CONCILIUM, Vol. 23, Nº 3, 2023
DOI: 10.53660/CLM-756-23A63
ISSN: 0010-5236

Simulação numérica do escoamento não-isotérmico em reservatórios de petróleo usando OpenACC

Numerical solution of non-isothermal flow in oil reservoirs using OpenACC

Ralph Alves Bini da Silva Almeida[1], Grazione de Souza[1*], Helio Pedro Amaral Souto[1]

Received: 2023-01-03 | Accepted: 2023-02-05 | Published: 2023-02-13

RESUMO

Neste trabalho, é feita a simulação numérica do escoamento não-isotérmico em um reservatório de petróleo. Utilizou-se a API OpenACC na paralelização de partes específicas do código original, o que permitiu a execução simultânea de tarefas em diferentes núcleos de uma placa de vídeo NVIDIA GTX 970 G1, em uma arquitetura de memória compartilhada. Estudou-se o problema utilizando-se um poço vertical produtor e poços aquecedores estáticos, em um domínio bidimensional. O método *Control Volume Finite Difference* (CVFD) foi empregado na discretização das equações governantes e o método dos Gradientes Conjugados foi o escolhido para se obter as soluções (pressão e temperatura) dos sistemas de equações algébricas. Como resultado, no que diz respeito ao desempenho computacional, foi obtida uma redução significativa do tempo de execução com a versão paralelizada.

Palavras-chave: Computação de Alto Desempenho; Escoamento Não-isotérmico; OpenACC; Simulação Numérica de Reservatórios; Recuperação terciária.

ABSTRACT

In this work, we performed a numerical simulation of the non-isothermal flow in an oil reservoir. Also, we used the OpenACC API to parallelize specific parts of the original code, which allowed the simultaneous execution of different tasks on a NVIDIA GTX 970 G1 video card in a shared memory architecture. The problem was studied using a vertical producer well and static heating wells in a two-dimensional domain. We used the Control Volume Finite Difference (CVFD) method to discretize the governing equations and the Conjugate Gradients method to obtain the solutions (pressure and temperature) of the systems of algebraic equations. As a result, concerning computational performance, a significant reduction in execution time was obtained with the parallelized version.

Keywords: High Performance Computing; Non-isothermal flow; OpenACC; Numerical Reservoir Simulation; Tertiary Recovery.

1 Universidade do Estado do Rio de Janeiro.
*E-mail: gsouza@iprj.uerj.br

INTRODUÇÃO

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

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.

Especificamente, em se tratando dos métodos térmicos, é possível citar 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 (AOUIZERATE; DURLOFSKY; 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, e.g., 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 (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.

Os modelos físico-matemáticos para 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 apenas para casos 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, pela realização de testes de pressão em poços (BOURDET, 2002)), as simulações numéricas do escoamento em reservatórios (ERTEKIN; ABOU-KASSEM; KING, 2001) são normalmente utilizadas 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 cenários de recuperação, a fim de se prever a produção de reservatórios ao longo de décadas.

No caso de reservatórios de óleo pesado, estudos têm sido realizados com o uso da simulação numérica (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 (REDONDO, 2017).

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) (AMARAL 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 tenda a ser muito elevado. Em contrapartida, melhorias no desempenho computacional podem ser alcançadas através da paralelização de todo o código numérico, ou de parte dele, implicando, assim, na utilização de arquiteturas possibilitando a execução usando memória distribuída ou compartilhada (JAQUIE, 1999).

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) ou de um coprocessador no caso do OpenMP. Em se tratando do OpenACC, uma CPU ou uma *Graphics Processing Unit* (GPU) podem ser utilizadas. 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 é o de melhorar o desempenho computacional, buscando atingir os maiores *speedups* possíveis (LOSADA et al., 2016). Neste trabalho, 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 petróleo na presença de poços aquecedores.

MODELAGEM DO ESCOAMENTO NÃO-ISOTÉRMICO DE PETRÓLEO

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. A partir destas equações, EDPs não-lineares, obtém-se as equações governantes escritas em termos das variáveis dependentes pressão e temperatura do óleo.

As seguintes hipóteses foram assumidas para o modelo físico-matemático utilizado na descrição do escoamento: 1) a permeabilidade absoluta é homogênea e anisotrópica; 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 desconsiderados; 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 se assume o equilíbrio térmico local; 11) o poço de produção é vertical e penetra totalmente a formação; 12) ausência de estocagem no poço.

A equação derivada do postulado da conservação de massa para o escoamento monofásico em meios porosos é dada por (ERTEKIN; ABOU-KASSEM; KING, 2001)

$$\frac{\partial}{\partial t}\left(\frac{\phi}{B}\right) + \nabla \cdot \left(\frac{\mathbf{v}}{B}\right) - \frac{q_m}{\rho_{sc}} = 0 \tag{1}$$

onde ϕ representa a porosidade, B o fator volume de formação, ρ_{sc}/ρ, sendo ρ a massa específica do fluido, $\mathbf{v}$ a velocidade superficial do fluido, q_m o termo de fonte que representa uma vazão mássica e o subscrito *sc* representa as condições padrão de temperatura e pressão.

Para escoamentos a baixas velocidades em meios porosos, em geral, a lei de Darcy é usada para descrever a conservação da quantidade de movimento (DAKE, 2001),

$$\mathbf{v} = -\frac{\mathbf{k}}{\mu}(\nabla p - \rho g \nabla D) \tag{2}$$

onde $\mathbf{k}$ é o tensor de permeabilidade absoluta (aqui considerado diagonal), μ a viscosidade do fluido, p a pressão do fluido, g o módulo da aceleração da gravidade e D a profundidade.

As Eqs. (1) e (2) podem ser combinadas de modo a se obter

$$\frac{\partial}{\partial t}\left(\frac{\phi}{B}\right) - \nabla \cdot \left(\frac{\mathbf{k}}{B\mu}\nabla p\right) - q_{sc} = 0 \tag{3}$$

sabendo-se que o termo de fonte foi reescrito como $q_m = \rho_{sc} q_{sc}$. Ademais, a derivada parcial em relação ao tempo pode ser expandida sabendo-se que as propriedades do fluido e da rocha dependem da pressão e da temperatura média do reservatório, T,

$$\begin{gathered} \rho = \rho^0[1 + c(p - p^0) - c_T(T - T^0)], \\ \phi = \phi^0\left[1 + c_\phi(p - p^0) - c_{\phi T}(T - T^0)\right], \qquad \mu = aexp\left(b/(T - T_{ref,\mu})\right), \end{gathered} \tag{4}$$

onde c e c_ϕ representam os coeficientes de compressibilidade do fluido e da rocha, enquanto que c_T e $c_{\phi T}$ são os respectivos coeficientes de expansão térmica, a e b são parâmetros para o óleo considerado e $T_{ref,\mu}$ uma temperatura de referência para o cálculo da viscosidade.

Portanto, dessas expressões é possível se obter a forma final

$$\Gamma_p \frac{\partial p}{\partial t} - \Gamma_T \frac{\partial T}{\partial t} - \nabla \cdot \left(\frac{\mathbf{k}}{B\mu}\nabla p\right) - q_{sc} = 0 \tag{5}$$

onde os coeficientes dos termos transientes são dados por

$$\Gamma_p = \left(\frac{\phi c_o}{B^0 + (\phi^0 c_\phi / B)}\right), \quad \Gamma_T = \left(\frac{\phi c_{oT}}{B^0 + (\phi^0 c_{\phi T}/B)}\right). \tag{6}$$

Utilizando-se a hipótese do não equilíbrio térmico local (MOYNE et al., 2000), na definição da temperatura média, em conjunto com uma modelagem incluindo termos fonte para representar aquecedores da jazida (ROUSSET, 2010), tem-se para a conservação da energia

$$\frac{\partial}{\partial t}\left(\overline{\rho c_p} T\right) + \nabla \cdot \left(\rho c_p T \mathbf{v}\right) - \nabla \cdot (\boldsymbol{\kappa} \nabla T) - q_H - \rho c_p T q_{sc} = 0 \tag{7}$$

onde $\overline{\rho c_p} = \phi \rho c_p + (1 - \phi)\rho_r c_{pr}$ é a capacidade térmica do meio e a temperatura média é definida a partir de $\overline{\rho c_p} T = \phi \rho c_p T_o + (1 - \phi)\rho_r c_{pr} T_r$, sabendo-se que os sobrescritos *o* e *r* se referem ao óleo e a rocha, respectivamente, e q_H.é o termo de fonte que contabiliza a energia fornecida ao reservatório via aquecedores. O tensor efetivo de dispersão térmica considera somente a contribuição da condução de calor $\boldsymbol{\kappa} = \phi \boldsymbol{\kappa}_o + (1 - \phi)\boldsymbol{\kappa}_r$, onde $\boldsymbol{\kappa}_o$ e $\boldsymbol{\kappa}_r$ são as

condutividades térmicas do óleo e da rocha, respectivamente. Portanto, não são contabilizados os efeitos devidos à tortuosidade e à dispersão hidrodinâmica (MOYNE et al., 2000).

Na resolução numérica das Eqs. (5) e (7) os valores inicias da pressão e da temperatura são fornecidos para todo o reservatório. Nas fronteiras, são impostas condições de contorno do tipo fluxo nulo. Um modelo de acoplamento poço-reservatório também é utilizado para relacionar vazão de produção com as pressões no poço e no reservatório (PEACEMAN, 1983),

$$q_{sc} = -J_w\left(p - p_{wf}\right), \tag{8}$$

onde J_w é o índice de produtividade (nulo onde não há poço) e p_{wf} a pressão no poço.

RESOLUÇÃO NUMÉRICA

Remontando à década de 1950 nas 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.

Na determinação da solução numérica emprega-se aqui uma malha de blocos centrados (WERNECK et al., 2019). A solução é obtida nos nós da malha computacional, localizados nos centros das células, sendo que n_x e n_y são 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, enquanto que os fracionários $i \pm 1/2$ e $j \pm 1/2$ indicam as faces das células.

Aproximações do tipo diferenças recuadas no tempo e centradas no espaço são empregadas na discretização das equações governantes (5) e (7). Na literatura, existem propostas de estratégias de resolução numérica, para o problema do escoamento não-isotérmico em meios porosos, que se baseiam no uso de uma decomposição de operadores (*operator splitting*), tais como a encontrada no trabalho de Vennemo (2016). Neste tipo de resolução, o sistema original acoplado é subdividido em dois subsistemas de equações, um para a determinação da pressão e outro para o cálculo da temperatura, com um estágio de troca de informações entre os subsistemas. Em geral, isso permite, por exemplo, que diferentes métodos de solução possam ser aplicados na determinação de cada variável dependente, em função da classificação matemática do tipo de equação diferencial. A estratégia apresentada em Vennemo (2016) foi a adotada neste trabalho.

A forma final da equação discretizada para a equação governante escrita em termos da pressão é dada por (DA SILVA ALMEIDA, 2021):

$$\begin{aligned}
&\tau_y\big|_{i,j-1/2}^{v,n+1} p_{i,j-1}^{v+1,n+1} + \tau_x\big|_{i-1/2,j}^{v,n+1} p_{i-1,j}^{v+1,n+1} \\
&+\left[\tau_y\big|_{i,j-1/2}^{v,n+1} + \tau_x\big|_{i-1/2,j}^{v,n+1} + V_b \frac{\left(\Gamma_p\right)_{i,j}^{v,n+1}}{\Delta t} + \tau_x\big|_{i+1/2,j}^{v,n+1} + \tau_y\big|_{i,j+1/2}^{v,n+1}\right] p_{i,j}^{v+1,n+1} \\
&+\tau_x\big|_{i+1/2,j}^{v,n+1} p_{i+1,j}^{v+1,n+1} + \tau_y\big|_{i,j+1/2}^{v,n+1} p_{i,j+1}^{v+1,n+1} = V_b \frac{\left(\Gamma_p\right)_{i,j}^{v,n+1}}{\Delta t} p_{i,j}^{n} \\
&+V_b \frac{\left(\Gamma_T\right)_{i,j}^{v,n+1}}{\Delta t}\left(T_{i,j}^{\overline{w},n+1} - T_{i,j}^{n}\right) + \left(q_{sc}\right)_{i,j}^{n+1}
\end{aligned} \tag{9}$$

enquanto que para a equação de energia obtém-se

$$\begin{aligned}
&K_y\big|_{i,j-1/2}^{w,n+1} T_{i,j-1}^{w+1,n+1} + K_x\big|_{i-1/2,j}^{w,n+1} T_{i-1,j}^{w+1,n+1} \\
&+\left[K_y\big|_{i,j-1/2}^{w,n+1} + K_x\big|_{i-1/2,j}^{w,n+1} + V_b \frac{\left(\overline{\rho c_p}\right)_{i,j}^{w,n+1}}{\Delta t} + K_x\big|_{i+1/2,j}^{w,n+1} + K_y\big|_{i,j+1/2}^{v,n+1}\right] T_{i,j}^{w+1,n+1} \\
&+K_x\big|_{i+1/2,j}^{w,n+1} T_{i+1,j}^{w+1,n+1} + K_y\big|_{i,j+1/2}^{w,n+1} T_{i,j+1}^{w+1,n+1} = -V_b \frac{\left(\overline{\rho c_p}\right)_{i,j}^{w,n+1}}{\Delta t} T_{i,j}^{n} \\
&+\left(\rho c_p T q_{sc}\right)_{i,j}^{w,n+1} + \Phi_{i,j}^{\overline{v},n+1} + \left(q_{sc}\right)_{i,j}^{n+1}
\end{aligned} \tag{10}$$

onde foram introduzidas as transmissibilidades $\tau_x \equiv A_x k_x/\mu B \Delta x$ e $\tau_y \equiv A_y k_y/\mu B \Delta y$, determinadas nas interfaces das células da malha numérica via média harmônica e onde k_x e k_y são, respectivamente, as permeabilidades absolutas nas direções x e y. Nessas equações $A_x = \Delta y L_z$, $A_y = \Delta x L_z$, $V_b = \Delta x \Delta y L_z$, L_z representa a profundidade do reservatório, Δx, Δy e Δt são os incrementos de espaço e tempo, respectivamente e, finalmente,

$$\Phi_{i,j}^{n+1} = \rho_{sc}\left[\left(\rho c_p T \tau_x\right)_{i-1/2,j}^{n+1}\left(p_{i,j} - p_{i-1,j}\right)^{n+1} - \left(\rho c_p T \tau_x\right)_{i+1/2,j}^{n+1}\left(p_{i,j} - p_{i-1,j}\right)\right.$$
$$\left. + \rho_{sc}\left[\left(\rho c_p T \tau_y\right)_{i,j-1/2}^{n+1}\left(p_{i,j} - p_{i,j-1}\right)^{n+1} - \left(\rho c_p T \tau_y\right)_{i,j+1/2}^{n+1}\left(p_{i,j} - p_{i,j-1}\right)^{n+1}\right.\right. \quad (11)$$

Os índices v e w referem-se, respectivamente, aos níveis iterativos para a obtenção das pressões e temperaturas. Já o $\overline{v}$ indica que o termo $\Phi_{i,j}^{n+1}$ está sendo avaliado com o uso das propriedades determinadas no nível iterativo v, no qual se obteve as pressões em $v+1, n+1$, quando da resolução da Eq. (9). O mesmo raciocínio pode ser estendido para o termo $T_{i,j}^{\overline{w},n+1}$.

Em se tratando do acoplamento poço-reservatótio, a equação discretizada é a seguinte:

$$\left(q_{sc}\right)_{i,j}^{n+1} = -\left(J_w\right)_{i,j}^{n+1}\left[p_{i,j}^{n+1} - \left(p_{wf}\right)_{i,j}^{n+1}\right] \quad (8)$$

sendo o índice de produtividade é calculado via

$$\left(J_w\right)_{i,j}^{n+1} = \left[\frac{2\pi\sqrt{k_x k_y}L_z}{B\mu ln\left(\frac{r_{eq}}{r_w}\right)}\right]_{i,j}^{n+1} \quad (8)$$

onde r_w é o raio do poço e o raio equivalente é obtido da expressão (PEACEMAN, 1983)

$$r_{eq} = 0{,}28\left[\sqrt{\sqrt{\frac{k_y}{k_x}}(\Delta x)^2 + \sqrt{\frac{k_x}{k_y}}(\Delta y)^2}\Bigg/\left(\sqrt[4]{\frac{k_y}{k_x}} + \sqrt[4]{\frac{k_x}{k_y}}\right)\right]_{i,j}. \quad (8)$$

Neste trabalho, o método dos Gradientes Conjugados (SAAD, 2003) foi empregado na resolução dos sistemas de equações algébricas. Trata-se de um método iterativo, desenvolvido de modo a se 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. No caso do armazenamento dos elementos da matriz de coeficientes, optou-se aqui pelo uso da técnica *Compressed Sparse Row* (CSR) (WERNECK et al., 2019).

PROCESSAMENTO PARALELO USANDO OpenACC

No campo da computação paralela, um dos recursos conhecidos disponível é a *Aplication 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). 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 disponíveis e memória dedicada.

Os dados usados nos cálculos são rotulados em dois tipos básicos: *shared* e *private*. No caso do tipo *shared* há apenas uma instância de dados, sendo que todas as *threads* podem acessá-los e modificá-los simultaneamente, a menos que alguma restrição seja imposta via comando específico do OpenACC. Todas as alterações são visíveis para todas as *threads*. Por outro lado, adotando-se o tipo *private*, 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 das diretivas de compilação encontra-se em OpenACC Organization (2015).

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).

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 trabalho, foram acrescentadas ao código as diretivas: *#pragma acc parallel loop* de modo a transmitir-se ao preprocessador da linguagem C a ordem de que se deve paralelizar o laço de repetição for que se encontra na próxima linha.

Uma diretiva fundamental para a paralelização do código computacional é a *data*. Assim como quando se usa a estratégia de paralelização com a API CUDA, deve-se copiar, por exemplo, vetores e/ou matrizes que se encontram armazenados na memória acessada pele 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. Portanto, 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)*.

Neste trabalho, 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). 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.

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 a compilação com o *PGI Compiler* (KIM; KANG; JOH, 2021). Aqui ele foi 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.

RESULTADOS NUMÉRICOS

Passa-se, agora, à apresentação dos resultados obtidos com as simulações do escoamento bidimensional não-isotérmico de óleo mediante o uso do simulador paralelizado neste trabalho, cujo método do Gradiente Conjugado foi paralelizado via uso da API OpenACC. Admite-se que o método convergiu quando o erro for menor que a tolerância estipulada, tol.

O reservatório considerado tem a forma de um paralelepípedo de dimensões L_x, L_y e L_z, sendo a última menor do que as demais. Esses e os demais parâmetros utilizados nas simulações podem ser vistos na Tabela 1. As coordenadas indicando a posição dos poços aquecedores 1 e 2, no plano xy são, respectivamente, x_{aq1}, y_{aq1}, x_{aq2}, y_{aq2} e o número de aquecedores é n_{aq}. Além disso, as coordenadas do poço produtor no mesmo plano são x_{pp} e y_{pp}. Os poços aquecedores e produtor possuem o mesmo comprimento do reservatório na direção *z*. O tempo de produção é t_{max}. A simulação inicia com um passo de tempo inicial, Δ_{ini}, que é multiplicado pela razão $F_{\Delta t}$ para a obtenção do próximo incremento de tempo. Repete-se o processo até que o incremento de tempo máximo, Δ_{max}, é alcançado e mantido constante até o final da simulação.

Para o estudo do desempenho computacional foram escolhidas 6 malhas uniformes e as suas especificações são mostradas na Tabela 2. A placa de vídeo utilizada na execução em paralelo do código numérico possui as seguintes características: *chipset* Nvidia Geforce GTX-970 G1; memória GDDR5; memória de 4 GB; *clock* da GPU de 1.050 MHz; 1664 núcleos CUDA; e interface PCI Express 3.0. Como compilador, utilizou-se o PGI Compiler 20.4 (NVIDIA HPC SDK Versão 20.11). No caso da CPU utilizada, tem-se 3.6 GHz, memória DDR4 e 16 GB de RAM.

Tabela 1 – Parâmetros para o caso padrão

Parâmetro	**Valor**	**Parâmetro**	**Valor**	**Parâmetro**	**Valor**
a	0,2 Pa.s	$L_x = L_y$	6.400 m	x_{pp}	3.200 m
b	600 K	L_z	40 m	y_{aq1}	3.200 m
B^0	1,3 m^3/std.m^3	n_{aq}	2	y_{aq2}	3.200 m
c	7,25 10^{-7} kPa^{-1}	$p_{ini} = p^0$	1 10^4 kPa	y_{pp}	3.200 m
c_p	2.100 J/(kg.K)	q_H	40 kW	κ_o	0,45 W/(m.K)
c_{pr}	1.800 J/(kg.K)	tol	1 10^{-6}	κ_r	3,5 W/(m.K)
c_T	7,25 10^{-7} K	t_{max}	60 dias	ρ_r	2.500 kg/m^3
c_ϕ	4,35 10^{-7} kPa^{-1}	$T_{ini} = T^0$	330 K	ρ^0	840 kg/m^3
$c_{\phi T}$	4,35 10^{-7} K	T_{ref}	550 K	ϕ^0	0,2
$F_{\Delta t}$	1,1	x_{aq1}	3.060 m	Δt_{ini}	0,1 dia
$k_x = k_y$	0,02 μm^2	x_{aq2}	3.340 m	Δt_{max}	1,0 dia

Fonte: Os autores (2023)

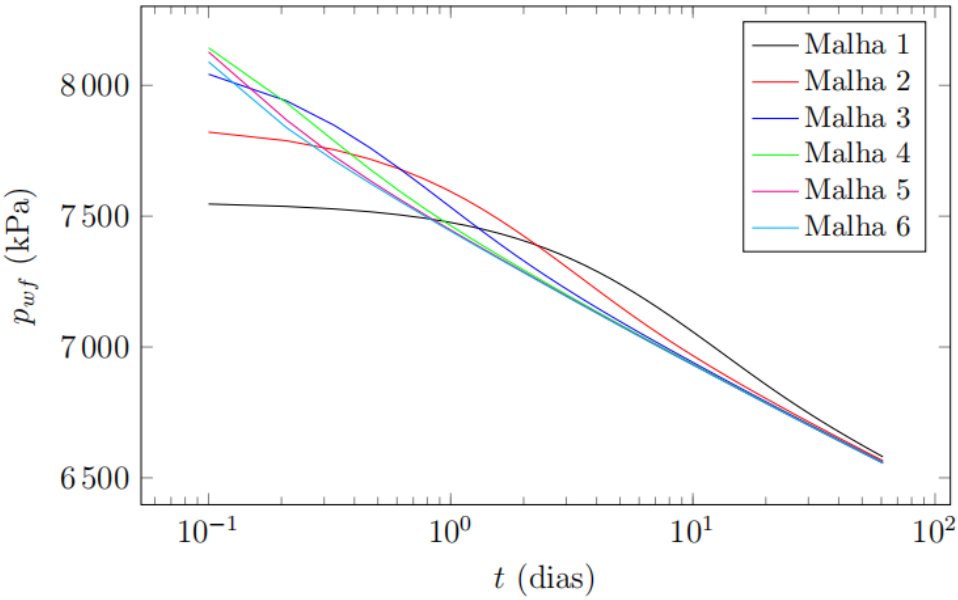
Tabela 2 – Malhas computacionais

Malha:	1	**2**	**3**	**4**	**5**	**6**
$n_x = n_y$:	47	93	185	369	737	1.473
Total de células:	2.209	8.649	34.225	136.161	543.169	2.169.729

Fonte: Os autores (2023)

Um estudo de refinamento de malha foi realizado e os resultados são apresentados na forma de um gráfico da variação da pressão no poço em função do tempo de produção. Conforme pode ser visto na Figura 1, à medida que as malhas são refinadas as curvas de pressão se aproximam cada vez mais uma das outras. Nos tempos iniciais há um maior distanciamento, que também diminui com o refinamento de malha, oriundo da aplicação do tipo de técnica de acoplamento poço-reservatório aplicada (PEACEMAN, 1983).

Figura 1 – Pressão no poço produtor: diferentes malhas

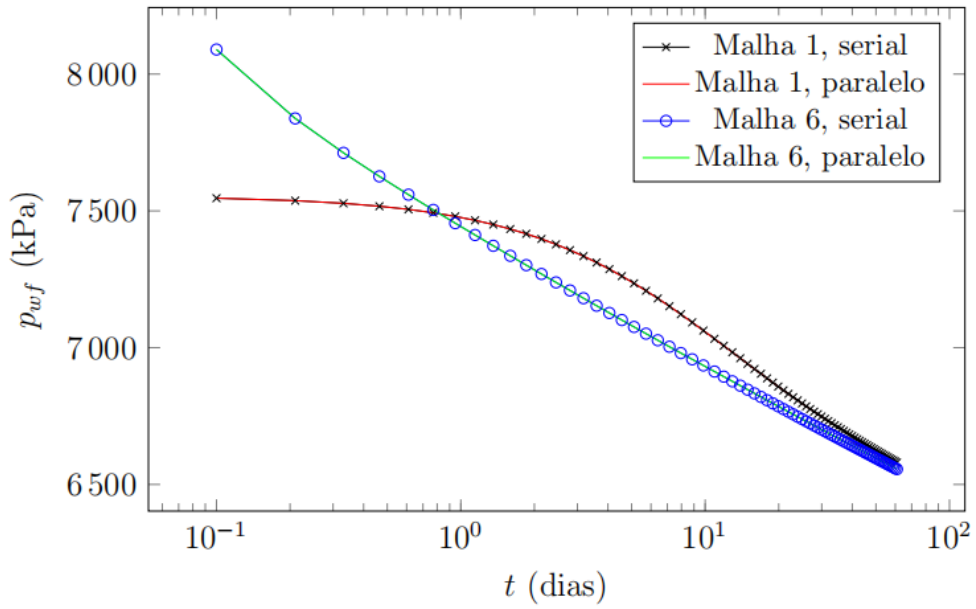


Fonte: Os autores (2023)

A fim de se verificar que a paralelização não afetou a solução numérica, comparou-se os valores fornecidos pelas versões serial e paralelizada do código numérico para a variação da pressão no poço, empregando respectivamente as Malhas 1 e 6. Atesta-se, a partir da Figura 2, que as curvas se encontram praticamente sobrepostas. Portanto, os valores oriundos da resolução numérica não foram afetados quando da paralelização do simulador numérico.

Neste ponto, constatada a acurácia 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 3 mostra os tempos (em segundos) de simulações correspondentes às execuções realizadas com a versão paralelizada (t_P) e serial (t_S).

Figura 2 – Pressão no poço produtor: códigos serial e paralelizado



Fonte: Os autores (2023)

Tabela 3 – *Speedup*

Malha:	1	2	3	4	5	6
t_S**(s):**	0,291	1,599	11,199	70,981	462,323	3183,673
t_P**(s):**	1,902	3,534	9,678	31,057	133,631	709,135
Speedup:	0,153	0,452	1,157	2,286	3,460	4,490

Fonte: Os autores (2023)

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. 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.

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 a Tabela 4, 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 um comportamento muito semelhante ao da Tabela 3 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, GPU%, indicando que a sua capacidade limite de processamento está sendo alcançada.

Tabela 4 – *Percentual de uso da GPU (%)*

Malha:	***1***	***2***	***3***	***4***	***5***	***6***
GPU%:	26	35	49	62	76	97

Fonte: Os autores (2023)

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 970 G1.

Conforme já avançado, constatou-se uma tendência de aumento do *speedup* à medida que se aumentou o número total de 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 com maiores recursos de *hardware* 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.

REFERÊNCIAS

AMARAL, V. et al. Programming languages for data-intensive HPC applications: A systematic mapping study. Parallel Computing, v. 91, p. 102584, 2020.

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.

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).

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.

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.

DA SILVA ALMEIDA, R. A. B. Solução numérica do escoamento não-isotérmico em reservatórios de óleo pesado empregando computação paralela. Dissertação (Mestrado) — Universidade do Estado do Rio de Janeiro, Brasil, 2021.

DAKE, L. P. The Practice of Reservoir Engineering (Revised Edition). Amsterdam, The Netherlands: Elsevier, 2001. Developments in Petroleum Science 36.

ERTEKIN, T.; ABOU-KASSEM, J.; KING, G. Basic Applied Reservoir Simulation. Richardson, USA: Society of Petroleum Engineers, 2001.

JAQUIE, 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.

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.

LOSADA, N. et al. Portable application-level checkpointing for hybrid MPI-OpenMP applications. Procedia Computer Science, v. 80, p. 19–29, 2016.

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.

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.

OPENACC ORGANIZATION. OpenACC Programming and Best Practices Guide. https://www.openacc.org/resources, 2015.

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.

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.

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.

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.

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

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.