Optimizing performance of data processing

Samuel Brown[1*] and Tony Martin[1] describe the process of optimizing data throughput, to reduce turnaround and enable both full automation and interactivity of geophysical tasks in an architecture agnostic environment.

Introduction

The industry associates data processing with the application of geophysical algorithms to seismic data, but this definition is too narrow. We should consider a broader meaning particularly in these times of increased automation, and time and cost consciousness. If processing is a method to produce information from the manipulation of data, then we should consider the framework that enables this process. A system must exist to facilitate the work. Generally, systems are a collection of computers, networks, processes and to a lesser extent people. For any given input, they create an output, which in seismic data processing can be the application of a geophysical algorithm, or it can be reordering, reducing or aggregating data. In this system, the headline grabbing geophysical process may only be a small part of the work. The underlying framework needs to be as efficient as it can be, either managing the data, enabling the geophysical process, or both.

While effective seismic data processing systems can reduce seismic project turnaround times, data volumes and algorithm complexity are increasing. This is occurring in an environment where pressure is growing to reduce data delivery times. As a consequence there is a growing interest in all forms of automation. Advanced computational science is more important than ever to help achieve the goals of near instantaneous seismic data processing. In this context, Martin and Bell (2019) presented an example of automating depth imaging velocity model building. A Monte Carlo simulation, using advanced inversion and migration algorithms, enables an order of magnitude improvement in turnaround. The mass accumulation, sorting and processing of data in the simulation requires an efficient processing system, without which both the automation and timeline reduction would not be achievable. Alternatively, there are examples of using pragmatic and pure machine learning algorithms to build models with Full Waveform Inversion, including Araya-Polo et al. (2018); Øye and Dahl (2019) and Yang and Ma (2019) all of which perform various flavours of deep neural networks, convolutional neural networks or fully convolutional neural networks.

The outlined examples are attempting to reduce turnaround for isolated processes within a data processing project. A typical modern seismic data set may have around 20 trillion samples. The data volume will be manipulated, processed, validated, sorted, transformed, assembled, accumulated, reduced, increased and transcribed many times over. While optimization of single processes, using advanced algorithms, and access to almost limitless resources is important, it is only as important as the ability to both effectively apply the algorithm, and manipulate the underlying data. Long and Martin (2019) outline a method of data mining for parameter selection to reduce the impact of manual processing of data. To fulfil their work, they require dynamic manipulation of the data and a productive implementation of the geophysical tasks. They also need this to happen in a programme and platform independent way, where more resources mean ever more fruitful returns on turnaround.

In this paper we describe a data processing system that is platform agnostic, and enables the optimal management of both data and algorithms. The system has an engine that will readily accommodate the rise and implementation of both pragmatic and advanced machine learning algorithms, and enable the near instantaneous processing of data, such that interactive processing, independent of the algorithm, is achievable. The system incorporates a constant statistical feedback loop enabling automated restarts and real-time adjustments to compute resource requirements. The statistical information produced by the system is used to optimize live project performance as well as future project planning and automation.

Method

Multi-dimensional workflow engine

The raw output of today's sophisticated imaging algorithms generally contains five or more dimensions. In addition to three spatial dimensions, angle and azimuth or vector offsets provide two additional dimensions. The dimensionality can be extended farther by other binning considerations such as specularity. As a result the output images quickly become very large, often in the range of hundreds of terabytes. The sheer size of the data sets produces significant challenges in post-processing. Several high-dimensional operators must be applied to the raw output in succession to produce a final image or set of common image point gathers. In a typical workflow, operators of different dimensions are applied to different combinations of sub-planes, volumes and hyper-cubes in the raw gathers. Often the dimensions used in one process are different than both the preceding and following steps. An easy way of addressing this complexity is to implement each post-processing filter as a separate program which reads data in

[1] PGS
[*] Corresponding author, E-mail: samuel.brown@pgs.com
DOI: 10.3997/1365-2397.2019040

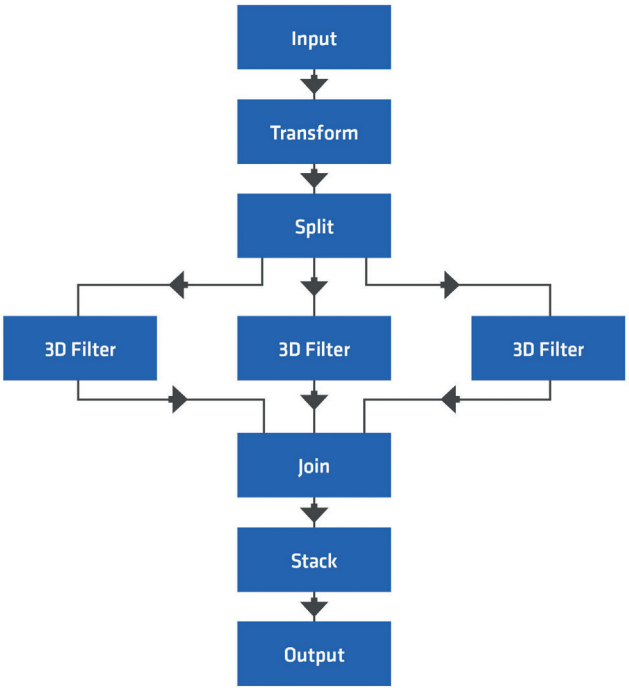

Figure 1 Schematic workflow detailing a dimensionality expansion and contraction within a single sequence.

the order presently needed and writes an intermediate data set which will be read by the next step. While simple to implement, this approach introduces an order of magnitude of more input and output (I/O) and disk storage than would be needed if the entire processing sequence could be performed in a single flow. The disk I/O becomes a major bottleneck, exploding the wall clock time necessary for post-processing.

Engine benefits

Trace-based flow engines are straightforward to implement and have long been standard in seismic processing. They are not, however, well suited to processing flows where multi-dimensional operators must be applied to large data sets of even higher dimensionality. Given the size of modern data sets and the need for parallelization, piping a connected set of processes also becomes problematic. To address this dilemma, we have implemented a high dimensional flow engine with the following key design features:

1. Keep data in memory and pass it between processing modules which have different data requirements such that there are no intermediate data sets written to disk.
2. Automatically handle large-scale parallelism and domain decomposition.
3. Perform dynamic work allocation and load balancing.
4. Provide checkpoint restart capabilities and statistical feedback.

Engine flexibility

In this system a geophysical user is able to create a flow consisting of any number of geophysical operators. These may range in complexity from simple 1D filters to complex 4D and 5D operators that work directly on gathers. The dimensionality of the entire data set may change several times in the flow. For example, a stack would reduce the dimensionality, while a decomposition

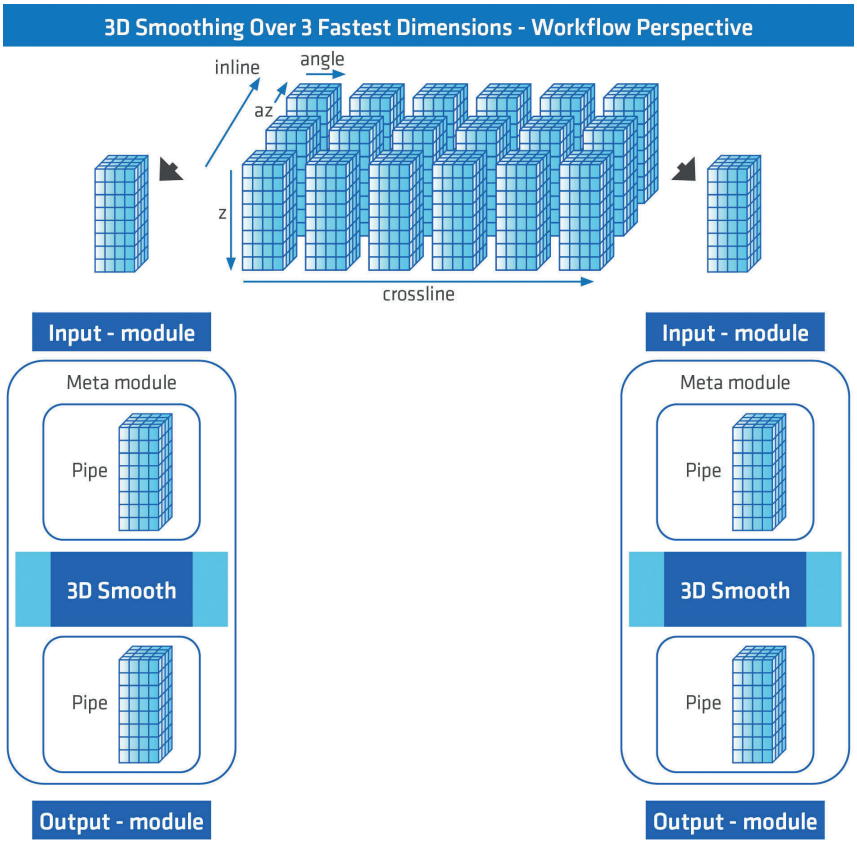

Figure 2 Workflow depiction of a multi-dimensional process applied to higher dimensionality data.

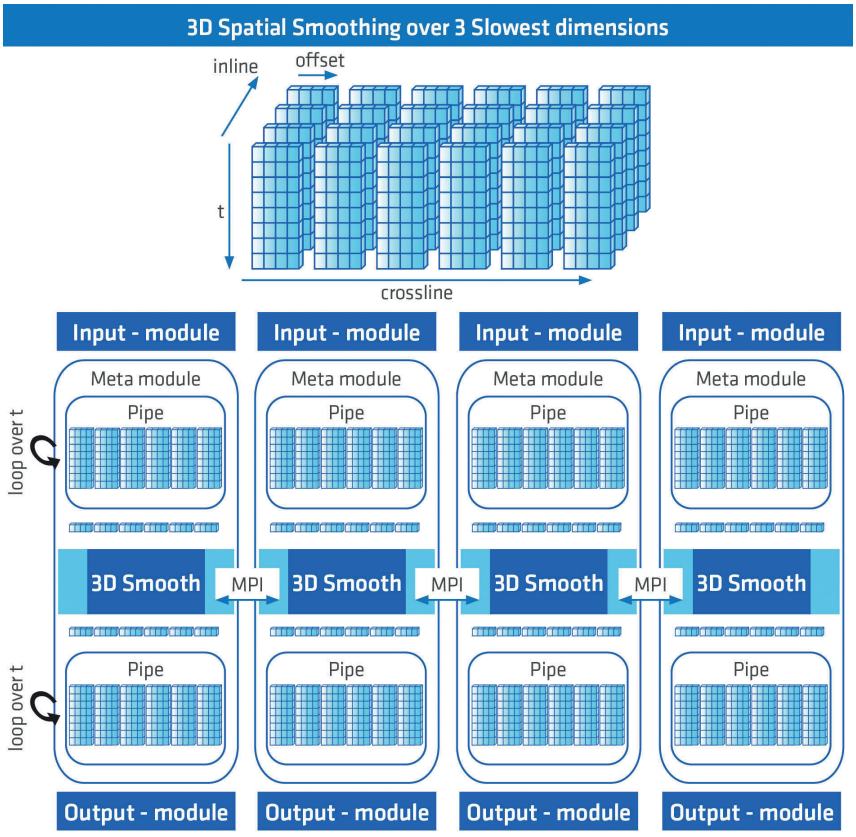

Figure 3 Workflow schematic detailing the application of a multi-dimensional process to higher dimensionality data, where the application is to the slowest data dimensions.

into frequency panels would increase the dimensionality. Each module is parameterized by the user to specify which dimensions it will act on. For example, a 3D smoother could be applied to any three dimensions in a 5D data set.

During execution, each processing module expects to see data that is compatible with its design and parameterization. For instance, a 2D filter parameterized to operate on dimensions x and z expects to be fed x-z planes. The processing system automatically buffers and loops through higher dimensional data in the pipeline to provide each module with atomic data. The key to this system is a two-step initialization process, in which the metadata of the input data set is analysed along with the data needs of all the operators. The metadata makes two passes through the workflow. On the forward pass modules apply any modifications they may make to the metadata, such as dropping a dimension for a stack, or modifying an axis when interpolation occurs. Then the metadata traverses the workflow in reverse and each module annotates its data requirements. Once this task has been completed, the system knows the smallest unit of data that can feed the workflow. It can then decompose the input data set into discrete units that can be processed independently and make decisions about how to perform domain decompositions when necessary. Each independent unit of work is represented as a work ticket which is dynamically assigned to a node or group of nodes. The shape of tickets can be further refined to optimize I/O. Since each ticket can be independently computed, they provide a natural granularity for check pointing. Statistics regarding I/O, inter-node communication and computation are automatically gathered at each step. This information is used to identify system issues and bottlenecks, and to identify and remove ailing compute nodes.

Workflow examples

Figure 1 depicts a workflow in which an image or set of common image point gathers are processed in a frequency-dependent manner by three instances of a 3D filter module. The workflow begins with an input module, which defines the input data set. In this workflow, the input can be of any dimensionality greater than or equal to three. In addition to naming the data set, this module can be parameterized to choose only a portion of the file on disk to be processed if desired, which can be especially handy for running a test line. If the dimensionality is greater than three, the input data will be automatically broken into sub-volumes as described by the parameters given to the 3D operators which will consume the data. The first module, Transform, increases the dimensionality by decomposing each sub-volume into three frequency bands, passing on a four-dimensional hypercube to the Split module. The Split module will route each frequency volume to the appropriate instance of 3D filter. Each instance is parameterized independently to optimally handle the frequency content in each frequency band. The three frequency bands are then rejoined before a Stack module combines the frequency bands and presents the Output module with data matching the original dimensions of the input.

This flow implements several complex data handling challenges including data selection and decimation; iterating over multiple dimensions, and dimensionality changes, all with simple modules which expect to be given the atomic type of data they are designed for. Similar workflows are built to perform time-varying, depth-varying, mask-based or horizon-based processing for all modules without introducing any data handling complications in the filters and computational kernels comprising the modules.

Figures 2 and 3 graphically depict the difference in applying a three-dimensional filter to different sub-volumes of data sets comprising five-dimensional azimuth sectored angle gathers in depth and four-dimensional offset gathers in time. In Figure 2, the data set has five dimensions, which are stored on disk in the order: *z, angle, azimuth, crossline and inline*. In this example, the 3D smoothing is applied on the fastest three dimensions*, z, angle, and azimuth*. The processing in this case is straightforward. The minimum amount of data needed to feed the pipeline is an azimuth-sectored angle volume at a single common image gather point (CIGs), for which I/O will be efficient. Tickets can be built from one or more CIGs and each computational node can work on different CIGs independently. In Figure 3, things are more complicated. In this case the four dimensional data set is stored on disk in the order: *time, offset, crossline and inline*, and the smoothing is to take place over the three slowest dimensions. It would be disastrous from an I/O perspective to read and write one sample at a time as the data needs of the workflow suggest. Instead all four dimensions are read at once and the sub-volumes for each time sample are staged through memory. The loop over time is implicitly handled by the flow engine. Due to the memory requirements, the individual compute nodes are no longer independent. In this case an automatic domain decomposition is performed and all the nodes work together to accomplish the desired workflow. As a result, the composite run time for filtering the three slowest dimensions is virtually identical to the run time for processing the three fastest dimensions. These examples use a single module, but the same principles apply to more complex workflows with multiple modules exhibiting different data requirements.

Discussion

In the post-processing stage of a seismic processing project, a number of piped workflows are used to apply geophysical algorithms as part of a sequence. A single component of that sequence is shown in Figure 1, where a 3D Filter is applied after a domain transform. The type and number of workflows used in post-processing depends on the data; what it will be used for, and the contractual obligations. More often than not a sequence is a collection of data conditioning, editing and refining, requiring multiple data outputs as well as dimensional variations and domain transforms.

To achieve a successful outcome using a conventional approach a geophysicist needs to interact with the data by modifying each workflow. All individual components need to be optimized, both for I/O and for the computer resources required to efficiently complete the task; what might be appropriate for one component might not be suitable for another. The new automated engine optimizes the resource requirements, while minimizing the network load. As a consequence, a simple post-processing sequence of workflows including denoising; muting, stacking, filtering and whitening can be reduced to a single sequence with each individual task optimized, reducing the time taken for the entire chained workflow by orders of magnitude.

Singular workflows, such as those found in Figure 1, might also be used during the testing phase of a project. Many post-processing tasks are 'near instantaneous', particularly when they contain simple components that apply a process without dimensionality reduction and expansion, or domain transforms. More convoluted workflows that required these variations become significantly more challenging for applications of multi-dimensional processes on high-dimensional data. As previously described the new platform automatically manages the complications of dimensional expansion and domain transform, and enables 'near instantaneous' applications to complex algorithms, which in turn can reduce testing turnaround.

Conclusions

Seismic data processing means more than the application of geophysical algorithms to discretely sampled data. As computer power is consolidated into mega-centres, and individual processor speeds grow exponentially, we allow the complexity of the geophysical tasks to increase as we seek to honour the theoretical physics. At the same time, seismic data processing volumes are increasing, both volumetrically, and dimensionally as we explore ever more innovative ways of applying geophysical algorithms.

The primary focus of automation in seismic processing is to remove human intervention using creative and pragmatic machine learning. It is a growing aspect of our industry, as seismic contractors investigate and implement new and innovative ways to reduce the time it takes to deliver data.

In conclusion, we face an industry where, in the midst of increasing pressure on delivery, compute resources are being centralized and geophysical algorithms are becoming more complex and resource intensive. To enable an effective throughput of conglomerated geophysical processes we have developed a platform that can seamlessly handle any data, regardless of its dimensionality or the system architecture on which it is being applied. Resources are optimized, and the user interaction with the data is minimized. Self-regulation of resource needs, whether dynamically balancing work, or rejecting and correcting failing components, is a key element of the engine. A detailed statistical feedback loop allows the user to understand minute-by-minute progress of the workflow, and enables more accurate future forecasting of project work using data analytics on automatically populated databases. Whether the system is resolving singular compute-intensive processes, or convoluted piped workflows, the main benefit of the new engine is a significant improvement in project turnaround.

Acknowledgements

The authors wish to thank PGS for permission to publish the paper.

References

Araya-Polo, M., Jennings, J., Adler, A. and Dahlke, T. [2018]. Deep-learning tomography. *The Leading Edge*, **37**(1), 58-66.

Long, A. and Martin, T. [2019]. Automation of marine seismic data processing. *The Leading Edge*, **38**(12), submitted to publication.

Martin, T. and Bell, M. [2019]. An innovative approach to automation for velocity model building. *First Break*, **37**(6), 57-65.

Øye, O.K. and Dahl, E.K. [2019]. Velocity model building from raw shot gathers using machine learning. *Second EAGE/PESGB Workshop on Velocities*, Extended Abstracts, Th VW 11.

Yang, F. and Ma, T. [2019]. Deep-learning inversion: A next-generation seismic velocity model building method. *Geophysics*, **84**(4), R583-R599.