Final Report

June 2021

Development of a Framework for Data Integration, Assimilation, and Learning for Geological Carbon Sequestration (DIAL-GCS)

SUBMITTED UNDER FUNDING OPPORTUNITY ANNOUNCEMENT (DE-FOA-0001240)

WORK PERFORMED UNDER AGREEMENT

DE-FE0026515 (10/01/2015 – 03/31/2021)

SUBMITTED BY

The University of Texas at Austin
Jackson School of Geosciences
Bureau of Economic Geology
University Station, Box X
Austin, Texas, 78713

DUNS #170230239

PRINCIPAL INVESTIGATOR

Alex Sun, PhD
PH: 512-475-6190
alex.sun@beg.utexas.edu

SUBMITTED TO
PROJECT MANAGER
Bruce M. Brown
Bruce.Brown@NETL.DOE.GOV

U. S. Department of Energy
National Energy Technology Laboratory

Final Report

June 2021

Development of a Framework for Data Integration, Assimilation, and Learning for Geological Carbon Sequestration (DIAL-GCS)

Leading Authors:

Alexander Y. Sun (Principal Investigator)
Katherine Romanak,
Sergey Fomel

Bureau of Economic Geology, Jackson School of Geosciences,
The University of Texas at Austin

DISCLAIMER

ABSTRACT

This project aimed to develop and demonstrate a Data Integration, Assimilation, and Learning framework for geologic carbon sequestration projects (DIAL-GCS). DIAL-GCS is an intelligence monitoring system (IMS) for automating GCS closed-loop management by leveraging recent developments in machine learning technologies, complex event processing (CEP), and reduced-order modeling. The safe and efficient operation of GCS repositories requires integrated monitoring to track the injected CO2 as it moves within a storage reservoir. GCS projects are data intensive, as a result of proliferation of digital instrumentation and smart-sensing technologies. GCS projects are also resource intensive, often requiring multidisciplinary teams performing different monitoring, verification, accounting (MVA) tasks throughout the lifecycle of a project to ensure secure containment of injected CO2. The success of GCS thus depends in a large part on our ability to access, assimilate, and analyze heterogeneous data and information sources in a timely manner.

This project included a number of meaningful and necessary tasks to transform the human domain knowledge into machine-interpretable rules for automating knowledge extraction and discovery in GCS. The specific technical objectives of the proposed DIAL-GCS project were to

- Task 2: Develop an ontology-driven GCS data management module for storing, querying, and exchanging GCS data (both historic and live sensor data) from multiple sources and in heterogeneous formats
- Task 3: Incorporate a CEP engine for detecting abnormal situations by seamlessly combining expert knowledge, rule-based reasoning, and machine learning
- Task 4: Enable uncertainty quantification and predictive analytics using a combination of coupled-process modeling, AI/ML methods, and reduced-order modeling, and
- Task 5: Integrate and demonstrate the system's capabilities with both real and simulated data

As far as we know, this is one of the first projects aimed to develop intelligent monitoring systems (IMS) targeting the GCS. Under this project, the team had developed a large number of web applications and scientific algorithms that contribute the main theme of intelligent monitoring. The team has published more than a dozen peer reviewed papers and disseminated the research results at multiple technical meetings.

ACKNOWLEDGMENTS

The authors would like to thank DOE NETL for funding this research and the Jackson School of Geosciences at The University of Texas at Austin for providing the cost share. We are grateful to the DOE NETL project manager, Mr. Bruce Brown, for his guidance throughout during the project. We thank colleagues at the Gulf Coast Carbon Center at Bureau of Economic for endless insightful discussion. Finally, the project objectives could not have been completed without the contribution of postdoctoral scholars (Drs. Hoonyoung Jeong and Zhi Zhong).

TABLE OF CONTENTS

1. INTRODUCTION

The safe and efficient operation of GCS repositories requires integrated monitoring to track the injected CO2 as it moves within a storage reservoir. GCS projects are data intensive, as a result of proliferation of digital instrumentation and smart-sensing technologies. GCS projects are also resource intensive, often requiring multidisciplinary teams performing different monitoring, verification, accounting (MVA) tasks throughout the lifecycle of a project to ensure secure containment of injected CO2. The success of GCS thus depends in a large part upon the operator's ability to access, assimilate, and analyze heterogeneous data and information sources in a timely manner.

The main aim of this project was to develop and demonstrate a data integration, assimilation, and learning framework for geologic carbon sequestration projects (DIAL-GCS). DIAL-GCS was designed to be an intelligence monitoring system (IMS) for automating GCS closed-loop management by leveraging recent developments in artificial intelligence/machine learning (AI/ML) technologies, complex event processing (CEP), and reduced-order modeling.

The DIAL-GCS project included a number of meaningful and necessary tasks to help transform the human domain knowledge into machine-interpretable rules for automating knowledge extraction and discovery in GCS. In terms of functional requirement, the IMS focused on 3 I's that are critical to the success of real time anomaly detection, namely Instantaneity, Intelligence, and Integration. Instantaneity means real time—the IMS needs to be able to handle high volume, high-frequency data. Intelligence means that the IMS needs to have significant machine learning capability, and Integration means that the IMS needs to integrate all the components in a seamless manner.

1.1 Project objectives

The project consists of the following main objectives/tasks (see also Figure 1):

- Task 2: Develop an ontology-driven GCS data management module for storing, querying, and exchanging GCS data (both historic and live sensor data) from multiple sources and in heterogeneous formats
- Task 3: Incorporate a CEP engine for detecting abnormal situations by seamlessly combining expert knowledge, rule-based reasoning, and machine learning
- Task 4: Enable uncertainty quantification and predictive analytics using a combination of coupled-process modeling, ensemble-based data assimilation, and reduced-order modeling, and
- Task 5: Integrate and demonstrate the system's capabilities with both real and simulated data

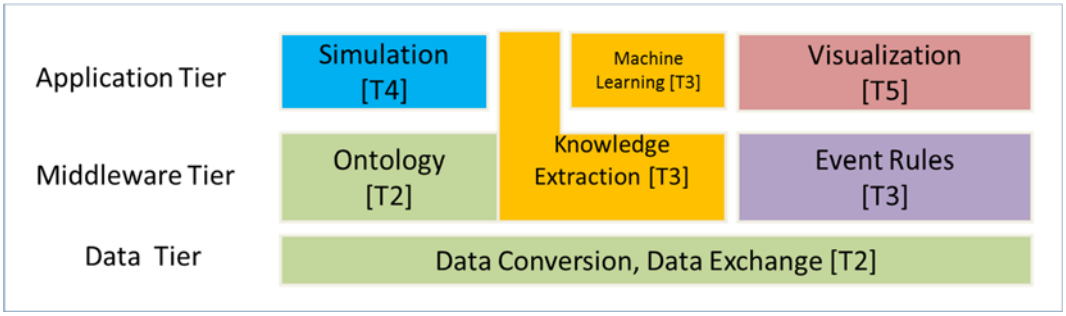

Figure 1. DIAL-GCS project is designed as a multi-tiered system: data tier (Task 2) handles raw data transformation and storage; middleware tier handles complex event processing (Task 3) and multiphysics modeling (Task 4), application tier handles presentation and visualization.

1.2 Organization of the report

This report is organized as follows. The technical details and accomplishments are summarized for each of the tasks in Sections 2 to 5, followed by lessons learned and conclusions. The list of are publications and presentations are given in the appendix.

2. TASK 2: DATA MANAGEMENT MODULE

2.1 Database selection process

Continuous monitoring sensors can generate significant volume information over time. For IMS to ingest the information, the raw data need to be cleansed and homogenized. In data science this is referred to as the data wrangling process. In the context of this IMS, it entails developing a set of scripts to transform the raw sensor data stream to event stream, which can then be analyzed (e.g., event correlation or event causal analysis) by the complex event processing engine. Datastreams are infinite, as opposed to finite-length data collected during some discrete sampling period. Therefore, the database chosen for the IMS backend must possess the following capabilities:

- Ability to handle high-volume streams arising from real-time GCS sensing;
- Ability to extract events from large temporal windows.

The second requirement in the above has implications on the first requirement, namely, how much historical data the database should archive for historical event lookup or for training the ML codes.

The relational database servers (e.g., Microsoft SQL Server, Oracle DB, PostgreSQL) have been extensively used to store sensor data in tabular forms, which are linked to each other through the so-called database keys to describe the relationship among data. Many of today's online applications have high data storage requirements that exceed the capabilities of the legacy relational databases. Specifically, database schemas can slow down database performance even though they provide conceptual encapsulation of the data relationships.

A new generation of scalable database servers have been designed without using the relational database paradigm to reduce query latency and increase data retrieval performance. This new generation of databases are collectively known as NoSQLs, indicating their departure from the key design concept behind the legacy relational databases.

Table 1 summarizes the main differences between a relational database and NoSQL server. It can be seen that a NoSQL database is ideal for handling data streams that are typified by high volume but relatively simple data structures. NoSQL databases are schema-free and two types of design are used in those databases: (a) key-value stores, which store the key-value pairs, and (b) wide column stores (also called extensible record stores), which store data in records with an ability to hold very large numbers of dynamic columns. Unlike a relational DB, the names and format of the columns can vary from row to row in a wide-column store.

Table 1. Comparison between relational DB and NoSQL DB. Adopted from https://academy.datastax.com/resources/brief-introduction-apache-cassandra

Relational DB	NoSQL DB
Handles moderate incoming data velocity	Handles high incoming data velocity
Data arriving from one/few locations	Data arriving from many locations
Manages primarily structured data	Manages all types of data
Supports complex/nested transactions	Supports simple transactions

Supports moderate data volumes	Supports very high data volumes
Centralized deployments	Decentralized deployments
Data written in mostly one location	Data written in many locations
Supports read scalability (with consistency sacrifices)	Supports read and write scalability

At the beginning of this project, we surveyed existing datastream storage servers. An extra focus was on those servers that are open-source, but provide certain event processing capabilities. Table 2 lists some of the databases that were surveyed. On the basis of IMS development needs, the team chose InfluxDB (https://www.influxdata.com) to serve as the backend server because it not only comes with out-of-box support for datastream storage and high-performance aggregate query, but also is seamlessly integrated with a complex event processing engine that is related to Task 3 of the project (Note: during later stage of the project, the team switched to a more integrated platform developed under the open-source Apache Superset project).

Table 2. Examples of NoSQL products surveyed as part of Task 2.

Database	**Open Source**	**Main Feature**	**Store**	**Event Processing Capability**
Apache Cassandra	Y	Linear scalability and high availability	Schema free; wide column	Provides Cassandra Query Language, a limited set of full SQL, poor for aggregation query
Amazon SimpleDB	N	Database as a service. Automatically creates multiple geographically distributed copies of each data item, providing high availability and durability	Key-value	
InfluxDB	Y	Purpose-built for time series data, no special schema design or custom app logic required; key-value store	Key-value	Kapacitor, is InfluxDB's data processing engine. It allows plugin of custom logic or user defined functions to process alerts with dynamic thresholds, match metrics for patterns or compute statistical anomalies.
Hypertable	Y	A high performance, open source, massively scalable database modeled after Bigtable, Google's proprietary, massively scalable database	Wide column	
Apache Flink	Y	A streaming dataflow engine that provides data	N/A	Supports stream processing and windowing

		distribution, communication, and fault tolerance for distributed computations over data streams		with Event Time semantics
Tibco Streambase	N	A high-performance system for rapidly building applications that analyze and act on real-time streaming data	N/A	Support real-time stream analytics

2.2 Database development using InfluxDB

Under Task 2, we developed a suite of Python scripts for persisting monitoring data to InfluxDB. These scripts correspond to the data adaptor or middleware development mentioned under Subtask 2.2 of the project SOPO. Figure 2 shows the main components used, including InfluxDB for data persistence, and Grafana for data stream visualization.

During the course of the project, these Python scripts were constantly evolved and expanded so that different sensor data formats can be accommodated. Essentially, for each different type of sensor data, a data dictionary was developed that maps the content of the raw data files to key/value tuples for InfluxDB. For example, for pressure gauge data, the key would be timestamp and the value would be observed pressure values. For distributed temperature sensing (DTS) data, some complexity arose because of the additional dimension of "length-along-the cable." We store the length-along-the-cable information as an additional column. The classic software design pattern, façade, was used to provide a generic interface for initiating data processing (Figure 3). The corresponding sensor names and their manufactures are given in Table 3. Best software practices were followed when developing the data importing routines so that they can be reused.

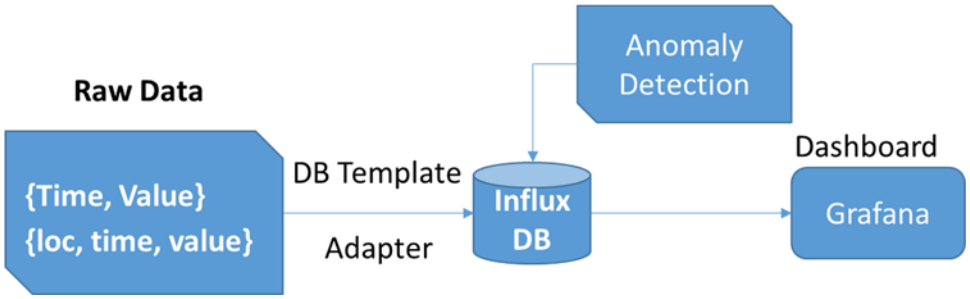

Figure 2. DIAL data processing and visualization workflow.

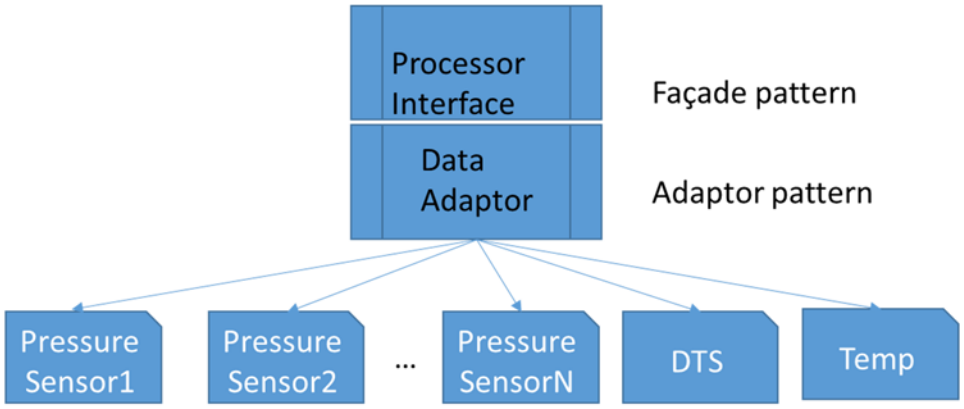

Figure 3. Design of data adaptors or middleware using No-SQL schema.

Table 3. List of data sources used under Task 2 for demonstration.

Manufacturer / Instrument	**Type**
Ranger Gauge System (formerly Panex) /SRO downhole gauge	Pressure, temperature
Silixa / ULTIMA™ DTS	Temperature
Emerson / Micro Motion™ flow meter	Mass flow rate
(Vendor unknown) / SRO wireline pressure gauge	Pressure and temperature
Omega / pressure transducer	Pressure

2.3 Data analytics/visualization portal 1.0

Visual analytics is an essential component of any IMS. The team invested significant amount of time in identifying and developing a versatile data portal for data visualization. Based on the initial survey of web-based systems, the team chose Grafana (https://grafana.com), which provides a powerful and elegant tool to create, explore, and share data. Grafana is a data portal with full-fledged user management capability. Users can make their own custom dashboard to display the time series in a number of styles. In addition to InfluxDB, Grafana can connect to a number of other datastream servers.

Figure 4 shows screenshots of the Grafana dashboard created for displaying downhole pressure and temperature time series collected during a field experiment. At the original 0.1s readout frequency, the amount of data can easily slow down many web-based visualization servers. With InfluxDB/Grafana, the time series are aggregated on the fly and temporal windowing of the data is easy by using slider control. Little latency was observed for all data series we tested.

GCS sensor data often have spatial context. Thus, it is very desirable to show data with respect to their spatial locations to enhance visual analytics experience. For this purpose, the team expanded Grafana dashboard to handle geospatial data, for example, to display vector data and remote layers served through the web map service (WMS) (e.g., the lower-left and upper-right screen shots in Figure 4).

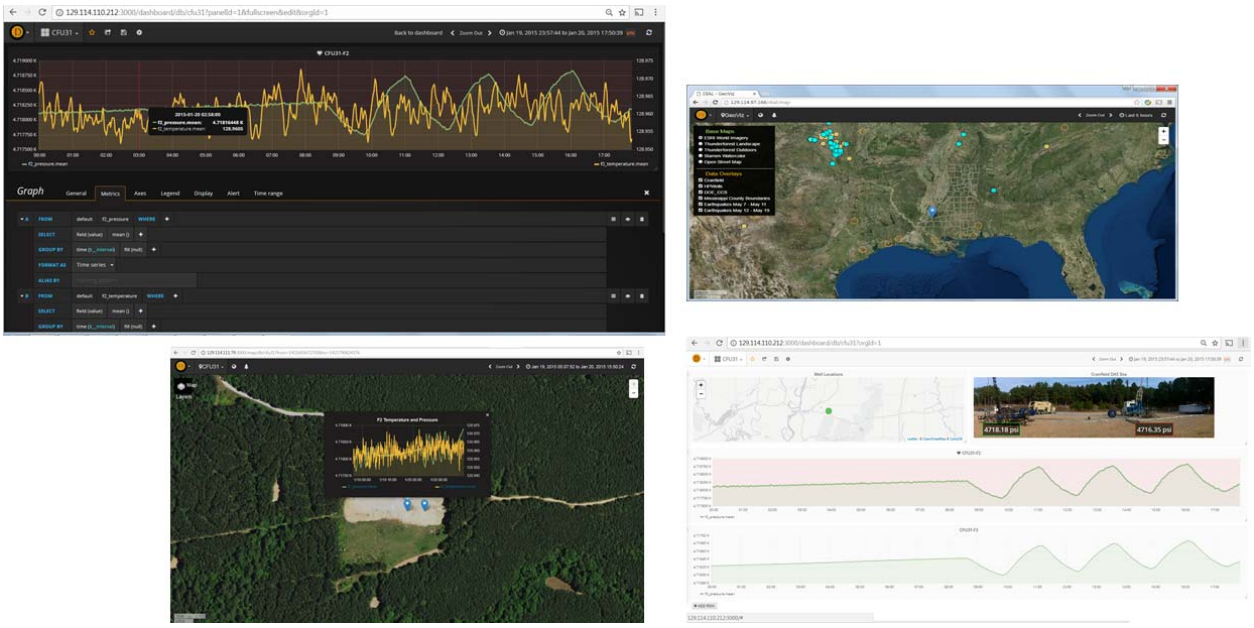

Figure 4. Grafana dashboard for displaying downhole pressure and temperature data.

2.4 Data analytics/visualization portal 2.0

While the combination of InfluxDB/Grafana allows for high-performance data visualization, customization of Grafana became more challenging to accommodate the needs of complex event processing activities in Task 3. Grafana was programmed in Go, which is different from the Python scripting language that used by the project team. Starting from Year 2 of the project, the team switched to a more user-friendly platform, Apache Superset (https://superset.apache.org/). Apache Superset is an enterprise-ready business intelligence web application. It comes with a large number of data analytics features and can connect to a number of commonly used relational and NoSQL databases, including Amazon Athena, Amazon Redshift, Apache Druid, Apache Spark SQL, Snowflake, SQLite, and SQL Server. A comprehensive list can be found on the Superset site.

We would to emphasize the decision to migrate from Grafana to Superset was made after a careful consideration of multiple factors, including team software expertise, development time, and more importantly, the potential to integrate with other tasks of this project.

Figure 5 shows a demo of DIAL-GCS2.0 built using Superset. The time series are from the same data sources as those used in DIAL-GCS1.0 in the last subsection. In this case, the backend complex event processing engine examines each pressure data instance and assign it with either normal or abnormal flag. More details can be found in (Sun et al., 2019) and in next section. The Superset dashboard can be easily customized by the end user. Light programming is required to connect to custom complex event processing engines.

2.5 Task Summary

In summary, Task 2 is a major component of the DIAL project, the outcome of which provides necessary infrastructures to support follow-on implementation of complex event processing using AI/ML. We have successfully developed a web-based data analytics portal for storing and displaying CCS related sensor data by combining a number of state-of-the-art data technologies. Many features of the web system surpassed the original functional spec given in the original project SOPO. Thus, the objectives of Task 2 were successfully met.

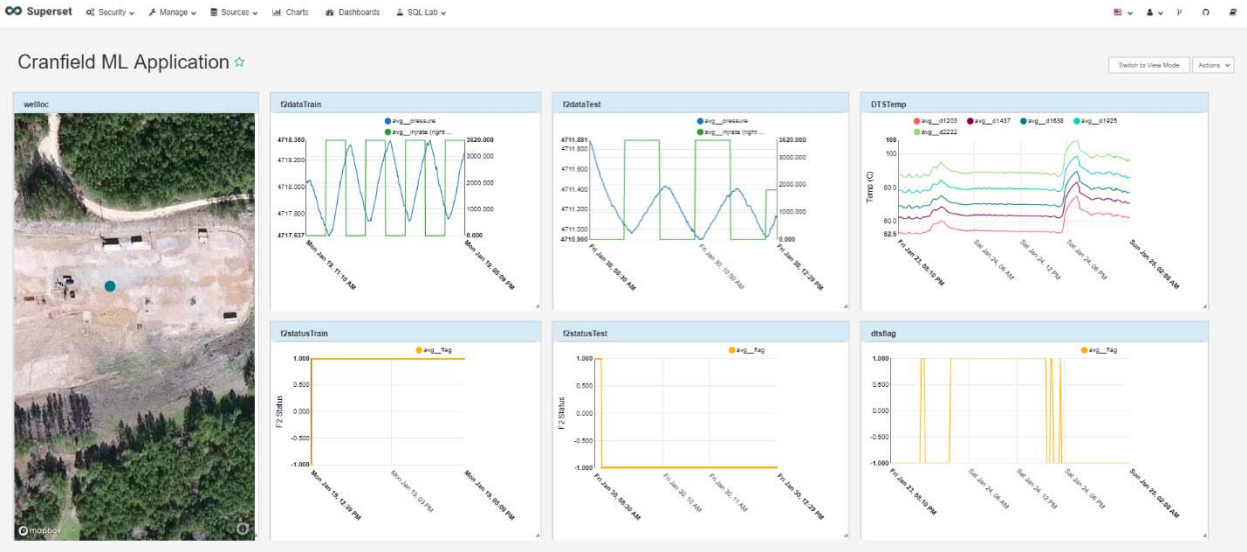

Figure 5. A dashboard developed using Apache Superset.

3. DEVELOPMENT OF COMPLEX EVENT PROCESSING CAPABILITIES

The purpose of Task 3 was to develop a CCS-specific complex event processing (CEP) capabilities for anomaly detection. In particular, the following subtasks were proposed in the SOPO,

- Rule definition: define a set of rules related to event filtering, extraction, and reaction;
- Reasoning and machine learning: identify events using human reasoning and machine learning from multiple sources;
- Testing: validate the CEP using real site data

Although the notion of CEP is not new and has been widely studied in many fields such as credit fraud detection, cyberintrusion detection, and machinery monitoring, its use in CCS industry had not been widely explored at the time of the project. The team perceived the following CCS-specific challenges for implementing CEP,

- Data are both spatial and temporal
- No single anomaly detection algorithm fits all purposes
- "Normal model" may be elusive in many situations
- Anomaly can be "shadowed" by noise
- Training data are rare
- String domain knowledge is usually required to process and understand monitoring signals

With these challenges in mind, the team has developed a suite of anomaly detectors. It is worth pointing out the project duration coincides with the fast evolution of AI/ML technologies in recent years. Thus, the anomaly detection methods developed under this project also evolved, ranging from the traditional ML algorithms to more advanced algorithms. This section summarizes the main activities conducted under Task 3.

3.1 Principles of complex event processing

A CEP engine processes continuous monitoring data to detect possible anonymous events. The CEP is domain dependent. Thus, a robust CEP can only be built with a solid understanding of anomalies and their causes. In this project, anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior (Chandola et al., 2009). In the literature,

these nonconforming patterns are often referred to as anomalies, outliers, discordant observations, exceptions, aberrations (Chandola et al., 2009). Below is simple taxonomy of anomalies in physics domains,

- Point anomaly: an event that lies outside a predefined region. For example, if the maximum daily temperature is outside some long-term climatology, then it is considered an anonymously hot day.
- Contextual anomaly: a type of event that can only be labeled abnormal when examined using multiple pieces of contextual attributes. For example, a pressure anomaly that is caused by increased injection rate is normal, but a pressure anomaly that appears in the absence of operation changes is abnormal.
- Collective anomaly: The individual data instances in a collective anomaly may not be anomalies by themselves, but their occurrence together as a collection is anomalous. For example, in a pressure monitoring data stream, the single instance of pressure perturbation is probably due to noise, while the appearance of consecutive anomalies in a short time period is alerting. In other cases, the appearance of multiple types of anomalies (e.g., pressure and thermal) may also be considered a possible anomaly.

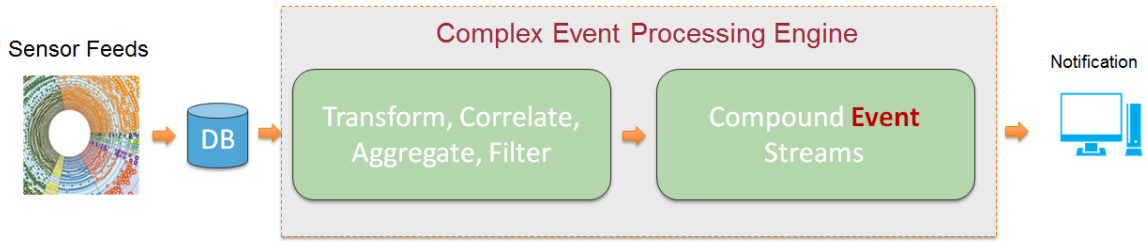

Figure 6. Generic workflow related to a complex event processing (CEP) engine. A CEP engine may perform multiple functions, including transforming, filtering, correlating, and aggregating the raw data, followed by formulating of event streams.

As mentioned before, significant challenges in CCS-related anomaly detection are related to the fact that anomaly signals may be hidden or shadowed by site noise. So far, CCS monitoring has relied mostly on human workflow to process anomalous signals. In a typical scenario, an operator or researcher processes a single case at a time. While such manual process may suffice for pilot scale research projects, it can quickly become overwhelmed as the type/volume of monitoring data increases in a project. Therefore, CCS can greatly benefit from automation of anomaly detection processes. Figure 6 shows such an automated workflow that seamlessly translate raw sensor data into human interpretable results.

For each type of signal, a rule set may be developed to define normal and anomalous signals. This is often referred to as data labeling in machine learning. Labeling is required for supervised learning algorithms, which relates each instance in the training dataset to one or more classes through a parametric or nonparametric model. The rules can be in the form of statistical thresholds or categorical classes. The limitation of labeling is that it can be extremely challenging, often requiring involvement of human experts who are knowledgeable about the subject matter. Even so, labels are usually nonexclusive depending on data availability and subject to ambiguity related to process-level understanding. In CCS, labeling of leakage signals is rarely possible because of the lack of known incidents. Researchers have turned to analogs to increase sample

database. For example, Jordan and Benson (2009) reported that "well blowout rates in oil fields undergoing thermally enhanced recovery (via steam injection) in California Oil and Gas District 4 from 1991 to 2005 were on the order of 1 per 1,000 well construction operations, 1 per 10,000 active wells per year, and 1 per 100,000 shut-in/idle and plugged/abandoned wells per year." Alternatively, a model-based approach may be used to simulate the rare events and quantify its attributes. During detection time, the model predicted outcome is compared to the data instance. If the deviation of the data instance from model prediction is greater than a certain threshold, then the new data is classified as an anomaly.

In unsupervised algorithms, training data are grouped into clusters based on certain similarity measure. Data instances that are far away from the "normal" clusters are considered anomalies. Thus, clustering techniques do not require labeling (i.e., unsupervised). Limitations of clustering are that they do not use a priori knowledge on anomalies, and anomalies are more of a byproduct of the classification process that attempts to partition the data. For algorithms that are not designed to uncover anomalies, the anomalous data patterns may be hidden inside different clusters.

Common challenges to all anomaly detection algorithms include data dimensionality and data volume. Most anomaly detection algorithms are designed for handling low-dimensional data. Uncovering hidden patterns in high-dimensional data is still a challenging research topic.

Under this project, we have taken a number of different approaches for extracting anomalies, with a special focus on real-time online anomaly detection algorithms.

3.2 Algorithm and Results

The capabilities of CEP were demonstrated using a real dataset we collected at Cranfield, MS, during a series of controlled CO2 release field experiment. Details of the field experiments may be found in (Sun et al., 2016).

3.2.1 Sequential z-score algorithm

Z-score algorithm is one of the simplest and, yet, most commonly used online anomaly detection algorithm. Given an instance of x of a data stream, X_t, its z-score is defined as

$$z = \frac{|x - \bar{x}|}{s}$$

where $\bar{x}$ and s represent the mean and standard deviation of the data stream, respectively. If z is greater than a certain threshold, say, 3, then the data instance is labeled as an anomaly. Other test statistics may be used when the sample size is small.

As a demonstration, we applied the z-score anomaly detection to pressure monitoring data. First, the z-score algorithm was trained using data from the first 1.5 hours. The period represents a baseline scenario where no operations were performed other than continuous CO_2 injection. These data are then labeled as normal in our case. Later, we modulated the injection rate to introduce sinusoidal patterns in the signals. The idea is to train an algorithm using the baseline data and then test whether the algorithm can pick up significant pressure changes. Results show that the algorithm successfully detected the pressure changes during the later period (note that the time axis in the bottom panel was used for animation purpose and does not correspond to the actual time). In this case, the anomalies may be considered "point anomalies" because we used a pre-defined threshold to filter each data instance.

Figure 7. Use of z-score algorithm to detect point anomalies. Top panel: raw pressure data stream from one of the monitoring wells during pulse experiments, in which the injection rate was modulated to create pressure anomalies (i.e., the sinusoidal waves shown in the later times). The entire data length is about 16 hours and the first 90 min of data were used to train the algorithm; Bottom panel: detected anomaly (dashed pink vertical lines).

3.2.2 Sequential causality analysis

Causality analysis generally seeks to understand the causal relationships, if there is any, between two types of events. Therefore, they are suitable for observing the "contextual anomalies" mentioned previously. In one of the field experiments, we performed CO2 venting in one of the monitoring wells to simulate leakage while observing pressure responses in a nearby monitoring well. In addition to the leak, the other source of pressure perturbations is injector. The idea of this sequential causality analysis was to test whether pressure responses in the monitoring well can be attributed to leaks. In other words, whether the algorithm was smart enough to differentiate between perturbations caused by injection (normal) and that caused by leak (abnormal). The result of causality test is given in *p* value, which is the probability, under the null hypothesis, of sampling a test statistic at least as extreme as that which was observed. The null hypothesis can be rejected in favor of the alternative hypothesis if *p* value is smaller than a preset threshold. In our case, the null hypothesis is pressure change in the monitoring well is not related to leaks. Figure 8 shows the p value of hypothesis testing is small in the first 2 hours of the experiment, suggesting strong linkage between leak and monitored pressure. As the leak rate was reduced toward the end of the experiment, the causal relationship became less obvious, as shown by large *p* values. Thus, this numerical experiment demonstrated that the causality test can be used to uncover contextual anomalies. This test may be extended to multivariate case, in which multiple pieces of evidence (or predictors) can be combined to detect causal relationships.

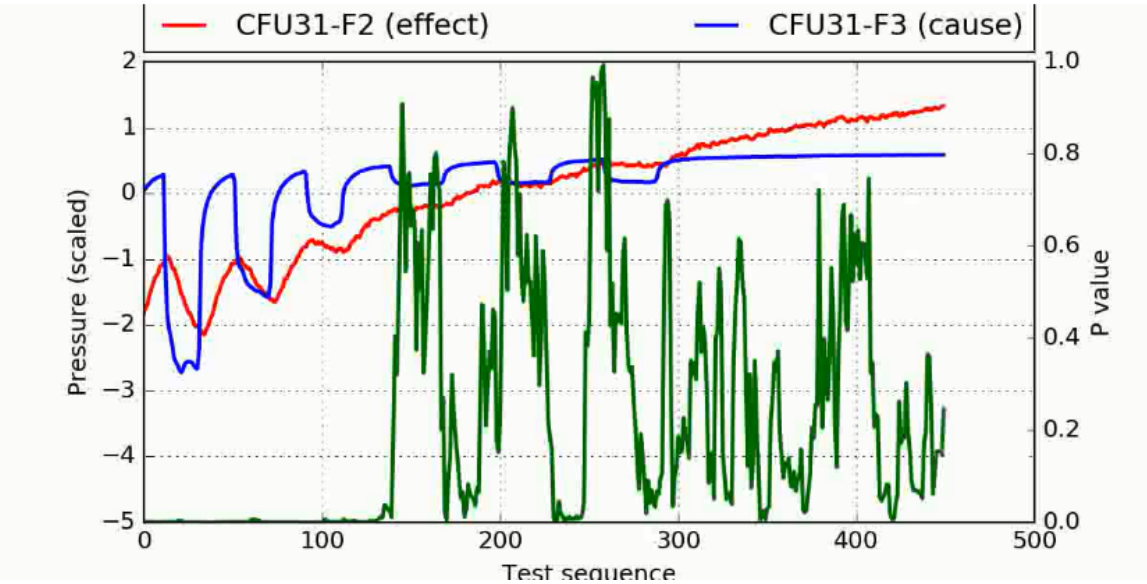

Figure 8. Use of causality test to examine causal relationship between events. Right axis: bottom hole pressure of leak well (blue line, cause) and pressure response in the monitoring well (red line, effect). Left axis: p-value of the causality test on each test data. Here p-value is the result from Granger causality's hypothesis testing, and lower p-value indicates high probability of invalidating the null hypothesis.

3.2.3 Isolation forecast algorithm

Isolation forecast (IFO) is an unsupervised ML algorithm specially designed to detect anomalies. It is based on the premises that anomalies have attribute values that are very different from the rest of the data instances and that anomaly instances are relatively few. To isolate anomalies from a data set, IFO partitions data samples recursively using an ensemble of random trees (Figure 9). When a sample has anomalous attributes, the number of partitions required to isolate the data sample is smaller than that for a "normal" sample. In other words, anomalies are more susceptible to isolation under random partitioning. IFO calculates an anomaly score by averaging path lengths (equivalent to number of partitions) over all random trees. The algorithm only requires two user parameters, namely, the number of trees to build and the subsample size. Subsampling is devised to alleviate the effect of masking (too many anomalies concealing their own presence) and swamping (normal instances located too close to anomalies), thus helping to build better trees more efficiently. We used the IFO function from the open-source machine learning package, scikit-learn (Pedregosa et al., 2011).

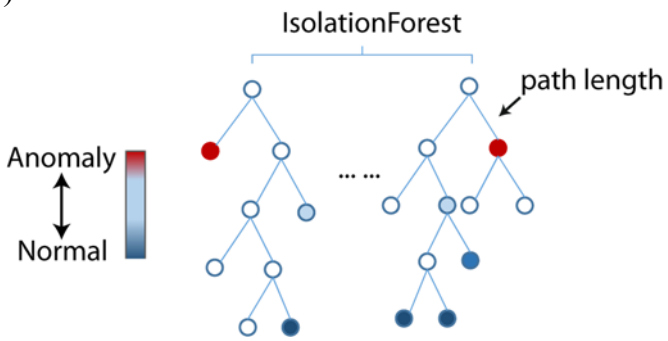

Figure 9. The main idea behind the IsolationForest (IFO) algorithm is that anomalies are more susceptible to isolation under random partitioning.

In Figure 10, the IFO training and testing results are demonstrated. IFO detected no anomalies on the training data, during which no leaks were activated. When the trained IFO model was applied to the controlled release data sequentially, the model correctly labels almost every data instance as anomaly (Figure 10b). In (Sun et al., 2019), we showed that IFO outperformed the traditional support vector machine on this dataset.

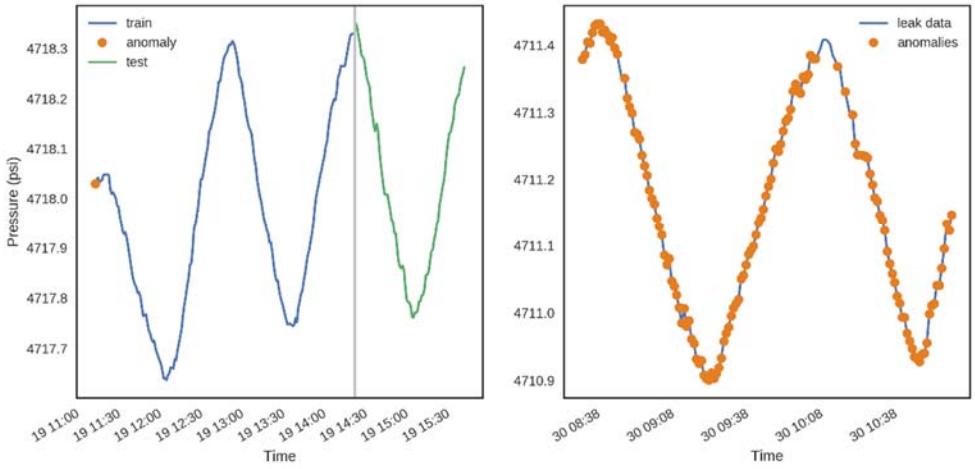

Figure 10. Anomaly detection on pressure data using IsolationForest: (a) results on trainingand testing data using the base case data (Jan 19, 2015, 11:00–15:30), where the vertical gray line indicates the separation of training and testing periods; (b) results on controlled release data (Jan 30, 2015, 8:30–11:30). Anomalies are labeled with filled circles.

3.3 Implementation of Complex Event Processing Engine

We automated AI/ML algorithms using Apache Kafka, which includes tools for managing data schema, connectors, and servers. The ecosystem described in Figure 11 comprises the CEP engine behind DIAL-GCS2.0.

Kafka is designed around four basic concepts:

- *Broker,* a broker orchestrates message flow between different entities involved
- *Topic,* a Kafka topic provides a way of organizing messages, which in turn serves as intermediate data containers for records to be transmitted between applications/systems. The topic data schema is defined by the user for different sensor types. Internally, each topic is organized in a number of partitions for faster information retrieval and data redundancy.
- *Producer,* a Kafka producer pushes content to a topic
- *Consumer*, a Kafka consumer pulls content from a topic

To use Kafka, the user is responsible for defining producer(s) and consumer(s). Kafka provides application programming interfaces (API) for developers to create customized producers and consumers. This where the AI/ML anomaly detection algorithms reside. In addition, the Kafka connectors allow configuration of sources/sinks that connect Kafka topics to known applications or data systems via standard interfaces such as JDBC (relational databases), HDF (Hadoop and Hive), and Amazon S3. Custom connectors are used to link the CEP to the data layer and knowledge discovery layer, automating the information exchange between CEP and those layers.

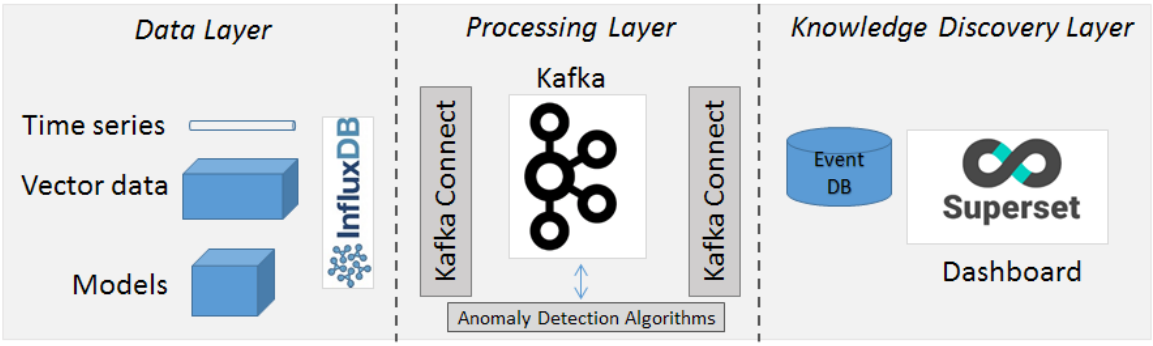

Figure 11. System design of IMS includes a data layer, a processing layer, and a knowledge discovery layer. Complex event processing engine is located in the processing layer. All layers are loosely coupled through Apache Kafka connectors and web services. InfluxDB is used as a temporary datastream store, Confluent Kafka is used for event processing, and Apache Superset is used for data visualization.

3.4 Task Summary

Under Task 3 of the project, a number of anomaly detection algorithms were implemented to deal with different types of CCS data and anomalies. All algorithms were designed to work with data streams. The algorithms were integrated into a Web-based system consisting both the backend and frontend. Thus, all original objectives in the project SOPO were either met or surpassed.

4. COUPLED MODELING AND ML-BASED SURROGATE MODEL DEVELOPMENT

Task 4 was designed to support the online simulation capability of the IMS. Specifically, Task 4 consisted of the following subtasks,

- Subtask 4.1 —Coupled modeling
- Subtask 4.2—Data assimilation
- Subtask 4.3—Model reduction and metamodeling

The main purpose of Subtask 4.1 was to develop coupled modeling capabilities for simulating commonly seen coupled GCS processes such as flow, transport, and geomechanics. Uncertainties are ubiquitous in GCS applications. Therefore, it is important to continuously improve the computational engine using newly acquired information in real time. This is supported by the data assimilation module. Finally, Web-based applications require that efficient reduced-order or surrogate models be developed to support real-time decision support, which was the main objective of subtask 4.3. When integrated together, the three subtasks were envisioned to enable online decision making using process-based surrogate models.

4.1 The need for coupled modeling in CCS

Like many other geological repositories, GCS operations require a thorough characterization of the target site and design of monitoring network for site permit. The main purpose of GCS monitoring network is to reduce the potential leakage risks and maximize the protection of public safety. During GCS operations, the reservoir pressure can significantly increase, which may drive the fluid migration from storage formations. Thus, it is important to use a reliable predictive model to simulate reservoir response and identify potential risks. Under Task 4, the team developed a number of coupled modeling capabilities to support the IMS. The capabilities generally fall under

- CCS monitoring network optimization
- Carbon reservoir modeling using deep learning techniques (DL)

4.2 Coupled modeling results

4.2.1 Monitoring network design

Under this subtask, the team considered two types of pressure-based optimization problems, namely, optimization of monitoring under geologic uncertainty and optimization of the location of brine extraction wells.

Leakage from geologic faults and abandoned wells represents one of the major risks to commercial-scale GCS projects. Current CCS regulations and best practice guidance recommend that operators develop risk-informed MVA plans to protect public safety and reduce property and environmental damage. Deep subsurface pressure monitoring is regarded as one of the most cost-effective technologies for early leakage detection in CCS projects. In practice, the number of deep-pressure monitoring wells that an operator can drill often remains limited because of the high costs associated with drilling, instrumentation, and operations. The former two costs belong to capital expenses (CAPEX), while the latter cost is considered operation expenses (OPEX). Thus, optimal design of the pressure monitoring network is essential to minimize monitoring and liability costs and gain public support. In this project, we developed a general, binary integer programming approach to solve an optimal monitoring well network design problem under multiple constraints. Specifically, our approach enables CCS operators to design a cost-optimal monitoring network that can cover all potentially leak locations (in a worst-case-scenario sense) while satisfying a prescribed carbon dioxide storage performance criterion (e.g., 99% storage permanence) and considering geological uncertainty. It is worth mentioning that in general such an optimization problem falls into the category of combinatorial problems that are hard to solve.

A common practice in previous studies is to use risks as surrogate of costs. However, most of the decision space is supported by monetary values than probability values. Instead of using cost surrogates as has been done in many previous studies, our formulation allows the user to directly assess total costs in terms of monitoring cost and potential economic losses associated with brine and CO2 leakage.

User inputs to the binary integer programming problem include costs related to brine and CO2 leakage, the maximum number of wells to be placed, the storage performance criterion, the CAPEX and OPEX related to monitoring wells, and pressure-based detection threshold. A synthetic numerical example was developed using CMG-GEM. The optimization problem was solved on a parallel computing cluster available at Texas Advanced Computing Center. Our numerical examples demonstrated that a cost-optimal monitoring network may save millions of dollars in total costs, including well construction and leakage costs. Factors exerting the most influence on cost-optimal monitoring network design are unit leakage damage costs, pressure threshold for leakage detection, and geological uncertainty. Results were published in (Jeong et al., 2018).

In a separate effort, the team developed an ensemble-based method for identifying optimal well locations for brine extraction. An improved ensemble-based stochastic gradient method was developed. Ensemble-based stochastic gradient methods, such as the ensemble optimization method (EnOpt), the simplex gradient method (SG), and the stochastic simplex approximate gradient method (StoSAG), approximate the gradient of an objective function using an ensemble of perturbed control vectors. These methods are increasingly used in solving reservoir optimization problems because they are not only easy to parallelize and couple with any simulator, but also computationally more efficient than the conventional finite-difference method for gradient calculations. In this work, we showed that EnOpt may fail to achieve sufficient improvement of

the objective function when the differences between the objective function values of perturbed control variables and their ensemble mean are large. On the basis of the comparison of EnOpt and SG, we proposed a hybrid gradient of EnOpt and SG to save the computational cost of SG. As a use case, we considered pressure management in carbon storage reservoirs, for which brine extraction wells need to be optimally placed to reduce reservoir pressure buildup while maximizing the net present value. Results show that our improved schemes reduced computational cost significantly (Jeong et al., 2020).

4.3 Data assimilation

The team adopted an ensemble smoother algorithm for assimilating pressure data. Two ensemble-based data assimilation algorithms can be identified, the smoother and filter. The main difference is that the smoother incorporates all observations to calibrate parameters. It has been shown iterative smoother may achieve good performance while avoiding the restarting problem (i.e., the simulation needs to be run from the beginning time every time after an assimilation step is done). Figure 12 illustrates the workflow used to assimilate pressure data. We designed a 3D reservoir model to demonstrate this workflow, in which a pulse injection pattern is applied at the injector and pressure observations from three monitoring wells are used. Results indicate that data assimilation reduced the uncertainty in model prediction (gray hairlines) compared to the "true" pressure history (solid red line) Figure 13. After data assimilation, the mean of the ensemble gave reasonable estimate of the "true" plume shape. The team also implemented a scheme to automate ensemble simulation.

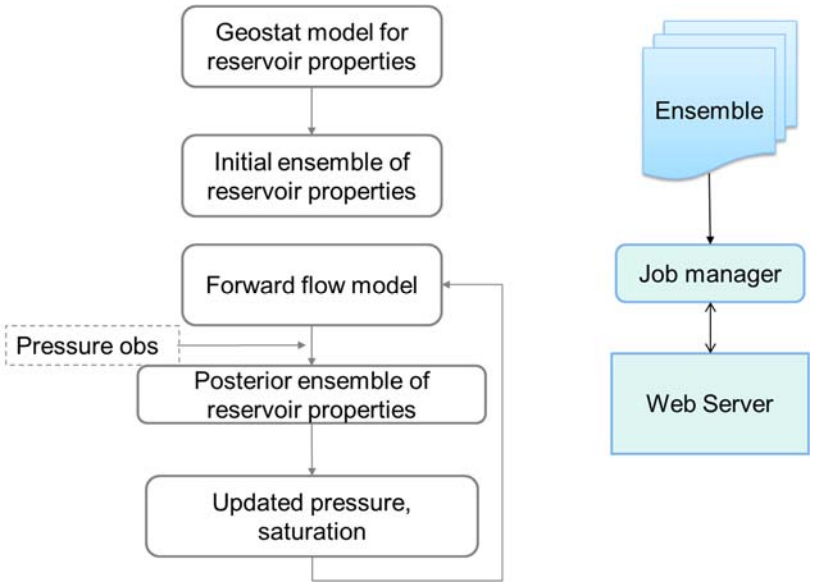

Figure 12. Workflow for using ensemble smoother to assimilate pressure data.

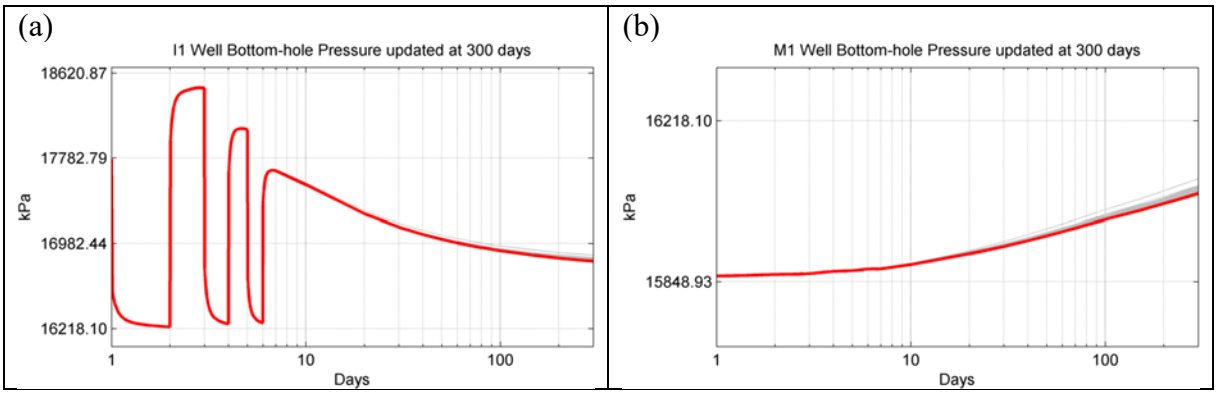

Figure 13. (a) bottom hole pressure at the injector, (b) bottom hole pressure at the monitoring well.

4.4 Surrogate modeling through deep learning

Surrogate modeling or reduced-order modeling refers to methods for obtaining proxies of the physically based models. Surrogate modeling is of significant interest to many subsurface modeling projects because of the needs for quantifying model uncertainty. Fully coupled physics based models can be time consuming to run. The team was one of the first in using deep learning techniques to develop surrogate models (Sun, 2018).

The framework we developed allows for fast forward and inverse modeling. Figure 14 shows an example of surrogate modeling, in which a deep convolutional neural network (CNN) was trained to learn the CO2 plume evolution in a heterogeneous reservoir. Results show that the CNN-based model was able to achieve very high accuracy in predicting the shape of the plume. The full set of results was reported in (Zhong et al., 2019).

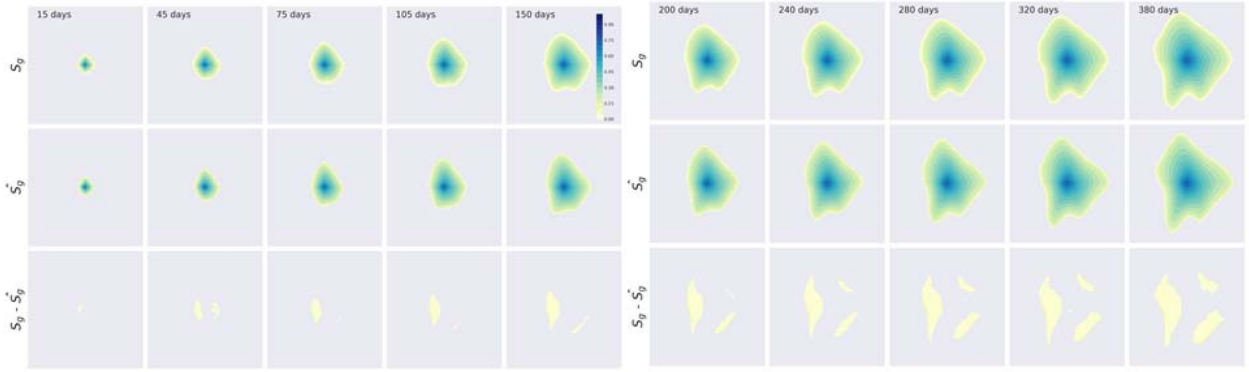

Figure 14. Comparison between simulated (1st row) and ML-predicted (2nd row) CO2 plumes for different output times. The residual is shown in the 3rd row.

In reality, reservoir forward modeling is inseparable from history matching and inverse modeling. A longstanding research question is how to combine data assimilation and online prediction. While the traditional data assimilation offers one approach for doing this, it becomes less feasible when the ensemble model runs require signficant computational time. Deep learning provides a viable alternative. In (Zhong et al., 2020), a multiphysics forward and inverse modeling framework was proposed for the first time, which allows the traditionally separated rock petrophysics, 4D seismics, and reservoir modeling steps to be united under one framework. This work, detailed in (Zhong et al., 2020), is visionary and paves the way for operational assimilation

of multi-source, multi-domain data. Figure 16 gives an example of using the proposed multiphysics surrogate model to perform forward and inverse modeling between seismic acoustic impedance (the product of seismic velocity and density) and reservoir CO2 plume saturation.

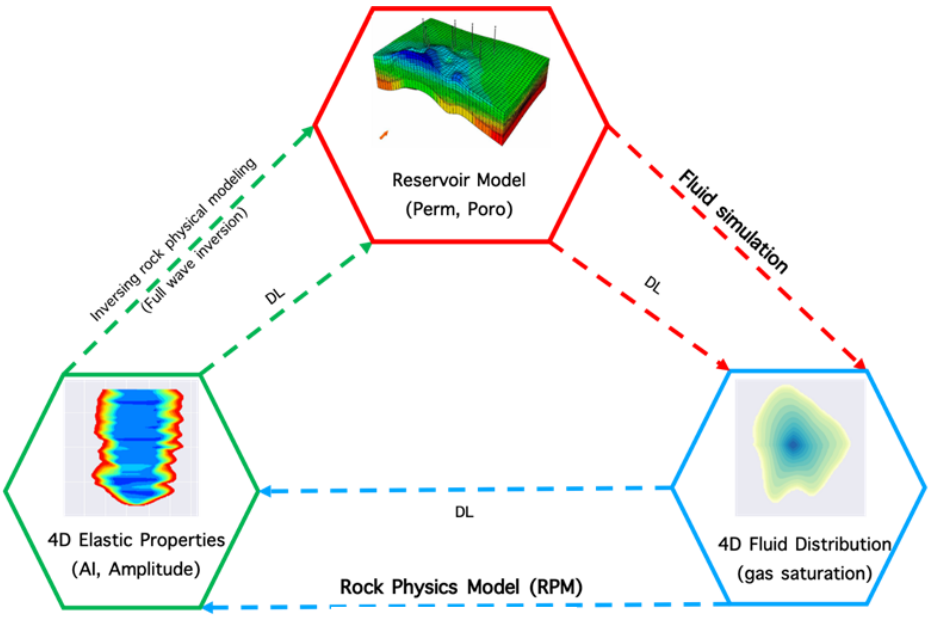

Figure 15. A machine learning enabled multiphysics surrogate modeling for connecting rock petrophysics, 4D seismics, and multiphase reservoir flow modeling.

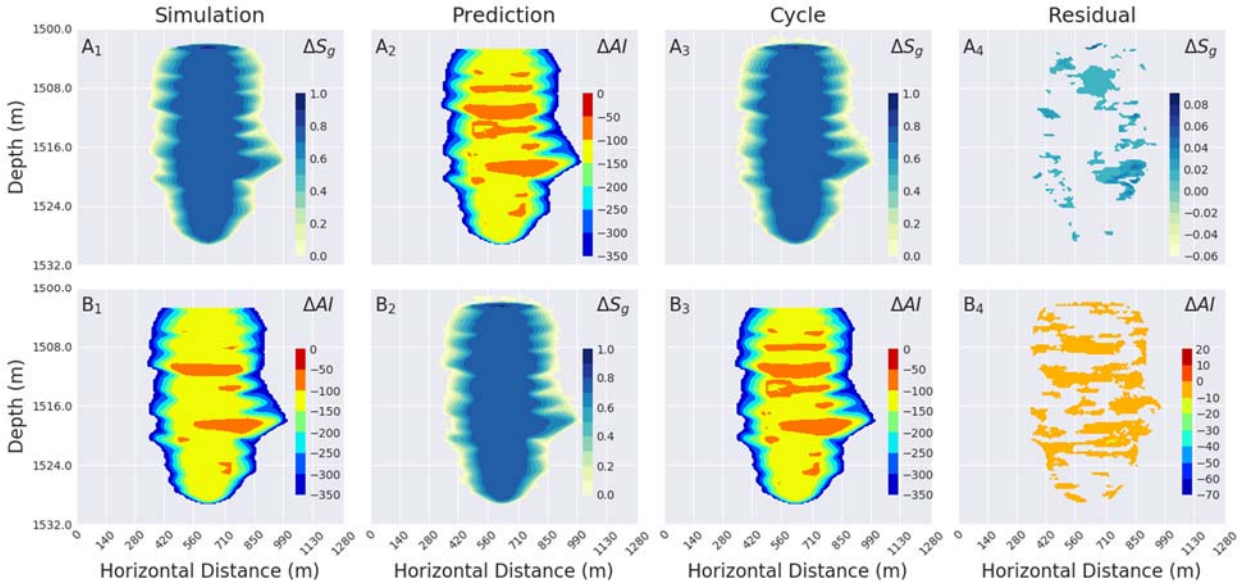

Figure 16. Demonstration of the multiphysics surrogate model on a 2D reservoir model (vertical slice), where a CycleGAN was applied to map from acoustic impedance in seismic domain to CO2 saturation in porous flow domain.

4.5 Optimal reservoir management

In addition to predictive online modeling, the DL-based surrogate model can also be wrapped in an optimization framework. In (Sun, 2020), I adopted elements of the recently developed Deep Q learning (DQL) algorithms to tackle a model-based sequential optimization problem, which can be found in many applied energy applications such as microgrid energy demand management, and reservoir production planning. Specifically, the sequential optimization problem is formulated as a Markov Decision Process (MDP), which involves an agent interacting with a dynamic environment through a sequence of actions, observations, and rewards. The agent is not told what to do, but instead seeks to maximize the cumulative future reward by discovering an optimal policy through interactions. This type of problem falls under reinforcement learning, which is a general class of algorithms in machine learning that aims to help an agent learn how to behave in a dynamic environment, where the only feedback consists of a scalar reward.

The main contributions of (Sun, 2020) include (a) developed a DQL-based framework for optimal multiperiod planning involving high-dimensional, multistate geosystem models and (b) formulated a deep multitask learning (DeepMTL) approach to approximating multistate transition functions under variable forcing conditions, thus reducing total computational costs. Figure 17 shows a high-level schematic plot for training the DQL model. As a case study, I applied the DQL framework to multiperiod carbon storage planning, which is a geosystem-based measure for greenhouse gas emission reduction that has received significant renewed interest in the U.S. because of the recently passed carbon tax incentives, Section 45Q (Internal Revenue Service, 2018). In a synthetic example, I showed how the DQL can be applied to identify the optimal injection schedule by taking various operational constraints into consideration.

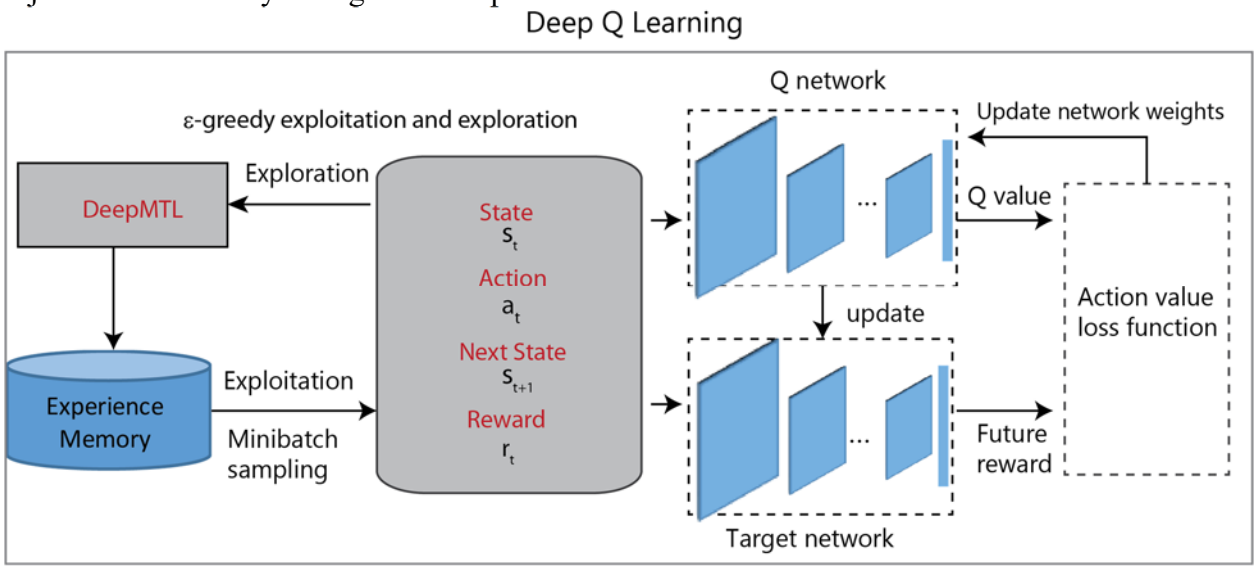

Figure 17. A deep reinforcement learning framework for optimizing carbon reservoir injection schedule.

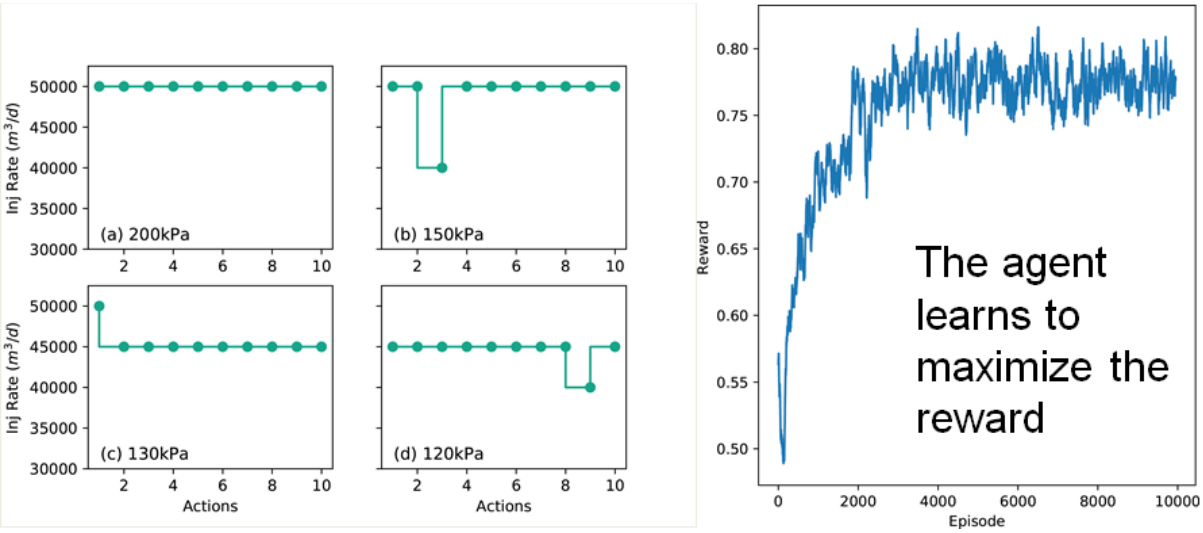

Figure 18. Optimal injection schedule recommended by the DQL framework for a demonstration problem.

4.6 Task Summary

Under Task 4 the team worked on enriching the DIAL system capability using latest AI/ML technologies. The new capabilities are complementary to the project's overarching goal, namely, improving the current MVA capability and enabling the access, assimilation, and analysis of heterogeneous data and information sources in a timely manner. The team successfully demonstrated the proposed workflows for surrogate modeling, data assimilation, and coupled modeling via meaningful use cases in the field of GCS.

5. INTEGRATION AND WEB DEVELOPMENT

Integration and web development were continuously performed throughout this project. The data analytics dashboard was already described in the previous sections. Under this research, web applications or modules were often developed as part of the research application. For example, Figure 19 shows a web module that uses a deep learning model as backend to predict reservoir CO2 plume evolution. Figure 20 shows a web-based module that uses a CO2 flow surrogate model as backend to predict leakage risk during the design stage (Sun et al., 2018).

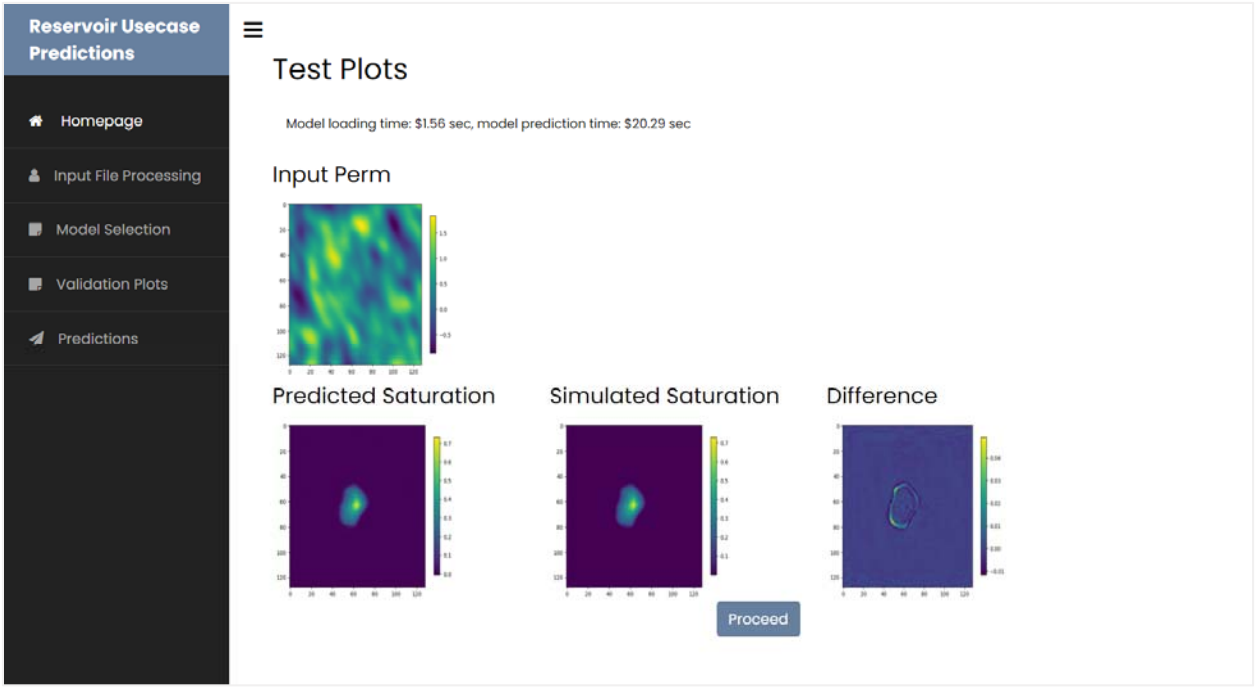

Figure 19. A web-based module for CO2 plume saturation prediction.

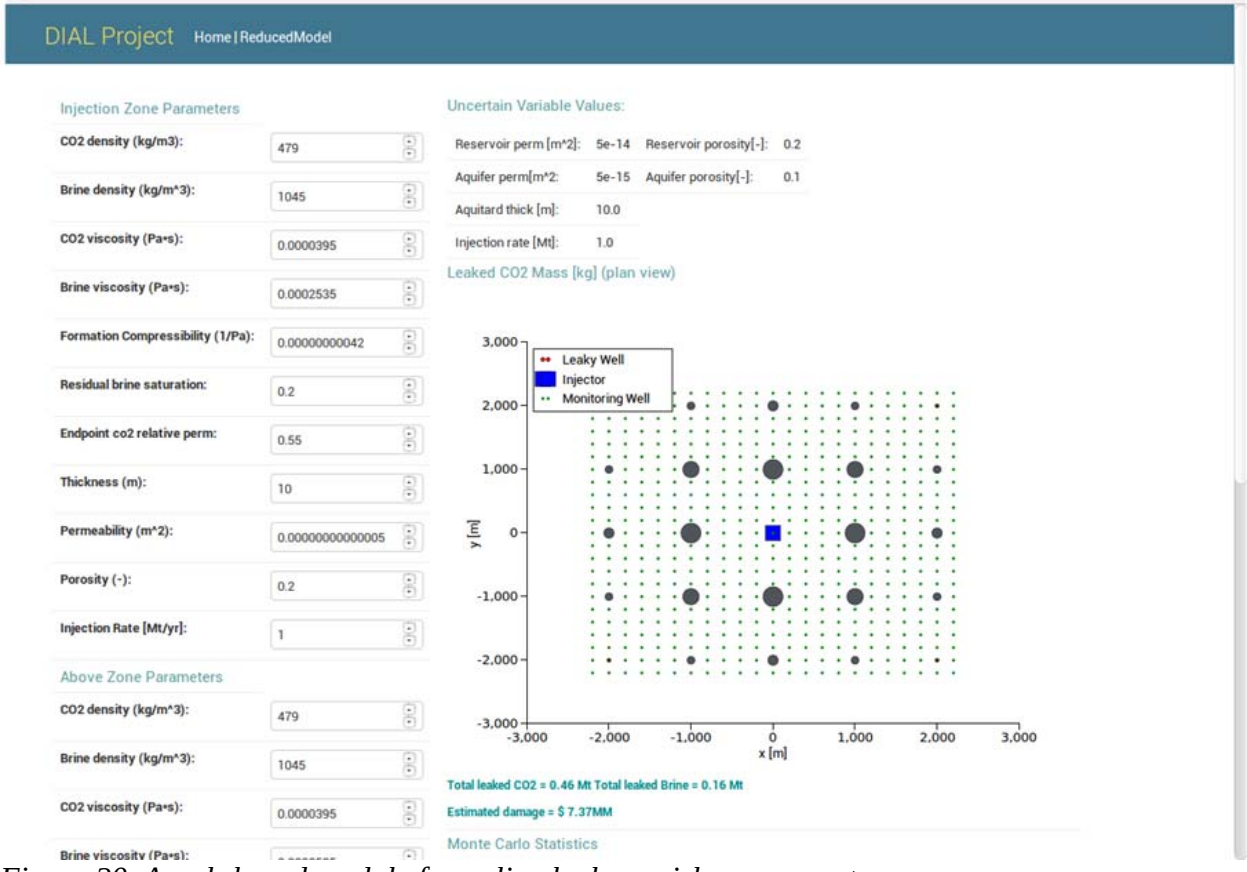

Figure 20. A web-based module for online leakage risk assessment.

6. SUMMARY

In summary, a multi-faceted research was carried out under this project that all contributes to the development of the first-of-its-kind CCS intelligent monitoring system. As part of the project, the team had developed a suite of tools for automating common MVA tasks in CCS projects. The tools combined state-of-the-art machine learning technologies with domain knowledge, and were demonstrated over real and synthetic use cases and data. Throughout the project, the team was actively involved in dissemination of the research results, through peer-reviewed publications and meeting presentations. Our results suggest that adapting off-the-shelf open-source platforms for CCS monitoring projects is cost effective, but domain knowledge is critical for testing and validating the products. The experience gained from this project allowed us to contribute directly the recent SMART initiative led by NETL.

Some of the lessons learned from this project are

- Combining AI/ML with domain knowledge may significantly improve efficacy of GCS management and risk mitigation
- Time series anomaly detection can be automated effectively with the current open-source technologies, but require domain knowledge for software product design and testing
- High-dimensional datasets (e.g., distributed acoustic sensing) present more challenges to data storage and anomaly detection
- All anomalies are different and no single method works for all cases
- In lieu of developing a tightly coupled IMS system, a loosely coupled IMS is probably the best way to go for maximizing system reusability. The Superset data dashboard we selected clearly demonstrated this aspect.

REFERENCES

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3): 1-58.

Jeong, H., Sun, A.Y., Jeon, J., Min, B., Jeong, D., 2020. Efficient Ensemble-Based Stochastic Gradient Methods for Optimization Under Geological Uncertainty. Frontiers in Earth Science, 8: 108.

Jeong, H., Sun, A.Y., Zhang, X., 2018. Cost-optimal design of pressure-based monitoring networks for carbon sequestration projects, with consideration of geological uncertainty. International Journal of Greenhouse Gas Control, 71: 278-292.

Jordan, P.D., Benson, S.M., 2009. Well blowout rates and consequences in California Oil and Gas District 4 from 1991 to 2005: implications for geological storage of carbon dioxide. Environmental Geology, 57(5): 1103-1123.

Pedregosa, F. et al., 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12: 2825-2830.

Sun, A.Y., 2018. Discovering state-parameter mappings in subsurface models using generative adversarial networks. Geophysical Research Letters, 45(20): 11,137-11,146.

Sun, A.Y., 2020. Optimal carbon storage reservoir management through deep reinforcement learning. Applied Energy, 278: 115660.

Sun, A.Y., Jeong, H., González-Nicolás, A., Templeton, T.C., 2018. Metamodeling-based approach for risk assessment and cost estimation: Application to geological carbon sequestration planning. Computers & geosciences, 113: 70-80.

Sun, A.Y., Lu, J., Freifeld, B.M., Hovorka, S.D., Islam, A., 2016. Using pulse testing for leakage detection in carbon storage reservoirs: A field demonstration. International Journal of Greenhouse Gas Control, 46: 215-227.

Sun, A.Y., Zhong, Z., Jeong, H., Yang, Q., 2019. Building complex event processing capability for intelligent environmental monitoring. Environmental modelling & software, 116: 1-6.

Zhong, Z., Sun, A.Y., Jeong, H., 2019. Predicting co2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network. Water Resources Research, 55(7): 5830-5851.

Zhong, Z., Sun, A.Y., Wu, X., 2020. Inversion of time-lapse seismic reservoir monitoring data using cycleGAN: a deep learning-based approach for estimating dynamic reservoir property changes. Journal of Geophysical Research: Solid Earth, 125(3): e2019JB018408.

APPENDIX

List of peer-reviewed publications that resulted from the support or partial support of this project (§ postdoc supported under the project):

1. §Zhong, Z., Sun, A. Y., Ren, B., & Wang, Y. (2021). A Deep-Learning-Based Approach for Reservoir Production Forecast under Uncertainty. SPE Journal, 1-27.
2. Sun, A. Y. (2020). Optimal carbon storage reservoir management through deep reinforcement learning. Applied Energy, 278, 115660.
3. §Zhong, Z., Sun, A. Y., Wang, Y., & Ren, B. (2020). Predicting field production rates for waterflooding using a machine learning-based proxy model. Journal of Petroleum Science and Engineering, 194, 107574.
4. §Zhong, Z., Sun, A. Y., & Wu, X. (2020). Inversion of Time-Lapse Seismic Reservoir Monitoring Data Using CycleGAN: A Deep Learning-Based Approach for Estimating Dynamic Reservoir Property Changes. Journal of Geophysical Research: Solid Earth, 125(3), e2019JB018408.
5. §Jeong, H., Sun, A. Y., Jeon, J., Min, B., & Jeong, D. (2020). Efficient Ensemble-Based Stochastic Gradient Methods for Optimization Under Geological Uncertainty. Frontiers in Earth Science, 8.
6. Sun, A.Y., §Zhong, Z., §Jeong, H., Yang, Q., 2019. Building complex event processing capability for intelligent environmental monitoring. Environmental Modelling & Software, 116: 1-6.
7. §Zhong, Z., Sun, A.Y., §Jeong, H., 2019, Predicting CO2 Plume Migration in Heterogeneous Formations Using Conditional Deep Convolutional Generative Adversarial Network. Water Resources Research, 55(7), 5830-5851.
8. §Zhong, Z., Sun, A. Y., Yang, Q., and Ouyang, Q., 2019, A deep learning approach to anomaly detection in geological carbon sequestration sites using pressure measurements. Journal of Hydrology, 573, 885-894.
9. Sun, A. Y., 2018, Discovering state-parameter mappings in subsurface models using generative adversarial networks, Geophysical Research Letters, 45(20): 11,137-11,146.
10. Sun, A. Y., §Jeong, H., §Gonzalez, A., and §Templeton, T., 2018, Metamodeling-based approach for risk assessment and cost estimation: application to geological carbon sequestration, Computers and Geosciences, v. 113, p. 70-80
11. §Jeong, H., Sun, A. Y., Lee, J., and Min, B., 2018, A learning-based, data-driven forecast approach for predicting future reservoir performance. Advances in Water Resources, 118, 95-109.
12. §Jeong, H., Sun, A. Y., and Zhang, X., 2018, Cost-optimal design of pressure-based monitoring networks for carbon sequestration projects, with consideration of geological uncertainty, International Journal of Greenhouse Gas Control, v. 71, p. 278-292.

List of main presentations

- Sun, A. Y. (2020, December). A Deep Reinforcement Learning Approach for Managing Carbon Storage Reservoirs (invited). In AGU Fall Meeting 2020. AGU.

- Sun, A. Y., Zhong, Z., & Jeong, H. (2019, December). Adversarial Learning for Subsurface Flow and Transport Modeling (invited). In AGU Fall Meeting Abstracts (Vol. 2019, pp. H31K-1854).
- Sun, A. Y., A geospatial intelligence use case, presented at the NSF Workshop on Advanced Cyberinfrastructure (invited), Chicago, July, 2018.
- Sun, A. Y. (2017, December). Development of anomaly detection models for deep subsurface monitoring. In AGU Fall Meeting Abstracts (Vol. 2017, pp. NG33A-0183).
- Sun, A. Y. (2017, October), Development of intelligent monitoring system for geologic carbon sequestration, presented at the Fall Seminar series of Texas A&M Kingsville.
- Sun, A. Y., Kelleher, C., & Rasmussen, R. (2016, December). Transforming Hydrologic Prediction and Decision Making: Intelligent Decision Making V. In 2016 AGU Fall Meeting. AGU.