

Introduction to SWIT

Haipeng Li (李海鹏) (haipengl@mail.ustc.edu.cn)

Supervisor: Prof. Junlun Li (李俊伦) (lijunlun@ustc.edu.cn)

USTC

Aug. 2021

Introduction to SWIT

1. Features of SWIT
2. Installation of SWIT
3. How to run SWIT
4. Cases of SWIT

Introduction to SWIT

- 1. Features of SWIT**
2. Installation of SWIT
3. How to run SWIT
4. Cases of SWIT

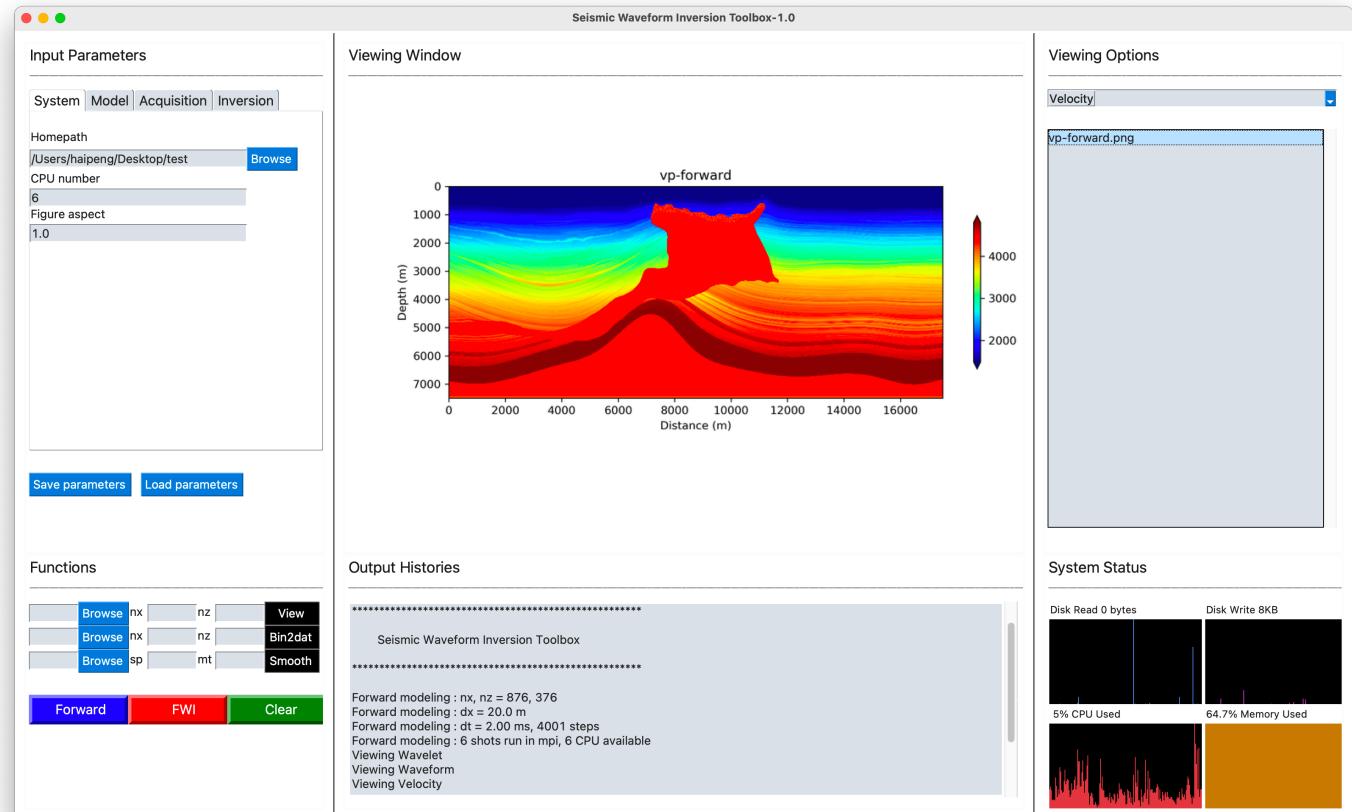
Features of SWIT

- ✓ Forward wavefield modeling (FD)
- ✓ Adjoint wavefield modeling (FD)
- ✓ Calculation of gradient (on-the-fly)
- ✓ Preconditioning & Regularization
- ✓ L-BFGS & NLCG
- ✓ Effective Step Length Search
- ✓ Various misfit functions
- ✓ Multi-scale inversion

Homepage: <https://github.com/Haipeng-ustc/SWIT-1.0>

(Please star the project for latest updates)

Seismic Waveform Inversion Toolbox



Features of SWIT

Homepage: <https://github.com/Haipeng-ustc/SWIT-1.0>

(Please star the project for latest updates)

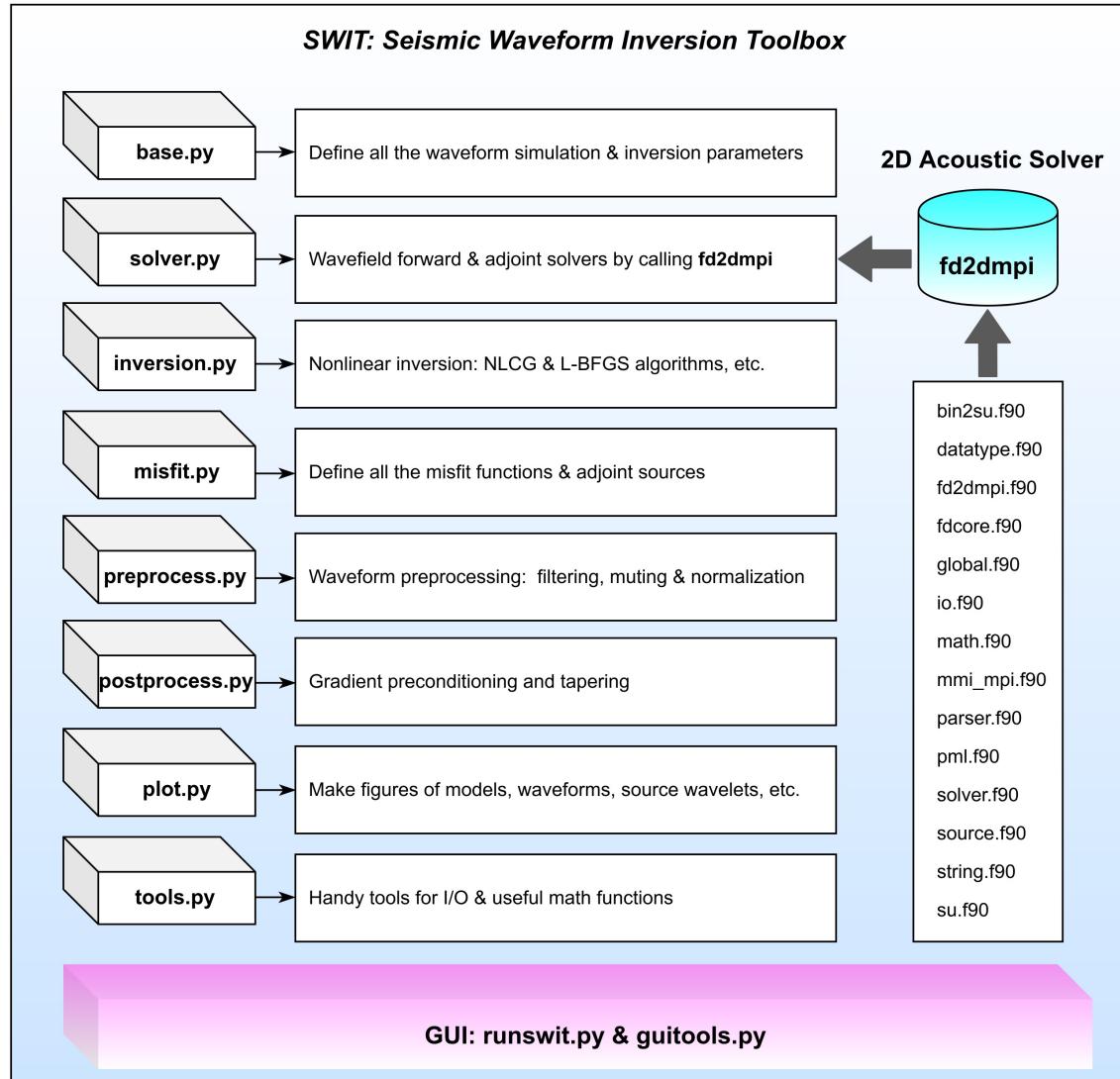
- ✓ Our in-house FWI platform (**Keep updating**)
- ✓ FD modeling engine (**Efficient Fortran, MPI**)
- ✓ State-of-the-art algorithms (**Flexible Python**)
- ✓ Easy definition of parameters (**GUI**)
- ✓ Industry-standard data format (**SU data stream**)
- ✓ Cross platform (**Linux & MacOS**)

Contents of SWIT

 bin
 doc
 examples
 fd2dmpi
 toolbox
 LICENSE
 README.md

- The compiled wavefield solver (fd2dmpi)
- The Manual (keep updating)
- The provided examples (keep updating)
- Wavefield solver code in Fortran
- Full-waveform inversion code in Python
- License
- Readme

Structures of SWIT



➤ **Forward: Fortran, MPI**

➤ **Inversion: Python**

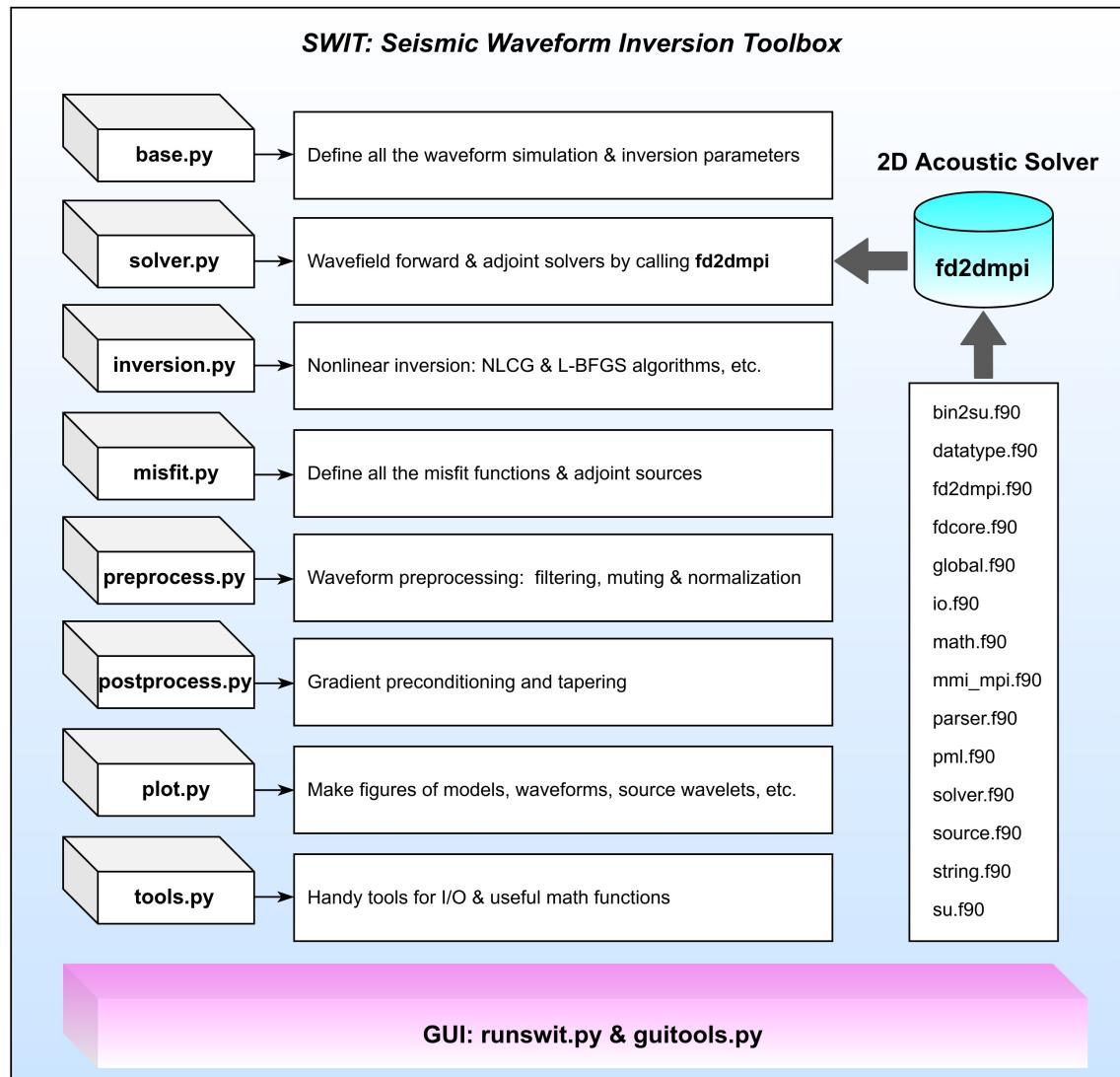
➤ **Usage:**

✓ opt 1: GUI

✓ opt 2: Python script

✓ opt 3: Jupyter Notebook

Structures of SWIT



➤ Forward: Fortran, MPI



➤ Inversion & Usage: Python

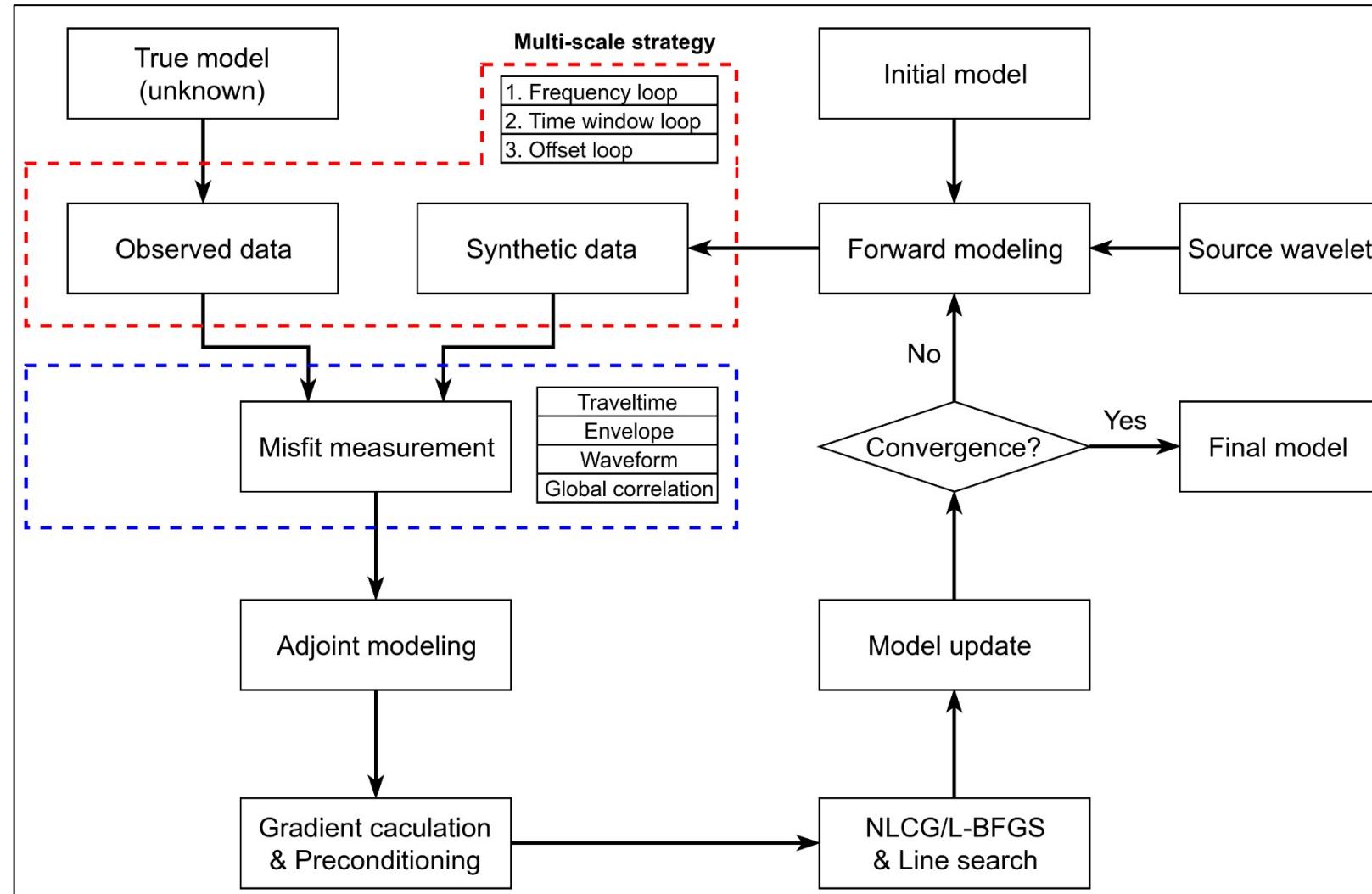


matplotlib

...

Workflow of SWIT

SWIT solves the **variable-density acoustic wave equation** in the stress-velocity form
SWIT uses a **4th-order Butterworth filter (zero-phase)**



Introduction to SWIT

1. Features of SWIT

2. Installation of SWIT

3. How to run SWIT

4. Cases of SWIT

Installation of SWIT

(Tested on Ubuntu 16.04, 18.04, 20.04)

Homepage: <https://github.com/Haipeng-ustc/SWIT-1.0>

Step 1.

```
sudo apt-get install build-essential
```

```
sudo apt install gfortran
```

If you have already installed gfortran, gcc, MPI, Anaconda (recommended), go to”

- **Step3:** install Python dependencies, and then
- **Step4:** compile fd2dmpi

Installation of SWIT

(Tested on Ubuntu 16.04, 18.04, 20.04)

Step 2. Install OpenMPI

Down load OpenMPI, and then install (Home website: <http://www.open-mpi.org/software/ompi>)

```
wget https://download.open-mpi.org/release/open-mpi/v4.1/openmpi-4.1.1.tar.gz
```

```
tar xvfz openmpi-4.1.1.tar.gz
```

```
cd openmpi-4.1.1
```

configure & make (takes some time)

```
./configure --prefix=/usr/local/openmpi CC=gcc FC=gfortran
```

```
make # make -j8 (use 8 cores to accelerate the process)
```

```
sudo make install
```

add environment path to `~/.bashrc` and check whether you successfully install OpenMPI or not

```
export PATH=/usr/local/openmpi/bin:$PATH
```

```
source ~/.bashrc
```

```
which mpirun
```

Installation of SWIT

(Tested on Ubuntu 16.04, 18.04, 20.04)

Step 3. Install Anaconda and Python Dependencies

```
# For install Anaconda, refer to: https://docs.anaconda.com/anaconda/install/linux/

# download package from: https://www.anaconda.com/products/individual/download-success

# bash ~/your_Anaconda_package

# Create new Anaconda env for SWIT

conda create --name SWIT python=3.7

conda activate SWIT

# Install python dependencies using pip with USTC resources

pip install numpy obspy scipy matplotlib -i https://pypi.mirrors.ustc.edu.cn/simple/

pip install multiprocessing PySimpleGUI psutil Pillow -i https://pypi.mirrors.ustc.edu.cn/simple/
```

Installation of SWIT

(Tested on Ubuntu 16.04, 18.04, 20.04)

Step 4. Compile the wavefield solver fd2dmpi

```
# Compile the Fortran code
```

```
cd ~/SWIT-1.0/fd2dmpi/
```

```
rm *.mod; make clean; make
```

```
# add environment path to ~/.bashrc
```

```
export PATH=~/SWIT-1.0/bin:$PATH
```

```
export PYTHONPATH=~/SWIT-1.0/toolbox
```

```
source ~/.bashrc
```

```
which fd2dmpi
```

```
python >> import inversion # test whether you can import this module from ~/SWIT-1.0/toolbox/
```

Introduction to SWIT

1. Features of SWIT
2. Installation of SWIT
- 3. How to run SWIT**
4. Cases of SWIT

How to run SWIT

(Tested on Ubuntu 16.04, 18.04, 20.04)

Opt 1: run SWIT via GUI

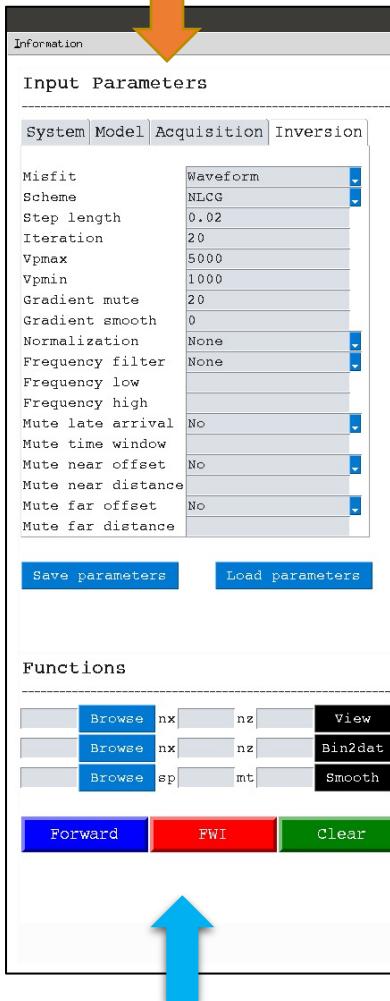
```
cd ~/SWIT-1.0/toolbox/  
python runswit_Linux.py  
# or python runswit_MacOS.py
```

Opt 2: run SWIT via Python script

```
cd ~/example/some_case/  
python -W ignore SWIT_workflow.py
```

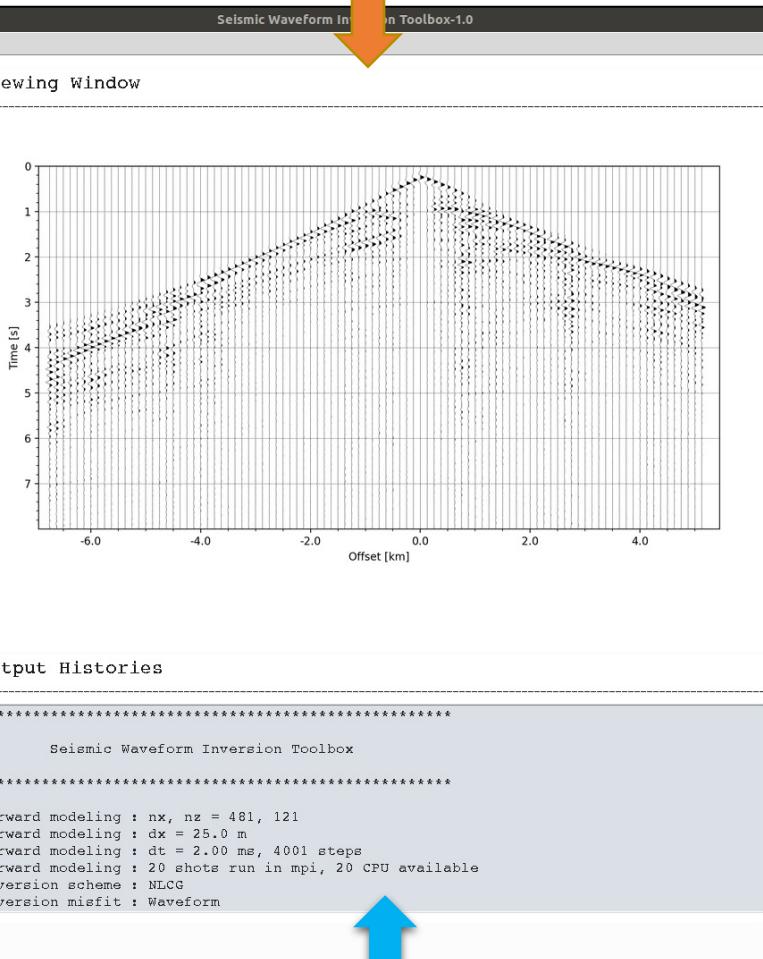
Run SWIT via GUI

1. Input parameters



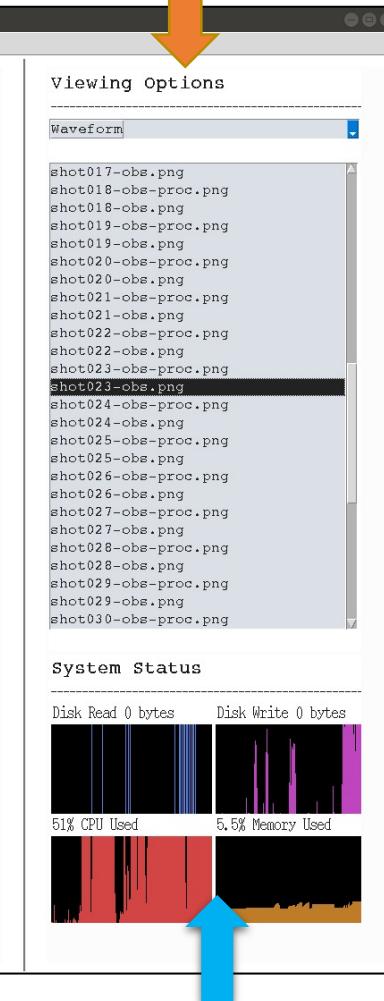
2. Functions

3. Viewing Window



4. Output Histories

5. Viewing Options



6. System Status

Input Parameters: System

Input Parameters

The screenshot shows a software interface for input parameters. At the top, there is a navigation bar with tabs: System (selected), Model, Acquisition, and Inversion. Below the tabs, there are three configuration fields:

- Homepath:** The value is /data1/TempData. To the right is a blue "Browse" button.
- CPU number:** The value is 20.
- Figure aspect:** The value is 1.

At the bottom of the interface are two buttons: "Save parameters" (blue) and "Load parameters" (blue).

Annotations on the right side of the interface provide descriptions for each setting:

- Project working folder (next to Homepath)
- CPU number for MPI (next to CPU number)
- Figure aspect for plotting (next to Figure aspect)

Two large black arrows point upwards from the "Save parameters" and "Load parameters" buttons towards the respective "Save all the current parameters into file: XXX.swit" and "Load the previously saved parameter file: XXX.swit" text labels.

Save all the current parameters into file: **XXX.swit**

Load the previously saved parameter file: **XXX.swit**

Save parameters Load parameters

Input Parameters: Model

Input Parameters

System		Model	Acquisition	Inversion
Nx	481			
Nz	121			
Dx	25			
Dt	0.002			
Nt	4001			
PML	50			
Free surface	Yes			

Vp true (forward)
/home/haipeng/Nutstore Files

Vp init (inversion)
/home/haipeng/Nutstore Files

SU data (inversion)

Model size along x direction

Model size along z direction

Grid size (in m)

Temporal step (in s)

Total time steps

PML size, use a relatively large one, i.e., 50

Set free surface or not

True Vp model saved in the **txt file**

Initial Vp model saved in the **txt file**

Field-data path, data should be named as: **src1_sg.su, src2_sg.su, ..., src40_sg.su, ...**

For model study: **Vp true & Vp init**

For field-data: **SU data & Vp init**

Input Parameters: Acquisition

Input Parameters

System Model Acquisition Inversion

Receiver coordinate
/home/haipeng/Nutstore Files [Browse](#)

Source coordinate
/home/haipeng/Nutstore Files [Browse](#)

Land or Marine

Source wavelet

Opt 1: F0 (Hz) 5.0

Opt 2: File [Browse](#)

Receiver file, see examples for detailed format

Source file, see examples for detailed format

Acquisition type: land or marine

Source wavelet: Ricker or from file

If Ricker, set the dominant frequency

If from file, set the file path, txt files

[Save parameters](#)

[Load parameters](#)

Input Parameters: Inversion

Input Parameters

	Inversion
Misfit	Waveform
Scheme	NLCG
Step length	0.02
Iteration	20
Vpmax	5000
Vpmin	1000
Gradient mute	20
Gradient smooth	0
Normalization	None
Frequency filter	None
Frequency low	
Frequency high	
Mute late arrival	No
Mute time window	
Mute near offset	No
Mute near distance	
Mute far offset	No
Mute far distance	

- Set misfit function: Waveform, Traveltime, etc.
- Set optimization scheme: NLCG or L-BFGS (**NLCG is recommended**)
- Set step length, i.e., 0.01 or 0.02 (with respect to Vp model), **small step length is recommended**
- Set number of iteration
- Maximum Vp (in m/s)
- Minimum Vp (in m/s)
- Set gradient mute. **In marine case, it's the water layer size; in land case, set as ~10 (in grid)**
- Set gradient smoothing size, i.e., 10 (in grid)
- Set whether to normalize the data. If so, set which type
- Set whether to filter the data. If so, set which type
- Set the low frequency bound (in Hz)
- Set the high frequency bound (in Hz)
- Set whether to mute late arrivals based on the automatically picked first break
- Set the time window after the first break, i.e., 1.0 (in s)
- Set whether to mute traces of near offset
- Set the distance to mute near offset distance (in m)
- Set whether to mute far offset
- Set the distance to mute traces of far offset distance (in m)

Save parameters

Load parameters

Functions

Functions



View a txt file (donot require nx & nz) or binary file (**require nx & nz**)

Convert a binary file to a txt file (**require nx & nz**)

Smooth a txt file. **sp**: the smoothing radius; **mt**: the top mute for any water layer

Forward Perform forward modeling based on the provided Vp true.

The generated shot gathers are saved in `~/data/obs/xxx.su`

FWI Perform inversion based on the provided Vp initial and SU data

Clear **Clear output histories**

Few more words

- **Sythetic model FWI :**

Step 1: set all the parameters (**must provide Vp true & Vp init**);

Step 2: run **Forward** to generate the ‘true’ data;

Step 3: run **FWI** to perfrom the inversion.

- **Field-data FWI :**

Step 1: set all the parameters (**must provide Vp init and SU data**);

Step 2: run **FWI** to perfrom the inversion.

Please keep in mind that the provided **SU data** should:

- Keep the same **source** and **receiver coordinates** with those set in the **Acquisition**
- Keep the same **Nt** and **Dt** with with those set in the **Model**
- For the varying locations of receivers with sources, use **zero traces** to keep all the receivers are the same for all sources
- Fill the bad traces with the zero traces, all the zero traces will be automatically skipped.

Functions

Browse	nx	nz	View
Browse	nx	nz	Bin2dat
Browse	sp	mt	Smooth

Forward	FWI	Clear
---------	-----	-------

Run SWIT via Python script

See the Python script under each example folder. All the parameters are well commented and thus should be axiomatic.

```
# import modules
import numpy as np
import base
from inversion import inversion, source_inversion
from plot import plot_geometry, plot_model2D, plot_stf, plot_trace
from preprocess import process_workflow
from solver import forward, source_wavelet
from tools import saveparjson, smooth2d

### system setup
homepath = '/data/haipeng/SWIT-1.0/examples/land-case1-marmousi/' # working path
mpiprocs = 41 # mpi process for fd2dmpi
figaspect = 1.0 # Figure aspect

### model setup
nx, nz = [481, 121] # Grid number along x and z directions
pml, fs = [50, True] # Grid number for PML layers (use a large one)
dx, dt, nt = [25, 0.002, 4001] # Grid size, time interval, and time step

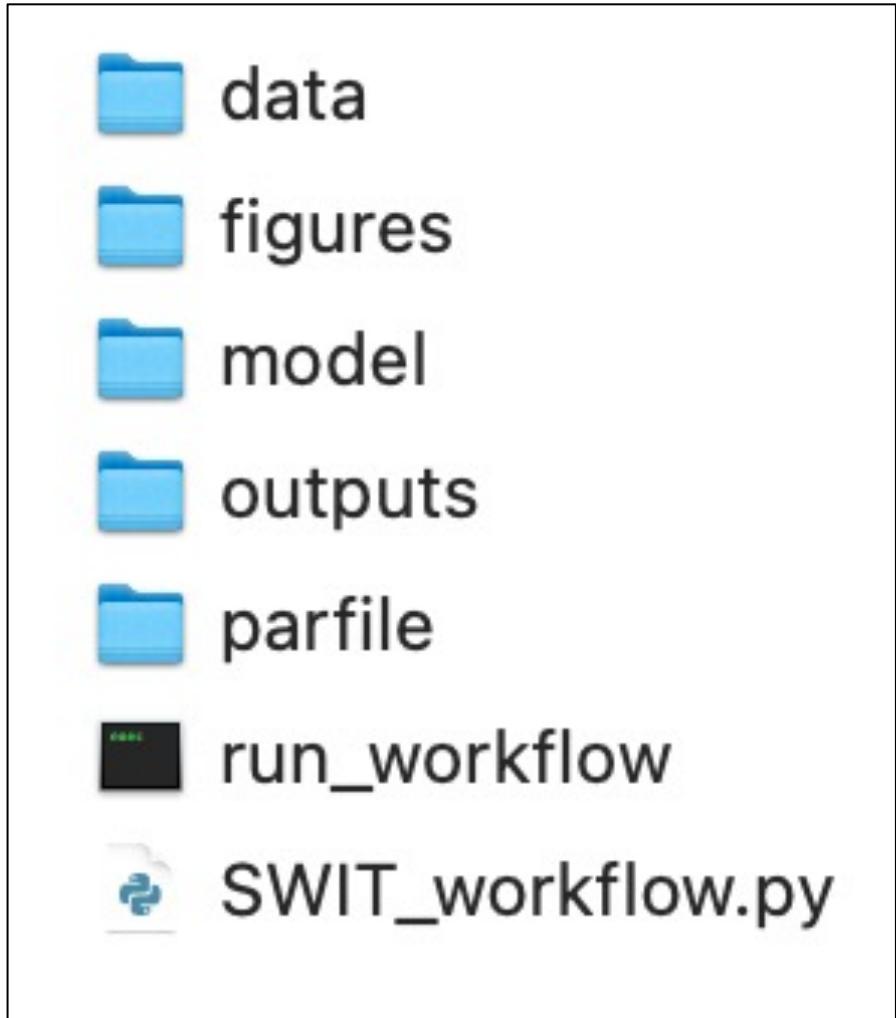
# velocity and density
vp_true = np.zeros((nx, nz))
vp_init = np.zeros((nx, nz))
rho_true = np.zeros((nx, nz))
rho_init = np.zeros((nx, nz))

vp_true = np.loadtxt(homepath + 'model/Marmousi_481_121_25m_True.dat')
```

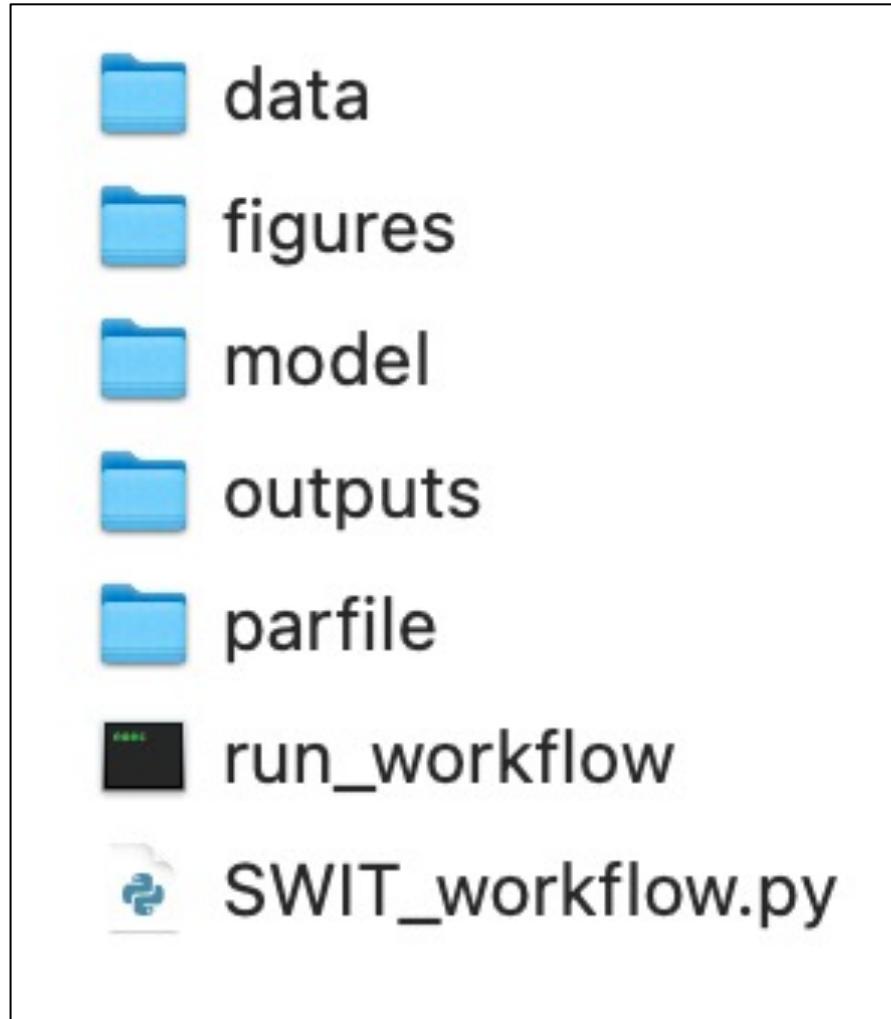
First activate SWIT Anaconda env, and then

python -W ignore SWIT_workflow.py

Outputs of SWIT



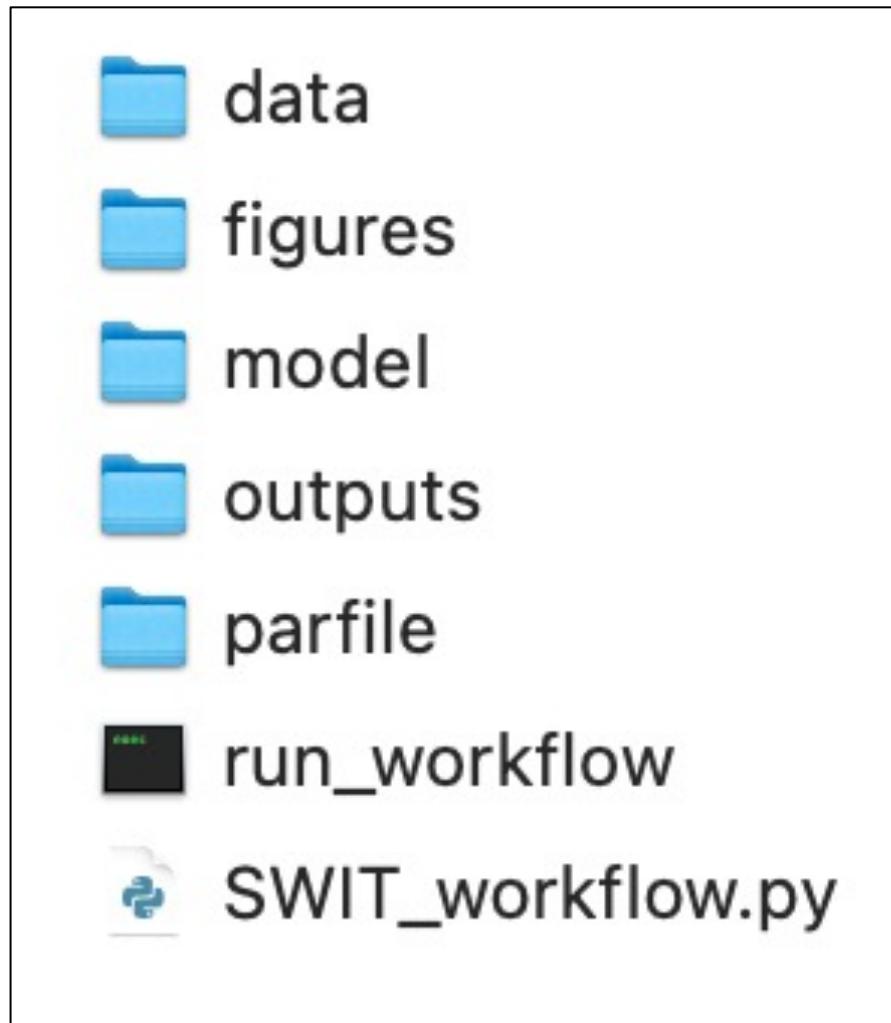
Outputs of SWIT



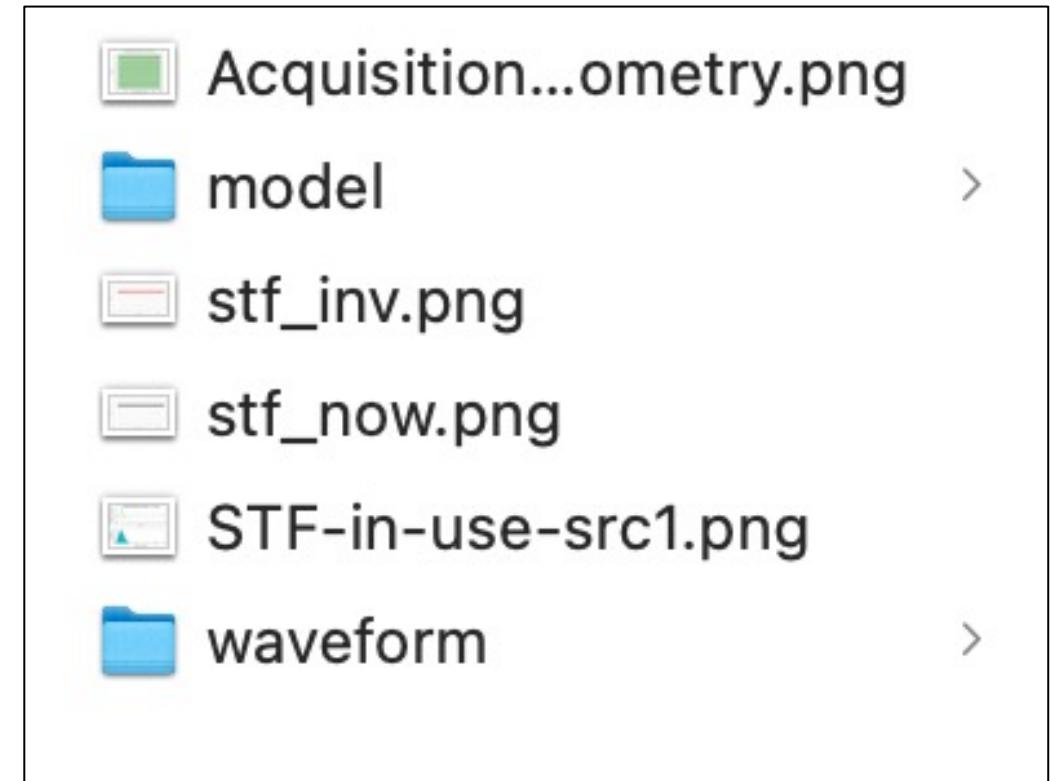
Waveform/wavefield data

- obs >
- syn >
- tempdata >

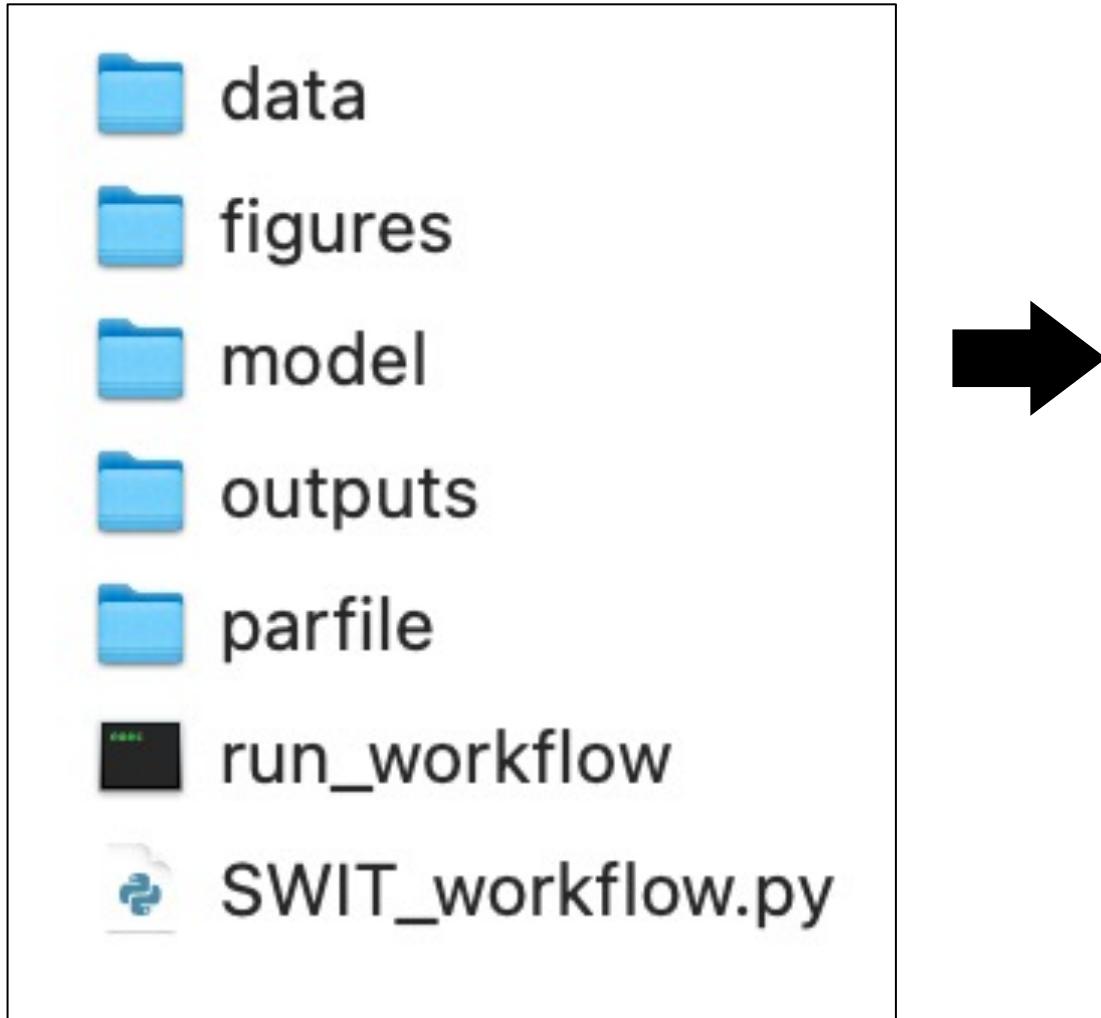
Outputs of SWIT



Visulization: acquisition, model (velocity, gradient, illumination), wavelet, waveform

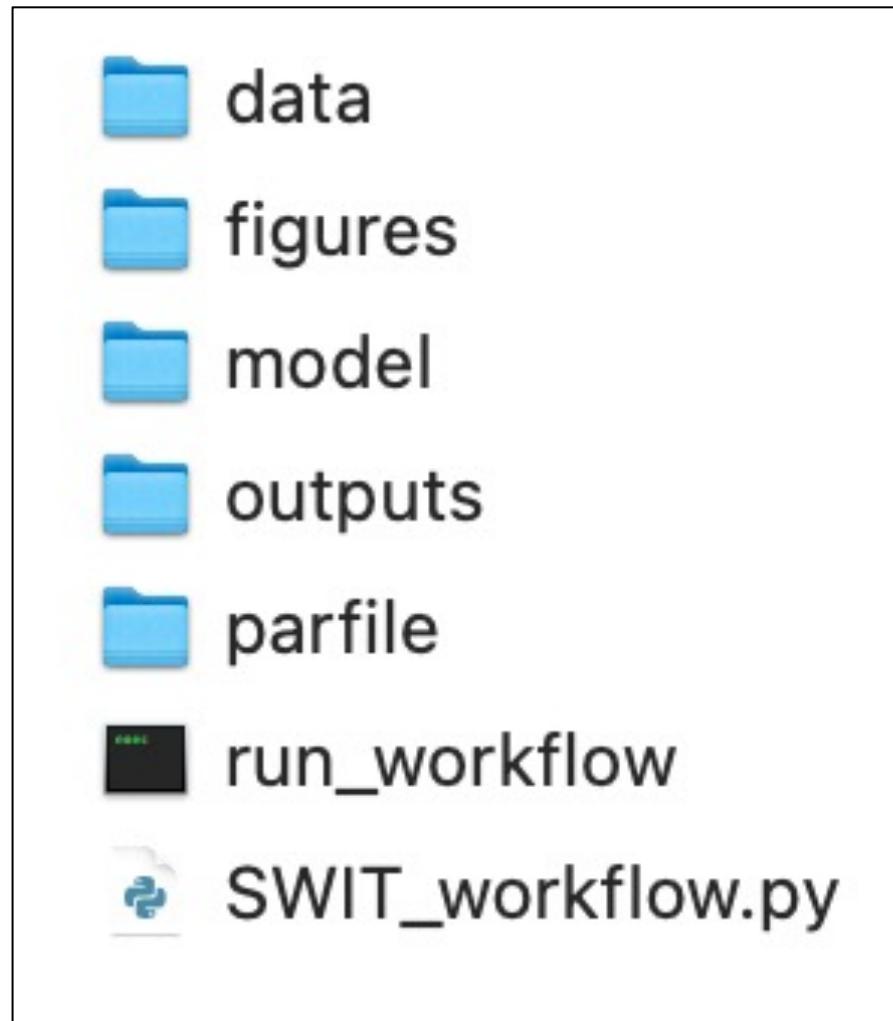


Outputs of SWIT

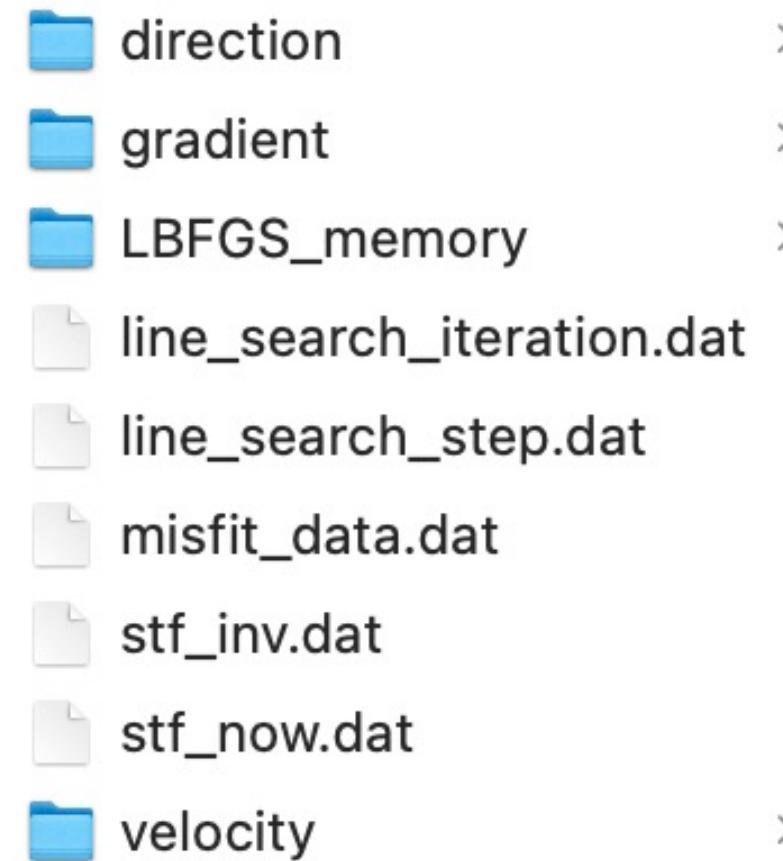


Where you put your true/initial
velocity models

Outputs of SWIT

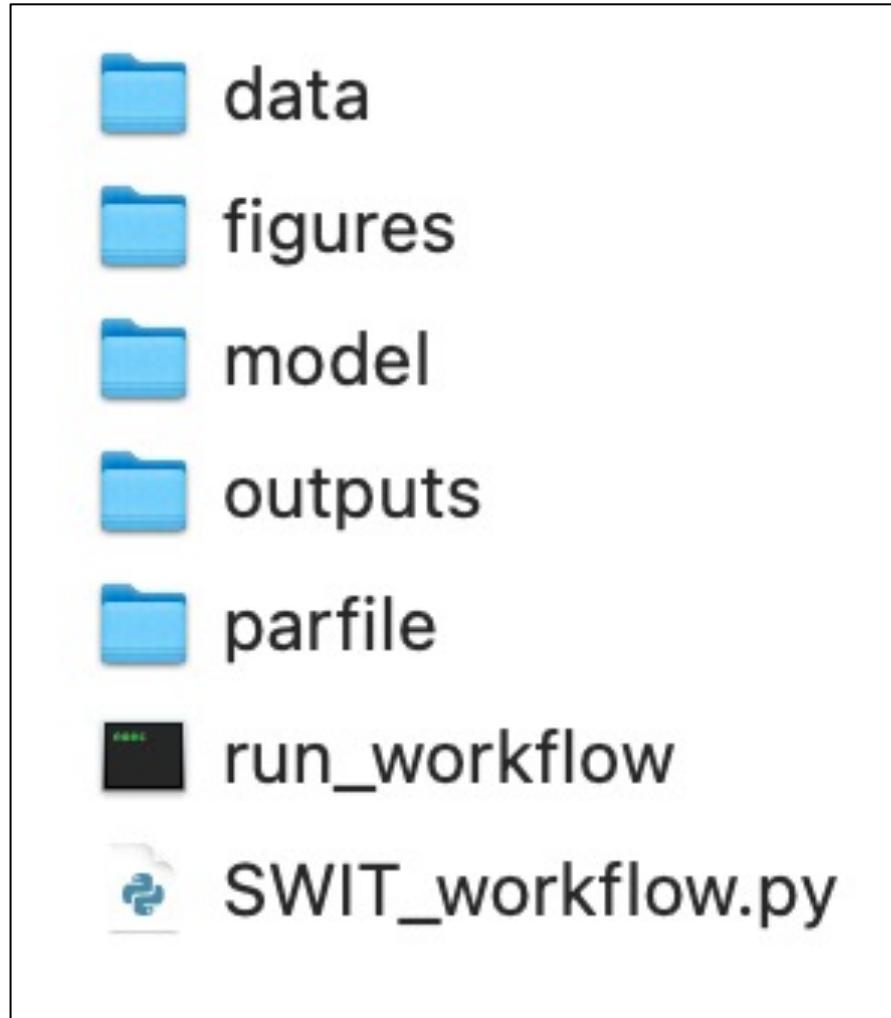


Inversion results: direction, gradient, line length, misfit, velocity, etc.



Outputs of SWIT

Parameters used during the running. All parameters
are saved into json files in case for later use.



Introduction to SWIT

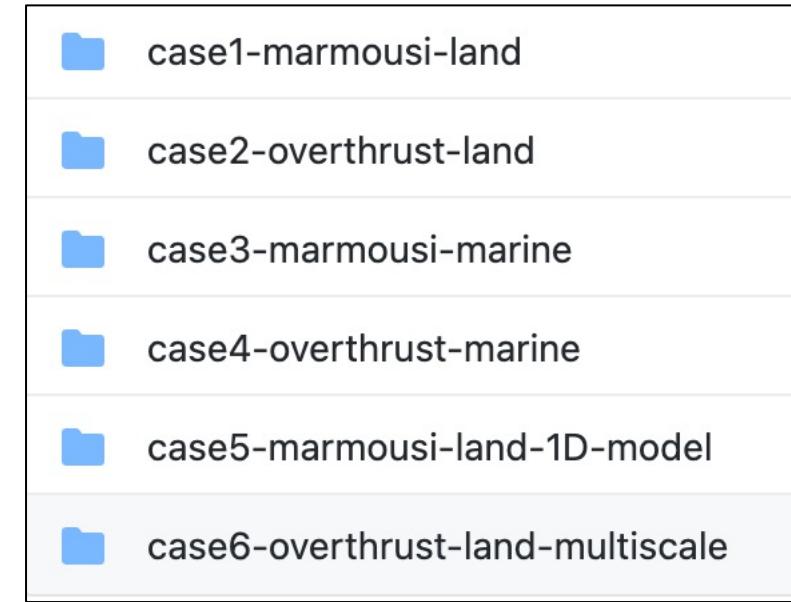
1. Features of SWIT
2. Installation of SWIT
3. How to run SWIT
- 4. Cases of SWIT**

Cases of SWIT

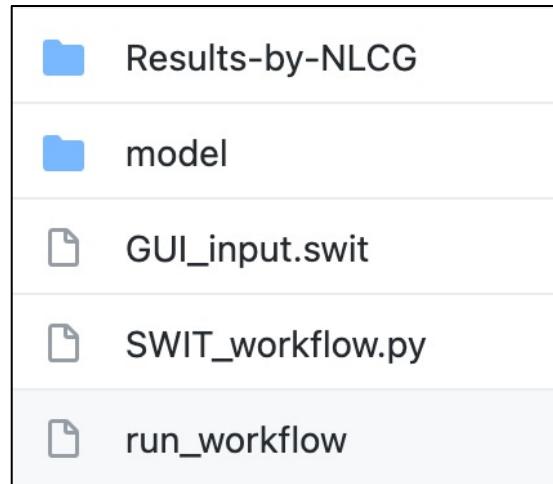
Homepage : <https://github.com/Haipeng-ustc/SWIT-1.0>

Provided examples

No.	Acquisition	Model	Misfit	Features	Optimization	Size
1	Land	Marmousi	Waveform	-	NLCG	481x121, 25 m
2	Land	Overthrust	Waveform	-	NLCG	401x101, 25 m
3	Marine	Marmousi	Waveform	-	NLCG	481x141, 25 m
4	Marine	Overthrust	Waveform	-	NLCG	401x121, 25 m
5	Land	Marmousi	Traveltime & Waveform	1D initial model	NLCG	401x121, 25 m
6	Land	Overthrust	Waveform	Multi-scale Inversion	NLCG	401x101, 25 m



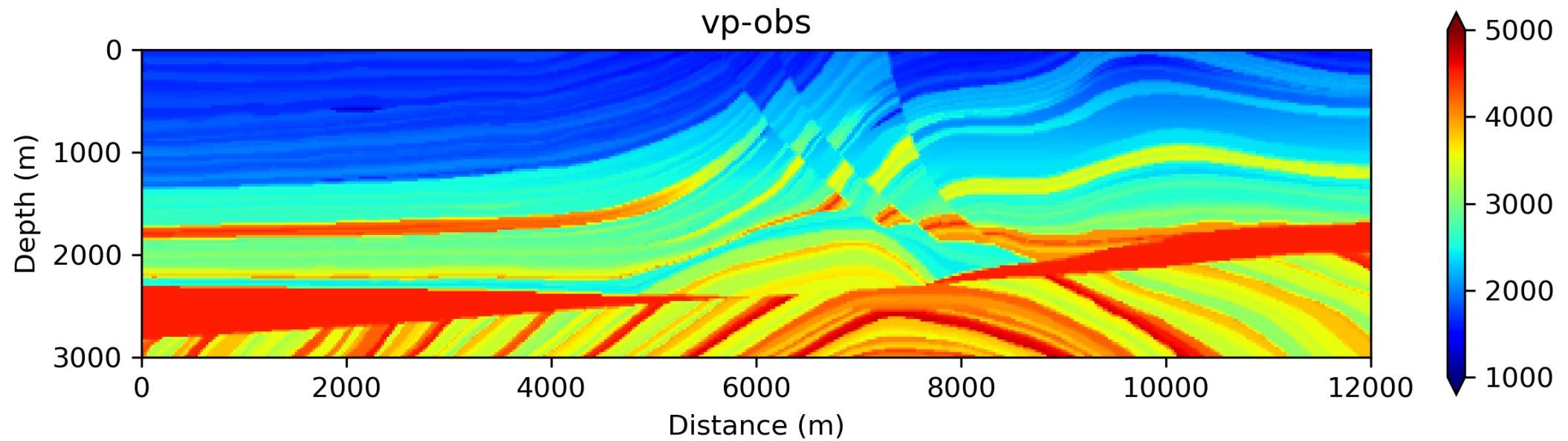
Content of each example



- Result for reference
- model
- Input parameters into GUI (you may change all the PATHs)
- Python script (you may change all the PATHs)
- Bash script

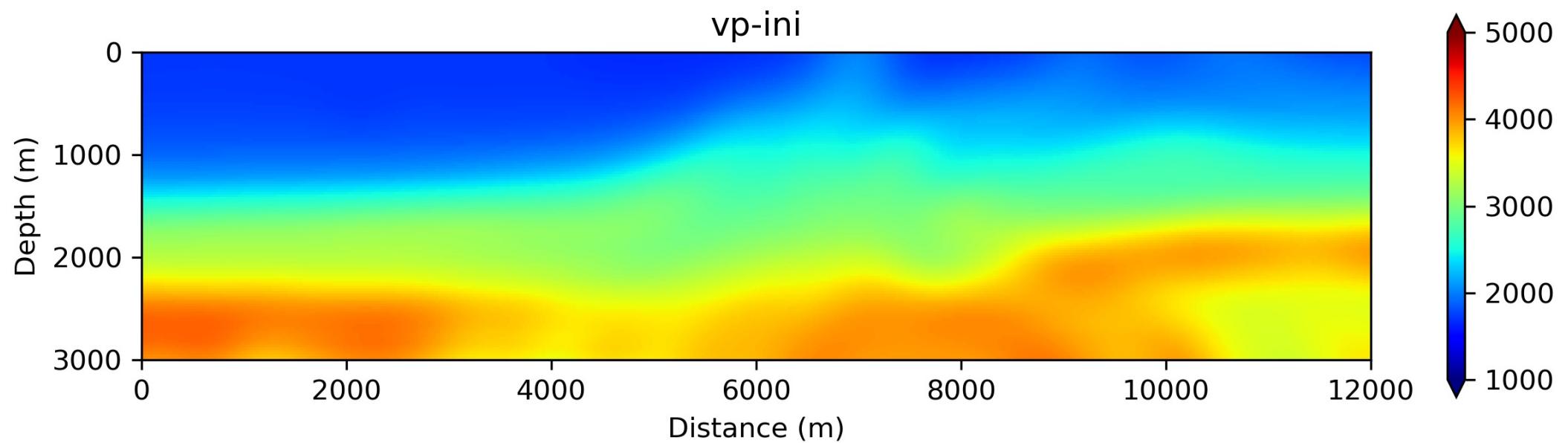
SWIT-Marmousi

True model



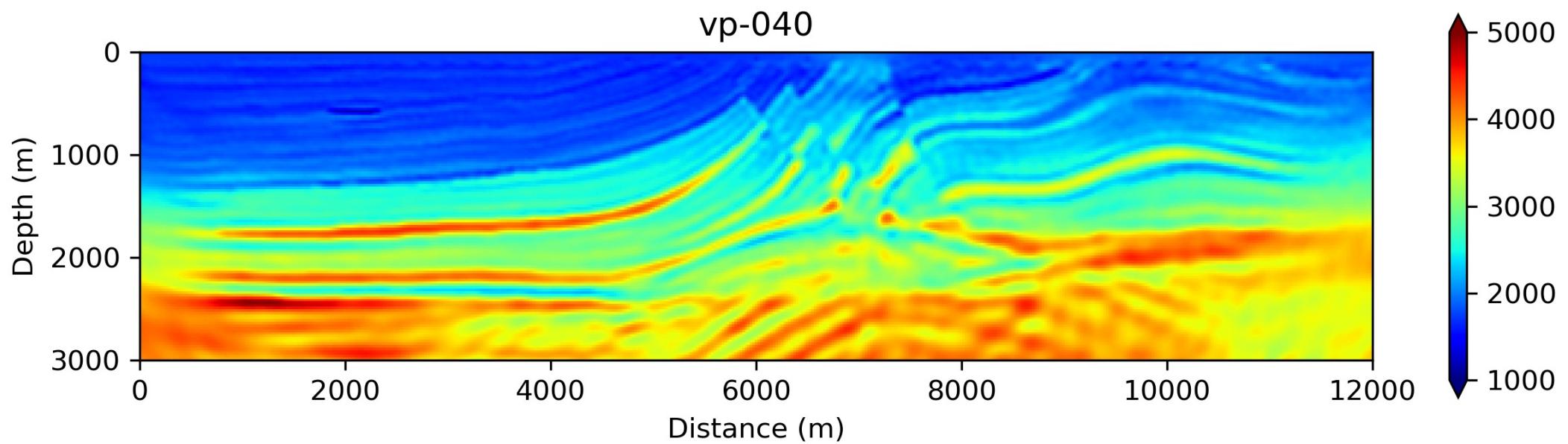
SWIT-Marmousi

Initial model



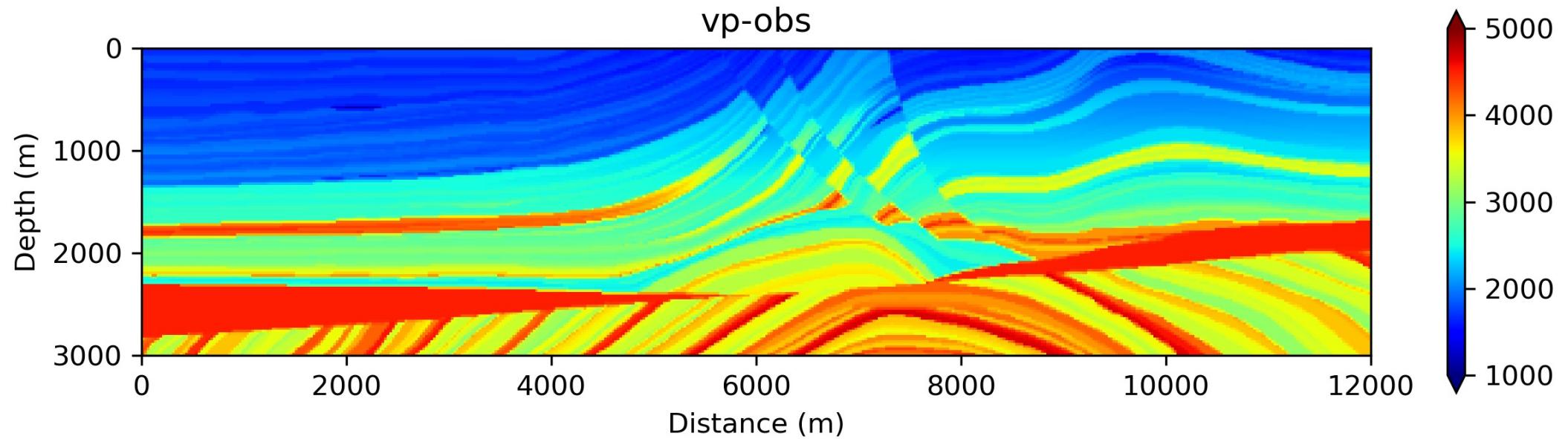
SWIT-Marmousi

Inversion result (40 iter.)



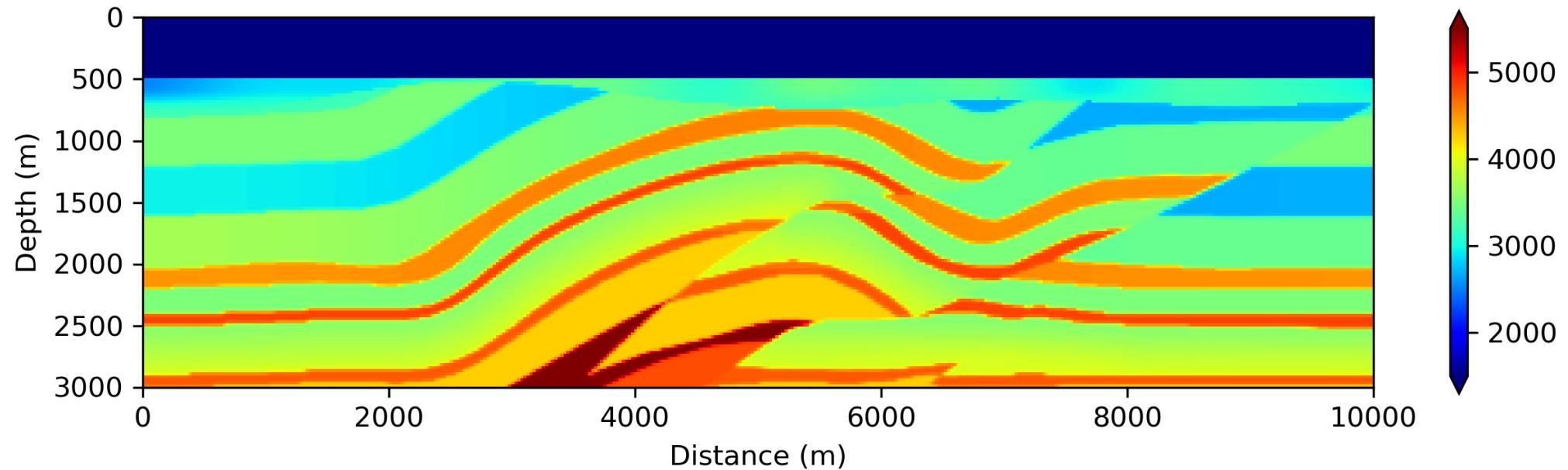
SWIT-Marmousi

True model



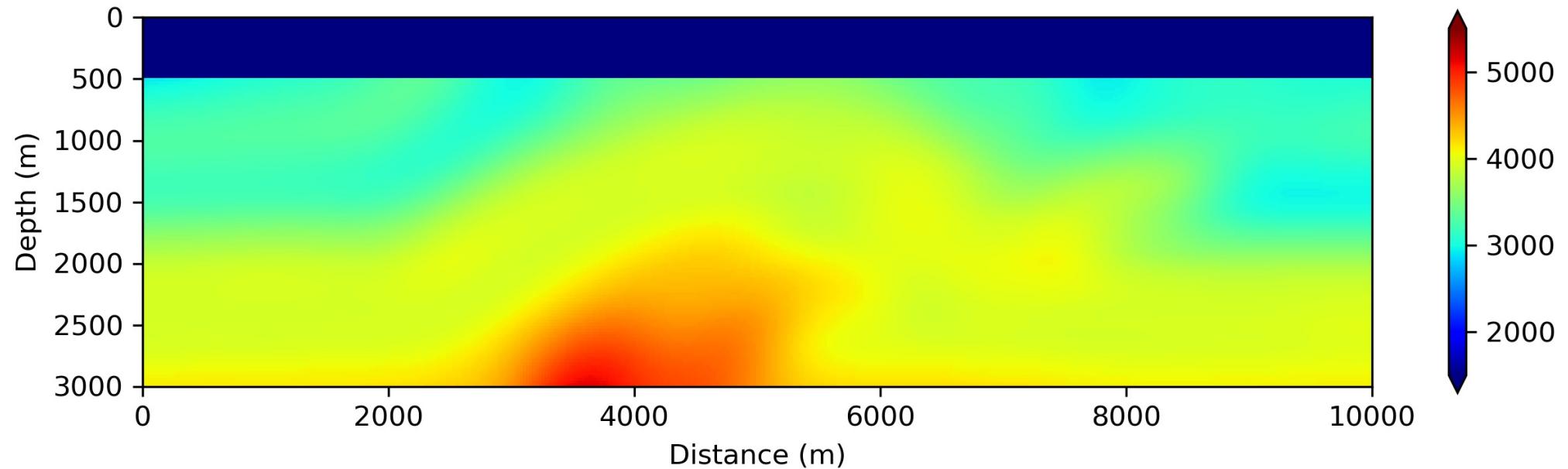
SWIT-SEG/EAGE Overthrust

True model



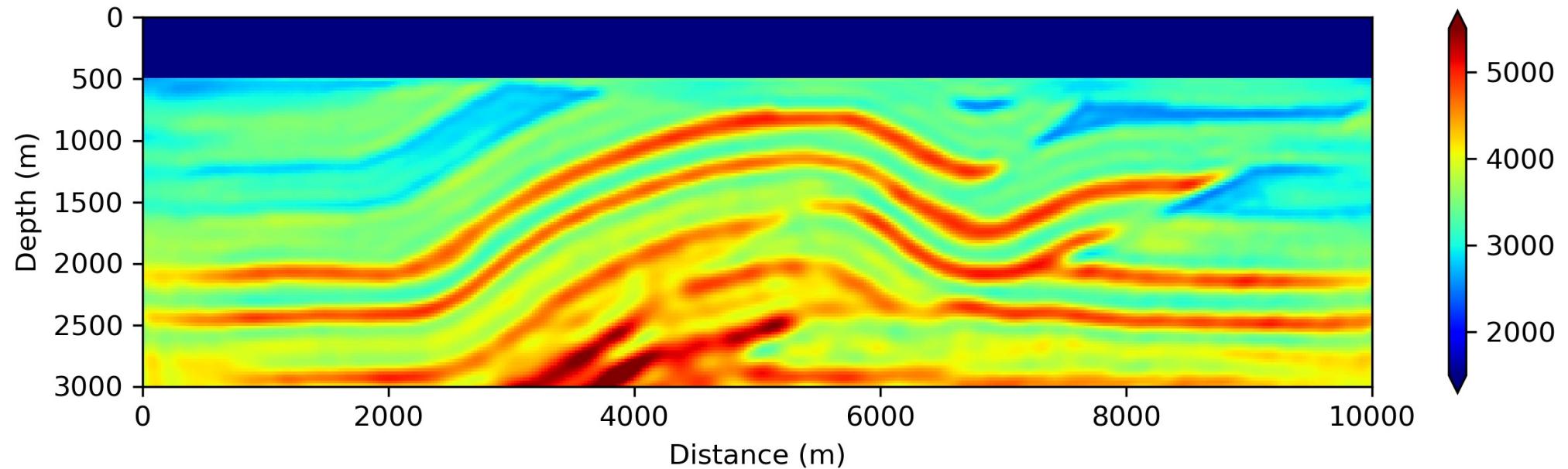
SWIT-SEG/EAGE Overthrust

Initial model



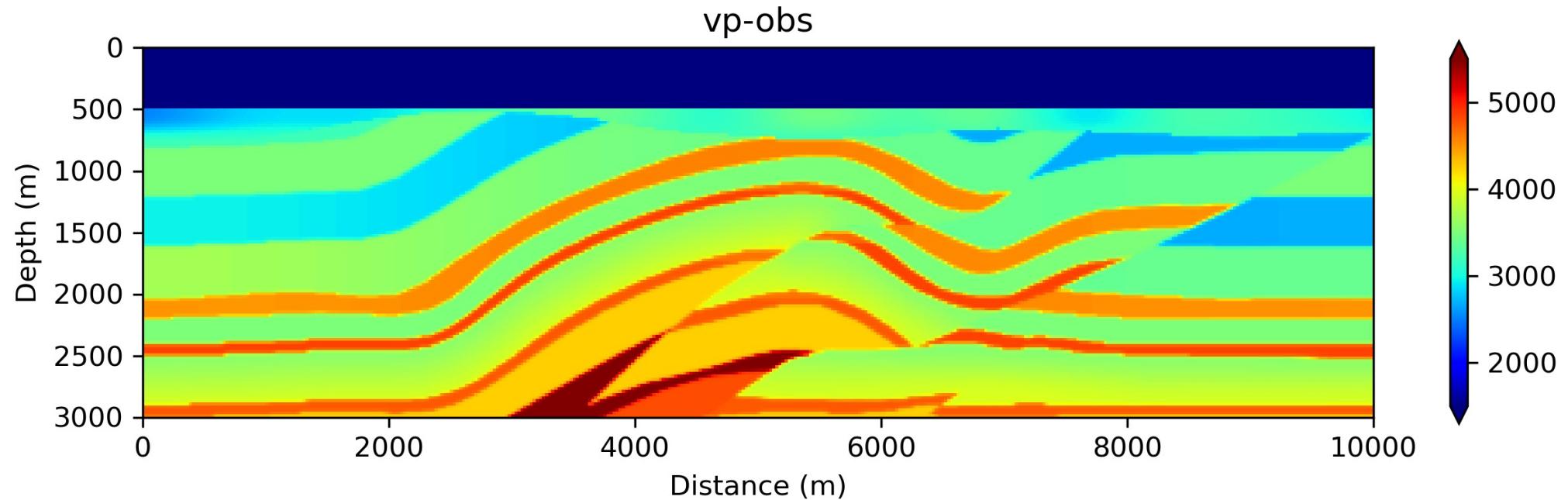
SWIT-SEG/EAGE Overthrust

Inversion result (40 iter.)



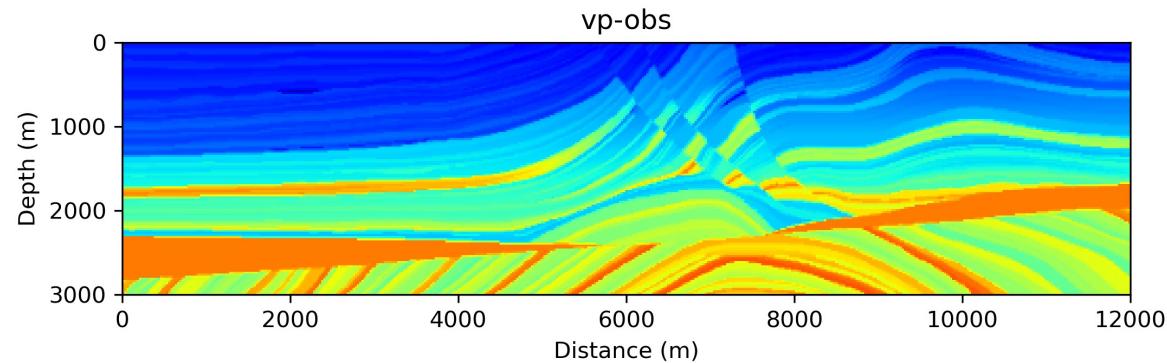
SWIT-SEG/EAGE Overthrust

True model

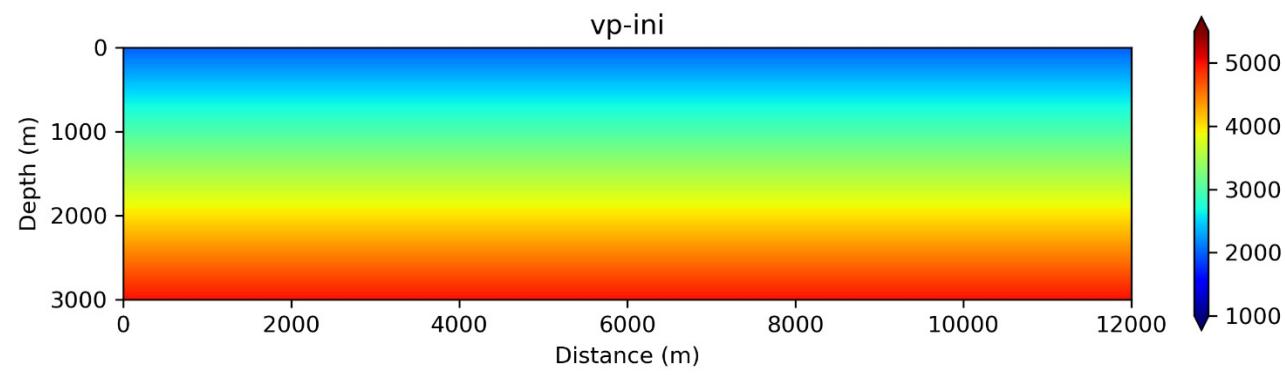


SWIT-From 1D ‘blind’ initial model

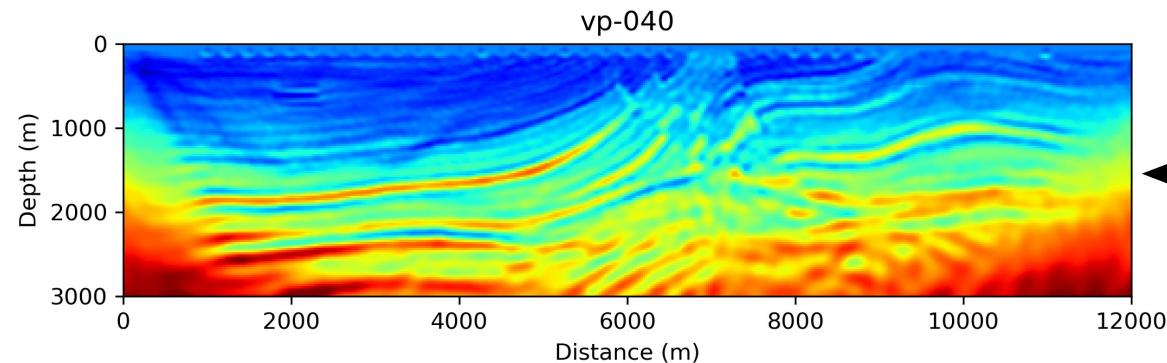
True model



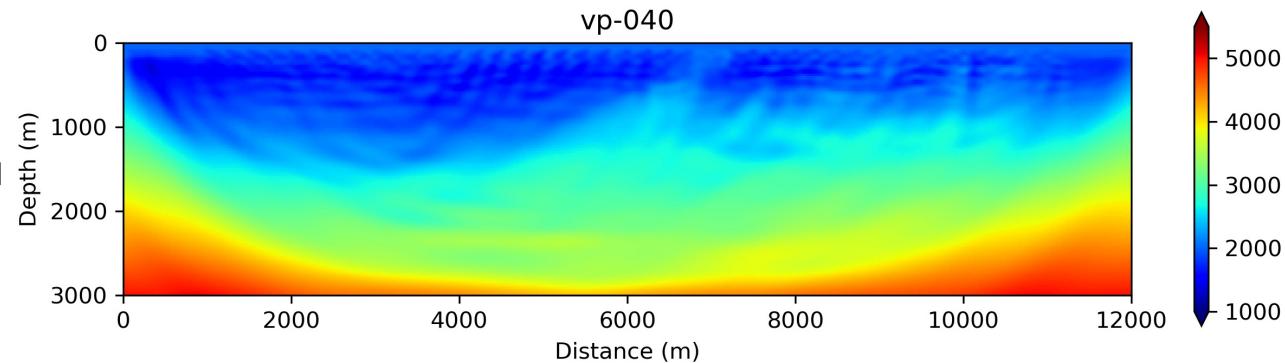
1D model



Waveform-FWI

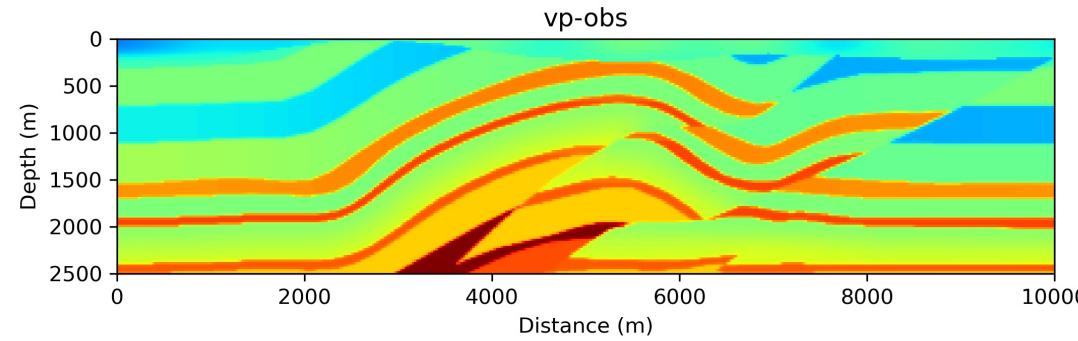


Travelttime-FWI

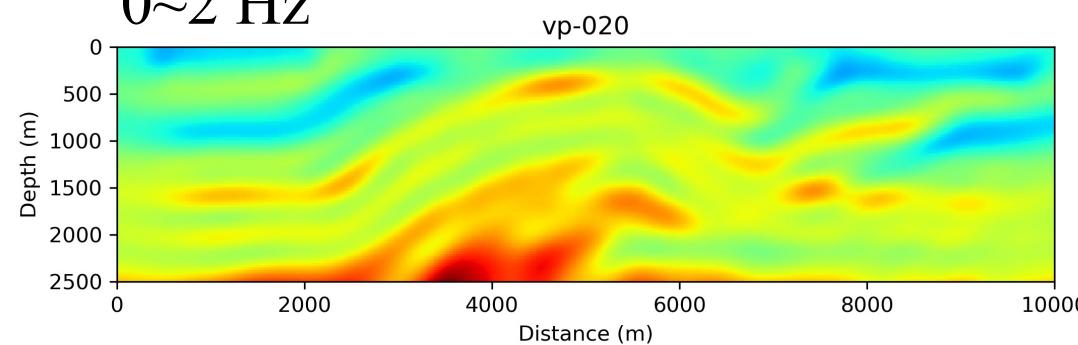


SWIT-Multiscale Inversion

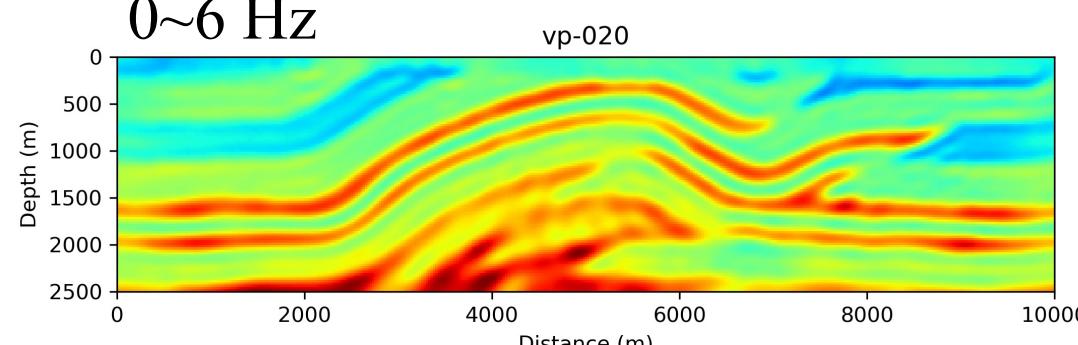
True model



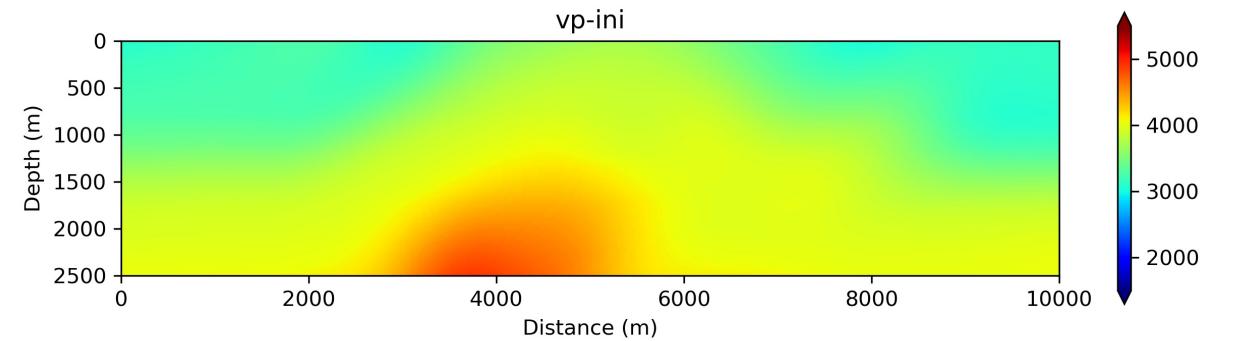
0~2 Hz



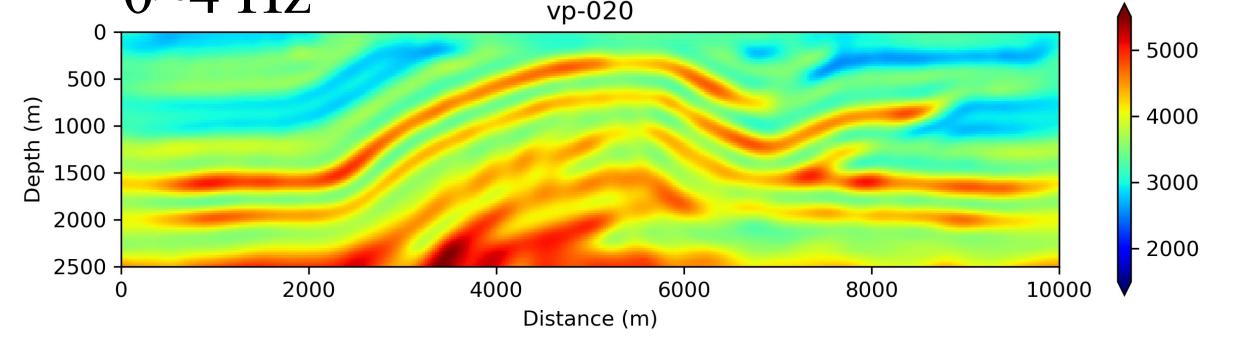
0~6 Hz



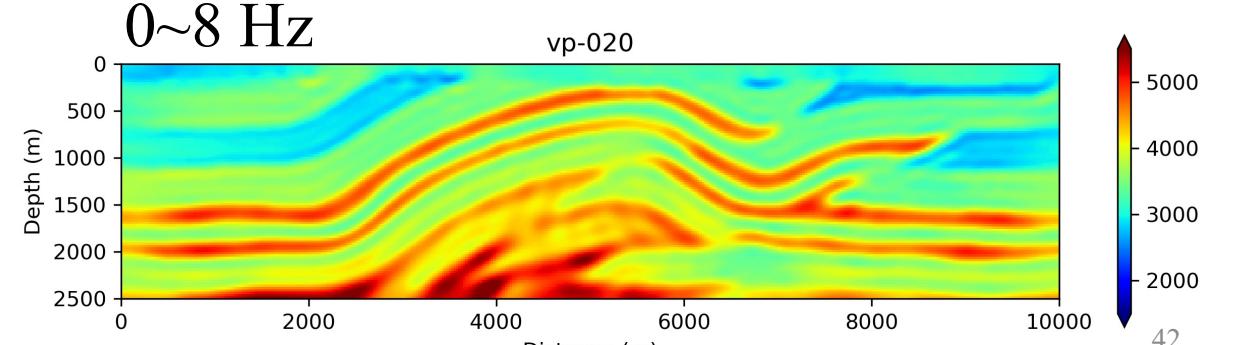
Initial model



0~4 Hz



0~8 Hz



Acknowledgement

- **Computational Toolkit** by Prof. Schuster
- **SeisFlows** by Dr. Ryan Modrak
- Python packages: Numpy, Obspy, Scipy, Matplotlib, PySimpleGUI, etc.
- Prof. Jianping Huang for very helpful discussions at China University of Petroleum-East China
- Dr. Kai Chang, graduate students Zeqiang Chen & Bao Deng at USTC for testing SWIT

References:

1. Li, H., Li, J., Liu, B., Huang, X. (2021). Application of full-waveform tomography on deep seismic profiling dataset for tectonic fault characterization. *International Meeting for Applied Geoscience & Energy*.
2. Schuster, G. T. (2017). Seismic inversion. *Society of Exploration Geophysicists*.

Thanks

Haipeng Li (李海鹏) (haipengl@mail.ustc.edu.cn)

Supervisor : Prof. Junlun Li (李俊伦) (lijunlun@ustc.edu.cn)

USTC

Aug. 2021

Few more words

I always believe that *Simplicity is the Greatest Virtue Ever*, which I try to obey in conceiving SWIT.

The *WIT of Seismic waveforms* has never been endowed. Rather, it lies within.

SWIT is also dedicated to my favorite singer-songwriter, *Taylor Swift*.