condições de contorno externas são as de fluxo de calor nulo nas fronteiras do reservatório

$$\left(\frac{\partial T}{\partial x} \right)_{x=0, L_x} = \left(\frac{\partial T}{\partial y} \right)_{y=0, L_y} = 0.$$

O termo fonte q_H será utilizado para a representação numérica da energia transferida pelo poço de aquecimento ao reservatório.

Agora, como as Eqs. (7) e (12) podem ser reescritas como

$$\Gamma_p \frac{\partial p}{\partial t} - V_b \nabla \cdot \left(\frac{\mathbf{k}}{B_\mu} \nabla p \right) = \frac{q_{sc}}{V_b} + \Gamma_T \frac{\partial T}{\partial t}$$

e

$$\frac{\partial}{\partial t} [(\rho c_p)T] - \nabla \cdot (\kappa \nabla T) = \frac{q_H}{V_b} + \frac{\rho_o c_{p_o} T q_{osc}}{V_b} + \nabla \cdot (\rho_o c_{p_o} \mathbf{v}_o T), \quad (13)$$

torna-se mais clara as suas semelhanças de forma, que serão exploradas quando das suas discretizações no Capítulo 2, no que diz respeito aos esquemas de aproximações das derivadas temporais, espaciais e dos termos de fonte/sorvedouro.

2 RESOLUÇÃO NUMÉRICA

Remontando à década de 1950 as suas aplicações iniciais, a simulação numérica de reservatórios de petróleo se tornou, hoje em dia, uma ferramenta padrão na indústria do petróleo (ERTEKIN; ABOU-KASSEM; KING, 2001). O uso generalizado da simulação numérica decorre do fato de que soluções analíticas só podem ser obtidas com o uso de hipóteses simplificadoras como, por exemplo, no caso do escoamento isotérmico unidimensional de um fluido incompressível ou ligeiramente compressível em um meio poroso. No entanto, para problemas mais realísticos, a alternativa viável atualmente é a utilização de códigos numéricos que forneçam soluções aproximadas suficientemente acuradas.

Neste capítulo é apresentada a metodologia numérica aqui aplicada. Nas Seções 2.1 e 2.2, respectivamente, discute-se a discretização das equações diferenciais parciais (equações do *momentum* e da energia) usando o Método das Diferenças Finitas. A utilização de uma decomposição de operadores, para resolver numericamente os sistemas de equações algébricas resultantes do processo de discretização, é abordada na Seção 2.3. Por fim, na Seção 2.4 discute-se o uso do método dos Gradientes Conjugados e introduz-se o algoritmo de resolução da versão serial utilizado.

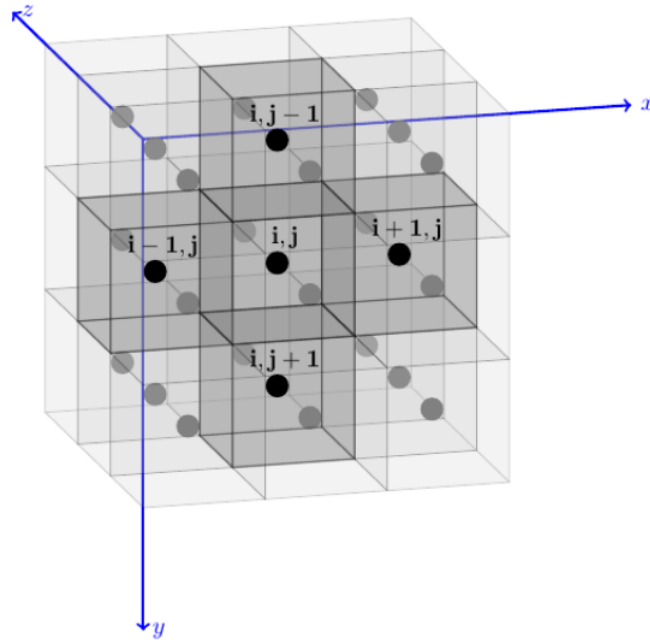
2.1 Discretização da equação do *momentum*

Na determinação da solução numérica emprega-se uma malha de blocos centrados (ERTEKIN; ABOU-KASSEM; KING, 2001; ABOU-KASSEM; ALI; ISLAM, 2006; CHEN; HUAN; MA, 2006). Uma representação esquemática de parte de um domínio bidimensional discretizado pode ser vista na Fig. 1 considerando o sistema de coordenadas cartesianas. A solução numérica é obtida nos nós da malha computacional, localizados nos centros das células, sendo que n_x e n_y representam as quantidades de células nas direções x e y , respectivamente. Os índices inteiros i e j representam as numerações das células nas respectivas direções x e y .

Para o problema do escoamento bidimensional, no plano xy e desconsiderando-se os efeitos gravitacionais, pode-se reescrever a equação do *momentum*, na célula designada por i, j e no nível temporal $n + 1$, no qual as pressões são desconhecidas, na forma discretizada

$$\left[\frac{\partial}{\partial x} \left(\mathbb{T}'_x \frac{\partial p}{\partial x} \right) dx + \frac{\partial}{\partial y} \left(\mathbb{T}'_y \frac{\partial p}{\partial y} \right) dy \right]_{i,j}^{n+1} = \left(\Gamma_p \frac{\partial p}{\partial t} + \Gamma_T \frac{\partial T}{\partial t} + q_{sc} \right)_{i,j}^{n+1}, \quad (14)$$

Figura 1 – Domínio bidimensional discreto



Nota: Parte de um domínio bidimensional discretizado.
 Fonte: [PESSANHA et al., 2020](#).

onde utilizou-se $V_b = dxdyL_z$ e $(V_b)_{i,j} = (\Delta x \Delta y)_{i,j} L_z$ nas Eqs. (7), (8) e (9) e também foram introduzidas as novas variáveis

$$\mathbb{T}'_x \equiv \frac{A_x k_x}{\mu B} \quad \text{e} \quad \mathbb{T}'_y \equiv \frac{A_y k_y}{\mu B},$$

onde $(A_x)_{i,j} = \Delta y_{i,j} L_z$ e $(A_y)_{i,j} = \Delta x_{i,j} L_z$, sendo $\Delta x_{i,j}$ e $\Delta y_{i,j}$, respectivamente, os espaçamentos da malha nas direções x e y na célula i, j e L_z é o comprimento do reservatório na direção z .

Considerando-se um arranjo de malha computacional como o apresentado na Fig. 1 e empregando-se o Método das Diferenças Finitas ([ERTEKIN; ABOU-KASSEM; KING, 2001](#)), obtém-se

$$\frac{\partial}{\partial x} \left(\mathbb{T}'_x \frac{\partial p}{\partial x} \right)_{i,j}^{n+1} \cong \frac{1}{\Delta x_{i,j}} \left[\left(\mathbb{T}'_x \frac{\partial p}{\partial x} \right)_{i+\frac{1}{2},j} - \left(\mathbb{T}'_x \frac{\partial p}{\partial x} \right)_{i-\frac{1}{2},j} \right]^{n+1}$$

que corresponde a um esquema de discretização de segunda ordem centrado.

No intuito de aproximar-se as derivadas espaciais, considera-se inicialmente o problema da discretização na direção x do espaço, no qual os nós $i - 1/2$ e $i + 1/2$ indicam as faces laterais da célula e o nó i o centro da célula. Utiliza-se, então, um esquema do tipo diferença centrada:

$$\left(\frac{\partial p}{\partial x} \right)_{i+\frac{1}{2},j}^{n+1} \cong \frac{p_{i+1,j}^{n+1} - p_{i,j}^{n+1}}{\Delta x_{i+\frac{1}{2},j}}$$

e

$$\left(\frac{\partial p}{\partial x}\right)_{i-\frac{1}{2},j}^{n+1} \cong \frac{p_{i,j}^{n+1} - p_{i-1,j}^{n+1}}{\Delta x_{i-\frac{1}{2},j}}$$

onde $\Delta x_{i\pm 1/2,j}$ é a distância entre os centros das células numeradas por i e $i \pm 1$. De maneira análoga, é possível obter-se as aproximações para as derivadas na direção y .

Introduz-se, em seguida, a variável que representa a transmissibilidade na direção x como sendo dada por

$$\mathbb{T}_{x,i\pm\frac{1}{2},j}^{n+1} = \left(\frac{A_x k_x}{\mu B \Delta x}\right)_{i\pm\frac{1}{2},j}^{n+1},$$

onde é utilizada para a cômputo da área e da permeabilidade, na posição $i \pm 1/2, j$, uma média harmônica, a partir dos valores conhecidos nas localizações i, j e $i \pm 1, j$. Por outro lado, para as propriedades do fluido emprega-se uma média aritmética (ERTEKIN; ABOU-KASSEM; KING, 2001). Bem entendido que uma expressão similar pode ser obtida para a transmissibilidade na direção y .

Utilizando-se uma formulação totalmente implícita no tempo, é possível obter-se a forma final discretizada da Eq. (14),

$$\begin{aligned} & \mathbb{T}_x \Big|_{i+1/2,j}^{n+1} (p_{i+1,j}^{n+1} - p_{i,j}^{n+1}) - \mathbb{T}_x \Big|_{i-1/2,j}^{n+1} (p_{i,j}^{n+1} - p_{i-1,j}^{n+1}) \\ & + \mathbb{T}_y \Big|_{i,j+1/2}^{n+1} (p_{i,j+1}^{n+1} - p_{i,j}^{n+1}) - \mathbb{T}_y \Big|_{i,j-1/2}^{n+1} (p_{i,j}^{n+1} - p_{i,j-1}^{n+1}) \\ & = \frac{(\Gamma_p)_{i,j}^{n+1}}{\Delta t} (p_{i,j}^{n+1} - p_{i,j}^n) + \frac{(\Gamma_T)_{i,j}^{n+1}}{\Delta t} (T_{i,j}^{n+1} - T_{i,j}^n) + (q_{sc})_{i,j}^{n+1} \end{aligned} \quad (15)$$

onde aplicou-se aproximações de Euler recuadas no tempo,

$$\left(\frac{\partial p}{\partial t}\right)_{i,j}^{n+1} \cong \frac{p_{i,j}^{n+1} - p_{i,j}^n}{\Delta t}$$

e

$$\left(\frac{\partial T}{\partial t}\right)_{i,j}^{n+1} \cong \frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t},$$

nas quais n indica o nível de tempo em que a pressão e a temperatura são conhecidas.

O termo $(q_{sc})_{i,j}^{n+1}$ pode ser utilizado para incluir, no modelo adotado, a pressão no poço p_{wf} . Neste caso, a forma discreta da Eq. (11) é dada por

$$(q_{sc})_{i,j}^{n+1} = - (J_w)_{i,j}^{n+1} \left[p_{i,j}^{n+1} - (p_{wf})_{i,j}^{n+1} \right],$$

enquanto que para o índice de produtividade, J_w , tem-se que

$$(J_w)_{i,j}^{n+1} = \left[\frac{2\pi \sqrt{k_x k_y} L_z}{B\mu \ln \left(\frac{r_{eq}}{r_w} \right)} \right]_{i,j}^{n+1}$$

onde r_w é o raio do poço e o raio equivalente (r_{eq}) é calculado através da equação (PE-ACEMAN, 1983)

$$r_{eq} = 0,28 \left[\frac{\sqrt{\frac{k_y}{k_x} (\Delta x)^2 + \frac{k_x}{k_y} (\Delta y)^2}}{\sqrt[4]{\frac{k_y}{k_x}} + \sqrt[4]{\frac{k_x}{k_y}}} \right]_{i,j}$$

2.2 Discretização da equação da energia

A discretização é realizada de forma similar a que foi empregada no tratamento da equação do *momentum*. Inicialmente, reescreve-se a Eq. (13) na forma

$$\frac{\partial}{\partial t} [(\rho c_p)T] - \nabla \cdot (\kappa \nabla T) = \frac{q_H}{V_b} + \frac{\rho_o h_o q_{osc}}{V_b} + \varphi, \quad (16)$$

onde o termo de fonte é $\varphi = \nabla \cdot (\rho_o \mathbf{v}_o h_o)$.

A partir da Eq. (16), considerando-se a determinação da temperatura na célula i, j e no nível temporal $n + 1$, tem-se

$$\left[\frac{\partial}{\partial x} \left(\kappa_x \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\kappa_y \frac{\partial T}{\partial y} \right) \right]_{i,j}^{n+1} = \left\{ \frac{\partial}{\partial t} [(\rho c_p)T] - \frac{q_H}{V_b} - \frac{\rho_o h_o q_{osc}}{V_b} - \varphi \right\}_{i,j}^{n+1}, \quad (17)$$

e nota-se que a mesma possui uma forma muito parecida com a da Eq. (14). Então, para o primeiro e o segundo termos da Eq. (17) aplica-se uma metodologia análoga à utilizada na discretização da Eq. (14):

$$\begin{aligned} & \mathbb{K}_x \Big|_{i+1/2,j}^{n+1} (T_{i+1,j} - T_{i,j})^{n+1} - \mathbb{K}_x \Big|_{i-1/2,j}^{n+1} (T_{i,j} - T_{i-1,j})^{n+1} \\ & + \mathbb{K}_y \Big|_{i,j+1/2}^{n+1} (T_{i,j+1} - T_{i,j})^{n+1} - \mathbb{K}_y \Big|_{i,j-1/2}^{n+1} (T_{i,j} - T_{i,j-1})^{n+1} \\ & = V_b \left[\frac{(\rho c_p)}{\Delta t} \right]_{i,j}^{n+1} (T_{i,j}^{n+1} - T_{i,j}^n) + (\rho_o h_o q_{sc})_{i,j}^{n+1} + \Phi_{i,j}^{n+1} + (q_H)_{i,j}^{n+1} \end{aligned} \quad (18)$$

onde

$$\mathbb{K}_x \Big|_{i\pm 1/2,j}^{n+1} = \left(\frac{A_x \kappa_x}{\Delta x} \right)_{i\pm 1/2,j}^{n+1},$$

$$\mathbb{K}_y \Big|_{i,j\pm 1/2}^{n+1} = \left(\frac{A_y \kappa_y}{\Delta y} \right)_{i,j\pm 1/2}^{n+1},$$

e

$$\begin{aligned} \Phi_{i,j}^{n+1} = & \rho_{osc} [(h_o \mathbb{T}_x)_{i-(1/2),j}^{n+1} (p_{i,j} - p_{i-1,j})^{n+1} - (h_o \mathbb{T}_x)_{i+(1/2),j}^{n+1} (p_{i+1,j} - p_{i,j})^{n+1}] \\ & + \rho_{osc} [(h_o \mathbb{T}_y)_{i,j-(1/2)}^{n+1} (p_{i,j-1} - p_{i,j})^{n+1} - (h_o \mathbb{T}_y)_{i,j+(1/2)}^{n+1} (p_{i,j+1} - p_{i,j})^{n+1}], \end{aligned}$$

com os termos tendo sido reescritos de forma a utilizar-se as transmissibilidades (introduzidas anteriormente) na definição de $\Phi_{i,j}^{n+1}$. Ela servirá de base para a estratégia de resolução numérica usando uma decomposição de operadores (DYRDAHL, 2014; MAES et al., 2015).

2.3 Decomposição de operadores

As Eqs. (15) e (18) formam um sistema de equações algébricas não-lineares que quando resolvido fornecerá os valores de p e T (ISLAM et al., 2010). Para tanto, a velocidade de Darcy na Eq. (18) pode ser usada para explicitar-se a pressão ou ela pode ser mantida como uma incógnita. A temperatura também pode ser explicitada a partir da função entalpia. Nos dois casos, a Eq. (18) permanece com uma não linearidade proveniente do termo que contém originariamente a velocidade de Darcy. A resolução desse sistema, para uma malha computacional com um número de células da ordem dos milhares exigirá uma quantidade de memória e de processamento significativas.

Uma outra questão importante é a forma final das equações algébricas, que pode ser diferente em função das escolhas feitas quando da modelagem físico-matemática e da discretização. Por exemplo, Singh, Goerke e Kolditz (2011) trataram do problema não-isotérmico, no escoamento em meios porosos, reescrevendo o termo de advecção, da equação do balanço de energia, de modo a explicitar o termo responsável pelo efeito de resfriamento Joule-Thompson. Como é sabido, existem métodos numéricos específicos para resolver diferentes tipos de EDPs em função da classificação das mesmas (DYRDAHL, 2014; VENNEMO, 2016). Em um mesmo sistema de EDPs é possível, por exemplo, existirem equações parabólicas e hiperbólicas.

Na literatura, existem propostas de estratégias de resolução numérica, para o problema do escoamento não-isotérmico em meios porosos, que baseiam-se no uso de uma decomposição de operadores (*operator splitting*), tais como as encontradas nos trabalhos de Rousset (2010), Dyrdaahl (2014), Maes et al. (2015) e Vennemo (2016). Neste tipo de resolução, o sistema original escrito em termos de p e T é dividido em dois subsistemas de equações, um para p e outro para T , com um estágio de troca de informações entre os subsistemas. No caso mais geral, isto permite, por exemplo, que diferentes métodos de solução possam ser aplicados na determinação de cada variável dependente, em função das características da EDP correspondente que deve ser resolvida.

No caso estudado por Vennemo (2016), por exemplo, mostra-se que o uso de iterações internas para determinar p e T devem estar condicionadas ao uso de iterações externas para verificar a convergência do sistema como um todo. Todos os testes nos quais a convergência foi regida pela verificação de critérios de tolerância e não por número de iterações prescritas levaram aos melhores resultados. Este tipo de estratégia foi também adotada na presente dissertação.

Considerando uma decomposição de operadores, que leve a uma sequência de resolução determinando primeiramente p e, em seguida, a temperatura T , sequência essa que segue o padrão de maior recorrência na literatura, as Eqs. (15) e (18) são reescritas respectivamente como

$$\begin{aligned}
& \mathbb{T}_y \Big|_{i,j-1/2}^{v,n+1} p_{i,j-1}^{v+1,n+1} + \mathbb{T}_x \Big|_{i-1/2,j}^{v,n+1} p_{i-1,j}^{v+1,n+1} \\
& - \left[\mathbb{T}_y \Big|_{i,j-1/2}^{v,n+1} + \mathbb{T}_x \Big|_{i-1/2,j}^{v,n+1} + \frac{(\Gamma_p)_{i,j}^{v,n+1}}{\Delta t} + \mathbb{T}_x \Big|_{i+1/2,j}^{v,n+1} + \mathbb{T}_y \Big|_{i,j+1/2}^{v,n+1} \right] p_{i,j}^{v+1,n+1} \\
& + \mathbb{T}_x \Big|_{i+1/2,j}^{v,n+1} p_{i+1,j}^{v+1,n+1} + \mathbb{T}_y \Big|_{i,j+1/2}^{v,n+1} p_{i,j+1}^{v+1,n+1} \\
& = - \frac{(\Gamma_p)_{i,j}^{v,n+1}}{\Delta t} p_{i,j}^n + \frac{(\Gamma_T)_{i,j}^{v,n+1}}{\Delta t} (T_{i,j}^{w,n+1} - T_{i,j}^n) + (q_{sc})_{i,j}^{n+1}
\end{aligned} \tag{19}$$

e

$$\begin{aligned}
& \mathbb{K}_x \Big|_{i,j-1/2}^{w,n+1} T_{i,j-1}^{w+1,n+1} + \mathbb{K}_x \Big|_{i-1/2,j}^{w,n+1} T_{i-1,j}^{w+1,n+1} \\
& - \left\{ \mathbb{K}_y \Big|_{i,j-1/2}^{w,n+1} + \mathbb{K}_x \Big|_{i-1/2,j}^{w,n+1} + V_b \left[\frac{(\rho c_p)}{\Delta t} \right]_{i,j}^{w,n+1} + \mathbb{K}_x \Big|_{i+1/2,j}^{w,n+1} + \mathbb{K}_y \Big|_{i,j+1/2}^{w,n+1} \right\} T_{i,j}^{w+1,n+1}
\end{aligned}$$

$$\begin{aligned}
& + \mathbb{K}_x \Big|_{i+1/2,j}^{w,n+1} T_{i+1,j}^{n+1} + \mathbb{K}_y \Big|_{i,j+1/2}^{w,n+1} T_{i,j+1}^{w+1,n+1} \\
& = -V_b \left[\frac{(\rho c_p)}{\Delta t} \right]_{i,j}^{w,n+1} T_{i,j}^n + (\rho_o h_o q_{sc})_{i,j}^{w,n+1} + \Phi_{i,j}^{\bar{v},n+1} + (q_H)_{i,j}^{n+1}
\end{aligned} \tag{20}$$

onde os índices v e w referem-se, respectivamente, aos níveis iterativos para a obtenção das soluções p e T . O índice $\bar{v}$ indica que o termo $\Phi_{i,j}^{n+1}$ está sendo avaliado com o uso de propriedades determinadas em um nível iterativo v , para o qual se obteve as pressões em $v+1, n+1$, também usadas em $\Phi_{i,j}^{n+1}$, obtidas quando da solução da Eq. (19). O mesmo tipo de raciocínio também pode ser estendido para o termo $T_{i,j}^{\bar{w},n+1}$.

Assim sendo, considerou-se uma linearização a partir do uso de coeficientes determinados em uma iteração externa, cuja convergência depende da verificação de uma dada tolerância, na presença de iterações internas para a determinação de p (iterações em v) e para T (iterações em w), seguindo de uma iteração de Picard (ERTEKIN; ABOU-KASSEM; KING, 2001). Assim, a ordem de solução é a que se segue:

1. cálculo dos coeficientes e dos termos fontes das equações;
2. resolução para a determinação de p^{n+1} considerando o valor de T^{n+1} mais recentemente calculado;
3. resolução para T^{n+1} empregando o valor mais recente de p^{n+1} ;
4. teste de convergência para o sistema de equações como um todo.

Na solução de cada subproblema, formado pelas equações algébricas (19) e (20), foi utilizado o método iterativo dos Gradientes Conjugados.

2.4 O método dos Gradientes Conjugados

Os sistemas algébricos de equações, oriundos da linearização das equações algébricas não-lineares, em geral, envolvem a determinação de um elevado número de incógnitas. Nos casos aqui tratados eles são da ordem de centena de milhares. Portanto, torna-se necessário, devido à grande dimensão destes sistemas, o uso de um método de resolução numérica de sistemas de equações algébricas eficiente. Além disso, deve-se prever o uso de recursos computacionais que disponibilizem uma grande quantidade de memória RAM e elevada capacidade de processamento.

Neste trabalho, o método dos Gradientes Conjugados (HESTENES; STIEFEL, 1952; BARRETT et al., 1993; SAAD, 2003) foi empregado com essa finalidade. Trata-se de um método iterativo, desenvolvido de modo a exigir menos cálculos que o método de Newton e a obter soluções acuradas em um número finito de iterações, em se tratando de sistemas lineares de alta ordem cujas matrizes dos coeficientes são simétricas e definidas positivas. Esse método vem sendo considerado, em geral, quando da aplicação do processamento em paralelo devido à sua facilidade de implementação e boa taxa de convergência na sua versão pré-condicionada.

O Algoritmo 1 mostra a sequência de operações associadas à implementação do método dos Gradientes Conjugados visando à obtenção da solução numérica de um sistema linear do tipo $Ax = d$.

Algoritmo 1: Gradientes Conjugados

```

1  Entrada :  $x^0$ ;  $A$ ;  $d$ ;  $r^0 = p^0 = d - Ax^0$  o número de iterações  $n$  e a tolerância
     $tol$                                      /*  $r^0$  = vetor de resíduos */

2  while  $x^{k+1} - x^k > tol$  do
3       $a^k = \frac{r^k \cdot r^k}{p^k \cdot Ap^k}$ 
4       $x^{k+1} = x^k + a^k p^k$ 
5       $r^{k+1} = r^k + a^k Ap^k$ 
6       $b^k = \frac{r^{k+1} \cdot r^{k+1}}{r^k \cdot r^k}$ 
7       $p^k = r^k + b^k p^{k-1}$ 

8  end

9   $x = x^{k+1}$ 

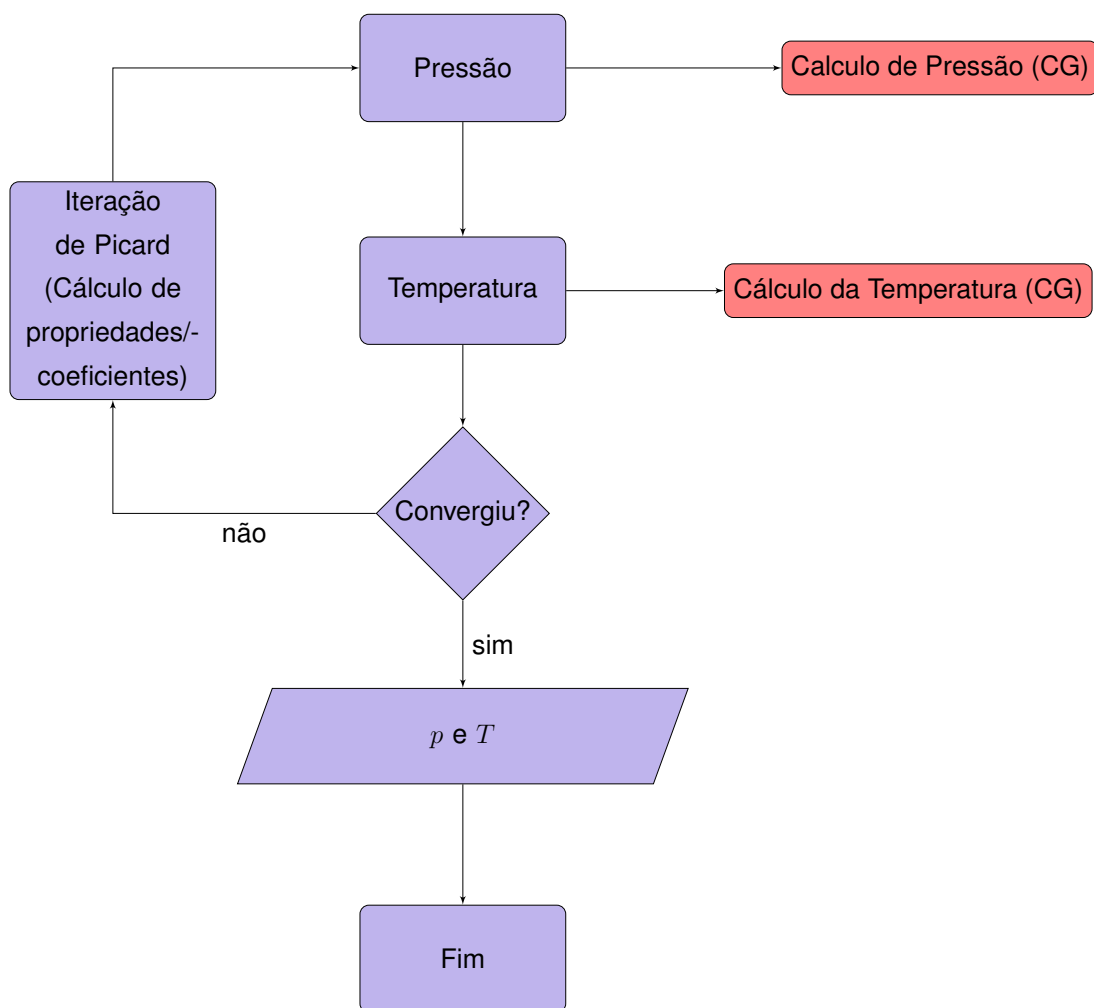
```

No caso do armazenamento dos elementos da matriz A , dentre as diferentes possibilidades de compressão de matrizes, optou-se aqui pelo uso da técnica *Compressed Sparse Row* (CSR) (SAAD, 2003; WERNECK et al., 2019). Através dela, almeja-se economizar espaço na memória e proporcionar um maior desempenho computacional. Nessa técnica, utiliza-se três vetores no simulador de modo a armazenar: os valores não nulos da matriz (*matrixValues*), os índices das colunas dos elementos não nulos (*mtxIndj*) e os índices da posição indicando o primeiro vetor citado, onde inicia-se uma nova linha da matriz (*nonzerosInRow*). Assim sendo, os produ-

tos matriz-vetor e matriz-matriz podem ser programados de forma mais eficiente do que quando os valores nulos de A são alocados na memória. A compressão facilita o emprego da paralelização usando a API OpenACC.

A Fig. 2 apresenta o fluxograma com os passos necessários para a obtenção numérica dos valores da pressão p e da temperatura T . Ele contempla a utilização da decomposição de operadores, da linearização do sistema de equações e do método dos Gradientes Conjugados. Ressalta-se que são empregados dois critérios de convergência, um para a iteração interna (tol_1) e outro para a externa (tol_2).

Figura 2 – Fluxograma para um passo de tempo



Nota: Fluxograma para a obtenção numérica dos valores de p e T .

Fonte: O autor, 2021.

O método de Picard foi originariamente utilizado na versão serial do código que foi o ponto de partida desta dissertação (LOPES, 2016). Ele também foi preservado na versão aqui desenvolvida, sendo que resolve-se primeiramente o subsistema para a determinação da pressão. Após a obtenção desta, seus valores são usados na atu-

alização do termo fonte, presente na equação de energia, que contém a velocidade de Darcy. Então, quando da resolução do subsistema para o cálculo da temperatura, caso a convergência para os dois subsistemas não seja alcançada, os valores mais recentes de pressão e temperatura são utilizados na atualização dos termos fontes e dos coeficientes nos subsistemas e retoma-se o processo iterativo de resolução dos dois subsistemas. Na literatura de simulação de reservatórios é comum, quando do uso de estratégias de desacoplamento, que seja primeiramente calculada a pressão e depois as demais variáveis, tais como a saturação (KOU; SUN, 2004), a concentração (DEBOSSAM et al., 2019) e a própria temperatura (HERINGER et al., 2019).

3 PARALELIZAÇÃO COM O OPENACC

Visto como foram obtidas as equações diferenciais parciais que governam o escoamento não-isotérmico de óleo, como elas foram discretizadas e alguns dos principais aspectos numéricos que devem ser observados quando da simulação numérica, passa-se agora a focar no problema específico da resolução numérica do sistema algébrico de equações não-lineares. A versão paralelizada do método dos Gradientes Conjugados é feita via a API OpenACC.

Na Seção 3.1 são introduzidas as características gerais do processamento utilizando a API OpenACC 2.7. Uma discussão sobre as diretivas e cláusulas que empregadas encontra-se na Seção 3.2. Por fim, a Seção 3.3 aborda os conceitos frequentemente empregados na avaliação dos códigos paralelizados: *Speedup*, Escalabilidade e Eficiência.

3.1 Processamento paralelo usando o OpenACC

No campo da computação paralela, um dos recursos conhecidos disponível é a *Application Programming Interface* (API) OpenACC. Quando da sua aplicação, em arquiteturas de memória compartilhada, faz-se necessário o uso de três componentes básicos: as diretivas de compilação, a biblioteca de execução e as variáveis de ambiente (SULZBACH, 2014). O emprego do OpenACC se baseia no modelo de execução ilustrado na Fig. 3. Em geral, em algum momento da execução do código numérico uma determinada diretiva dará início à paralelização (*thread* inicial), de um trecho do código computacional, distribuindo uma sequência de tarefas entre diversas *threads*. Em seguida, as *threads* executarão separadamente as tarefas designadas. O OpenACC permite também a paralelização utilizando placas gráficas, através de uma programação mais simples quando comparada àquela da API CUDA. Essa última característica representa o motivo principal para se ter escolhido o OpenACC, ou seja, a possibilidade de se contar com o melhor desempenho das GPUs na paralelização dos códigos numéricos, devido ao, em geral, grande número de núcleos (*threads*) disponíveis e memória dedicada.

Figura 3 – Modelo de paralelização usando o OpenACC



Fonte: O autor, 2021.

As diretivas do OpenACC podem ser empregadas em códigos escritos nas linguagens de programação Fortran, C ou C++. Como exemplo de aplicação a uma codificação empregando essas diretivas, em linguagem C, procede-se à paralelização de uma função que executa o produto escalar:

```
double produto_escalar(int N,double *a,double *b)
{
    int i;
    double soma;

    soma=0.0;

    for(i=1;i<=N;i++)
    {
        soma += (a[i]*b[i]);
    }
    return soma;
}
```

Então, após a sua paralelização empregando o OpenACC obtém-se o seguinte código:

```
double produto_escalar(int N,double *a,double *b)
{
    int i;
    double soma;
```

```

soma=0.0;

#pragma acc parallel loop reduction(+:soma)

    for(i=1;i<=N;i++)
    {
soma += (a[i]*b[i]);
    }
    return soma;
}

```

Observa-se que através da diretiva *#pragma* determina-se qual a região que deve ser paralelizada. No caso em questão, a paralelização aplica-se ao laço *for*.

Os dados utilizados nos cálculos são rotulados em dois tipos básicos: *shared* e *private*. No caso do tipo *shared* existe apenas uma instância de dados, sendo que todas as *threads* podem acessar e modificar os dados simultaneamente, a menos que alguma restrição seja imposta via um comando específico do OpenACC. Todas as alterações são visíveis para todas as *threads*. Por outro lado, quando o tipo *private* é adotado, cada linha de execução acessa uma cópia dos dados, que é particular. Nesse caso, as alterações são visíveis apenas para a *thread* que “possui” os dados. Uma descrição completa sobre as diretivas de compilação podem ser encontradas em [OpenACC Organization \(2015\)](#).

Tipicamente, nos simuladores de reservatórios desenvolvidos pelo grupo de pesquisadores do LABTRAN, a maior demanda em termos do tempo de execução tende a ser aquela dedicada à resolução do sistema de equações algébricas. Como consequência, o processo de paralelização concentrou-se no método iterativo de resolução do sistema algébrico escolhido, o método dos Gradientes Conjugados, mais precisamente nos laços de repetição da sua codificação no simulador. Portanto, nesta dissertação, as diretivas e os tipos de dados utilizados na criação de um ambiente paralelizado, *#pragma*, *shared* e *private*, foram aplicados exclusivamente com essa finalidade.

Conforme já aventado, a execução em paralelo de um código computacional pode ser feita em placas gráficas com, por exemplo, as da Nvidia. Nesse caso, pode-se optar por uma abordagem que implique o uso da API CUDA ou OpenACC. Assim, nesta seção aborda-se as vantagens e as desvantagens do OpenACC com relação à CUDA ([SULZBACH, 2014](#)). O OpenACC é uma API que permite, através de direti-

vas e cláusulas, relativamente simples, a paralelização de códigos computacionais e não exige que o usuário aprenda uma nova linguagem, como no caso da CUDA. O OpenACC possui grandes semelhanças com o OpenMP e, assim, dá ao usuário uma sensação de familiaridade caso ele já tenha uma experiência prévia de paralelização utilizando o OpenMP.

Vários estudos apontam que, no que diz respeito à diminuição do tempo de execução ou ganho de *speedup*, um código paralelizado com a API CUDA tende a ter um melhor desempenho do que aquele escrito em linguagem C e utilizando o OpenACC. Porém, em relação à correção de erros decorrentes da paralelização, como o OpenACC modifica pequenas partes do código, ele tende a ser mais simples do que quando depura-se um código paralelizado com a API CUDA (NORMAN et al., 2015).

3.2 Diretivas, cláusulas, *flags* e PGI Compiler

A paralelização com uso de OpenACC demanda, além da correta instalação dos compiladores e da placa gráfica, uma análise minuciosa do código computacional, de modo a definir quais são as partes ou as funções do código que exigem mais do computador, a identificar os trechos paralelizáveis e a escolher apropriadamente as diretivas, cláusulas e *flags* para a correta compilação do código numérico (MOREIRA, 2015).

Aqui, os resultados das simulações foram obtidos através do emprego da versão, da edição comunitária para Linux (baseada no Debian), do PGI Compiler 20.4 (NVIDIA HPC SDK Versão 20.11).

3.2.1 Diretivas e cláusulas

Após definir-se a região do código a ser paralelizada é necessário escolher, dentre as diferentes opções, as diretivas e cláusulas que são as mais apropriadas para que esta tarefa seja finalizada a contento. Neste contexto, foram acrescentadas ao código as diretivas:

```
#pragma acc parallel loop
```

de modo a transmitir-se ao pré-processador da linguagem C a ordem de que deve-se paralelizar o laço de repetição *for* que encontra-se na próxima linha. Conforme já dito, as diretivas acrescentadas ao código para a paralelização com o OpenACC são

semelhantes às utilizadas no OpenMP. Assim, a linha de código, exemplificada acima, em se tratando da API OpenMp seria substituída por

```
#pragma omp parallel for
```

Na primeira, os núcleos da placa de vídeo executarão as tarefas, ao passo que na segunda serão as *threads* do(s) núcleo(s) do(s) processador(es) do computador que farão este trabalho.

Uma diretiva fundamental para a paralelização do código computacional é a *data*. Assim como quando usa-se a estratégia de paralelização com a API CUDA, deve-se copiar, por exemplo, vetores e/ou matrizes que se encontram armazenados na memória acessada pela CPU para a memória da GPU, ou vice-versa. Portanto, em se tratando da API OpenACC, a diretiva responsável por essa troca de informações é, justamente, a *data*.

Para paralelizar-se laços de repetição do tipo *for* é possível também substituir-se a diretiva *parallel loop* pela *kernels*. O uso dessa última diretiva permite que o compilador escolha quais são as estratégias de paralelização mais seguras que deverão ser empregadas e se a paralelização é viável. Por conseguinte, ela retira do programador as suas prerrogativas de escolha.

Foram empregadas, associadas à diretiva *parallel loop*, diferentes cláusulas tais como a *private* e a *reduction*. A primeira especifica que cada iteração do laço tem a sua própria cópia das variáveis listadas. Já no caso da segunda, é gerada uma cópia privada das variáveis, mas existe uma redução ao final da execução em paralelo das cópias privadas em um único resultado final e ele é retornado quando a execução é concluída. As operações de redução possíveis são *max*, *min* e *sum*, por exemplo, *reduction(+:sum)*.

No que diz respeito à diretiva *data* foram utilizadas as cláusulas *copy* e *create*, no intuito de copiar os dados para a memória da GPU e para a CPU, respectivamente.

No intuito de elucidar o uso de algumas cláusulas bem como as suas respectivas funções, utilizadas com as diretivas *parallel* e *data*, foi criada a Tabela 1.

3.2.2 PGI Compiler

Embora tenham sido realizados avanços no sentido de possibilitar o uso do GNU *Compiler* e, dependendo da versão do OpenACC, já seja possível compilar certos códigos computacionais neste compilador, o uso mais difundido e consolidado do OpenACC preconiza que a compilação do código seja feita com o PGI *Compiler* (KIM;

Tabela 1 – Tabela de Cláusulas

Cláusulas	Função
private(variável)	Na diretiva <i>parallel</i> , essa cláusula especifica que cada iteração do laço tem a sua própria cópia das variáveis declaradas entre parênteses
reduction(operador:variável)	Na diretiva <i>parallel</i> , ela implica na redução do valor da variável nas operações de soma (+), multiplicação (*), máximo (<i>max</i>) e mínimo <i>min</i> .
async	Na diretiva <i>parallel</i> , ela é responsável pela retirada das barreiras presentes ao final da região paralela.
copy(variáveis)	Na diretiva <i>data</i> , a cláusula cria espaço para a alocação das variáveis no dispositivo, iniciando-as e copiando os dados para a placa gráfica no início da região. Ela também copia os resultados de volta para o <i>host</i> quando a execução chega ao final da região e libera o espaço no dispositivo.
create	Na diretiva <i>data</i> , ela cria espaço disponível para a alocação das variáveis e as libera ao final da região, mas não copia nenhum dado do dispositivo ou para ele.

Fonte: O autor, 2021.

KANG; JOH, 2021). Ele foi, inclusive, o compilador escolhido para gerar o arquivo executável. Para tanto, algumas *flags* específicas de compilação devem ser utilizadas de forma que o código execute tarefas de forma paralelizada na placa gráfica.

Neste trabalho, as execuções do simulador foram feitas em uma máquina contendo uma placa de vídeo Nvidia GTX 750 Ti com 640 núcleos CUDA. Após a modificação do código computacional, escrito em linguagem C, empregando as diretivas e as cláusulas do OpenACC, digita-se em um terminal Linux o seguinte comando:

```
pgcc -acc -ta=nvidia:managed -fast -Minfo=all -lm sim\_xy1.c -o sim\_parallel
```

O comando “pgcc” indica que o código fonte deve ser compilado, já a *flag* “-acc” informa ao compilador que ele deve considerar as diretivas e cláusulas do OpenACC presentes no código. Em contrapartida, a *flag* “-ta=nvidia:managed” indica que o có-

digo deve ser executado na placa gráfica (já o uso de “-ta:multicore” resultaria na execução do código, de maneira semelhante quando do uso do OpenMP, nas *threads* do processador) e que a alocação na memória usando *Malloc* e *Calloc* será substituída por rotinas que alocam memória de maneira unificada como, por exemplo, a *cudaMallocManaged*. Finalmente, a *flag* “-fast” tem a função de otimizar o código e a *flag* “-Minfo=all” faz com que o máximo de informações possíveis apareçam na tela do terminal após a compilação (MOREIRA, 2015).

3.3 Speedup, Eficiência e Escalabilidade

Apresenta-se, em seguida, as métricas e os conceitos que são normalmente utilizadas na análise de desempenho computacional na programação paralela e distribuída. Mais precisamente, serão discutidos o *Speedup*, Subseção 3.3.1, a Eficiência, Subseção 3.3.3, Subseção 3.3.3, e a Escalabilidade. Essas métricas e conceitos são fundamentais para que os programadores possam avaliar se de fato o código paralelizado deve ser levado adiante ou não (SULZBACH, 2014). No caso específico da simulação numérica de reservatórios, equipes compostas normalmente por Engenheiros, Pesquisadores e Programadores em geral tendem, com uma frequência cada vez maior, a usar códigos paralelizados a fim de reduzir o tempo excessivamente longo das simulações de problemas práticos de interesse na indústria de petróleo e gás.

3.3.1 Speedup

Speedup, ou aceleração, é a razão entre o tempo gasto na execução de um programa em sua versão serial, com um único processador e uma única *thread* (T_s) e o tempo gasto na execução com uso de técnicas de paralelização (T_p) (CHAPMAN; JOST; PAS, 2008),

$$S(P) = T_s / (T_p),$$

onde S representa o valor do *Speedup*. Uma maior ênfase é dada a essa métrica neste trabalho. Os valores da aceleração (*Speedup*) alcançada com a paralelização via a API OpenACC é o critério considerado para se definir se o objetivo aqui estipulado foi atingido ou não.

3.3.2 Eficiência

A eficiência de um código paralelizado pode ser representada pela função $E(P)$, onde P representa o número de processadores empregados na execução do programa computacional (OYARZUN et al., 2020),

$$E(P) = S(P)/P,$$

onde $S(P)$ representa o *Speedup*. Sabe-se que quanto mais próxima de 1 for a eficiência, mais eficaz será o código paralelizado. Tipicamente, o aumento do número de processadores diminui a eficiência do código e, portanto, deve-se sempre buscar uma configuração que desperdice menos os recursos disponíveis.

3.3.3 Escalabilidade

A Escalabilidade representa a capacidade de aumento do desempenho computacional de um código numérico conforme a complexidade do problema que se quer resolver também aumenta.

Um código numérico paralelizado escalável possui uma eficiência que deve permanecer constante à medida que o número de processadores usados na sua execução e a complexidade do problema a ser resolvido aumentam proporcionalmente. É sabido que, para um dado código computacional que resolve um mesmo problema, o aumento do número de processadores empregados na sua execução leva a um crescimento do *overhead* de comunicação, o que tenderia a fazer com que a eficiência diminua (SEARLES et al., 2018).

Assim, os programadores precisam levar em consideração a possibilidade do aumento da dificuldade de execução quando se aumenta o número de processadores, afim de compensar o crescimento do *overhead* de comunicação (OYARZUN et al., 2020). Caso o problema a ser resolvido tenha o dobro da complexidade inicial e sejam usadas na paralelização o dobro de processadores, o código paralelizado é dito escalável se o tempo de execução for o mesmo que o obtido com a metade da complexidade e do número de processadores.

É possível que, a depender da aplicação do código, seja mais importante a escalabilidade do que o valor do *Speedup*. Assim, cabe à equipe responsável por analisar o desempenho computacional decidir qual desses dois aspectos é o mais importante digo e se a paralelização do código é viável na prática.

4 RESULTADOS NUMÉRICOS

Passa-se, agora, à apresentação dos resultados obtidos com as simulações do escoamento não-isotérmico de óleo, empregando a versão do simulador contendo o método dos Gradientes Conjugados paralelizado via o uso da API OpenACC. Os resultados abordam tanto os aspectos do desempenho computacional quanto físico, abrangendo, por exemplo, a variação do número de células da malha computacional e o consequente aumento do percentual de uso da GPU.

Na Seção 4.1, faz-se um estudo de convergência numérica através do refinamento de malha além de uma confrontação entre os resultados obtidos com o simulador original (LOPES, 2016; LOPES; SOUZA; SOUTO, 2018), não paralelizado, e o deste trabalho. Em seguida, na Seção 4.2, discute-se especificamente o desempenho computacional advindo da aplicação do OpenACC na paralelização parcial do código numérico. Por fim, a Seção 4.3 é dedicada à análise de sensibilidade da variação de pressão e temperatura, em consequência da variação de alguns dos parâmetros de escoamento, assim como ao desempenho computacional.

A placa de vídeo utilizada na execução em paralelo do código numérico possui as seguintes características:

1. Modelo: PV-07;
2. Chipset: Nvidia Geforce GTX-750TI;
3. Tipo de memória: DDR5;
4. Tamanho efetivo da memória: 2 GB;
5. Velocidade do núcleo: 1020 MHz;
6. Clock de memória: 5400 MHz;
7. Núcleos CUDA: 640;
8. Interface: PCI Express 3.0.

Escolheu-se como caso de estudo, no contexto da recuperação terciária de hidrocarbonetos, o escoamento não-isotérmico bidimensional de óleo pesado, com a produção sendo assistida pelo uso de aquecedores estáticos. O reservatório considerado possui o formato de um paralelepípedo com dimensões L_x , L_y e L_z , sendo a última muito menor do que as demais. Para essa geometria do domínio, o número de

células utilizado nas malhas nas direções dos eixos x e y , n_x e n_y , são apresentados na Tabela 2, assim como o número total de células correspondente.

Tabela 2 – Malhas computacionais

Malha	$n_x = n_y$	Total de células
1	47	2.209
2	93	8.649
3	185	34.225
4	369	136.161
5	737	543.169
6	1473	2.169.729

Fonte: O autor, 2021.

Tendo sido apresentadas as seis malhas empregadas nas simulações, passa-se agora à apresentação dos demais parâmetros utilizados. Tais valores podem ser vistos na Tabela 3, sendo este o caso padrão adotado nesta dissertação. Então, esses parâmetros são usados em todas as simulações, a menos que seja especificado o contrário.

Tabela 3 – Caso padrão de simulação

Parâmetro	Valor	Parâmetro	Valor
L_x	6.400 m	L_y	6.400 m
L_z	40 m	n_{aq}	2
x_{aq1}	2.725 m	y_{aq1}	3.200 m
x_{aq2}	3.675 m	y_{aq2}	3.200 m
x_{prod}	3.200 m	y_{prod}	3.200 m
t_{max}	60 dias	k_x	$0,02 \mu\text{m}^2$
k_y	$0,02 \mu\text{m}^2$	ϕ^0	0,2
c_ϕ	$4,35 \times 10^{-7} \text{ kPa}^{-1}$	c_o	$7,25 \times 10^{-7} \text{ kPa}^{-1}$
$c_{\phi T}$	$4,35 \times 10^{-7} \text{ K}^{-1}$	c_{oT}	$7,25 \times 10^{-7} \text{ K}^{-1}$
$c_{p\phi}$	$1200 \text{ J}/(\text{kg} \cdot \text{K})$	κ_r	$3,5 \text{ W}/(\text{m} \cdot \text{K})$
ρ_r	$0,2500 \text{ kg}/\text{m}^3$	a	$0,2 \text{ Pa} \cdot \text{s}$
B^0	$1,3 \text{ m}^3/(\text{std m}^3)$	c_{vo}	$1800 \text{ J}/(\text{kg} \cdot \text{k})$
c_{po}	$2100 \text{ J}/(\text{kg} \cdot \text{k})$	κ_o	$0,45 \text{ W}/(\text{m} \cdot \text{K})$
Δt_{ini}	0,1 dia	Δt_{max}	1,0 (dia)
$F_{\Delta t}$	1,1	$tol_1 = tol_2$	$1,0 \times 10^{-6} \text{ kPa ou K}$
T_{ref}	500,0 K	b	600 K
$T_{inic} = T^0$	330 K	$p_{inic} = p^0$	6900,0 kPa

Fonte: O autor, 2021.

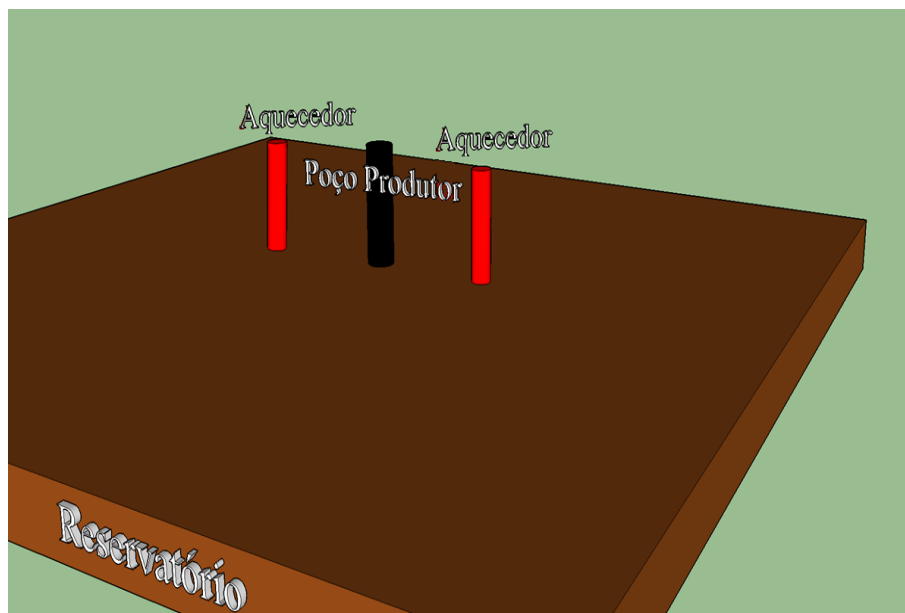
Na referida tabela, os pares de coordenadas em x e y que localizam as posições dos Aquecedores 1 e 2 são, respectivamente, (x_{aq1}, y_{aq1}) e (x_{aq2}, y_{aq2}) . Além disso,

no que diz respeito ao poço produtor, ele encontra-se posicionado nas coordenadas (x_{prod}, y_{prod}) . Por fim, o número de aquecedores é representado por n_{aq} .

O tempo máximo de produção é t_{max} , o incremento de tempo inicial é Δt_{ini} e a razão de crescimento do passo de tempo é $F_{\Delta t}$, tal que $\Delta t^{n+1} = F_{\Delta t} \Delta t^n$. O seu crescimento ocorre até que o valor Δt_{max} seja alcançado, permanecendo inalterado até o final da simulação. Procedimentos desse tipo são comuns na simulação de reservatórios, quando pretende-se ter um maior detalhamento dos resultados para os primeiros instantes de produção, de modo a, por exemplo, se estudar o comportamento inicial da pressão no poço produtor.

De modo a ilustrar o posicionamento dos poços aquecedores e produtor foi elaborada a Figura 4, que mostra o poço produtor no centro do reservatório com os dois poços aquecedores no seu entorno.

Figura 4 – Reservatório com poço produtor e aquecedores

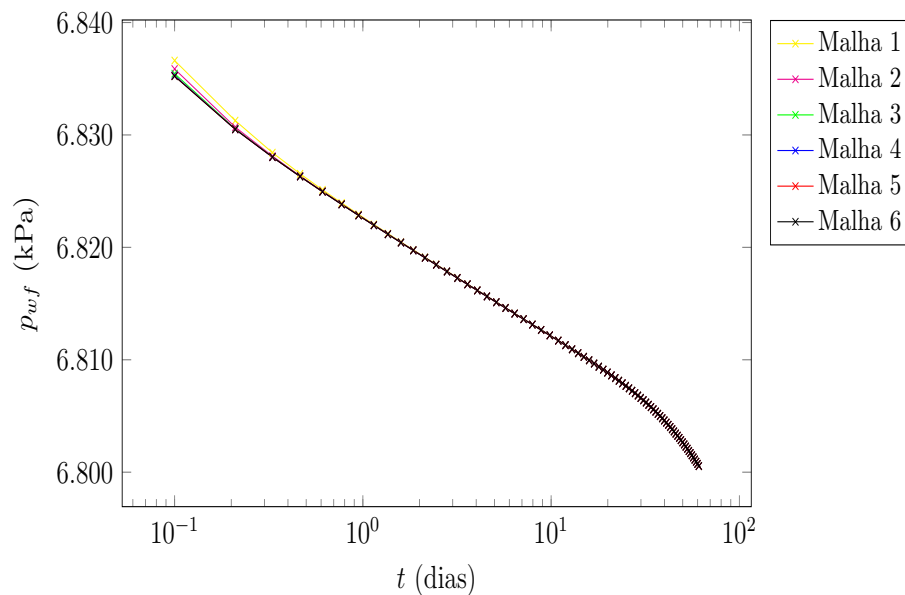


Fonte: O autor, 2021.

4.1 Verificação numérica

Primeiramente, é apresentado o resultado do estudo do refinamento de malha. Os gráficos apresentam os valores da pressão no poço produtor, p_{wf} , estimada via a técnica de acoplamento poço-reservatório utilizada, em função do tempo transcorrido t . Na Figura 5, vê-se um gráfico desse tipo, chamado de especializado, utilizado frequentemente na engenharia de petróleo.

Figura 5 – Pressão do poço produtor: seis diferentes malhas



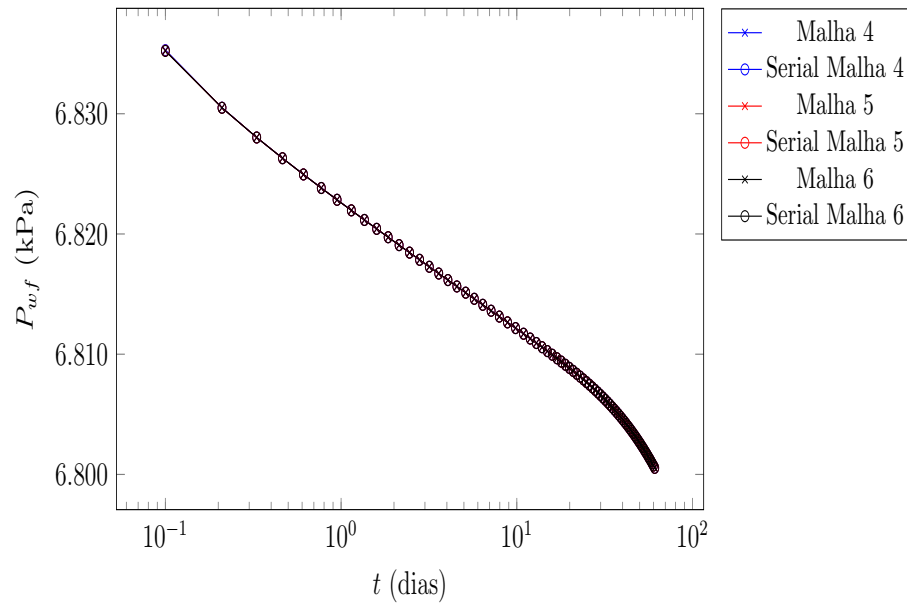
Fonte: O autor, 2021.

A partir da análise da figura, é possível notar que a partir da malha de número 4 pode-se que a convergência numérica foi verificada. Os resultados correlacionados às Malhas 4, 5 e 6 possuem curvas que aparecem praticamente sobrepostas umas as outras.

O artefato numérico, que surge em certas simulações usando a técnica de acoplamento poço-reservatório aqui empregada, para tempos curtos e permeabilidades baixas, não apareceu nas curvas mostradas na figura (SOUZA, 2013).

A seguir, é feita uma verificação com a finalidade de se constatar se a paralelização do código numérico, através do uso do OpenACC, pode ter alterado os resultados. Para tanto, compare-se os valores determinados com as versões paralelizada e serial. Deve-se chamar a atenção do leitor para o fato de que a versão serial do simulador já foi verificada anteriormente, mediante a comparação dos seus resultados com soluções analíticas, para a produção de óleo no caso do escoamento isotérmico e da condução de calor no caso isobárico (LOPES, 2016). Então, foi gerado o gráfico visto na Figura 6 e, em função dos resultados, está comprovado que a paralelização não alterou os resultados, visto que eles coincidem com os obtidos com a versão serial do simulador.

Figura 6 – Pressão do poço produtor: serial e em paralelo



Fonte: O autor, 2021.

4.2 Desempenho computacional

Neste ponto, constatada a exatidão dos resultados, prossegue-se com os estudos de modo a se verificar o desempenho computacional do simulador e o ganho em eficiência a partir do *speedup*. Foram realizadas simulações e registrados os seus respectivos tempos de duração. A Tabela 4 mostra os tempos de simulação correspondentes às execuções realizadas com a versão paralelizada ($t_{OpenACC}$) e serial (t_{Serial}).

Tabela 4 – Tempos de execução: seis diferentes malhas

Malha	t_{Serial} (s)	$t_{OpenACC}$ (s)	<i>Speedup</i>
Malha 1	3,054	13,821	0,221
Malha 2	17,861	30,427	0,587
Malha 3	112,646	85,373	1,325
Malha 4	774,619	307,675	2,518
Malha 5	5.483,819	1.445,18	3,385
Malha 6	37.763,475	8.757,326	4,312

Nota: Tempos de execução em serial e em paralelo.
Fonte: O autor, 2021.

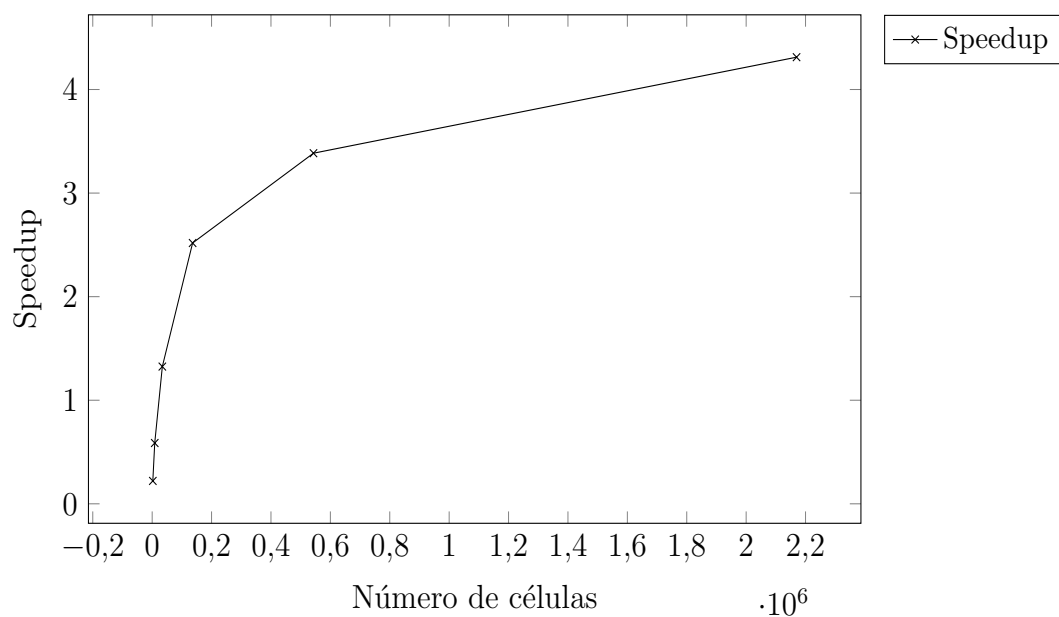
Para os casos estudados, a forma da implementação numérica e o *hardware* utilizado, conclui-se que a paralelização com uso do OpenACC não produz ganhos quando são empregadas malhas mais grosseiras, pois não há uma diminuição no

tempo de execução, pelo contrário, há um aumento desse tempo! Uma explicação para esse tipo de comportamento é devida ao fato de que o tempo ganho em eficiência durante a execução não é suficiente para compensar o tempo gasto pelo *host* na distribuição de tarefas entre os núcleos da GPU, bem como aquele destinado ao acesso das informações armazenadas nas memórias do *host* e da GPU.

Em contrapartida, quanto mais refinadas são as malhas, maior é o ganho em termos de aceleração. Vale notar que o refinamento de malha provoca o aumento exponencial das operações a serem efetuadas e, assim, a tendência de aumento do *speedup* deveria ser cada vez maior.

Para uma melhor compreensão da influência do aumento do número de células das malhas no valor do *speedup*, o gráfico da Figura 7 foi confeccionado. Da sua análise, percebe-se nitidamente que embora haja um aumento do *speedup* em função do refinamento de malha, a taxa de variação desse aumento vai diminuindo à medida que cresce o número total de células.

Figura 7 – *Speedup*: seis diferentes malhas

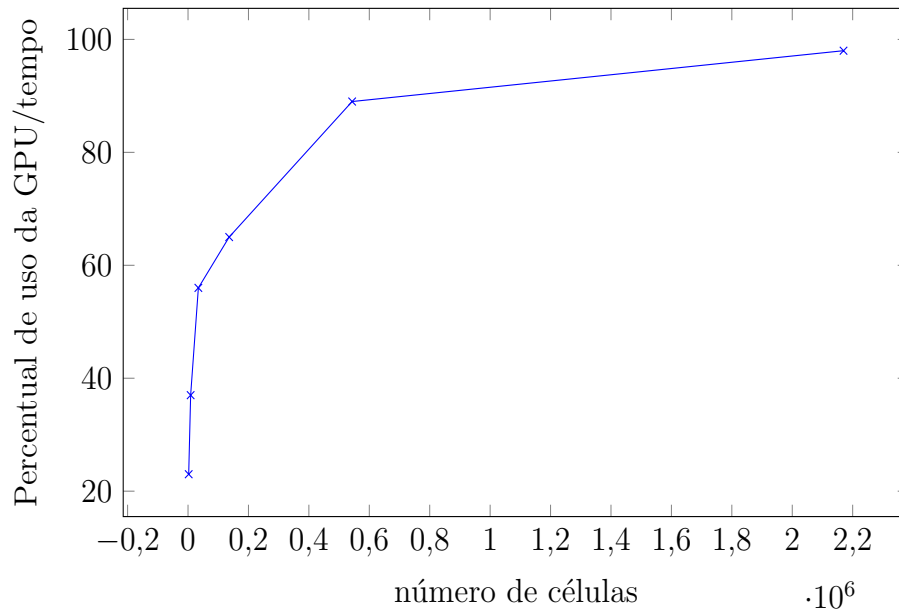


Fonte: O autor, 2021.

Outra informação relevante está relacionada ao percentual máximo de utilização da GPU por unidade de tempo registrada (geralmente 1 segundo, mas pode variar de acordo com a placa). Portanto, gerou-se o gráfico da Figura 8, que mostra a variação do percentual de utilização da GPU em função do aumento da quantidade de tarefas a serem realizadas, provocado pelo refinamento de malha. Nela, é possível notar o aspecto muito semelhante ao da Figura 7 e, portanto, pode-se perceber que a diminuição da taxa de crescimento do valor de *speedup* é acompanhada pelo aumento

do percentual de tempo de utilização da GPU, indicando que a sua capacidade limite de processamento está sendo alcançada.

Figura 8 – Uso da GPU por unidade de tempo



Fonte: O autor, 2021.

A Figura 9 mostra uma imagem do aplicativo da Nvidia no momento em que foi registrado o percentual de uso da GPU, além do já comentado aumento da *GPU utilization*, foi observado ao longo das simulações o aumento também do uso da memória dedicada da placa gráfica. No momento da captura de tela ela chegou a 98%. Indicando, mais uma vez, que os recursos do *hardware* da placa de vídeo estão próximos dos seus valores máximos.

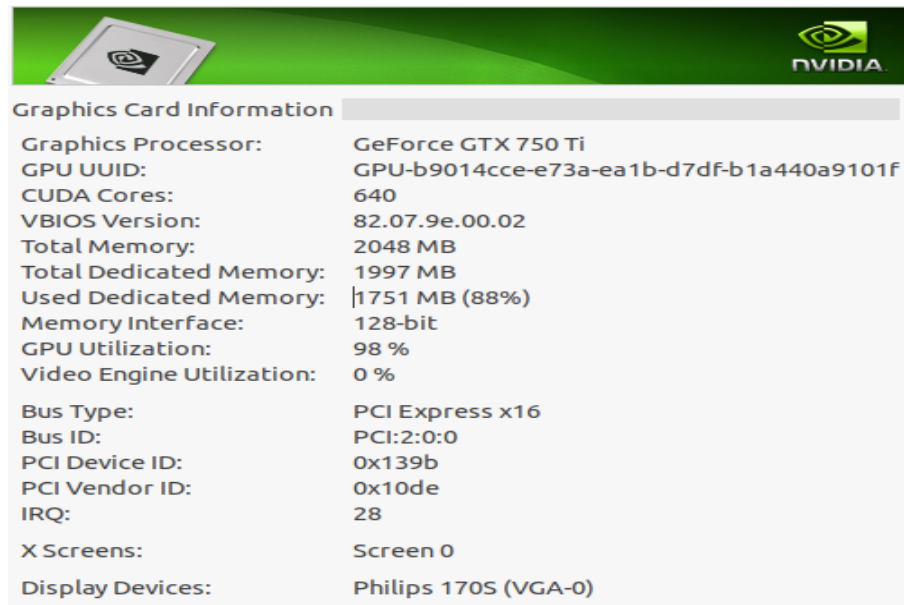
4.3 Análise de sensibilidade

Seção dedicada à análise de sensibilidade e desempenho computacional. Estuda-se as variações de pressão e temperatura em função da alteração de alguns dos principais parâmetros que caracterizam o escoamento não-isotérmico.

4.3.1 Pressão no poço produtor

Para analisar a aplicabilidade do simulador sob condições mais severas, utilizou-se a Malha 4 em um tempo de produção de 1.000 dias. A principal finalidade desse

Figura 9 – Imagem do NVIDIA X server

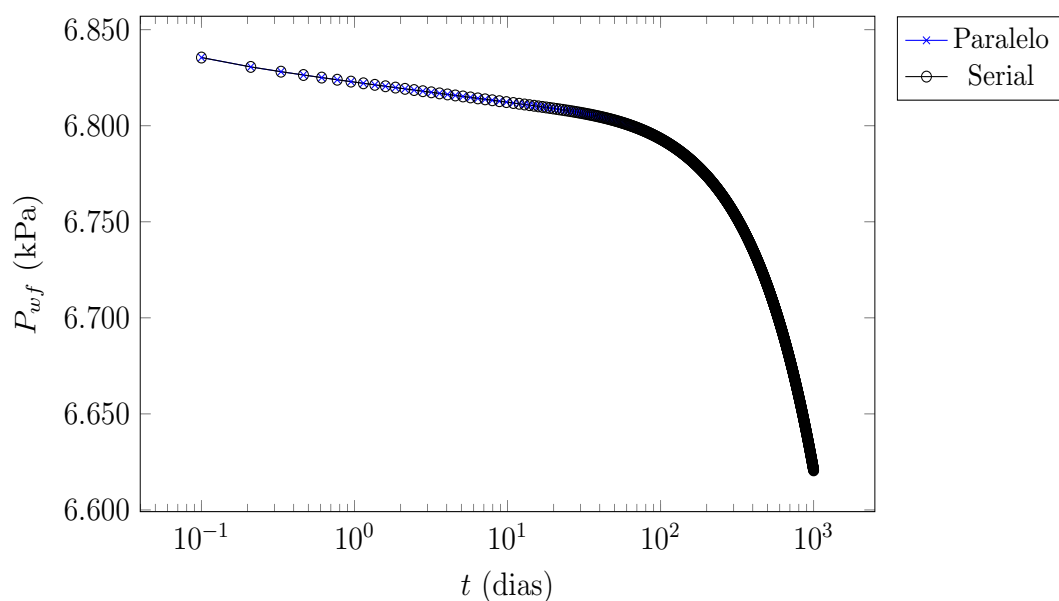


Fonte: O autor, 2021.

teste foi a de se certificar que a versão paralelizada forneceria os mesmos valores e comportamento, para a pressão no poço produtor, do aqueles determinados com a versão serial para um tempo maior de produção.

Da observação dos resultados da Figura 10, nota-se que os valores obtidos com as duas versões do simulador estão sobrepostos. Além disso, o perfil da pressão do poço condiz com aqueles previstos na literatura (ROSA; CARVALHO; XAVIER, 2006).

Figura 10 – Pressão do poço produtor: 1.000 dias de produção



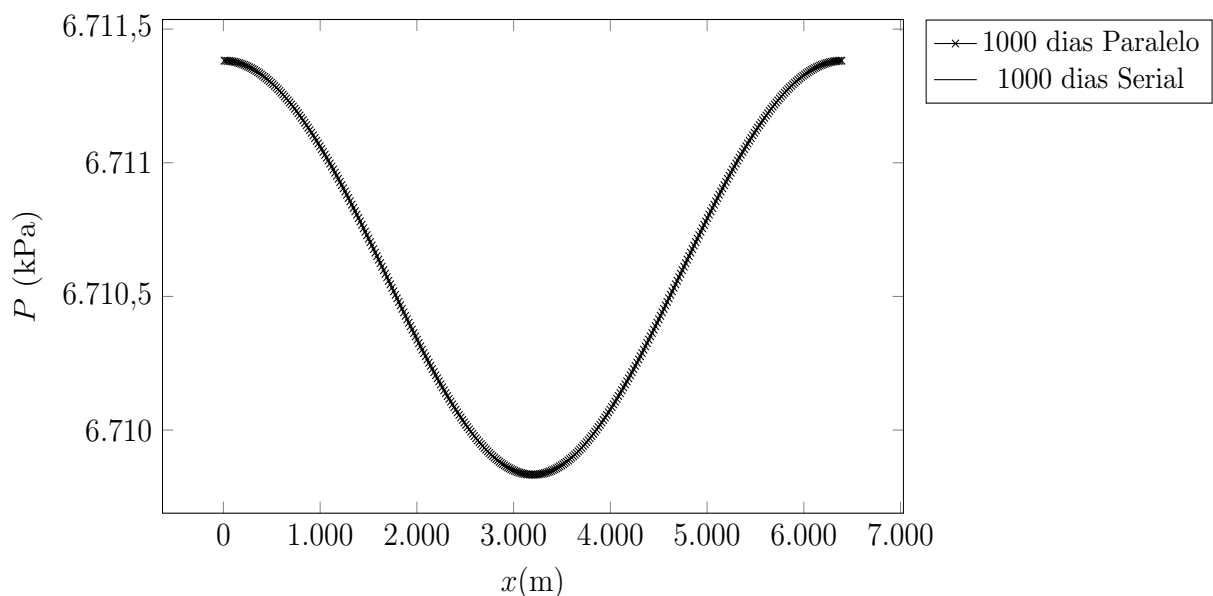
Fonte: O autor, 2021.

4.3.2 Pressão no reservatório

De modo a se verificar a exatidão do perfil de pressão no reservatório, ao longo do eixo x e após transcorridos 1.000 dias, os valores oriundos das duas versões do simulador foram comparados, vide a Figura 11.

Levando em conta que o poço produtor está posicionado no centro do reservatório, no plano xy , e que os aquecedores estão localizados na mesma abscissa y e encontram-se equidistantes do poço produtor (com relação a x), o perfil de pressão deveria ser simétrico. Isto é comprovado pelo perfil de pressão na Figura 11, sabendo-se que a pressão deve ser mínima no poço produtor, $x=3.200$ m, e máxima nas extremidades do reservatório. Portanto, há uma queda progressiva da pressão à medida que se aproxima do centro do reservatório, até que o seu valor mínimo seja atingido no poço produtor.

Figura 11 – Pressão do reservatório: ao longo do eixo x

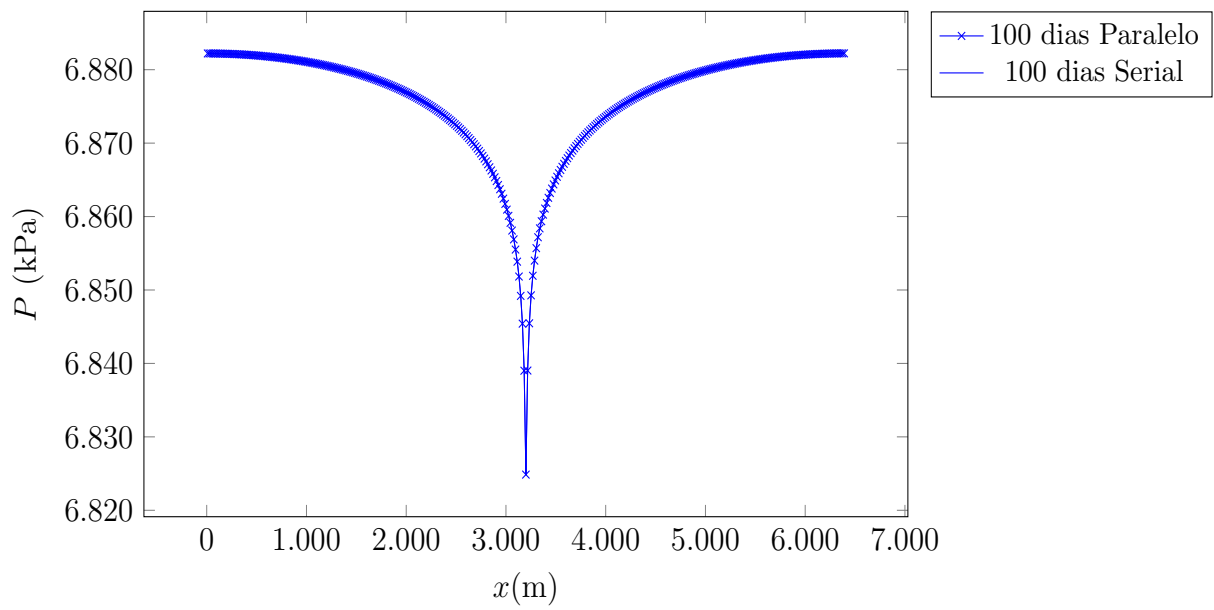


Fonte: O autor, 2021.

A seguir, de modo a se acompanhar a evolução do perfil de pressão até que sejam alcançados os 1.000 dias de simulação, foram geradas as curvas das Figuras 12, 13 e 14, tratando-se dos resultados correspondentes a 100, 500 e 1.000 dias de produção. Nelas, pode-se observar que há uma queda gradativa do valor da pressão, ao longo do reservatório, com o passar dos dias.

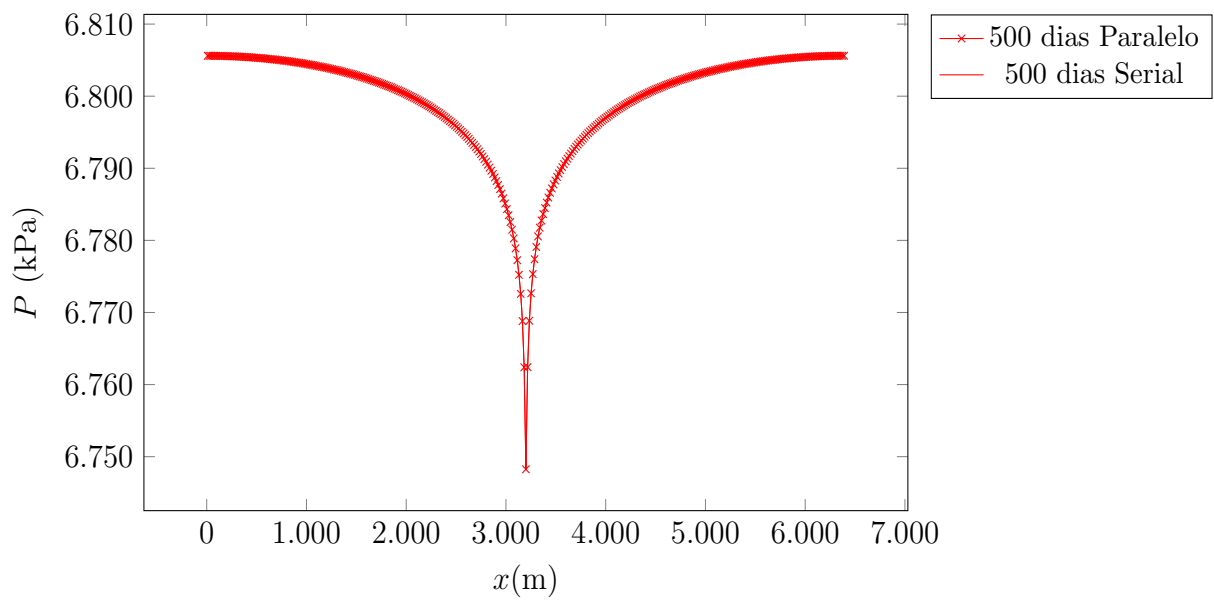
Além disso, constatou-se para os tempos de execução registrados os valores de *speedup* mostrados na Tabela 5, todos para a Malha 4 e 100, 500 e 1.000 dias de

Figura 12 – Pressão do reservatório: 100 dias de produção



Fonte: O autor, 2021.

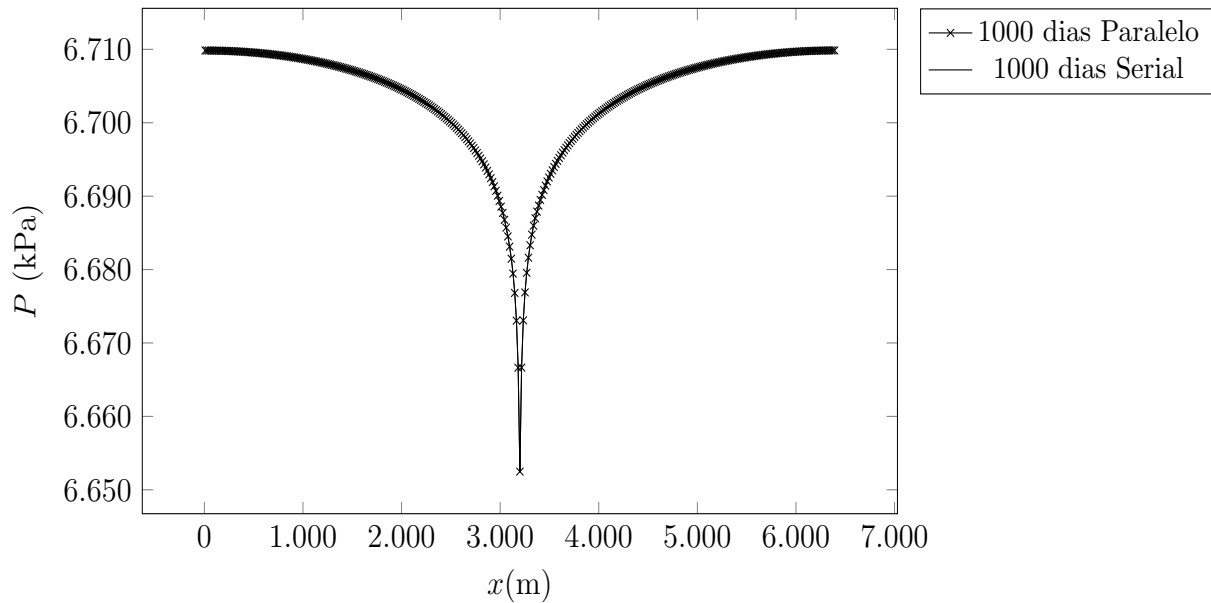
Figura 13 – Pressão do reservatório: 500 dias de produção



Fonte: O autor, 2021.

produção, respectivamente.

Figura 14 – Pressão do reservatório: 1.000 dias de produção



Fonte: O autor, 2021.

Tabela 5 – *Speedup*: 100, 500 e 1.000 dias de produção

Tempo de produção (dias)	<i>Speedup</i>
100	2,748
500	2,679
1.000	2,571

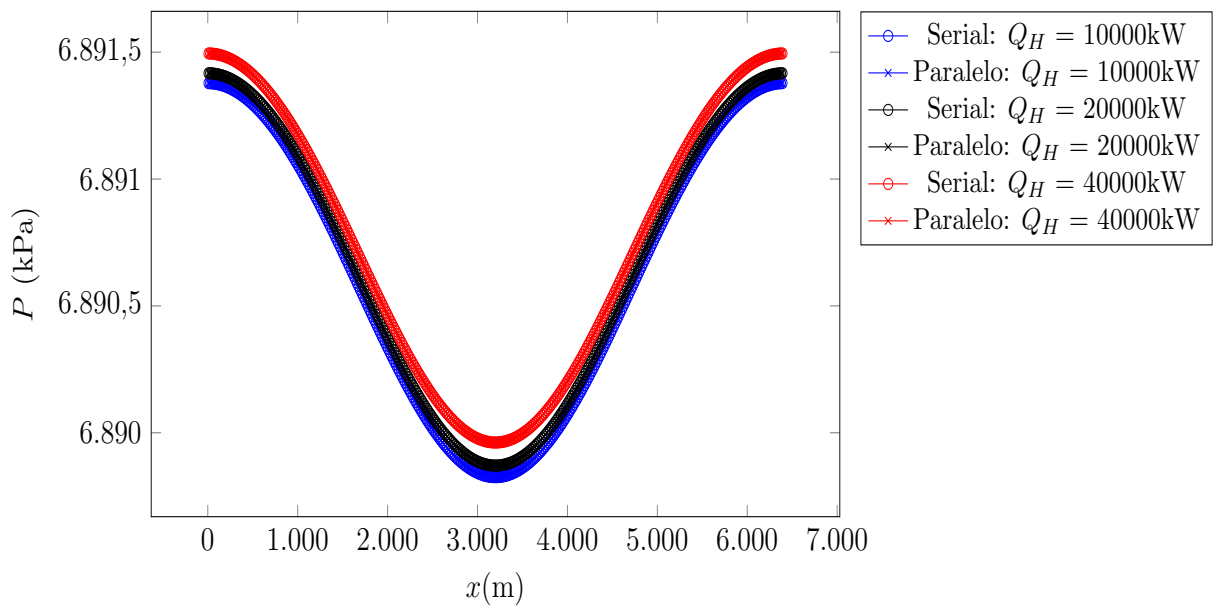
Fonte: O autor, 2021.

Percebe-se que os valores da tabela estão relativamente próximos ao *speedup* de 2,518 determinado para um período de produção de 60 dias. Entende-se que o tipo de critério escolhido para a evolução do passo de tempo deve ter influenciado esses valores. No entanto, mais simulações do tipo ainda seriam necessárias para que uma conclusão seja extraída.

Para se estudar as consequências da mudança da taxa de aquecimento (Q_H) gerada pelos dois aquecedores, pensou-se em realizar simulações onde o seu valor fosse alterado de 20 kW (caso padrão) para 10 kW e, em seguida, para 40 kW. Neste estudo, mostra-se a variação de pressão no reservatório para diferentes ordenadas x e para uma abscissa fixa: $y=6.400$ m (região de fronteira do reservatório). Os resultados desse estudo encontram-se no gráfico da Figura 15, para 100 dias de produção.

Após decorridos os 100 dias de produção, verifica-se que as mudanças em Q_H

Figura 15 – Pressão do reservatório: diferentes Q_H e $y=6.400$ m



Fonte: O autor, 2021.

faz com que a queda no valor da pressão no reservatório, ao longo do eixo x , vá diminuindo à medida que a potência nos aquecedores é elevada. Ou seja, para um $Q_H=10$ kW tem-se a maior queda de pressão no reservatório ao passo que com 40 kW tem-se a menor, favorecendo a produção no reservatório.

Com relação aos valores de *speedup*, eles podem ser vistos na Tabela 6. Assim, nota-se que o aumento do valor de Q_H não implicou, necessariamente, numa tendência de aumento do valor do *speedup* para os testes realizados.

Tabela 6 – *Speedup*: 10, 20 e 40 kW

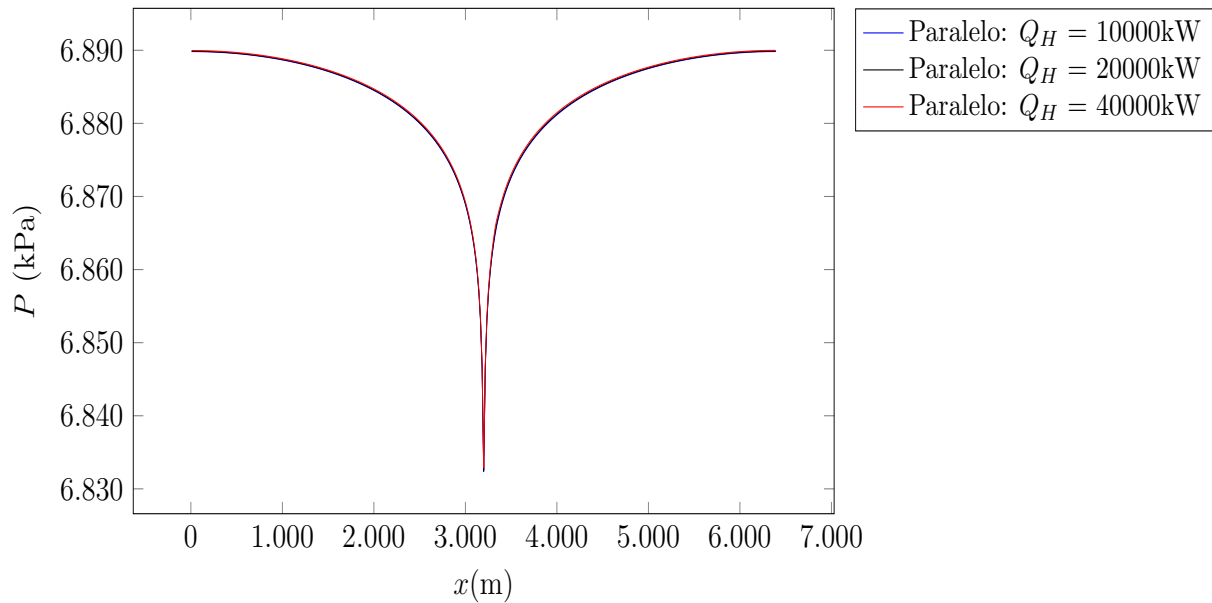
Potência do Aquecedor (kW)	<i>Speedup</i>
10	3,319
20	2,518
40	2,780

Fonte: O autor, 2021.

Em seguida, de modo semelhante, mas desta vez fixando-se a abscissa y em 3.200 m (centro do reservatório), foram gerados os novos gráficos para os perfis de pressão ao longo de x , vide a Figura 16.

Consequentemente, vê-se que o aumento de Q_H exerce uma pequena influência no perfil de pressão ao longo da faixa central do reservatório. Porém, o mesmo não pode ser dito com relação aos seus efeitos na região de fronteira do reservatório. Ou

Figura 16 – Pressão do reservatório: diferentes Q_H e $y=3.200$ m

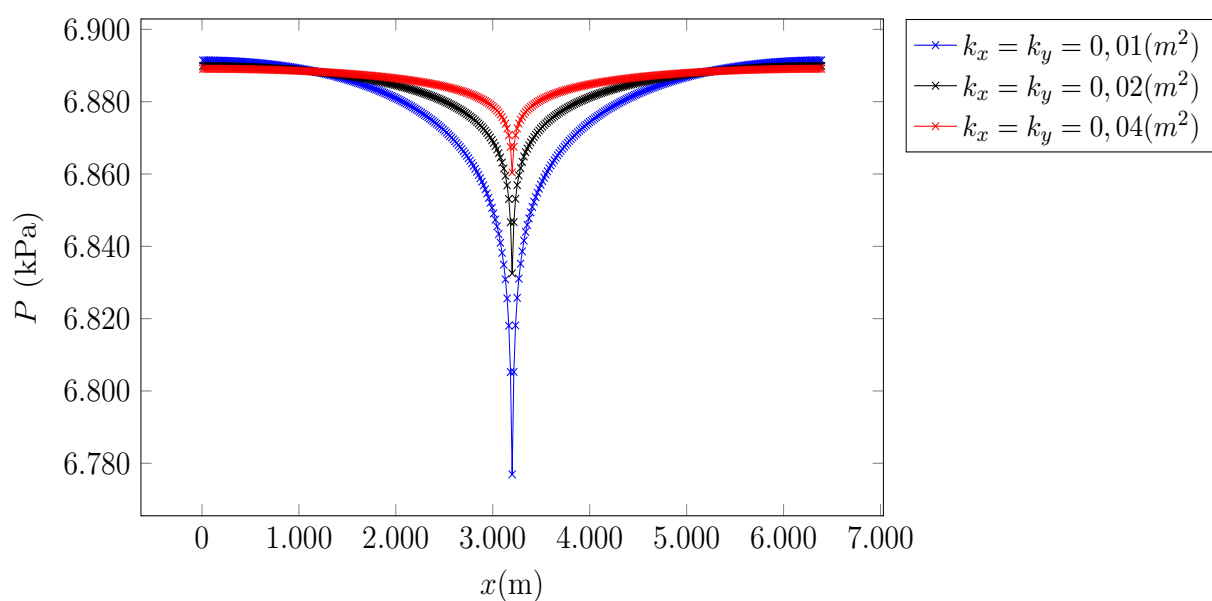


Fonte: O autor, 2021.

seja, a taxa de transferência de calor não influenciou significativamente os resultados para $y=3.200$ m, contrariamente ao que se viu para $y=6.400$ m.

Em seguida, de modo a verificar-se o comportamento dos perfis de pressão, ao longo do eixo x , quando da variação de permeabilidade e para $y=3.200$ m, gerou-se o gráfico da Figura 17.

Figura 17 – Pressão do reservatório: diferentes permeabilidades

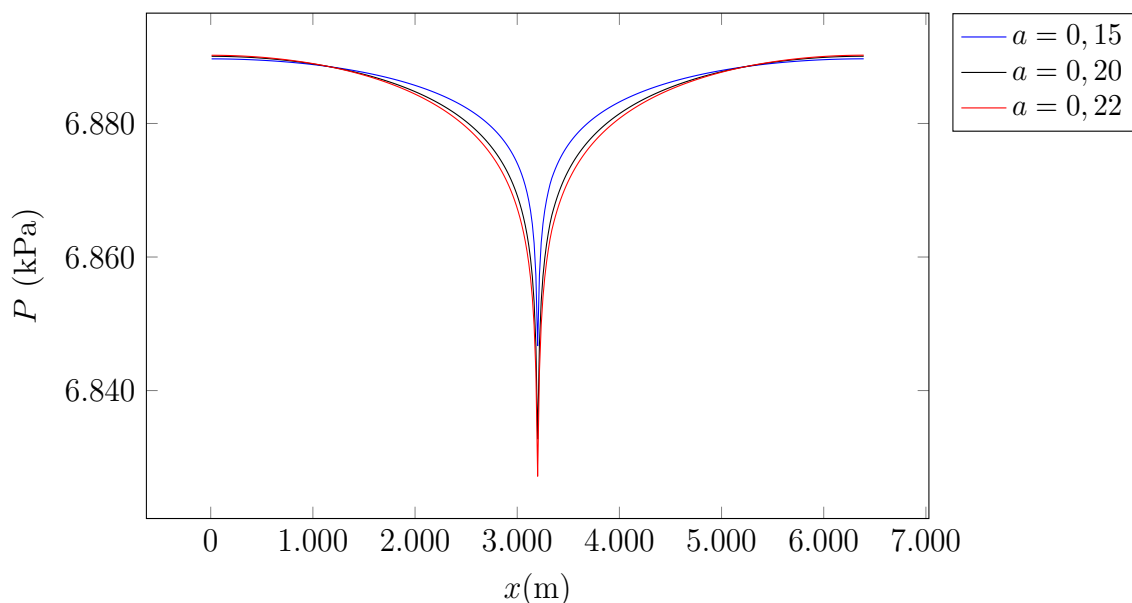


Fonte: O autor, 2021.

Para um tempo de simulação padrão (60 dias), o aumento da permeabilidade nas duas direções principais do reservatório (x e y) provoca uma diminuição na queda de pressão na região do poço produtor e um estreitamento do gráfico, pois há uma maior facilidade para que o fluido escoe no reservatório, de acordo com o previsto pela teoria e os experimentos para o escoamento em meios porosos (ROSA; CARVALHO; XAVIER, 2006). Também ressalta-se que quanto maiores forem os valores da permeabilidade, mais cedo ocorrerão os chamados efeitos de fronteira, conforme o esperado.

Ainda no âmbito dos testes de variação dos principais parâmetros, alterou-se os valores do coeficiente a de modo a modificar-se a viscosidade do fluido. Assim, na Figura 18 tem-se as curvas de pressão no reservatório, para o plano xy fixado no centro do reservatório, para três valores distintos da viscosidade, ou seja, do coeficiente a da Eq. (10). Quanto maior o valor desse coeficiente maior será o valor da viscosidade.

Figura 18 – Pressão do reservatório: diferentes viscosidades



Fonte: O autor, 2021.

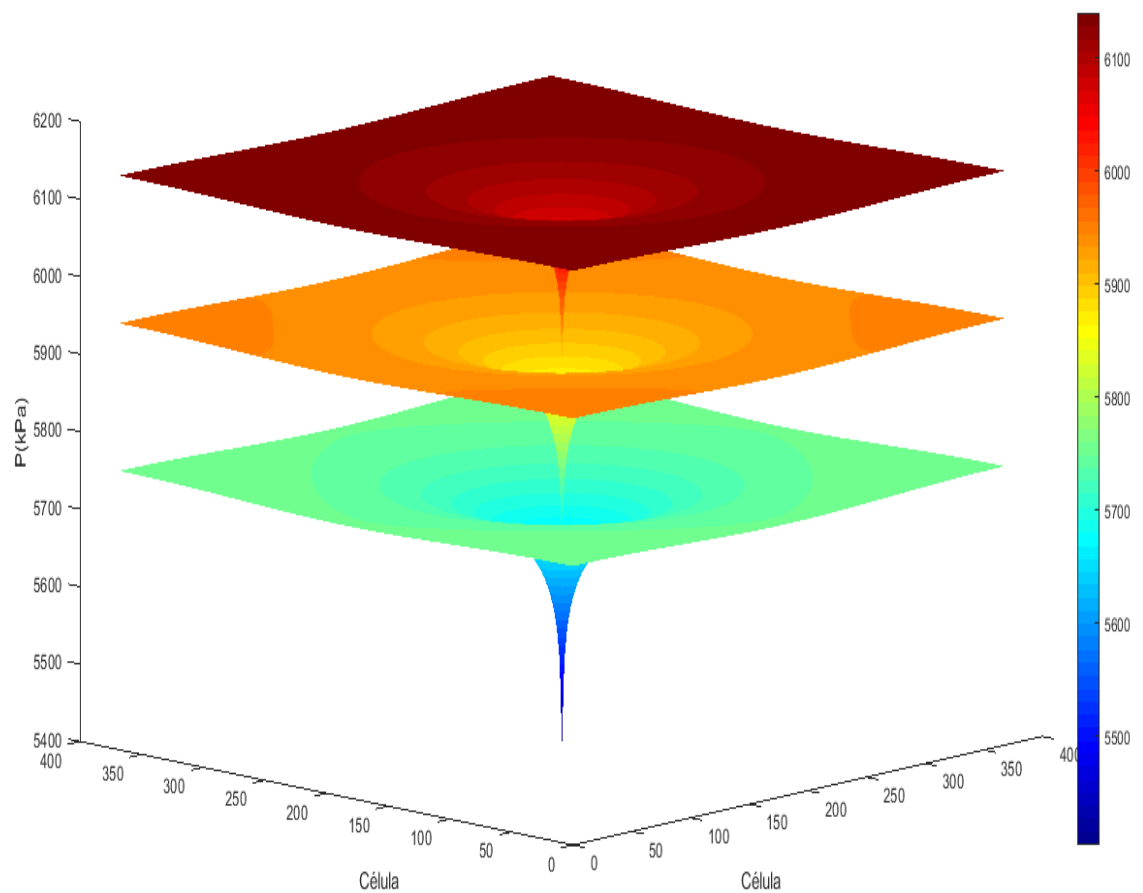
Então, conforme pode-se ver através das curvas apresentadas na Figura 18, o aumento do valor do parâmetro a (da viscosidade) faz com que o perfil de pressão tenda a ter sua queda mais acentuada ao redor do centro do reservatório, aonde encontra-se posicionado o poço produtor, e um efeito contrário quando da sua diminuição.

O gráfico da Figura 19 mostra os campos de pressão correspondentes a diferentes valores da vazão de produção. Observa-se, de cima para baixo, os gerados

com as vazões de 200 std m³/dia, 250 std m³/dia e 300 std m³/dia. Nestas simulações empregou-se a Malha 4 ($nx=ny=369$) e os eixos coordenados apresentam no plano xy as células da malha computacional e os valores de pressão no eixo z . As superfícies foram criadas para 1.000 dias de simulação.

Assim, percebe-se que, conforme o previsto pela física, o aumento no valor da vazão de produção provoca uma diminuição no valor da pressão em todo o reservatório. Além do mais, pode-se ver ainda que os menores valores de pressão estão sempre presentes na célula que contém o poço produtor.

Figura 19 – Campos de pressão: diferentes vazões

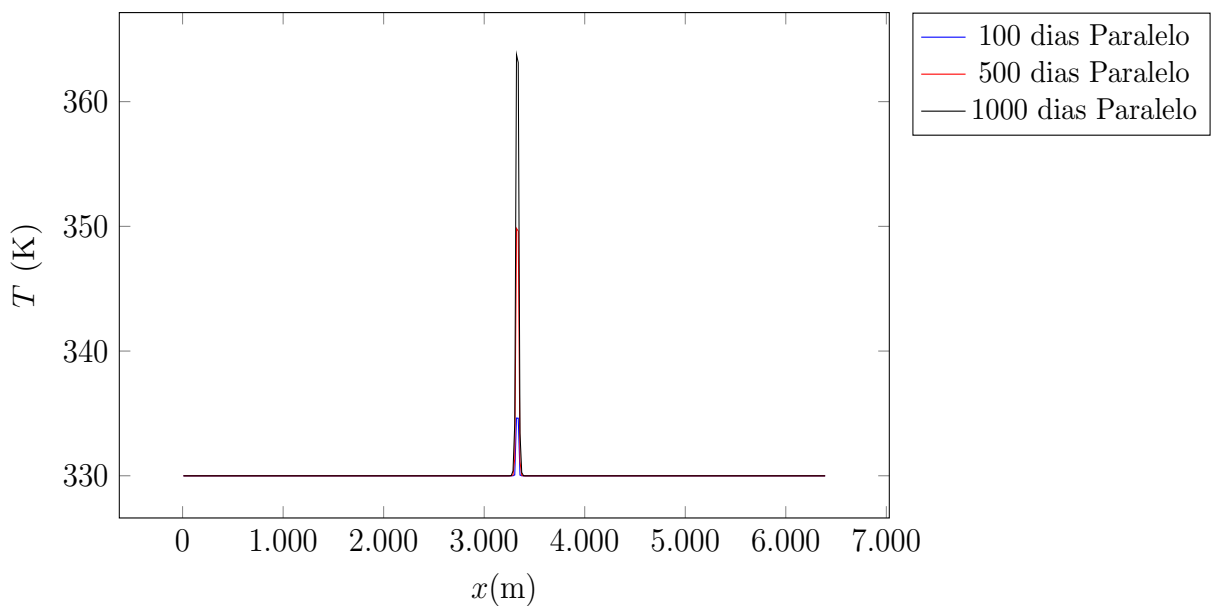


Fonte: O autor, 2021.

4.3.3 Temperatura no reservatório

Agora, passa-se ao resultados relacionados ao perfil de temperatura no reservatório. Primeiramente, reporta-se ao gráfico da Figura 20, que mostra o perfil de temperatura no reservatório, ao longo de x , transcorridos 100, 500 e 1.000 dias de simulação e para o centro do reservatório: $y=3.200$ m. Nota-se que na região aonde estão localizados os poços aquecedores há um aumento gradual do valor da temperatura à medida que os dias passam (100, 500 e 1.000 dias).

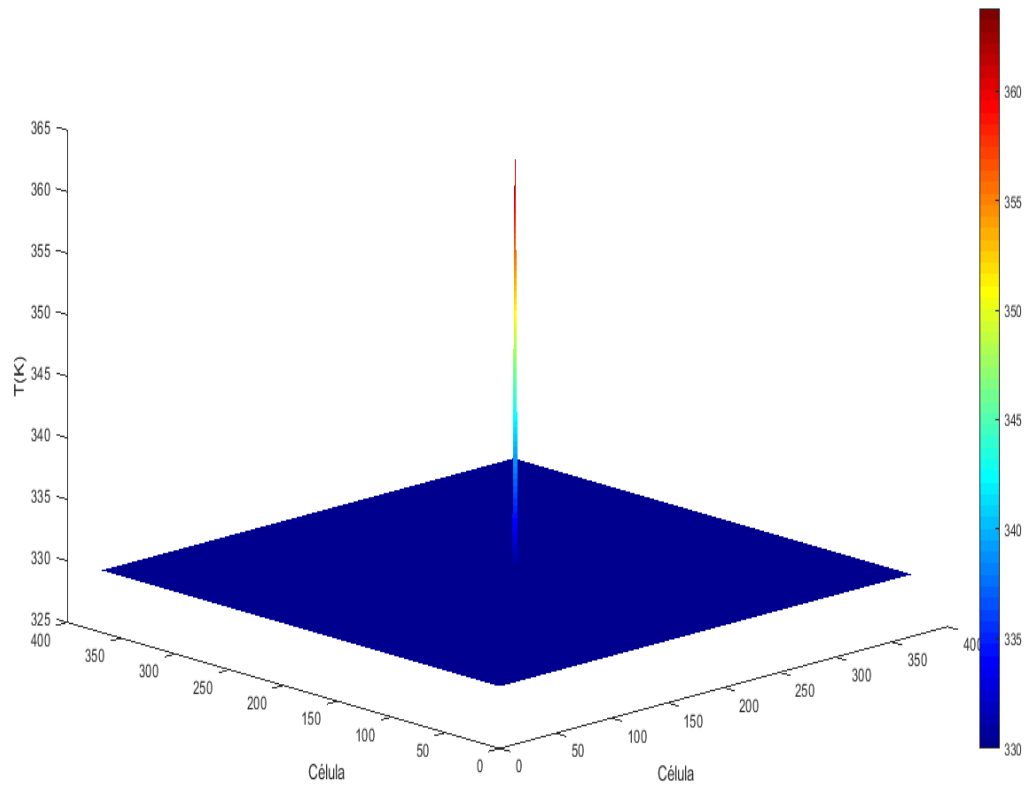
Figura 20 – Temperatura do reservatório: 100, 500 e 1.000 dias



Fonte: O autor, 2021.

Finalmente, foi produzido um gráfico tridimensional, Figura 21, cujos eixos coordenados x , y e z representam, respectivamente, as células da malha computacional em x , as respectivas células em y e o valor da temperatura no reservatório. Neste, as simulações foram realizadas com os mesmos parâmetros que os utilizados na determinação dos resultados da Figura 19. Entretanto, como há uma sobreposição das superfícies geradas, o único resultado perceptível na figura é o correspondente a uma vazão de produção de $300 \text{ std m}^3/\text{dia}$. Percebe-se que, praticamente, a única região que tem a sua temperatura aumentada passados 1.000 dias é a região vizinha ao poço produtor. A maior temperatura registrada foi 367 K, ou seja, 37 K a mais do que o valor inicial da temperatura no reservatório.

Figura 21 – Campos de temperatura: diferentes vazões



Fonte: O autor, 2021.

CONCLUSÕES E TRABALHOS FUTUROS

Após terem sido feitas a modelagem físico-matemática do escoamento não-isotérmico de óleo, a implementação da versão paralelizada do método dos Gradientes Conjugados, via a API OpenACC, e a obtenção e discussão dos resultados, sintetiza-se agora as principais conclusões tiradas. Também discute-se algumas propostas que possam vir a serem consideradas em trabalhos futuros.

As conclusões estão fundamentalmente direcionadas à questão do desempenho computacional do simulador, modificado através da introdução das diretivas da API OpenACC.

Quanto às proposições para trabalhos futuros, pensou-se em aproveitar a experiência aqui adquirida para a paralelização de outros códigos preexistentes, ou de partes deles, no sentido de se disponibilizar versões paralelizadas implementadas com técnicas e métodos distintos, de modo a se abordar outros problemas de interesse na engenharia de petróleo e gás.

Conclusões

A partir das modificações realizadas no simulador numérico, foi possível se obter ganhos de performance computacional, com aumento do *speedup* quando do refinamento de malha, para a simulação numérica do escoamento não-isotérmico de óleo em meios porosos.

Todo o processo foi realizado no contexto do uso da API OpenACC na paralelização do método dos Gradientes Conjugados. Sendo que as execuções foram realizadas em placa gráfica NVIDIA Geforce GTX 750 Ti.

Conforme já avançado, constatou-se uma tendência de aumento do *speedup* à medida que aumentou-se o número total células das malhas computacional. Como consequência, as simulações empregando a Malha 6 foram as que apresentaram o maior *speedup*. Tal fato já esperado, por se tratar da malha com o maior grau de refinamento e cujo o esforço computacional tenderia a ser o maior. Entretanto, entende-se que em função do *hardware* utilizado não se pode afirmar que a tendência de ganho em desempenho computacional se manteria para malhas ainda mais refinadas. Acredita-se que com placas mais “robustas” os resultados seriam mais expressivos em termos do *speedup*.

Foi comprovado que a paralelização do código computacional não prejudicou em nada na obtenção de resultados fisicamente corretos, uma vez que em todos os casos testados eles foram confrontados com aqueles calculados com a versão serial do simulador, já testada anteriormente por outros autores, e nenhuma diferença significativa foi observada entre eles.

Quanto à captura dos efeitos físicos, foram observados comportamentos condizentes com os problemas estudados e amplamente relatados na literatura. Assim, é possível afirmar que houve sucesso na tentativa de se obter uma versão paralelizada.

Trabalhos Futuros

Em função do conhecimento e dos resultados adquiridos ao longo do desenvolvimento do presente trabalho, algumas perspectivas surgiram e elas são passíveis de serem aproveitadas na realização de trabalhos futuros.

Uma primeira sugestão para que possam ser alcançados alguns de avanços é a de aprofundar a programação usando a API OpenACC, de maneira a melhorar a transferência de dados e o processamento entre o *host* e a placa gráfica (PICKERING et al., 2015). Além de se almejar a utilização de um *hardware* mais robusto nos testes futuros.

Por outro lado, também seria interessante se pensar na paralelização dos simuladores utilizando as bibliotecas da API CUDA (ZAZA et al., 2016).

Ainda, devido aos recentes estudos e ao crescente interesse de parte da indústria, há a perspectiva de se aplicar a paralelização usando a API OpenACC na resolução numérica do escoamento em reservatórios não convencionais do tipo: *tight* e *shale* (WANG et al., 2017).

Do ponto de vista físico-matemático, seria também pertinente o estudo de problemas de escoamento envolvendo, por exemplo, a existência de fraturas (KAMARI et al., 2015) ou escoamentos multifásicos (SIAVASHI et al., 2014).

REFERÊNCIAS

- ABOU-KASSEM, J. H.; ALI, S. M. F.; ISLAM, M. R. *Petroleum Reservoir Simulation, A Basic Approach*. Houston, USA: Gulf Publishing Company, 2006.
- ADO, M. R.; GREAVES, M.; RIGBY, S. P. Numerical simulation of the impact of geological heterogeneity on performance and safety of thai heavy oil production process. *Journal of Petroleum Science and Engineering*, v. 173, p. 1130 – 1148, 2019.
- AMARAL, V. et al. Programming languages for data-intensive hpc applications: A systematic mapping study. *Parallel Computing*, v. 91, p. 102584, 2020.
- AMIRIAN, E.; DEJAM, M.; CHEN, Z. Performance forecasting for polymer flooding in heavy oil reservoirs. *Fuel*, v. 216, p. 83 – 100, 2018.
- AOUIZERATE, G.; DURLOFSKY, L. J.; SAMIER, P. New models for heater wells in subsurface simulations, with application to the in situ upgrading of oil shale. *Computational Geoscience*, v. 18, n. 3, p. 183–194, 2015.
- AZIZ, M.; SETTARI, A. *Petroleum Reservoir Simulation*. New York, USA: Elsevier Applied Science, 1979.
- BARRETT, R. et al. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. 2. ed. USA: Society of Industrial and Applied Mathematics, 1993.
- BERA, A.; BABADAGLI, T. Status of electromagnetic heating for enhanced heavy oil/bitumen recovery and future prospects: A review. *Applied Energy*, v. 151, p. 206 – 226, 2015.
- BOURDET, D. *Well Test Analysis: the Use of Advanced Interpretation Models*. Amsterdam: Elsevier, 2002. (Handbook of Petroleum Exploration and Production 3).
- CARPEN-AMARIE, A.; HUNOLD, S.; TRÄFF, J. On expected and observed communication performance with mpi derived datatypes. *Parallel Computing*, v. 69, p. 98–117, 2017.
- CHAPMAN, B.; JOST, G.; PAS, R. van der. *Using OpenMP: Portable Shared Memory Parallel Programming*. Cambridge, USA: Massachusetts Institute of Technology, 2008.
- CHEN, Y. et al. A preliminary feasibility analysis of in situ combustion in a deep fractured-cave carbonate heavy oil reservoir. *Journal of Petroleum Science and Engineering*, v. 174, p. 446 – 455, 2019.
- CHEN, Z.; HUAN, G.; MA, Y. *Computational Methods for Multiphase Flows in Porous Media*. Philadelphia, USA: Society of Industrial and Applied Mathematics, 2006.
- CLASS, H. *Models for Non-Isothermal Compositional Gas-Liquid Flow and Transport in Porous Media*. Stuttgart, Germany: Fakultät für Bau- und Umweltingenieurwissenschaften der Universität Stuttgart, 2007.

- CREMON, M. A.; GERRITSEN, M. G. Multi-level delumping strategy for thermal enhanced oil recovery simulations at low pressure. *Fluid Phase Equilibria*, v. 528, p. 112850, 2021.
- DAKE, L. P. *The Practice of Reservoir Engineering (Revised Edition)*. Amsterdam, The Netherlands: Elsevier, 2001. Developments in Petroleum Science 36.
- DANDEKAR, A. Y. *Petroleum Reservoir Rock and Fluid Properties*. USA: CRC Press, 2013.
- DARCY, H. *Les fontaines publiques de la ville de Dijon. Exposition et application des principes à suivre et des formules à employer dans les questions de distribution d'eau*. [S.l.]: Victor Dalmont, 1856.
- DEBOSSAM, J. G. et al. Numerical simulation of single-phase flow in naturally fractured oil reservoirs. *Coupled Systems Mechanics*, p. 129–146, 2019.
- DYRDAHL, J. *Thermal flow in fractured porous media and operator splitting*. Dissertação (Mestrado) — Norwegian University of Science and Technology, Trondheim, Norway, 2014.
- ERTEKIN, T.; ABOU-KASSEM, J.; KING, G. *Basic Applied Reservoir Simulation*. Richardson, USA: Society of Petroleum Engineers, 2001.
- HERINGER, J. D. S. et al. Numerical simulation of non-isothermal flow in oil reservoirs using a two-equation model. *Coupled Systems Mechanics*, p. 147–168, 2019.
- HESTENES, M. R.; STIEFEL, E. Method of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau Standards*, v. 49, n. 6, p. 409–436, 1952.
- ISLAM, M. R. et al. *Advanced Petroleum Reservoir Simulation*. 1. ed. Salem, USA: Scrivener Publishing LLC., 2010.
- JACQUE, K. *Extensão da Ferramenta de Apoio à Programação Paralela (F.A.P.P.) para ambientes paralelos virtuais*. Dissertação (Mestrado) — Universidade de São Paulo, São Carlos, 1999.
- KAMARI, A. et al. On the evaluation of fast-sagd process in naturally fractured heavy oil reservoir. *Fuel*, v. 143, p. 155 – 164, 2015.
- KIM, J. Y.; KANG, J.-S.; JOH, M. GPU acceleration of MPAS microphysics WSM6 using OpenACC directives: Performance and verification. *Computers & Geosciences*, v. 146, p. 104627, 2021.
- KOU, J.; SUN, S. On iterative IMPES formulation for two-phase flow with capillarity in heterogeneous porous media. *International Journal of Numerical Analysis and Modeling*, v. 1, n. 1, p. 20–40, 2004.
- LAKE, W. L. *Enhanced Oil Recovery*. Englewood Cliffs: Prentice-Hall, 1989.
- LEE, S. et al. Performance portability study for massively parallel computational fluid dynamics application on scalable heterogeneous architectures. *Journal of Parallel and Distributed Computing*, v. 129, p. 1 – 13, 2019.

- LESHCHINSKIY, D. V. et al. Parallel implementation of a numerical method for solving a three-dimensional transport equation for a mesoscale meteorological model. *Procedia Computer Science*, v. 178, p. 47 – 54, 2020. 9th International Young Scientists Conference in Computational Science, YSC2020, 05-12 September 2020.
- LOPES, R. B. *Simulação Numérica de Escoamento Não-isotérmico em Reservatório de Óleo*. [S.l.], 2016.
- LOPES, R. B.; SOUZA, G.; SOUTO, H. P. A. Simulação numérica de escoamento não isotérmico em reservatórios de Óleo com poços aquecedores. *Revista Brasileira de Engenharia e Física Aplicada*, v. 3, p. 9–24, 2018.
- LOSADA, N. et al. Portable application-level checkpointing for hybrid MPI-OpenMP applications. *Procedia Computer Science*, v. 80, p. 19–29, 2016.
- MA, Y.; CHEN, Z. Parallel computation for reservoir thermal simulation of multicomponent and multiphase fluid flow. *Journal of Computational Physics*, v. 201, n. 1, p. 224 – 237, 2004.
- MAES, J. et al. Modelling in-situ upgrading of heavy oil using operator splitting method. *Computational Geoscience*, v. 18, n. 3, p. 183–194, 2015.
- MARQUEZ, S. G. et al. Delineation of most efficient recovery technique for typical heavy oil reservoir in the middle east region through compositional simulation of temperature-dependent relative permeabilities. *Journal of Petroleum Science and Engineering*, v. 186, p. 106725, 2020.
- MINDEN, V. *Improved Iterative Methods for NAPL Transport Through Porous Media*. Dissertação (Mestrado) — Tufts University, Medford, USA, 2012.
- MOHAMMADI, K.; AMELI, F. Toward mechanistic understanding of fast sagd process in naturally fractured heavy oil reservoirs: Application of response surface methodology and genetic algorithm. *Fuel*, v. 253, p. 840 – 856, 2019.
- MOREIRA, K. C. A. *Anotação automática de código com diretivas OpenACC*. Dissertação (Mestrado) — Universidade Federal de Minas Gerais, Belo Horizonte, 2015.
- MOYNE, C. et al. Thermal dispersion in porous media: one-equation model. *International Journal of Heat and Mass Transfer*, v. 43, n. 20, p. 3853–3867, 2000.
- NORMAN, M. et al. A case study of CUDA FORTRAN and OpenACC for an atmospheric climate kernel. *Journal of Computational Science*, v. 9, p. 1 – 6, 2015. Computational Science at the Gates of Nature.
- OPENACC ORGANIZATION. *OpenACC Programming and Best Practices Guide*. <https://www.openacc.org/resources>, 2015.
- OYARZUN, G. et al. A GPU-based algorithm for efficient LES of high reynolds number flows in heterogeneous CPU/GPU supercomputers. *Applied Mathematical Modelling*, v. 85, p. 141 – 156, 2020.

PEACEMAN, D. W. Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability. *Society of Petroleum Engineers Journal*, v. 23, n. 3, p. 531–543, 1983.

PESSANHA, M. L. O. et al. Comparative study and sensitivity analysis in simulation of non-Darcy flow in shale gas reservoirs. *International Journal of Advanced Engineering Research and Science*, v. 7, p. 109 – 121, 2020.

PICKERING, B. P. et al. Directive-based GPU programming for computational fluid dynamics. *Computers & Fluids*, v. 114, p. 242 – 253, 2015.

RANGEL-GERMAN, E. et al. Electrical-heating-assisted recovery for heavy oil. *Journal of Petroleum Science and Engineering*, v. 45, n. 3, p. 213 – 231, 2004.

REDONDO, C. *A fast IMPES multiphase flow solver in porous media for reservoir simulation*. Tese (Doutorado) — Universidad Politécnica de Madrid Escuela Técnica Superior de Ingenieros Aeronáuticos, 2017.

ROSA, A. J.; CARVALHO, R. S.; XAVIER, J. A. D. *Engenharia de Reservatórios de Petróleo*. Rio de Janeiro, Brasil: Interciência, 2006.

ROUSSET, M. *Reduced-order modelling for thermal simulation*. Tese (Doutorado) — Stanford University, 2010.

SAAD, Y. *Iterative Methods for Sparse Linear Systems*. 2. ed. Philadelphia: SIAM, 2003.

SEARLES, R. et al. MPI + OpenACC: Accelerating radiation transport mini-application, minisweep, on heterogeneous systems. *Computer Physics Communications*, v. 236, p. 176–187, 2018.

SIYAVASHI, M. et al. Three-dimensional streamline-based simulation of non-isothermal two-phase flow in heterogeneous porous media. *Computers & Fluids*, v. 103, p. 116 – 131, 2014.

SINGH, A. K.; GOERKE, U. J.; KOLDITZ, O. Numerical simulation of non-isothermal compositional gas flow: application to carbon dioxide injection into gas reservoirs. *Energy*, v. 36, p. 3446–3458, 2011.

SOUZA, G. de. *Acoplamento Poço-reservatório na Simulação Numérica de Reservatórios de Gás*. Tese (Doutorado) — Universidade Estadual do Norte Fluminense, Macaé, Brasil, 2013.

SULZBACH, M. *Programação Paralela Híbrida para CPU e GPU: Uma avaliação do OpenACC frente a OpenMP e CUDA*. Dissertação (Mestrado) — Universidade Federal de Santa Maria, Santa Maria, 2014.

TAMIM, M.; ABOU-KASSEM, J.; Farouq Ali, S. Recent developments in numerical simulation techniques of thermal recovery processes. *Journal of Petroleum Science and Engineering*, v. 26, n. 1, p. 283 – 289, 2000.

TIAB, D.; DONALDSON, E. C. *Petrophysics, Theory and Practice of Measuring Reservoir Rock and Fluid Transport Properties*. Second edition. Burlington, USA: Gulf Professional Publishing, 2004.

VENNEMO, S. B. *Multiscale Simulation of Thermal Flow in Porous Media*. Dissertação (Mestrado) — Norwegian University of Science and Technology, Trondheim, Norway, 2016.

WANG, L. et al. Advances in improved/enhanced oil recovery technologies for tight and shale reservoirs. *Fuel*, v. 210, p. 425 – 445, 2017.

WERNECK, L. F. et al. An openmp parallel implementation using a coprocessor for numerical simulation of oil reservoirs. *Computational & Applied Mathematics*, v. 38, p. 33, 2019.

WU, X.; TAYLOR, V. Performance modeling of hybrid MPI/OpenMP scientific applications on large-scale multicore supercomputers. *Journal of Computer and System Sciences*, v. 79, p. 1256–1268, 2013.

YAKUBOV, S. et al. Hybrid MPI/OpenMP parallelization of an Euler–Lagrange approach to cavitation modelling. *Computers and Fluids*, v. 80, p. 365–371, 2013.

ZAZA, A. et al. A cuda based parallel multi-phase oil reservoir simulator. *Computer Physics Communications*, v. 206, p. 2 – 16, 2016.