# Basin DNA: Gaining subsurface insight using unsupervised learning on heterogeneous data

Antoine Vial-Aussavy; Pooja Rani; Daniel Conlin; Joel Graves; Jeremy Vila

## Background

Studying the subsurface can rarely be done through direct observations. The data we gather for this task comes from a wide variety of tools with different purposes. One consequence is that despite observing the same medium, the data acquired can be highly heterogeneous in their size, dimensionality and domain.

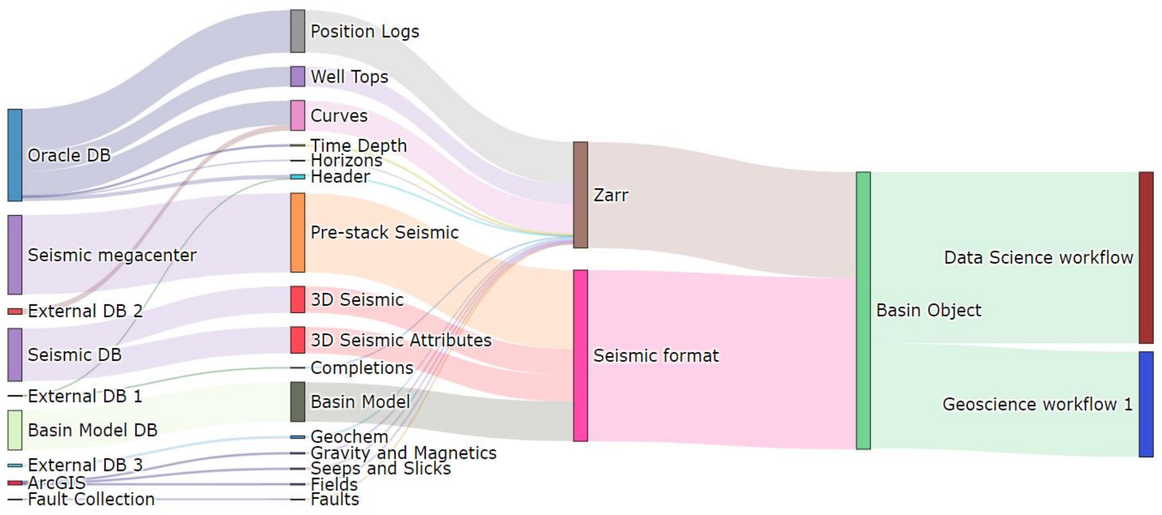
Basin DNA tries to leverage recent advancements in data science to consume this vast amount of heterogeneous data as a way to discovered correlations and patterns via capturing the essential information embedded in all these relevant data points.

Zarr is the powerhouse behind Basin DNA, it offers a comprehensive solution to gather different data types into a single Pythonic entity: the basin object. This approach abstract most of the data wrangling too often taking time away from the data scientists.

The architecture proposed makes use of multiple auto encoders for each data type in order to gain insights from the different data sources.

## Data Integration

The Basin object is built utilizing data from siloed databases, unified into one unique silo. Zarr serves as an easy medium for storing and organization the various dimensional arrays and metadata. Larger objects (> 1 Tb) are kept in their native formats but a wrapper is built in order to integrate them in the Zarr environment




## Basin Object

Once built, the basin object exhaustively regroups all the data for a given area. This can be used to build complex integrated workflows.

```
import pandas as pd
import plotly.graph_objects as go
```

```
from basindna import Basin
from basindna.visualize import Visualize
```

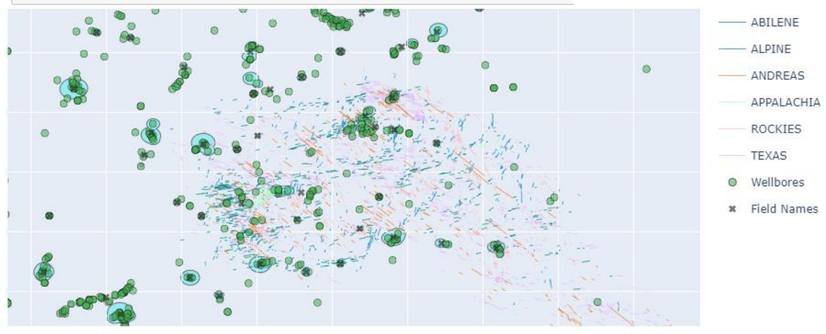
```
# Retrieve the data
the_basin = Basin.from_environment('dev')
the_basin
```

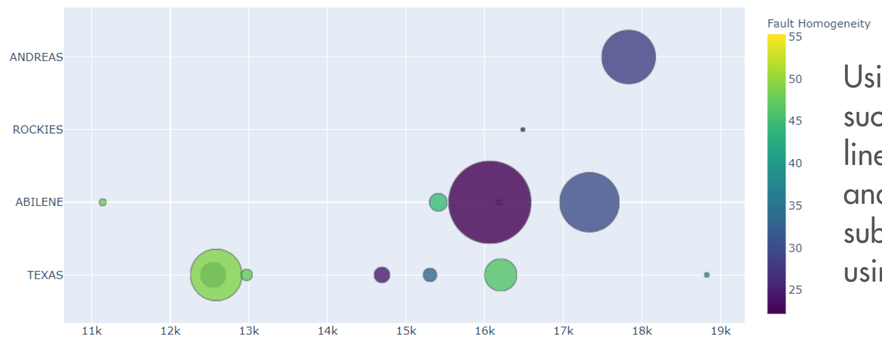
```
Basin(Wellbores: 3222, Horizons: 14, Faults: 6, Fields: 100)
```

The basin object regroups all the data from a given environment. Once loaded, all the different data types become directly accessible

This allow us to use Python to quickly test complex geoscience hypothesis or to visualize a subset of the data. Shown below is an example of such hypothesis. A geoscientist was looking for correlation between field depth (x-axis) and neighboring faults dominant orientation (y-axis).

```
# Show data used in the investigation
vis = Visualize(the_basin)
vis.map_basin()
```




Field depth vs fault orientation. Size is field size

Using the basin object, testing such hypothesis takes only a few lines of code while a similar analysis would have been substantially more complicated using conventional software.

## Representation Fusion

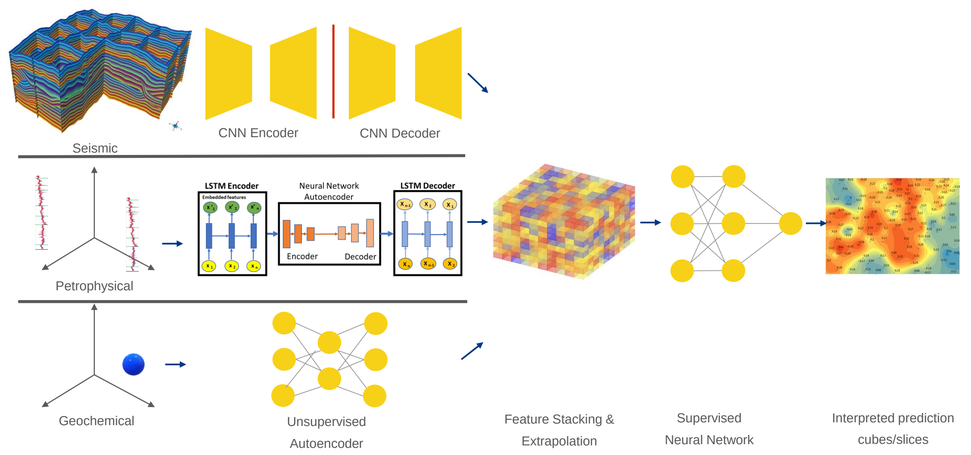
We outline a general approach to fusing representations from data sources.

1) Using each of the subsurface data sources as inputs, we propose to train unsupervised models in the form of autoencoders to extract the intrinsic representation of the data.

- **Seismic:** we implemented a w-net architecture with two sequences of compression and decompression. The hidden representation has the same number of time samples but compressed in other pre-stack dimensions.
- **Petrophysical:** different logs, e.g., gamma ray, resistivity are the input features into a Recurrent Neural Network. The compressed representation is used to predict the next sample (depth-wise) and to reconstruct via decompression.
- **Geochemical:** we implemented a neural network architecture.

2) For sparse data, the encoded features can be extrapolated to the seismic volume, then once registered, stacked to "fuse" encoded features.

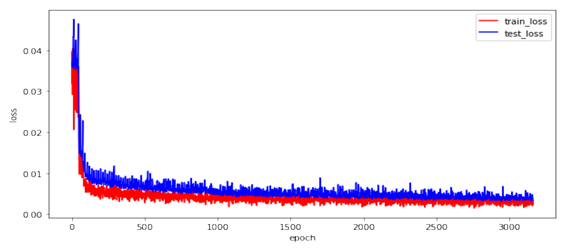
3) Then, given the labels we propose to train a simple neural network on each of the "fused" hidden vectors for each labeled voxels. Predictions are generated on any unlabeled voxels.




Propose Data Fusion Workflow

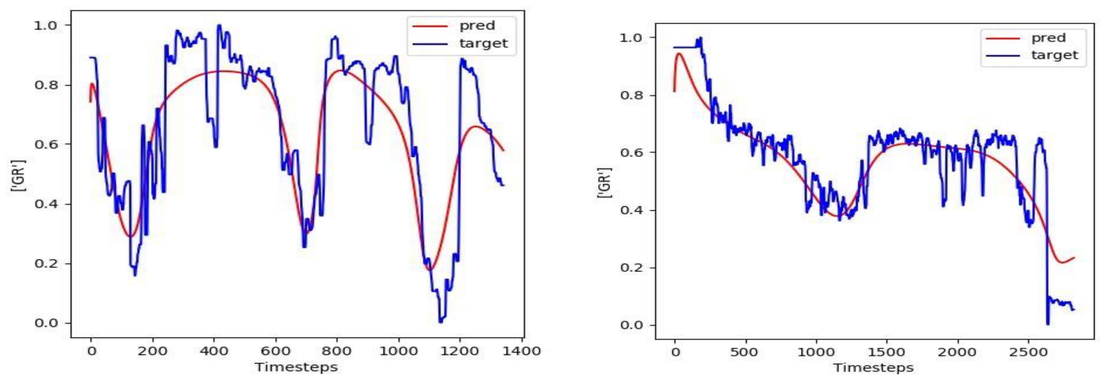
## Empirical Results

We show the RNN autoencoder model reconstruction loss for petrophysical data on training and hold-out testing sets. The model was forced to have a compression level of 40x, but still see a convergence in reconstruction.




Training and Testing validation loss

We also show the reconstructed Gamma Ray signature on two wells. Though not shown, relaxing compression rates leads to better reconstruction. However, having a compact representation typically makes the final supervised prediction more straightforward.




Two well logs True vs. Reconstruction

Final trained AE models for seismic and geochemical data is future work.

## Conclusion

Basin DNA is still being actively developed and despite facing some performance issues, the project team is confident that approach chosen is scalable thanks to, amongst other things, Zarr capabilities to directly access MongoDB storage.

Beyond the data architecture, Basin DNA has proven to be a great way to integrate and consume data by building new and innovative workflows. Given the geoscientist community growing appetite in using Python we hope that the solution will become a cornerstone of our subsurface workflows