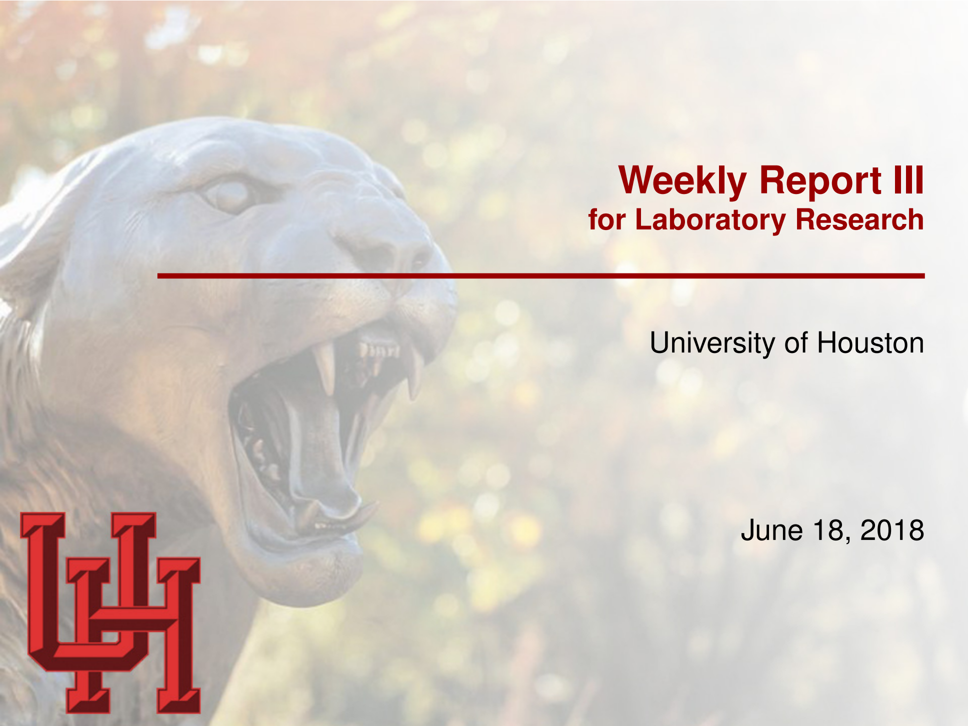

Weekly Report III
for Laboratory Research

University of Houston

June 18, 2018

Outline

1. Forward model: Curves
   - C++ Prototype
   - Matlab Result
   - Python Result
2. Plan for next step

Outline

1. Forward model: Curves
   - C++ Prototype
   - Matlab Result
   - Python Result
2. Plan for next step

Forward model: Curves

C++ Prototype

The forward model API

The basic API is

```
curves(Layers, Rh, Rv, Zbed, Dip, TVD)
```

- This project is aimed at simulating the response of an azimuthal resistivity LWD tool.
  - `Layers`: The number of layers.
  - `Rh`: [ohm * m] Resistivity $\in$ [0.1,1000].
  - `Rv`: Usually the same as `Rh`.
  - `Zbed`: [feet] Boundary Position $\in$ TVD+[-100, 100].
  - `Dip`: [deg] Dip Angle tool/boundary. When 90°, parallel to boundary.
  - `TVD`: True Vertical Depth. When 0, relative.

Forward model: Curves

Matlab Result

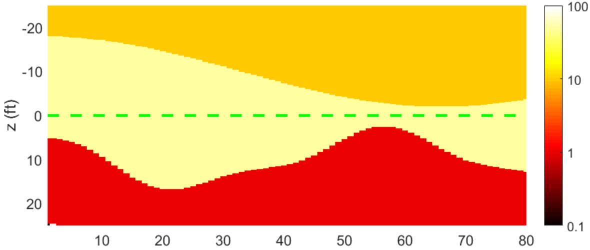

Figure 1: The generated random model.

- Use a script to generate a random 3-layer model.
- The resistivity is {10, 50, 1} respectively.
- The Z-bed position is adaptive according to *t*.

Forward model: Curves

Matlab Result

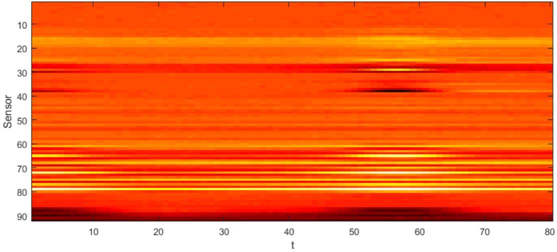

Figure 2: The response of the model.

- The sensor number is 92. `TVD` is kept 0 and `Dip` is kept 90 °.
- $t \in (1, 80)$.
- Each sensor gives a different response.

Forward model: Curves

Matlab Result

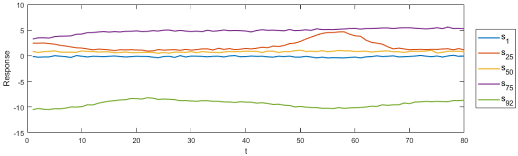

Figure 3: The response of the model (selective).

- We select s_1, s_{25}, s_{50}, s_{75} and s_{92} to show part of the results.
- The response is added with a small noise.

Forward model: Curves

Matlab Result

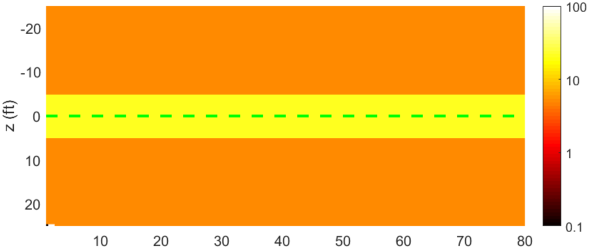

Figure 4: The initialized model during the inversion.

- We use a flat model to predict the whole model during the initialization.
- The target is to minimize $\|\mathscr{C}(\mathbf{p}) - \mathbf{r}\|$, where $\mathscr{C}$ is the model, $\mathbf{p}$ is the input parameters and $\mathbf{r}$ is the response.

Forward model: Curves

Matlab Result

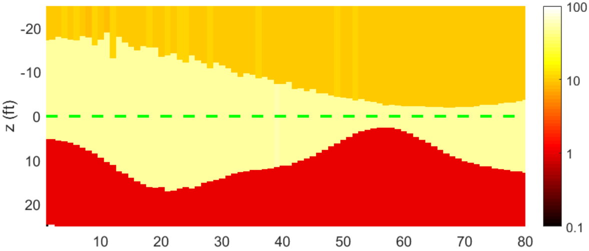

Figure 5: The inversed model.

- The inversed model is slightly different from the real one.

Forward model: Curves

Python Result

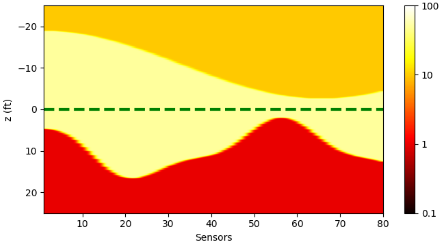

Figure 6: The python model.

- We have migrate the matlab-c-api to python-c-api.
- This is the result given by python and of the same real model.

Forward model: Curves

Python Result

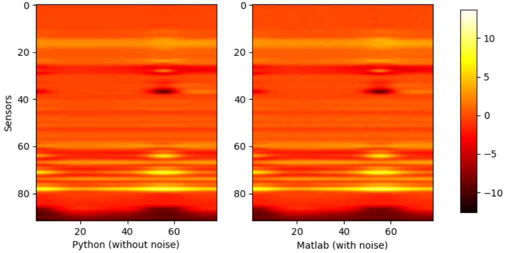

Figure 7: The python model response compared to the original one.

- We have compared the results of the forward model defined by python API. It is almost the same with the matlab one (the difference is caused by small noise).

Forward model: Curves

Python Result

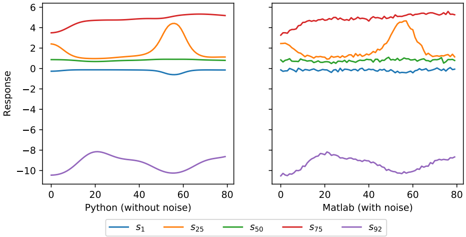

Figure 8: The python model response compared to the original one (selective).

- To show the details, we compare the results from same sensors.
- We does not add random noise to python results.

Forward model: Curves

Python Result

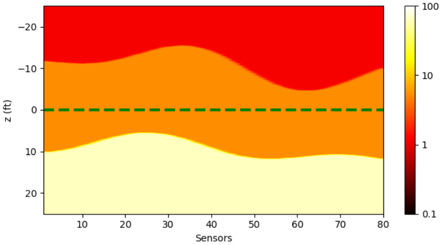

Figure 9: The model generated by migrated generator script.

- We have migrated the random generator to python script, too.
- This is another different model generated by the script.

Forward model: Curves

Python Result

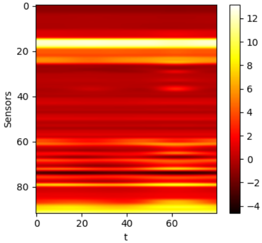

Figure 10: The response of generated model.

- The is the simulated response from the forward model.

Outline

1. Forward model: Curves
   - C++ Prototype
   - Matlab Result
   - Python Result

2. Plan for next step

Plan for next step

1. Try to use the tensorflow to simulating the primal inversion algorithm.
   - Maybe `tf.py_func` would be a solution.
2. Try to implement the deep-learning scheme by introducing the forward model.