Article from

CompAct

April 2019

Issue 59

A Smart Way to Accelerate Model Runs Through In-force Data Compression

By Ramandeep Nagi, Dean Kerr and Xin Yao Li

Liability in-force data **compression** is a solution to shorten model runtime by reducing the number of model points. In this article, we will dive into compression approaches, specifically clustering algorithms, and outline how compression can be implemented effectively.

Section 1 provides an overview of **cluster analysis** and describes two common clustering algorithms: K-means and hierarchical agglomerative clustering. Section 2 outlines how to implement a hierarchical agglomerative clustering algorithm. Section 3 illustrates runtime savings achieved by a compression model under different levels of in-force data compression.

Definitions of certain technical terms are provided on page 29; these terms are bolded the first time they are used.

Exhibit 1
Plot of data points based on two location variables

SECTION 1: CLUSTER ANALYSIS

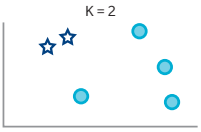
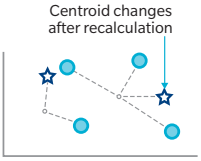
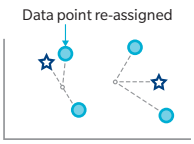
Compression is a type of cluster analysis that groups data points based on a set of characteristics. Clusters can be defined as a group of data points with short **distances** among members or as dense areas in the data space. While clustering algorithms differ in the methodology used to combine data points, all share common properties:

- Clustering is accomplished by setting specific characteristics of data points as **location variables.** (See Exhibit 1)
- The chosen clustering algorithm then iteratively groups data points to optimize a defined objective function.

Clustering Algorithms

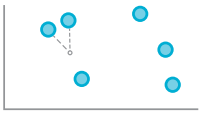
Two common clustering algorithms are K-means and hierarchical agglomerative clustering. (See Exhibit 2)

Exhibit 2: K-means Clustering Algorithm

Step 1:	Randomly select k data points as **centroids**, where k represents the desired number of clusters.	
Step 2:	Assign every data point to its nearest centroid.	
Step 3:	Redetermine the centroid of each cluster based on available data points in the cluster.	
Step 4:	Repeat steps 2 and 3 until clusters reach their target state, which is when additional iterations have no impact on the cluster selection.	

A K-means clustering algorithm is simple to define and illustrate. It partitions the data into a well-distributed set of clusters when k is relatively small. However, this technique can be sensitive to outliers and random initial assignment of the k data points. (See Exhibit 3)

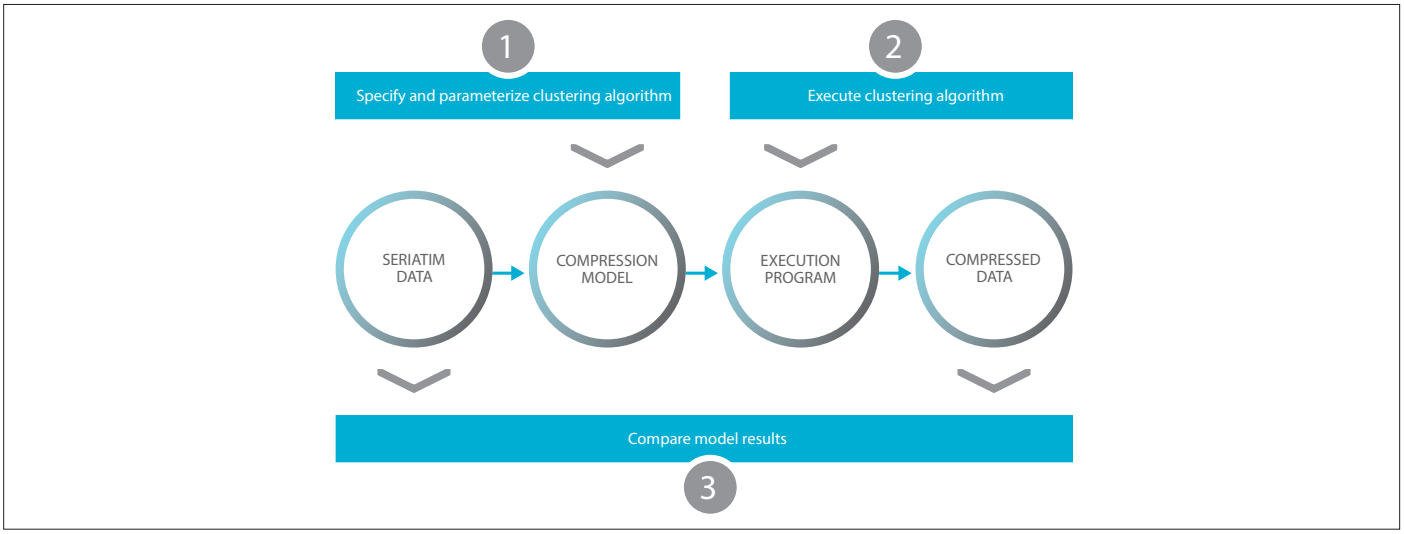
Exhibit 3
Agglomerative Hierarchical Clustering Algorithm

Step 1:	Treat every data point as an individual cluster. Calculate the distance between each cluster.	
Step 2:	Merge the closest pair of clusters.	
Step 3:	Repeat step 2 until the target clustering level is reached.	
Step 4:	The result is a set of clusters meeting the target clustering level.	

SECTION 2: PERFORMING COMPRESSION

Exhibit 4 outlines key steps involved in compressing in-force data with a hierarchical agglomerative clustering algorithm.

Exhibit 4
Compressing In-force Data



DEFINITIONS

Centroid: The arithmetic mean position of a given set of data points.

Cluster analysis: Data analysis technique that groups data points into clusters.

Compression: Type of cluster analysis technique that compresses large sets of data points into more compact sets.

Compression ratio: Number of data points (e.g., model points) after compression relative to the original number of data points (e.g., seriatim policies).

Distance: Normally the Euclidian distance between two data points in terms of their location variables.

Distortion: Alteration of the original characteristics of the data. As a clustering algorithm executes, distortion is inherently introduced into the data model.

Location variables: Location variables reflect policy characteristics or risk drivers of the underlying policies in the clustering algorithm.

Measure: A metric an actuary attempts to control, or preserve, between the full seriatim and compressed data models (e.g., total reserves).

Weight: Importance assigned to each location variable used to determine the measure metric.

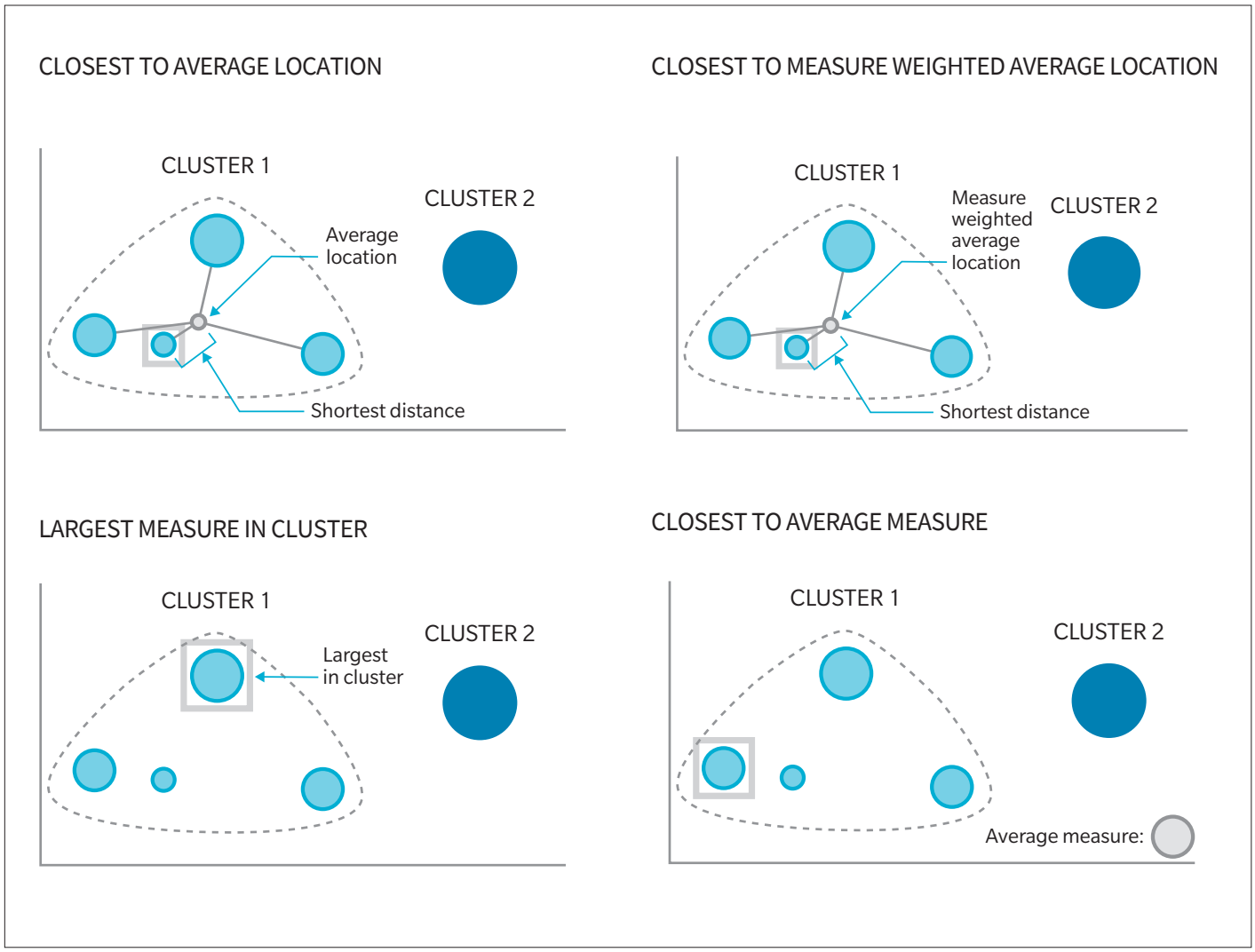
Step 1: Specify and Parameterize Clustering Algorithm

It is important to select a suitable compression algorithm for the problem at hand. K-means has an advantage of being a very fast algorithm but requires predetermination of how many natural clusters exist in the data at the beginning of the process. This information is unknown at the beginning and is generally gained through repetitions of the clustering algorithm. On the other hand, agglomerative hierarchical clustering does not require knowledge of the number of clusters at the beginning of the process but is a much slower algorithm compared to K-means.

The main inputs into the clustering algorithm are full seriatim data, location variables, **weight** of location variables and the **measure**. In addition, data segments can also be defined to achieve better compression results. Segmenting policies (e.g., by major product line, GAAP cohort, gender, etc.) and separately compressing each segment (e.g., different **compression ratios**, location variables, etc.) will generally lead to the best fit of results and decrease the time required to run the clustering algorithm.

Once the clustering algorithm determines which policies are compressed to create a cluster, it becomes important to determine which policy will represent the cluster. This is achieved by creating rules that determine the representative policy for each cluster and its characteristics. A cluster is thus represented by a real policy whose characteristics are already part of the seriatim data. Four possible output linkage rules are shown in Exhibit 5.

Exhibit 5
Examples of Output Linkage Rules



Step 2: Execute Clustering Algorithm

Many ways exist to program and execute a clustering algorithm. In addition to actuarial software, common approaches are to utilize SQL, VBA, R and Python.

Clustering functionality is available in most modern actuarial software platforms. Such functionality can be helpful when compressing model points for inner loop projections. Certain reserving standards (e.g., AG43, VM-20, SOP 03-1) require stochastic calculations. Performing stochastic reserve calculations in an actuarial forecast (often referred to as stochastic on deterministic) significantly increases the computational strain to generate financial results. To overcome this challenge, certain actuarial software platforms offer the functionality to perform reserve revaluations (i.e., inner loop projections) using compressed model points while maintaining the granularity of the main forecast (i.e., outer loop projection) with full seriatim data. This setup improves model runtime proportionally to the compression ratio of the inner loop data model.

A clustering algorithm can also be implemented in SQL, VBA, etc. This may provide additional transparency as a modeler can see the building blocks of the compression algorithm. However, it typically requires programming the clustering algorithms from first principles, which can be time-consuming and may also result in control or efficiency issues.

Finally, due to advancements in data science, clustering algorithms are also available in both R and Python ("scikit-learn" library). The modeler can leverage available libraries for existing code and create modified functions for a range of clustering algorithms.

Step 3: Compare Model Results

The compressed model should be evaluated by comparing model outputs between compressed and seriatim model runs. Experimentation may be necessary to determine optimal parameters: location variables, weights, measure, output linkage rule, segments, and compression ratios.

Careful consideration is required when choosing the location variables. The performance of a compression model depends on how well the location variables represent the underlying policies. For example, for a valuation model, one should choose location variables that drive reserve levels. If policies are not well represented by the location variables and weights, **distortion** will occur even with minimal compression.

Furthermore, once a compression process continues beyond compression ratios supported by the data and attempts to cluster policies that differ more significantly, distortion will increase. This is called over-clustering. As an example, consider the loss of accuracy when attempting to group all policies into a single model point.

Thus, the compression process should involve a tuning phase specific to the intended application. This phase involves selecting location variables and their respective weights based on trial runs and may require several iterations to achieve adequate calibration. However, once a suitable compression model is established, significant efficiency can be achieved without material loss of fidelity in results.

SECTION 3: ILLUSTRATIVE MODEL RESULTS

Compression was performed on an illustrative variable annuity product using a range of compression ratios and compressing on key risk drivers. The following charts show resulting statutory reserves under a range of compression ratios along with the reduced model runtime. (See Exhibit 6)

Exhibit 6
Statutory Reserves Under a Range of Compression Ratios

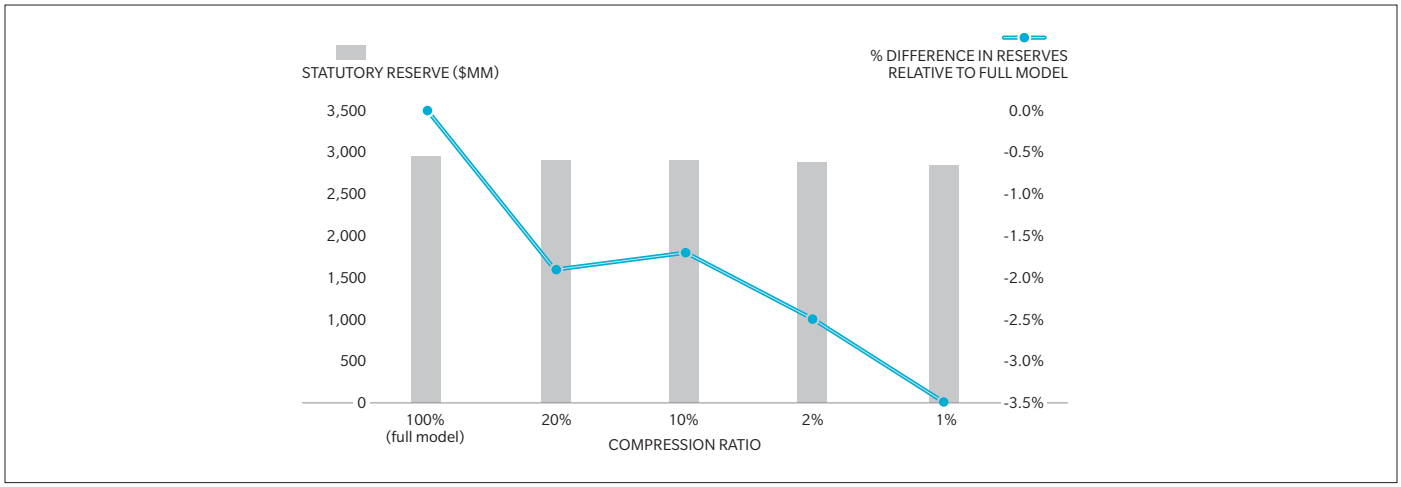


Exhibit 7
Model Runtime Under a Range of Compression Ratios

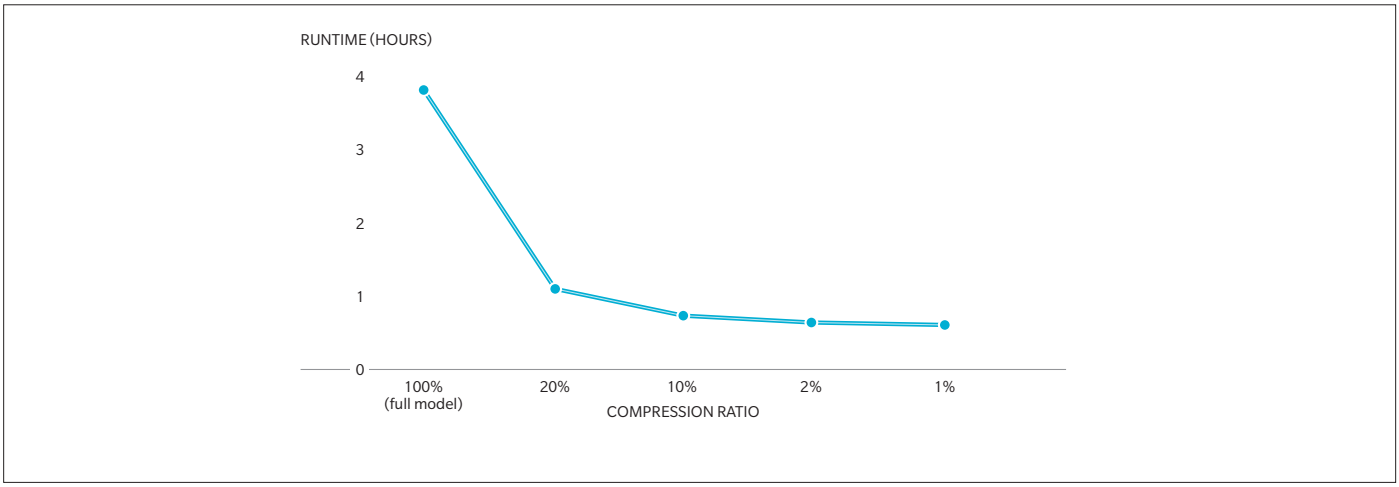


Exhibit 7 illustrates the significant benefit a company may realize by implementing an intelligent clustering algorithm. Valuation (i.e., calculation) runtime is reduced proportionally to the reduction in model points, while calculated reserves deviate by a reasonable margin. Note that overall runtime does not reduce proportionally due to model overhead, such as in-force loading and certain model aggregation and output processes.

CONCLUSION

In-force data compression provides insurers advanced data clustering techniques and a practical solution to reducing model runtime. For computationally intensive tasks such as stochastic modeling and forecasting, the efficiency achieved by developing a robust compression process could outweigh the loss in model fidelity and upfront development costs. ■

The views or opinions expressed in this article are those of the authors and do not necessarily reflect the official policy or position of Oliver Wyman.

Dean Kerr, FSA, MAAA, ACIA, is a partner at the Actuarial Practice of Oliver Wyman. He can be reached at *dean.kerr@oliverwyman.com.*

Ramandeep Nagi, FSA, MAAA, FCIA, is a senior consultant at the Actuarial Practice of Oliver Wyman. He can be reached at *ramandeep.nagi@oliverwyman.com.*

Xin Yao Li, ASA, is a consultant at the Actuarial Practice of Oliver Wyman. She can be reached at *xinyao.li@oliverwyman.com.*