

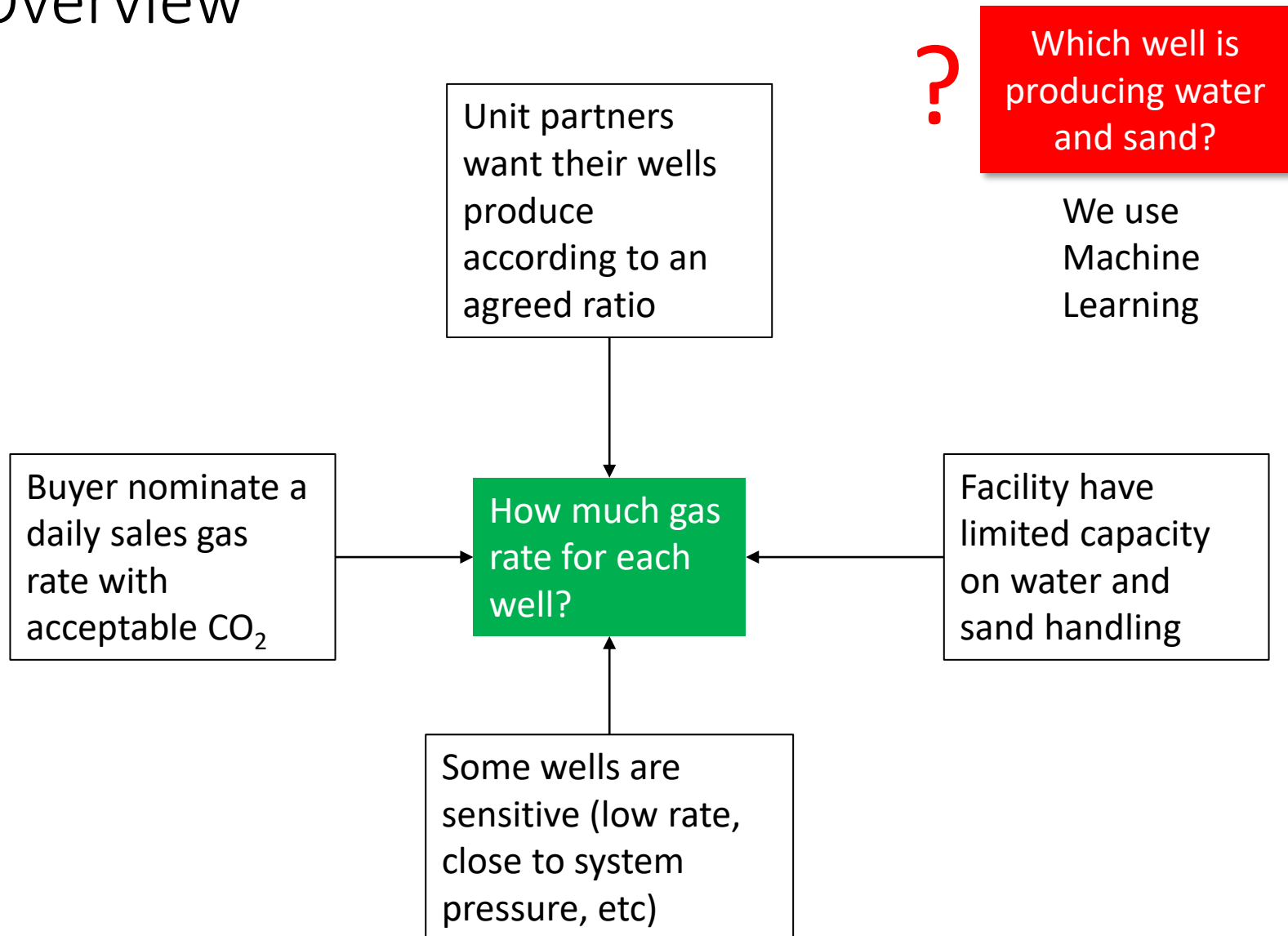
# Machine Learning-Based Gas Fields Management: Water and Sand Analytics

Akmal Aulia, PhD

Reservoir Engineer

January 27<sup>th</sup>, 2022

# Overview

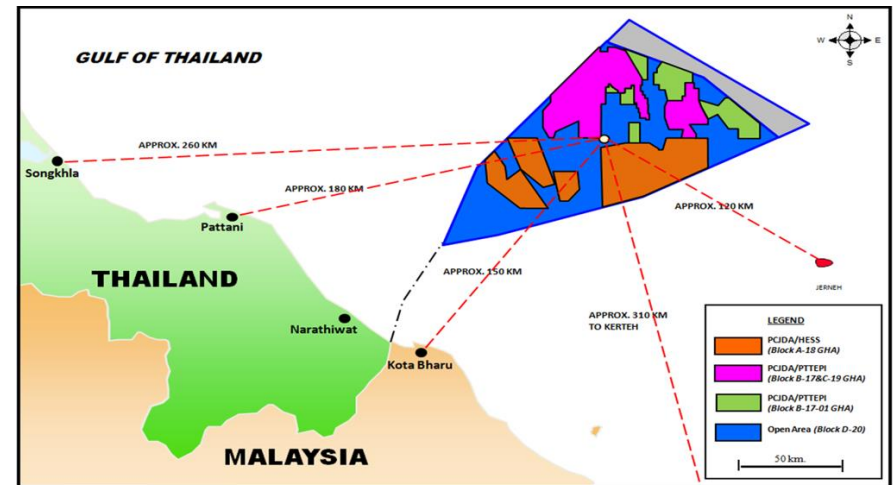


# Outline

- A brief overview of our operations
- The objectives of our operations
- Formulating the objectives as an optimization problem
- Major constraints:
  - Well water rate
  - Well sand rate
- Identifying water producing wells
- Identifying sand producing wells

# A Brief Overview of Our Operations (1)

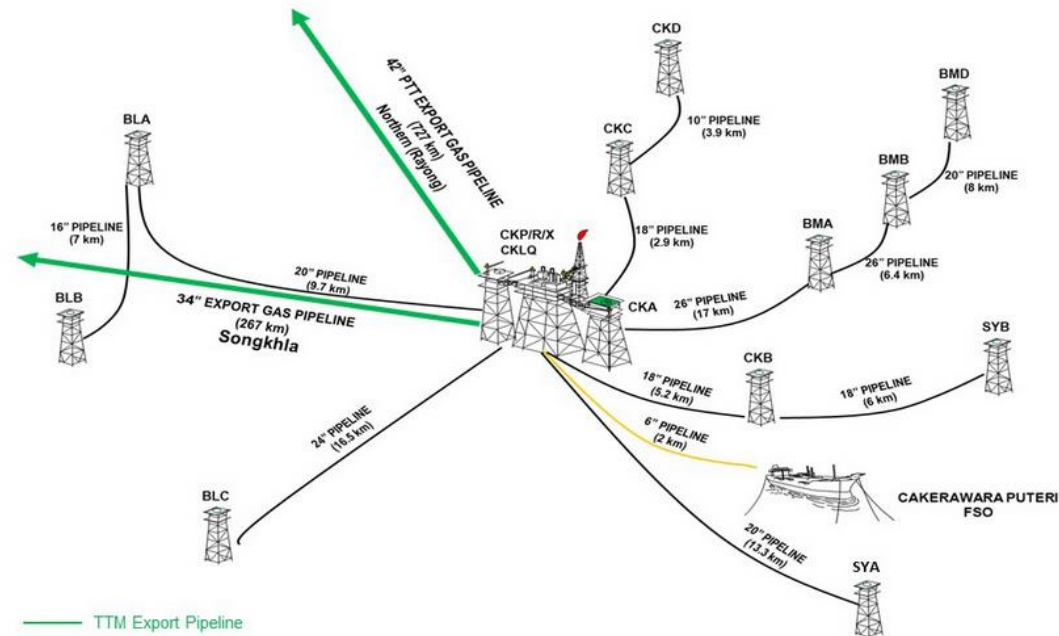
- Complex Geology:
  - Multiple stacked tidal to shallow marine sands
  - Facies ranging from:
    - Low quality bioturbated sand
    - Intermediate quality heterolithic sands
    - High quality massive channel sand



Source: mtja.org

# A Brief Overview of Our Operations (2)

- 12 wellhead platforms
- Numerous development wells
- Single processing hub
- Limited metering

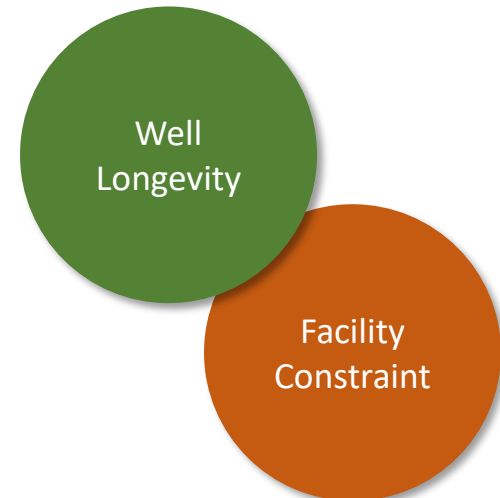
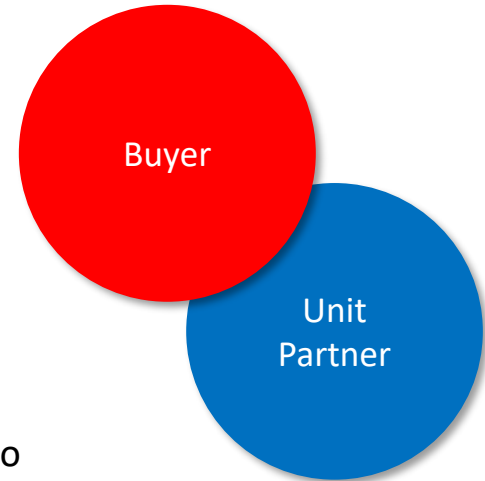


Source: mtja.org

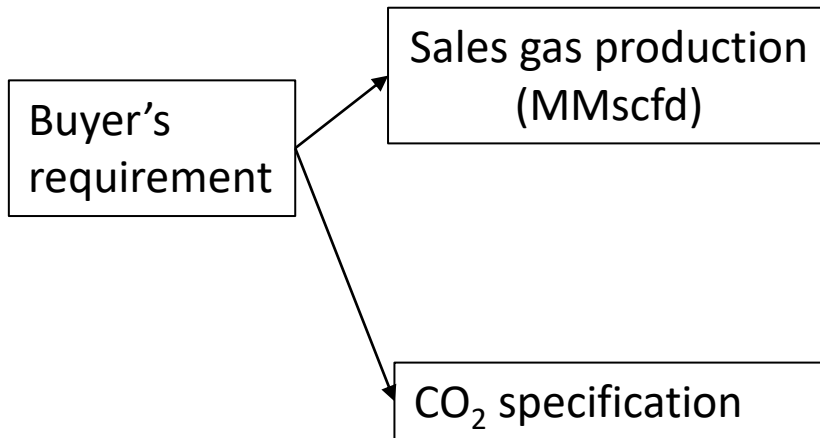
# The Objectives of Our Operations

- Objectives

- Meet buyer's demand (we have a dedicated buyer):
  - Daily sales gas nomination (fluctuating)
  - CO<sub>2</sub> specification
- Meet unit partners' demand:
  - Production meets agreed capital expenditure (CAPEX) ratio
- Ensure longevity of wells
  - Be conservative in changing choke size (especially for sensitive/cyclic wells)
- Meet facility constraint
  - Facility capacity in handling produced water
  - Facility capacity in handling produced sand



# Formulating The Objectives As An Optimization Problem (1)



Error = Demand - Base

$$\epsilon_S = |S_T(t) - S_B(t)| \leq \epsilon_T$$

$$x_T^o(t) = \frac{\sum_{i=1}^N G_i^o(t) x_i(t)}{\sum_{i=1}^N G_i^o(t)}$$

Weighted  
Average CO<sub>2</sub>

$$x_T^o(t) \leq x_{T,\max}$$

Facility constraints

Sales = f(Gross Rate, Feed CO<sub>2</sub>)

$$S_i(t) = G_i(t) (\alpha(y_T(t))x_i + \beta(y_T(t)))$$

Shrinkage factor  
(CO<sub>2</sub> removal)

Flare + Membranes



<http://dreamstime.com/royalty-free-stock-photo-flare-boom-nozzle-fire-offshore-oil-rig-image28254785>

# Formulating The Objectives As An Optimization Problem (2)

- Ensure profitability of unit partners

$$\epsilon_S = |S_T(t) - S_B(t)| \leq \epsilon_T$$

1<sup>st</sup> Unit Partner's  
requirement

$$\epsilon_1 = |u_1 S_B(t) - S_1|$$

2<sup>nd</sup> Unit Partner's  
requirement

$$\epsilon_2 = |u_2 S_B(t) - S_2|$$



# Formulating The Objectives As An Optimization Problem (3)

- Taking care of sensitive wells

Sensitive wells

$$\min(W_{G,i}) = \max(W_{G,i}) = W_{G,i}(t - 1) = W_{G,i}(t)$$

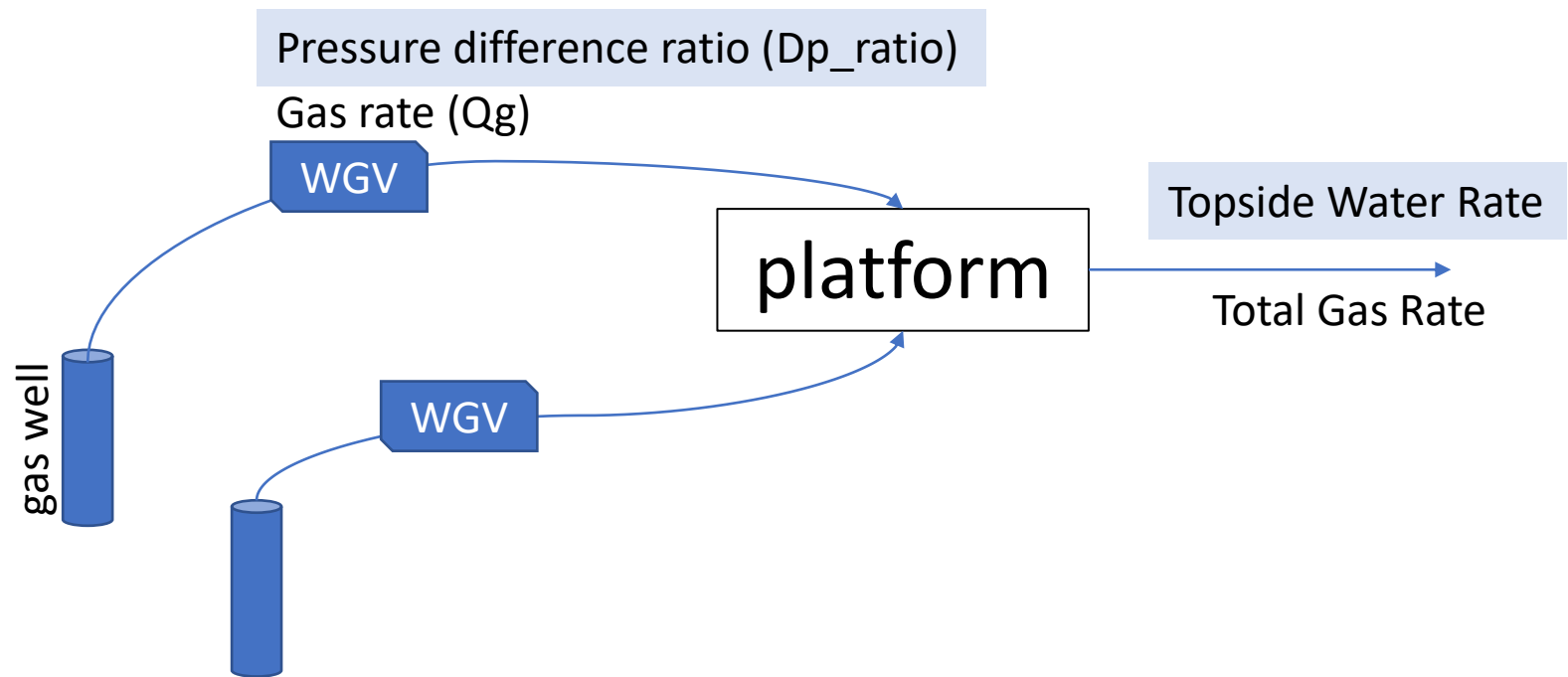
Non-sensitive wells

$$\min(W_{G,i}) \leq W_{G,i}(t) \leq \max(W_{G,i})$$

# Major Constraints

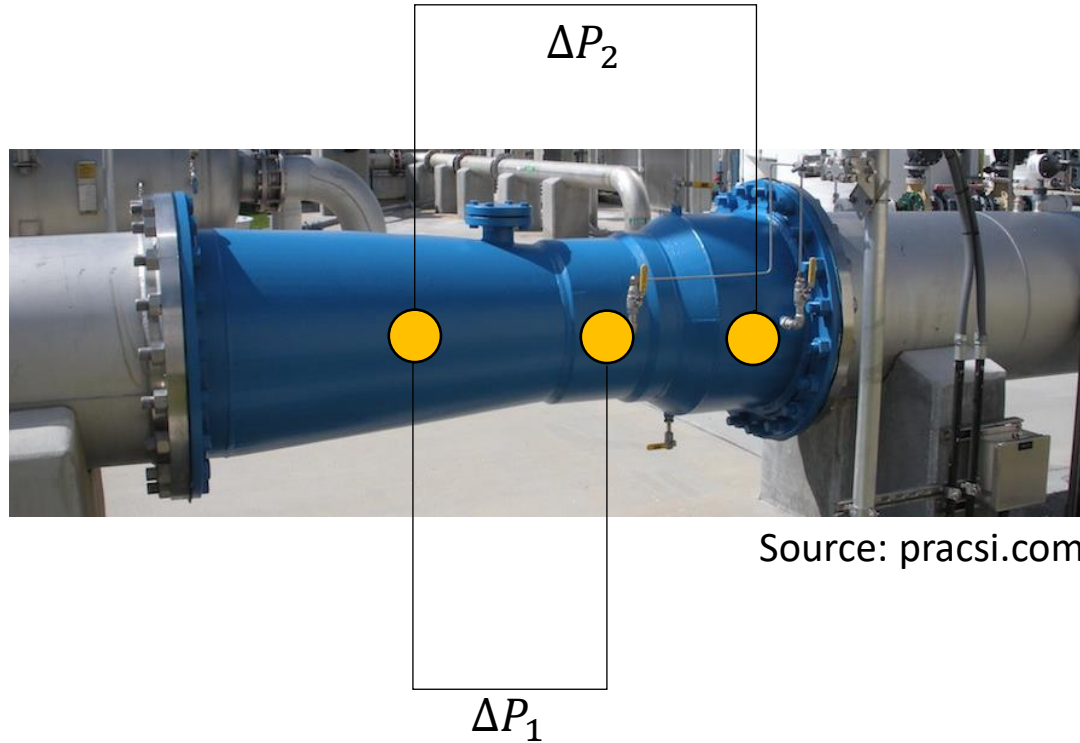
- Facility handling capacity
  - Water production
  - Sand production
- Issue: Limited information (measurements)
  - Water production rate for each well?
    - Available sensors: Wet Gas Venturimeter, Temperature
  - San production rate for each well?
    - Available sensors: Acoustic Sand Device

# Identifying Water Producing Wells



WGV: wet gas venturi meter

# Wet Gas Venturi Meter



$$DP \text{ Ratio} = \frac{\Delta P_2}{\Delta P_1}$$

# Converting DP ratio to Water Rate




$$Q_w = f(Q_g, CGR, DP \text{ Ratio}, \rho_g, \rho_w, \rho_c)$$

Assumption (throughout all wells):

- Constant CGR
- Constant fluid densities

Leeuw, R.C. (1997). Liquid correction of venturi meter readings in wet gas flow. North Sea Flow Measurement Workshop.

# Objectives

- Determine well-by-well water production rate using:
  - Pure AI approach 
  - Simple material balance + AI approach 
  - Time series approach 
- Develop AI apps that directly take input data from the database server.

# Pure AI Approach

- Random Forests for regression
  - Supervised learning
  - Ensembles of decision trees
  - Good for data with large input features and fewer samples
  - Shows ranking of input features

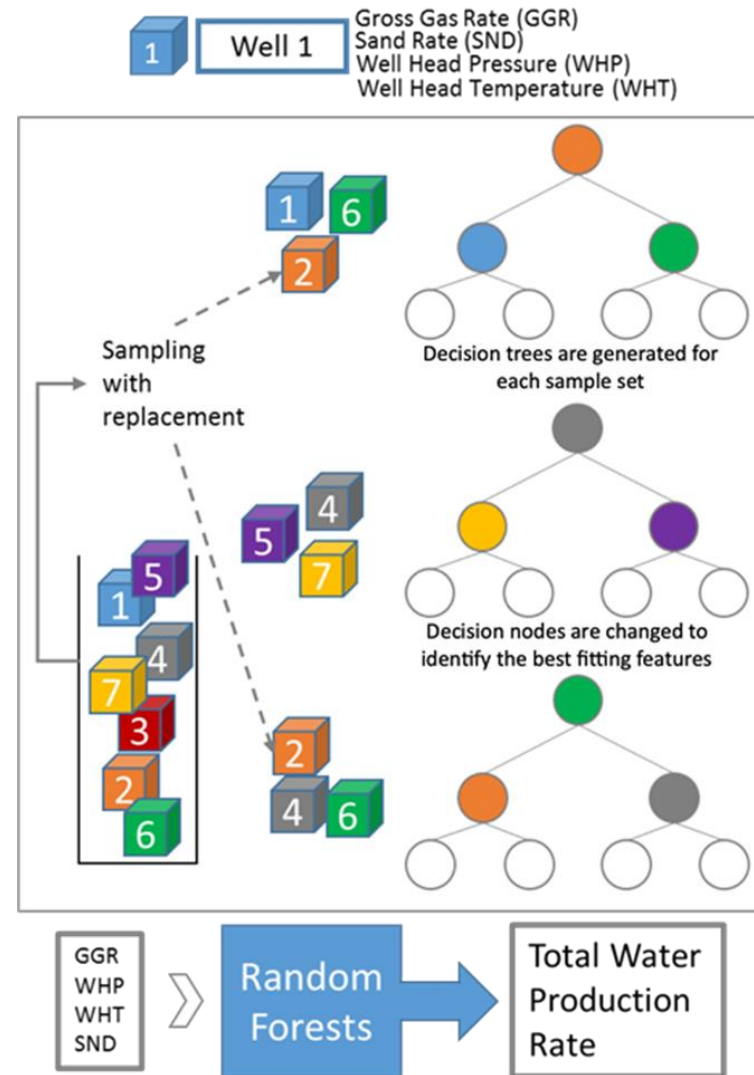
Well  
parameters

Random  
Forests

Topside  
Water Rate

Input

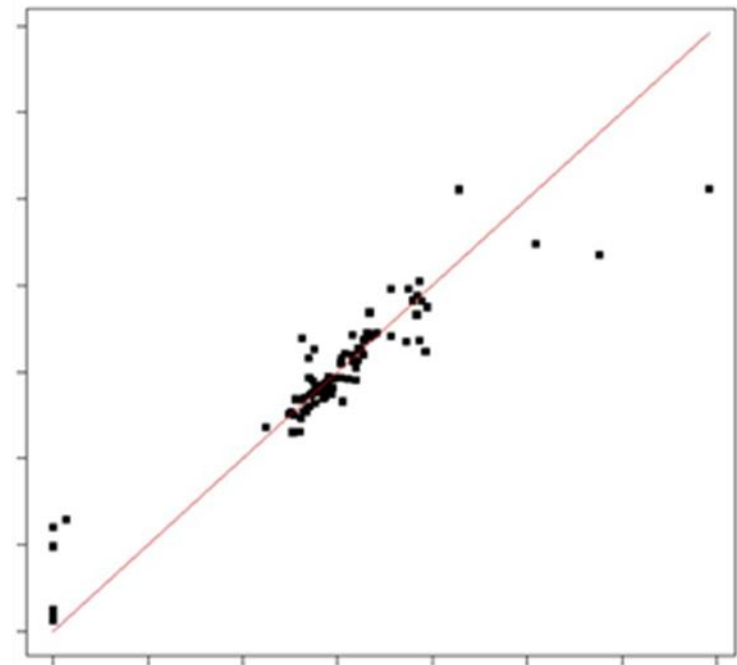
Output



# Random Forests Results

- Great predictive capability
- Shows ranking of input features, i.e. well parameters

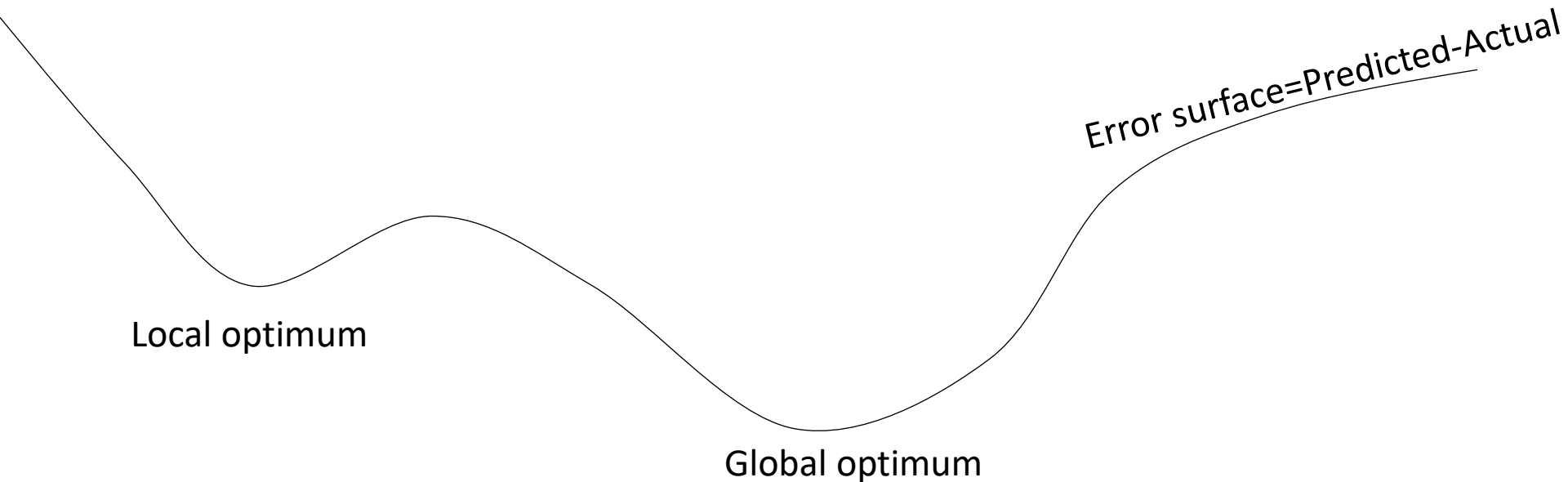
Well	Parameter	% Influence
15	WHT	8.9
9	GGR	8.2
15	WHP	7.1
89	WHP	6.6
16	WHP	6.4
71	GGR	6.3
11	WHP	6.0
9	WHT	5.8
71	WHT	5.6
37	WHP	5.6





# Another Approach: Simple Balance + AI

- Goal: Solve Water/Gas Ratio (WGR)
- Genetic Algorithm is known to solve for global optimum



## Description of Material Balance (1)

$$\mathcal{W}_k(t) = \omega_k G_k(t) \quad \longrightarrow \quad \mathcal{W}(t) = \sum_k \mathcal{W}_k(t)$$

Well by well water rate

Total Water rate

Where:

- $\mathcal{W}_k(t)$  is water production rate of  $k^{\text{th}}$  well at time  $t$ , measured in bbld (barrels per day)
- $\omega_k$  is WGR, in bbld/mmscfd
- $G_k(t)$  is gross gas production rate of  $k^{\text{th}}$  well at time  $t$ , in bbld.

## Description of Material Balance (2)

Taking the gross gas rate for all wells,

$$\mathbf{G} = \{G_1(t), G_2(t), G_3(t), \dots\},$$

the entire problem can be represented by:

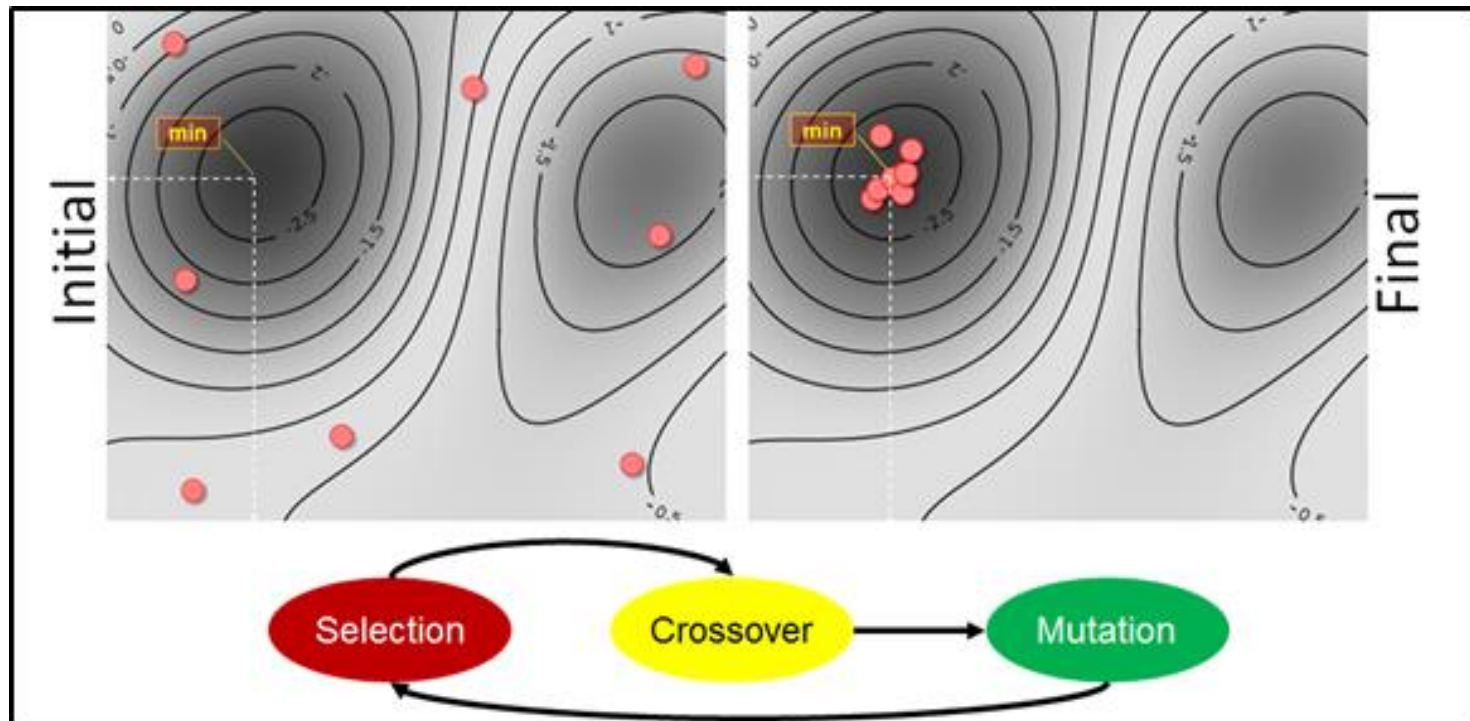
$$\mathbf{G} \cdot \omega_k = \mathcal{W}(t)$$

Well gross gas      WGR      Total Water rate

# Solving with Global Optimization Method

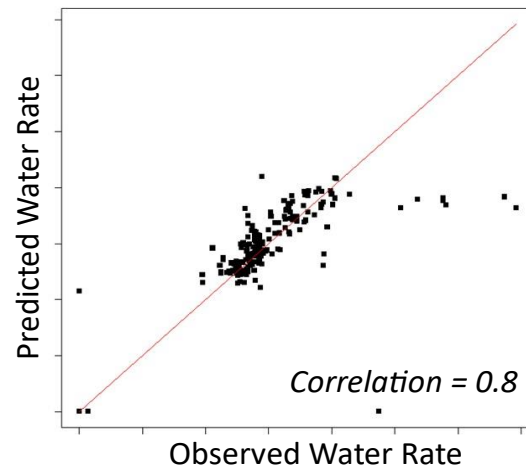
- Genetic Algorithm

$$\underset{\substack{\text{Well} \\ \text{gross} \\ \text{gas}}}{\mathbf{G}} \cdot \underset{\text{WGR}}{\omega_k} = \underset{\substack{\text{Total} \\ \text{Water} \\ \text{rate}}}{\mathcal{W}(t)}$$

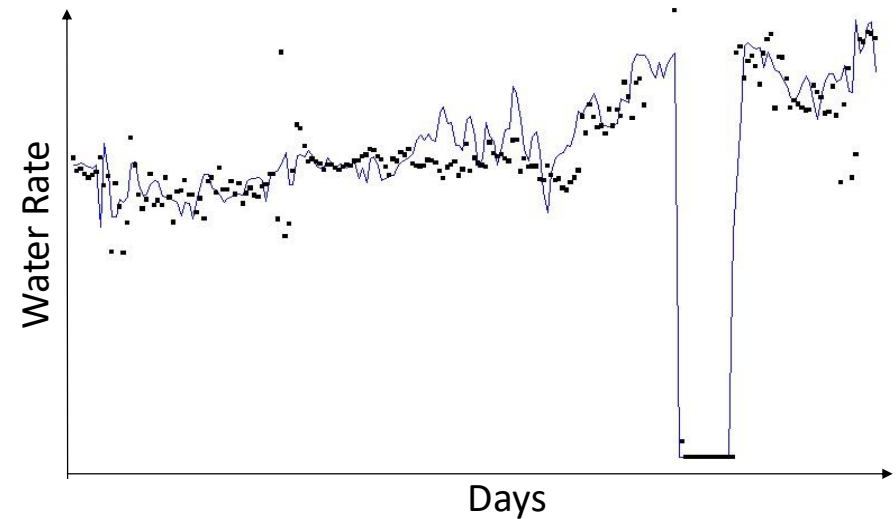


# GA-based WGR

Well	WGR
51	350
27	237
122	218
53	164
15	132
127	85
70	55
102	54
75	47
45	22



**Predicted (Line) vs Observed (Dot)**



# Water Analytics: Time Series Approach

Instead of modeling this with machine learning,

$$f(a_1(t), a_2(t), a_3(t), \dots, a_N(t)) \rightarrow y(t)$$

IN(t)

OUT(t)

We took another approach,

$$f(a_1(t), a_2(t), a_3(t), \dots, a_N(t), a_1(t-1), a_2(t-1), a_3(t-1), \dots, a_N(t-1), \dots, a_1(t-k), a_2(t-k), a_3(t-k), \dots, a_N(t-k)) \rightarrow y(t)$$

IN(t)

IN(t-1)

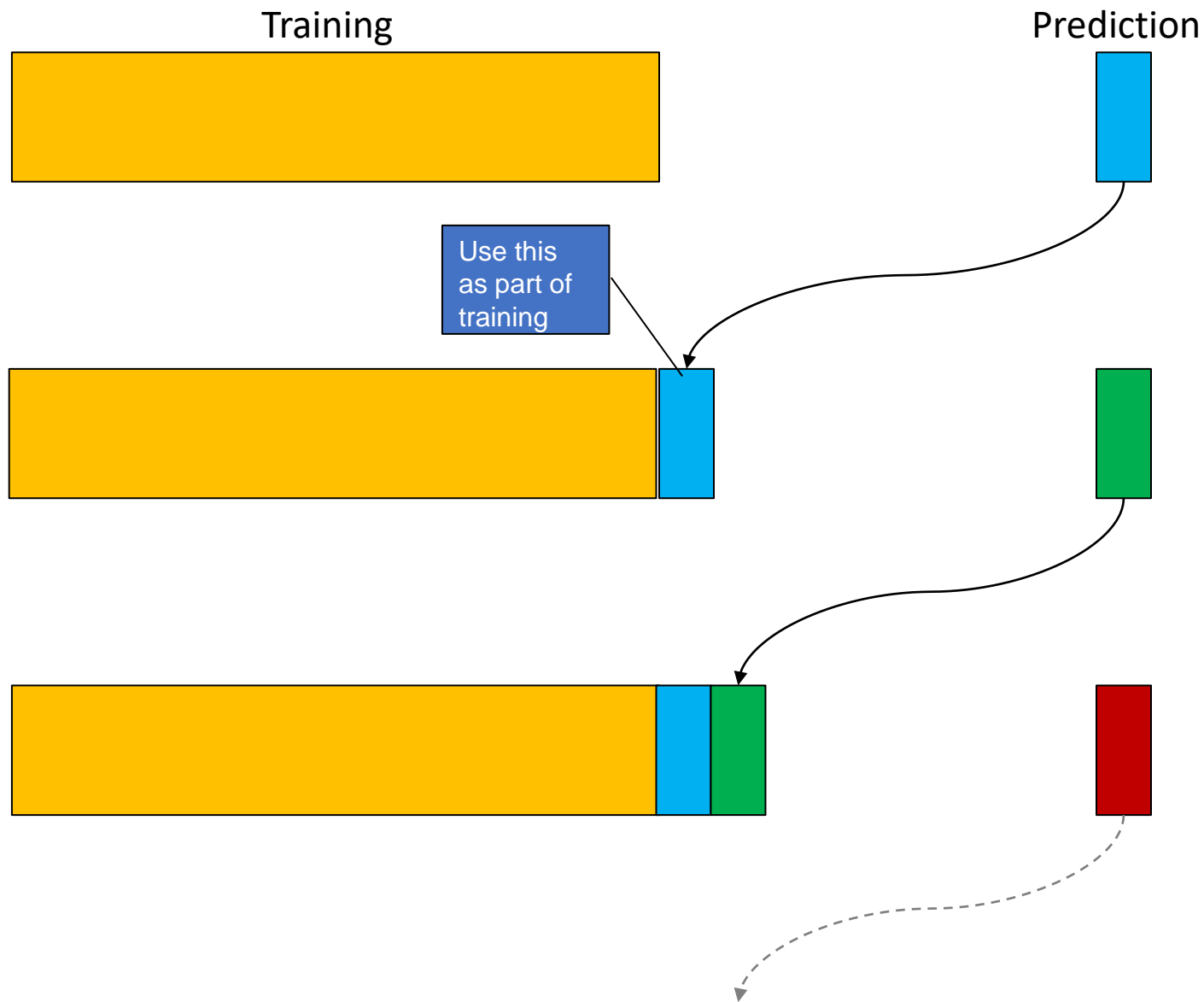
IN(t-k)

OUT(t)

Why do this?

Intuitive example: Due to the loss of pressure in the reservoir, the same choke opening today will not yield the gross gas rate of yesterday.

# Walk-In Validation Concept



# Experiments

- ML models: Random Forests, Gradient Boosting (XGBoost)

Type	Elements
Input	choke (t, t-1, ... )
	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	Random Forests

Type	Elements
Input	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	Random Forests

Type	Elements
Input	choke (t, t-1, ... )
	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	XGBoost

Type	Elements
Input	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	XGBoost

Type	Elements
Input	temp (t, t-1, ... )
	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	XGBoost

Type	Elements
Input	temp (t, t-1, ... )
	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	XGBoost



# Time Series with Walk-In: Results (1)

Type	Elements
Input	temp (t, t-1, ... )
	dp ratio (t, t-1, ... )
	water (t-1, t-2, ... )
Output	water (t)
Model	XGBoost

After numerous trials,

XGBoost with Temperature, DP Ratio, and Historical Topside Water seems to be promising.

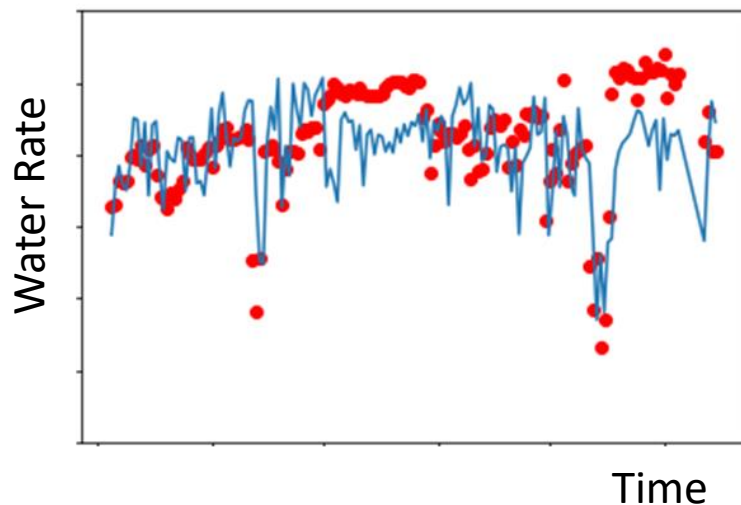
# Time Series with Walk-In: Results (2)

```
[1252]: mse = ((y_test - y_pr)**2).mean()
Rsqr = np.corrcoef(y_test,y_pr)

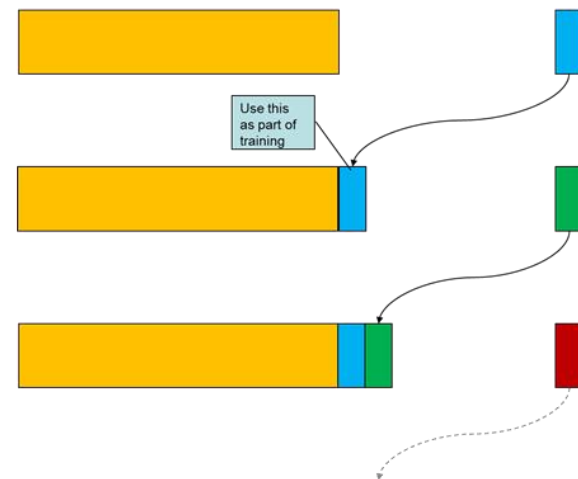
print("RMSE = ", np.sqrt(mse), " bbl/day")
print("R2 = ", Rsqr)
```

```
RMSE = 1457.058660103274 bbl/day
R2 = [[1.          0.43247048]
      [0.43247048 1.          ]]
```

```
[1253]: plt.scatter(X_test_date, y_test, color="red")
plt.plot(X_test_date, y_pr)
plt.xticks(rotation=90)
plt.ylim([10000,22000])
plt.show()
```



Training Prediction



test\_7.ipynb

```
[ ]: # perform walk-forward validation
# X_train, y_train, X_test, y_test
for j in range(0, len(y_pr)):
    temp = np.array(X_test[j,:]).copy() # take j^th row of X_test

    for k in range(0,j+1):
        col_idx = j+k*ncol_orig

        if col_idx < ncol_offset:
            temp[col_idx] = y_pr[j-k].copy() # offset: replace true water rate with predicted water rate
            #print("j+k*ncol_orig = ", j, "+", k, "=", ncol_orig, "=", j+k*ncol_orig)
        else:
            break

    y_pr[j] = regressor.predict(np.array([temp])) # predict water rate based on j^th row of X_test

# update training set for walk-forward validation
X_train = np.vstack([X_train, temp])
y_train = np.hstack([y_train, np.array(y_pr[j]).copy()])

# retrain model
regressor.fit(X_train,y_train)

print("j = ", j)
```

# Sand Analytics (1)

- We have installed acoustic sand device.
- The vendor provided the sand rate for each well based on proprietary equations.
- Sum of all sand rates did not match with the total sand produced.



[https://www.clampon.com/wp-content/uploads/2013/04/particle\\_monitor\\_2\\_500px.jpg](https://www.clampon.com/wp-content/uploads/2013/04/particle_monitor_2_500px.jpg)

# Sand Analytics (2)

- Formulation: Given  $N$  wells,

$$c_1 w_{1,ASD}(t) + c_2 w_{2,ASD}(t) + \cdots c_N w_{N,ASD}(t) = \sum_i c_i w_{i,ASD}(t) \rightarrow Q_{sand}(t)$$

Optimizer: GA Islands

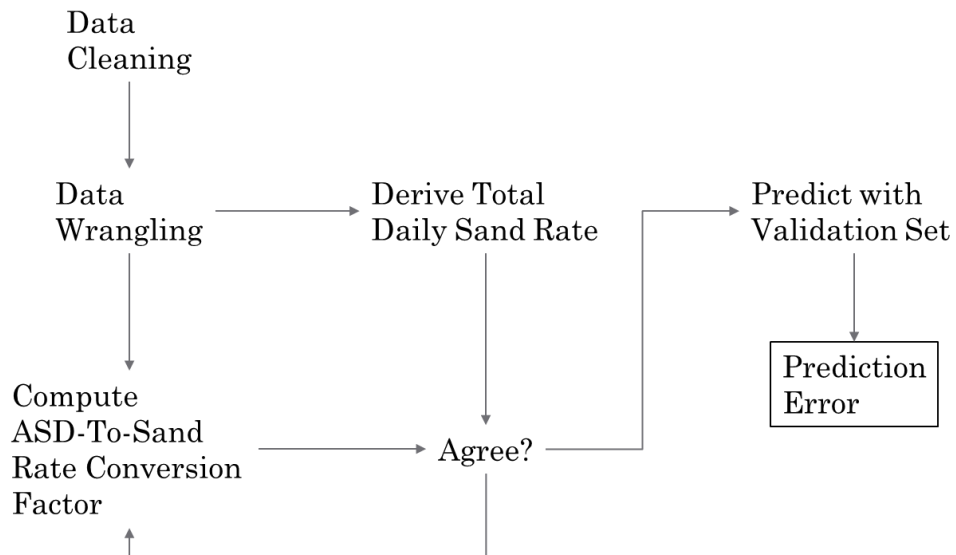
Challenges:

Multiple sand catch pot manually sampled at non-synchronous schedule.  
How to derive the daily sand rate  $Q_{sand}(t)$  ?

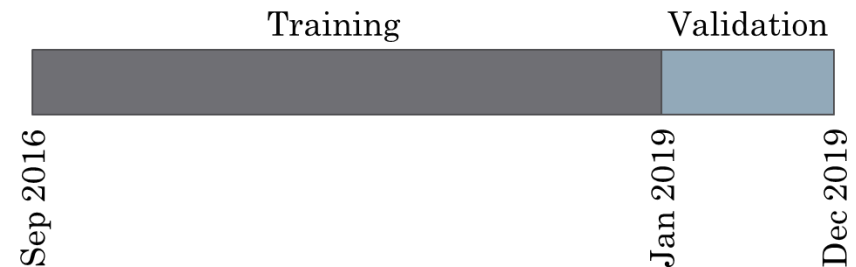
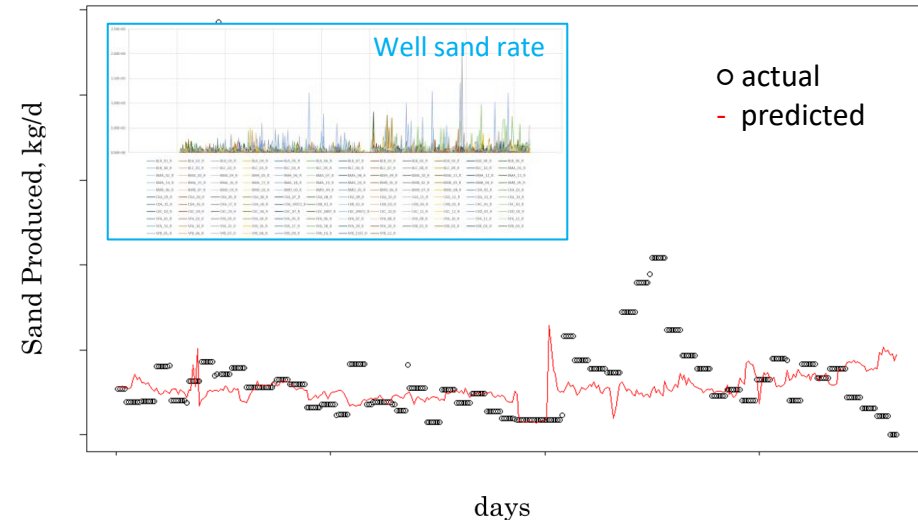
# Sand Analytics (3)

## Objectives

- Develop a computer program to derive total daily sand rate
- Develop a AI-based module to convert ASD (Acoustic Sand Device) signal of each well to a sand rate that obeys the total daily sand rate



## Predicted total sand production (2019)



# Conclusions

- Many different ML-based approach to quantify sources of water and sand.
- Computing water gas ratio (WGR) is highly interpretable to engineers. However, in reality, WGR for a well is not a constant value.
- The simple balance model (i.e. computing WGR) reflects good prediction for the training data, but not the test data. Hence, not reliable enough for prediction.
- Random Forests model can predict future water rate, and at the same time it can rank the importance of each well measurements. But, the engineers find it hard to interpret.
- Using time series method to predict future water rate, XGBoost outperformed Random Forests significantly. For daily dataset, XGBoost could reliable predict 3 to 6 months onto the future. Expect a low prediction accuracy when there is a facility shutdown.
- Converting acoustic signals to sand rate is quite reliable with a linear factor. The factor can be solved using Genetic Algorithm, although computationally expensive. Expect a low prediction accuracy when there is a facility shutdown.

Thank you.