Introduction

Reverse Time Migration (RTM) belongs to a family of seismic techniques that produces reliable subsurface imaging on complex media. But this reliability comes with the cost of a significant memory consumption and access, posing as challenges for current hardware. To overcome such challenges, multiple techniques appeared in the past few years to address the intense use of memory. Notably, optimal checkpointing (Griewank, 1992) was a milestone in the oil and gas industry, allowing the reduction of terabytes of data usage to a few gigabytes. Despite the elegant solution, this reduction comes at the expense of more calculations that easily can be converted into an unsatisfactory trade-off between memory usage and recomputations. As an alternative, boundary saving (Yang et al., 2014) presents a more acceptable memory trade-off with a fixed recomputation overhead. Nevertheless, the technique still uses a considerable amount of memory for large datasets, which leaves plenty of room for more improvements and new solutions to appear.

We propose a hybrid approach that uses both checkpointing and boundary saving techniques. Furthermore, we find the optimal number of checkpoints that minimizes the high memory demands of RTM. Numerical results show the reliability of the produced data. Afterwards, a discussion section describes how the technique efficiently uses the available resources. Finally, we conclude that our approach has a competitive trade-off between memory and recomputation, which allow it to become a relevant alternative to the existing methods.

Method

Reverse time migration

RTM is a cross-correlation at the same time level of a forward and a backward wavefield, expressed as

\[ I(x) = \sum_{s=1}^{N_s} \sum_{g=1}^{N_g} \int_0^T p_s(x, T-t) p_g(x, t) \, dt, \]  

where \( x \) is the spatial variable (2D or 3D), \( N_s \) is the number of shots in the seismic data, \( N_g \) is the number of receivers associated with shot \( s \), \( T \) is the maximum time recording with \( N_t \) time steps, \( p_s \) and \( p_g \) are the forward and backward wavefields, respectively, given by shot \( s \) and receiver \( g \). Considering the acoustic case in a constant density medium, the wave equation (Bleistein et al., 2001) can numerically be solved through a second order Time Domain Finite Difference (TDFD) approach, given by

\[ \sigma_1 P^{f+1}(x) = 2\sigma_2 P^f(x) + \sigma_3 P^{f-1}(x) + \Delta t^2 \nabla^2 \left( L^{2M}(P^f(x)) + f^f(x-x_0) \right), \]  

with \( \ell \) representing the specified time, \( x \) and \( x_0 \) the discrete spatial variables (2D or 3D), \( \nabla \) the discrete Laplacian operator specified by a finite-difference method (FDM) with centered scheme of order \( 2M \) for \( M \geq 1 \), and finally \( \sigma_1, \sigma_2 \) and \( \sigma_3 \) are the constant terms associated with the absorbing condition (Kosloff and Kosloff, 1986), that within the region of interest are zeros. From equation (2), it is specified that both \( P^f \) and \( L^{2M} \) have the same dimension of \( \nabla \) and that only two pointers are necessary at each step of the computation, such that \( P^{f-1}(x) \) is read and immediately overwritten by \( P^{f+1}(x) \) value.

Checkpointing

To apply the cross-correlation, a naive approach retains previously computed values and avoids recomputation through saving all \( N_t \) time snapshots of field \( p_s \), which are used for the reverse memory access of both \( p_s \) and \( p_g \) fields specified on equation (1). Due to the high memory requirements of saving all snapshots, the checkpointing strategy arises as an alternative, trading memory usage for recalculations.

The strategy consists of saving only \( 2N_c \) snapshots, because we need the two previous states to continue the TDFD propagation, in which \( N_c \) is the number of checkpoints equally spaced from the \( N_t \) time steps used to recompute the unsaved \((N_t - 2N_c)\) snapshots of field \( p_s \). In this approach, beyond the \( N_c \) computations to generate the snapshots, each checkpoint requires \(\lceil N_t/N_c \rceil(\lceil N_t/N_c \rceil+1)/2\) recomputations to recover the unsaved snapshots, which results in a total of \(\lceil (N_t^2/N_c+N_t)/2 \rceil\) computations, or in a recomputation ratio (which is the total number of computations divided by \( N_t \)) of \( (1 + \lceil N_t/N_c \rceil)/2\).
Boundary saving

Boundary saving (Yang et al., 2014) is another strategy that lowers memory usage by storing only the model borders and the two last wavefields. In this solution the recomputation ratio is fixed at 2.

The strategy works as shown in Figure 1, with a 2D forward propagation illustrated and a saved region (gray color) stored to memory at each time step. After the forward propagation, the last two snapshots are back-propagated together with the $p_g$ field construction. At each step of the backpropagation, points inside the model region (white color) are correctly reconstructed, whereas points from the other two regions are kept incorrect due to energy absorption coming from the absorbed region (red color). To secure the correct reconstruction of the model region in future backpropagations, the saved region must be corrected, which is done by substituting the incorrect cells by the saved ones, and because the absorbed region does not introduces fragments in the other two, the region is kept unchanged and therefore always incorrect.

Considering the memory usage, let us define $N_b$ as the thickness of saved region and $N_{abc}$ the thickness of absorbed region. Then, the number of cells saved in each 2D snapshot is

$$S_{\text{bound}} = 2N_b\left[ (N_x - 2N_{abc}) + (N_z - 2N_{abc} - 2N_b) \right].$$

(3)

Encompassing the six surfaces of a 3D snapshot box, the numbers of cells necessary to save is given by

$$S_{\text{bound}} = 2N_b\left[ (N_y - 2N_{abc})(N_z - 2N_{abc}) + (N_x - 2N_b - 2N_{abc})(N_z - 2N_{abc}) \right.\left. + (N_y - 2N_b - 2N_{abc})(N_x - 2N_b - 2N_{abc}) \right].$$

(4)

Hybrid strategy

Even though checkpointing can offer considerable savings, multiple schemes aiming to improve its high recomputation ratio have emerged, such as optimal checkpointing (Griewank, 1992). In this solution, the ratio and memory buffers are greatly reduced to a multiple of $\log_2 N_c$, as explained by Symes (2007). Despite its merits, the implementation is fairly complex and the recomputation is not always satisfactory when compared to memory usage. The same logic applies when boundary saving is considered, where the ratio is attractive but the number of saved cells is considerably high on large datasets.

To leverage the benefits of both checkpointing and boundary saving, a hybrid approach is proposed and shown in Figure 2. In such scheme, $N_t$ checkpoints are used, but instead of multiple recomputations of the unsaved snapshots, each checkpoint recovers $[N_t/N_c]$ snapshot borders, which are then backpropagated and a ratio of $(N_t + [N_t/N_c](N_c - 1) + N_t)/N_t$ is achieved. Also, the overall memory usage is given by

$$M_{\text{hybrid}} = (2(N_c - 1))S_{\text{model}} + [N_t/N_c]S_{\text{bound}})S_{\text{float}},$$

(5)

where $S_{\text{float}}$ is the number of bytes used to store each cell and $S_{\text{model}} = N_x \times N_y \times N_z$, with $N_y = 1$ on 2D. In this case, for a typical $N_t \leq 20000$ (i.e., 20 s for a time step set to 1 ms), a simple loop from 0 to $N_t$ can inexpensive find the value of $N_t$ that minimizes equation (5). As an alternative, we can remove the ceil function from equation (5) and derive it with respect to $N_t$ to find the critical point

$$N_t^* = \left\lceil \sqrt{N_c S_{\text{bound}}/(2S_{\text{model}})} \right\rceil$$

that minimizes it. Finally, it is important to highlight that two snapshots are saved in each checkpointing, since two fields are necessary to continue the TDFD propagation. Furthermore, only $(N_c - 1)$ checkpoints are actually stored, because the last $[N_t/N_c]$ model borders are already saved due to the forward propagation.
Numerical experiments

We developed a framework using CUDA 10 and C++, such that multiple adjoint state modules could uniformly benefit from savings brought by the hybrid strategy. To easily explore the embarrassingly parallel property of shot parallelization in such adjoint state methods, PY-PITS (Benedicto et al., 2017) is used as the fault-tolerant runtime system. To validate the proposed scheme, we performed the numerical experiment on the 2D Marmousi model resampled to an uniform mesh of 921×401 points illustrated in Figure 3 with a spatial discretization of 10 m. To prevent the migrated result from being damaged by the border absorption effects, the velocity model was extended by adding fifty points to each side, i.e. the final model used by the migration algorithm consisted of 1021×501 points. Regarding the hybrid strategy, we consider $N_{abc} = 50$, $N_b = 2$ (FDM order of 2). A total of 461 shots and 700 receivers were used for the migration execution. Figure 4 shows the migrated image after applying a Laplacian filter and removing the extended points from the borders. Numerical comparisons between the hybrid strategy and the naive approach show an error around $10^{-6}$ using $L_2$ norm, thus validating our implementation. Finally, the hardware used to execute all experiments runs Ubuntu 18.04 and has an Intel Core i7-7700HQ CPU coupled with 16 GB of RAM and a Nvidia GTX 1050 Ti GPU with 4 GB of VRAM.

**Figure 3** Exact velocity model Marmousi.  **Figure 4** Migrated data with Laplacian filter.

Discussion

Table 1 shows comparisons between the memory demands of the hybrid strategy, naive approach (i.e., all snapshots are entirely saved) and boundary saving (i.e., all snapshots border saved). We observe that our hybrid strategy consumes from 1.9 to 2.9 times less memory than the simple boundary strategy and from 218 to 1165 times less memory than the naive approach. We also note that not only are the memory reductions considerable but also the recomputation ratio stable in the small interval of 2.6 to 2.8, which is a consequence of the low number of checkpoints done in each dataset, that is, $3 \leq N_c \leq 5$. However, we must point that the best-case scenario is a ratio of 2 with $N_c = 1$, in which no snapshot is stored, and the worst-case scenario is a ratio of 3, where $N_c = N_t$. The recomputation ratio of 2.8 obtained for $N_t = 10000$ is the same as obtained by Symes (2007) when using the optimal checkpointing strategy, although the latter used 60 buffers, whereas our hybrid strategy required to store only $2 \times 4$ snapshots. This represents a reduction of 7.5 times the number of buffers when compared to the optimal strategy.

<table>
<thead>
<tr>
<th>$N_x$</th>
<th>$N_y$</th>
<th>$N_z$</th>
<th>$N_t$</th>
<th>Hybrid checkpoints (GB)</th>
<th>Hybrid borders (GB)</th>
<th>Recomputation Ratio Hybrid</th>
<th>Boundary saving (GB)</th>
<th>Boundary saving / Hybrid Memory</th>
<th>Naive (GB)</th>
<th>Naive / Hybrid Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>1021</td>
<td>1</td>
<td>501</td>
<td>3001</td>
<td>0.011</td>
<td>0.015</td>
<td>2.75</td>
<td>0.059</td>
<td>2.3</td>
<td>5.719</td>
<td>218.4</td>
</tr>
<tr>
<td>13701</td>
<td>1</td>
<td>2901</td>
<td>13000</td>
<td>0.592</td>
<td>1.059</td>
<td>2.67</td>
<td>3.177</td>
<td>1.9</td>
<td>1924</td>
<td>1165.7</td>
</tr>
<tr>
<td>500</td>
<td>500</td>
<td>500</td>
<td>2000</td>
<td>2.794</td>
<td>3.541</td>
<td>2.75</td>
<td>14.163</td>
<td>2.2</td>
<td>931.323</td>
<td>147.0</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>4000</td>
<td>22.352</td>
<td>36.049</td>
<td>2.75</td>
<td>144.197</td>
<td>2.5</td>
<td>14901</td>
<td>255.2</td>
</tr>
<tr>
<td>2000</td>
<td>2000</td>
<td>2000</td>
<td>10000</td>
<td>238.419</td>
<td>322.080</td>
<td>2.80</td>
<td>1610.401</td>
<td>2.9</td>
<td>298023</td>
<td>531.7</td>
</tr>
</tbody>
</table>

As reminded by Yang et al. (2014), the minimum number of $N_b$ for a correct wavefield reconstruction must be $M$, which is the half order of the discrete Laplacian scheme. Further, the hybrid strategy is easily extended to other adjoint state methods, such as FWI, but in this case $N_{abc}$ should be chosen wisely to not harm the gradient values, which could potentially lead to premature divergence. Nonetheless, it suffices to maintain the absorbing condition on the border region, while $N_b = (N_{abc} + M)$ and right after $N_{abc}$ is updated to zero. Despite the simplicity, the increase in memory usage turns it into a non-trivial solution.
In addition, some workaround has to be considered when there is an attenuating medium. To achieve stable solutions in wavefield reconstruction, an idea related to the hybrid strategy has already been described by Yang et al. (2016), where in each time step of the backpropagation the approach chooses to use either checkpointing or saving boundary in a visco-acoustic modeling with attenuation phenomena.

Regarding the technical aspects, adjoint state methods are much more attractive on accelerators (GPUs) due to their highly parallel TDFD. Despite this, it may be cumbersome to manage the scarce memory available and the still considerable memory requirements of the hybrid scheme, even more so on 3D datasets. The usage of CUDA unified memory was considered to mitigate this problem, at the cost of a possible increase in data movement overheads. Besides, only snapshot borders were accessed and stored on device memory to highlight savings brought by the technique, such that a separate thread was used to asynchronously copy the checkpoint data from auxiliary snapshots residing on the GPU to the CPU memory (during forwarding propagation) and from the CPU to the GPU memory (during backpropagation). The cudaMemPrefetch function of the CUDA toolkit was used to hide data movements, where all auxiliary snapshots were requested to go back to the device memory after copy completed. Henceforth, the GPU memory usage was reduced, while the data movement overhead was completely hidden due to the asynchronous and prefetch operations. It is important to remind that such an approach was only valid since $N_c \ll N_t$, otherwise, such overhead possibly would not be hidden.

Conclusions

Dealing with high memory demands of Reverse Time Migration and other adjoint state methods is not a trivial task. Aiming to alleviate such demands, checkpointing, and boundary saving were relevant strategies proposed over the years. However, their high recomputation and memory requirements left plenty of room for improvements. Henceforth, a hybrid approach is proposed to leverage the improved memory usage of checkpointing, and the attractive recomputation ratio of boundary saving. Numerical results validate our technique and comparative values demonstrate the gains of such a hybrid approach. Furthermore, a worst-case recomputation ratio of 3 highlights the low price to pay when memory reductions from two to three times exists when compared with the simple boundary saving approach. Therefore the hybrid strategy is a promising solution for non-attenuative media as it also partially alleviates the memory constraints on devices with low memory availability, such as GPUs.

Acknowledgements

This work was possible thanks to the support of Petrobras and Fapesp/Cepid #2013/08293-7-Brazil. The authors also thank the High-Performance Geophysics (HPG) team for technical support.

References


