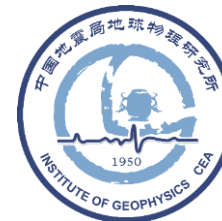


An Earthquake Detection and Location Architecture for Continuous Seismograms: Phase Picking, Association, Location, and Matched Filter (PALM)

speaker: Yijian ZHOU

Lihua FANG, Han YUE, Shiyong ZHOU, Abhijit GHOSH

contact info: yijian.zhou@email.ucr.edu



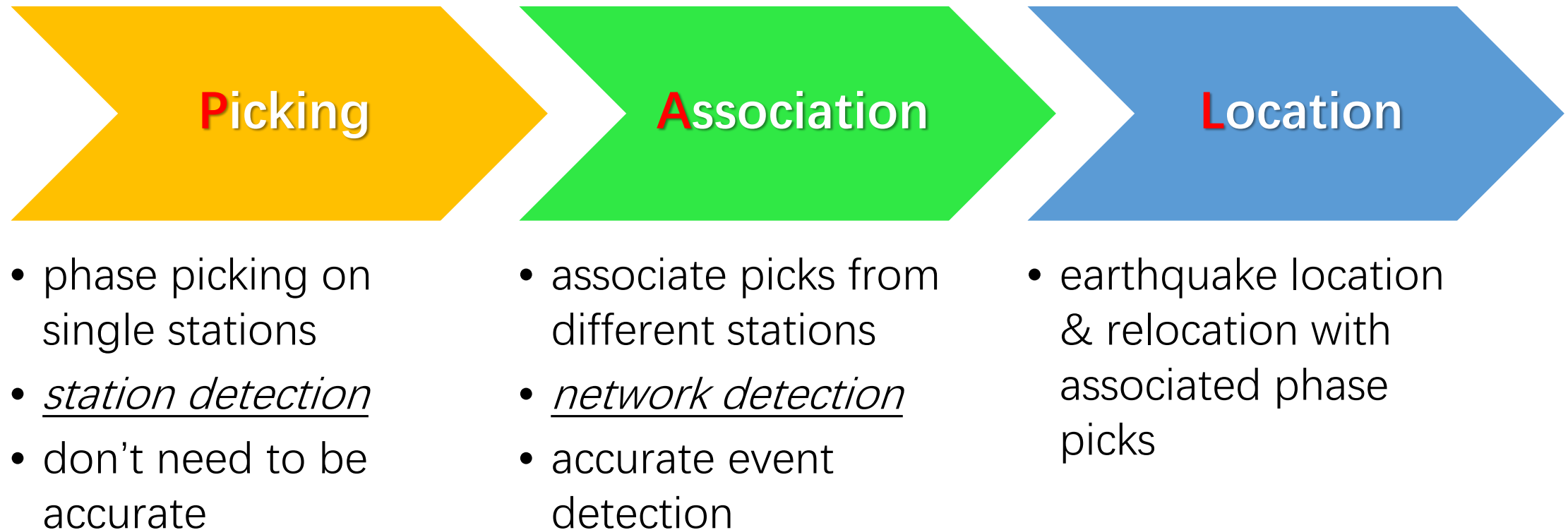
Outline

- PAL: phase picking, association, and location
- MESS: a new matched filter detector
- Apply PALM to build high-resolution catalog

Outline

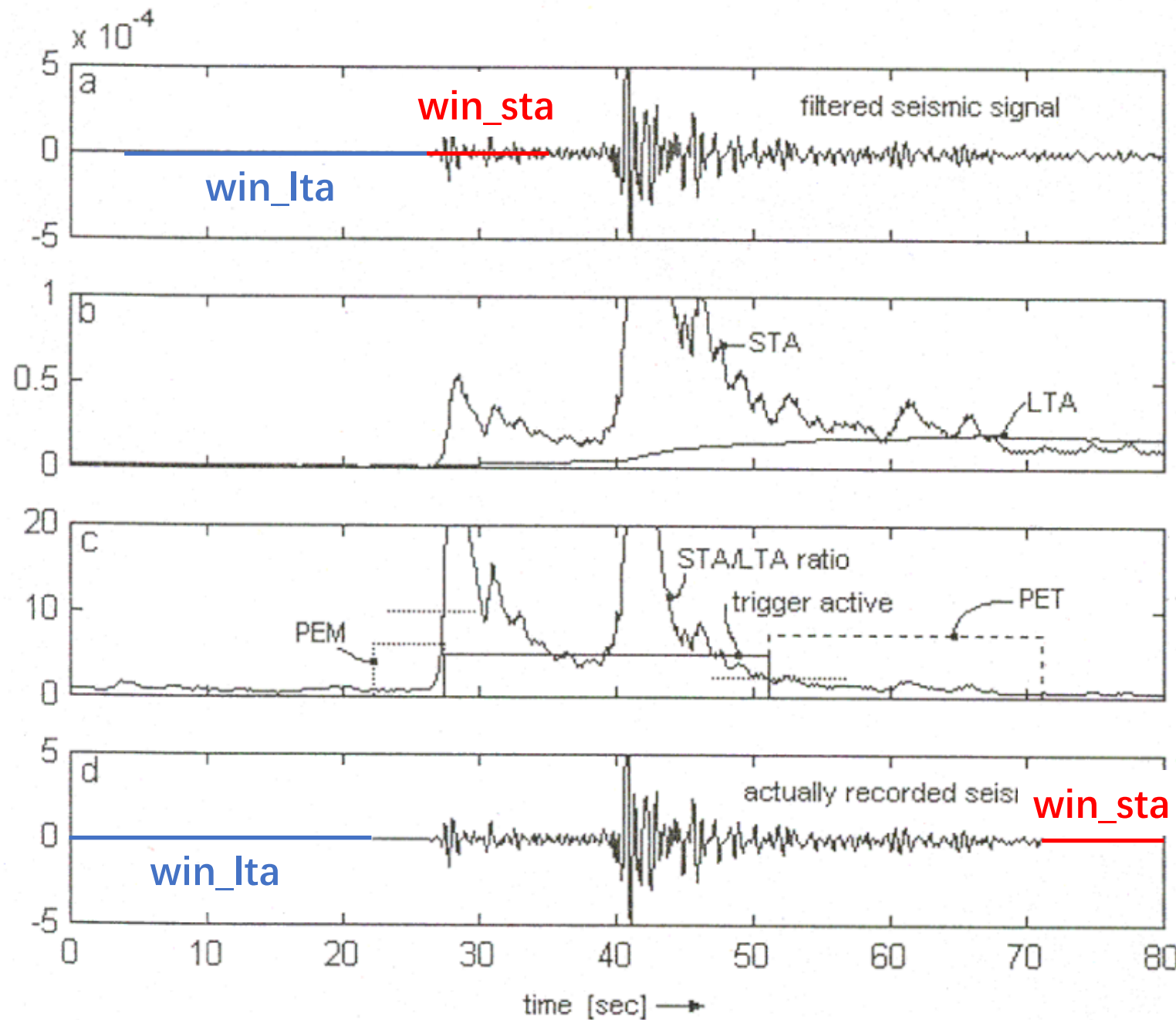
- **PAL: phase picking, association, and location**
- MESS: a new matched filter detector
- Apply PALM to build high-resolution catalog

Picking-based Workflow



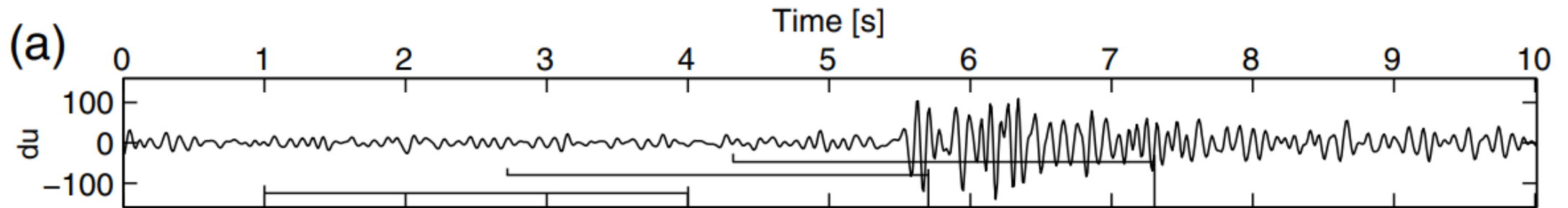
Phase Picking Algorithms

	STA/LTA	Kurtosis	AI-picker
Accuracy	<u><i>prediction < target</i></u>	<u><i>prediction > target</i></u>	prediction ~ target
Precision	<u><i>low</i></u>	high	high
Efficiency	high	<u><i>low</i></u>	high
Generalizability	high	high	<u><i>low</i></u>

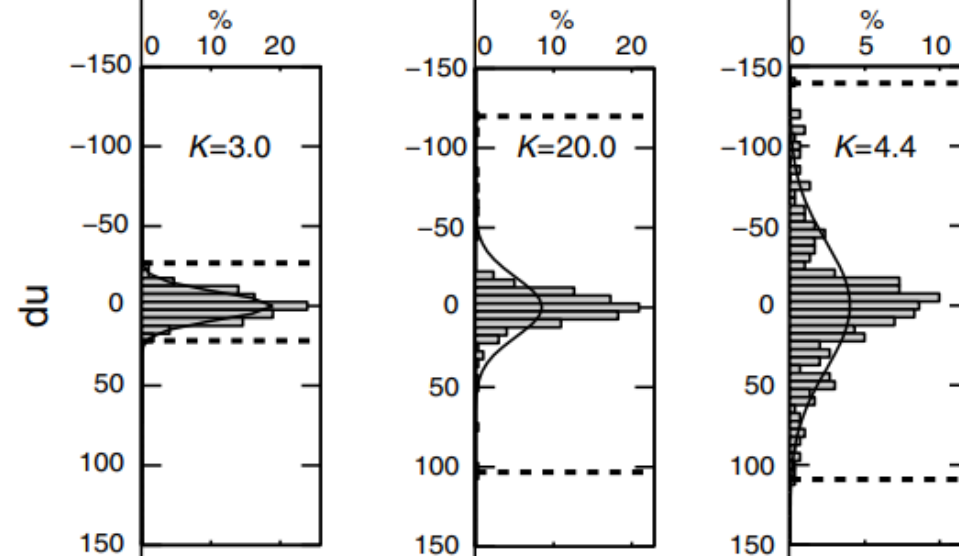


**STA/LTA
picks earlier
than real
phase arrival**

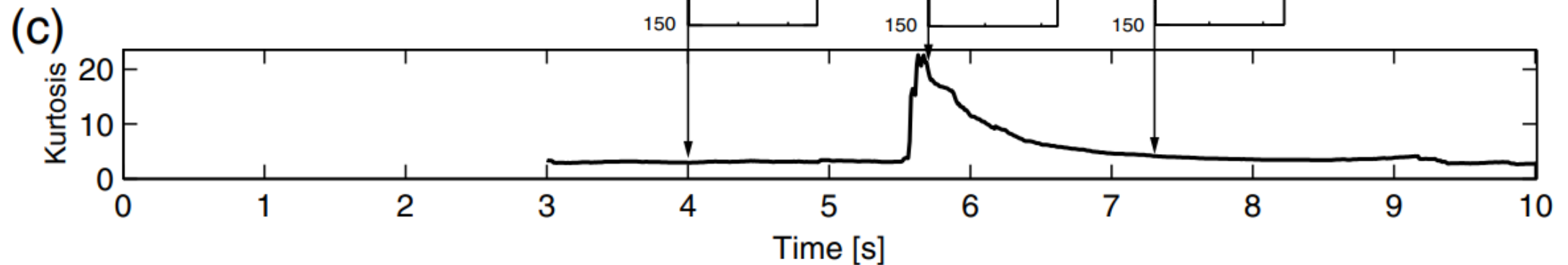
Trnkoczy,
NMSOP
1999



(b)

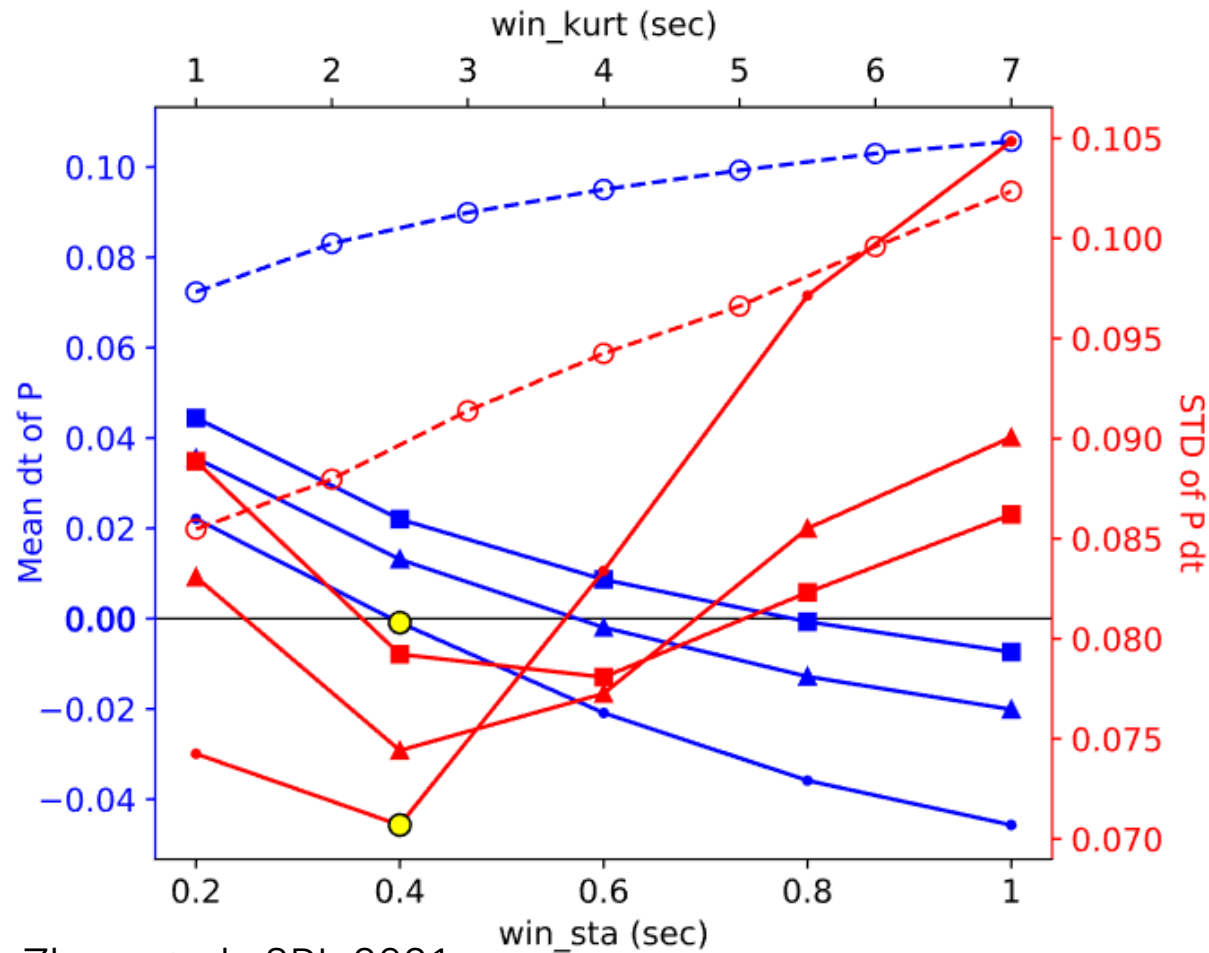


**Kurtosis
picks later
than real
phase arrival**



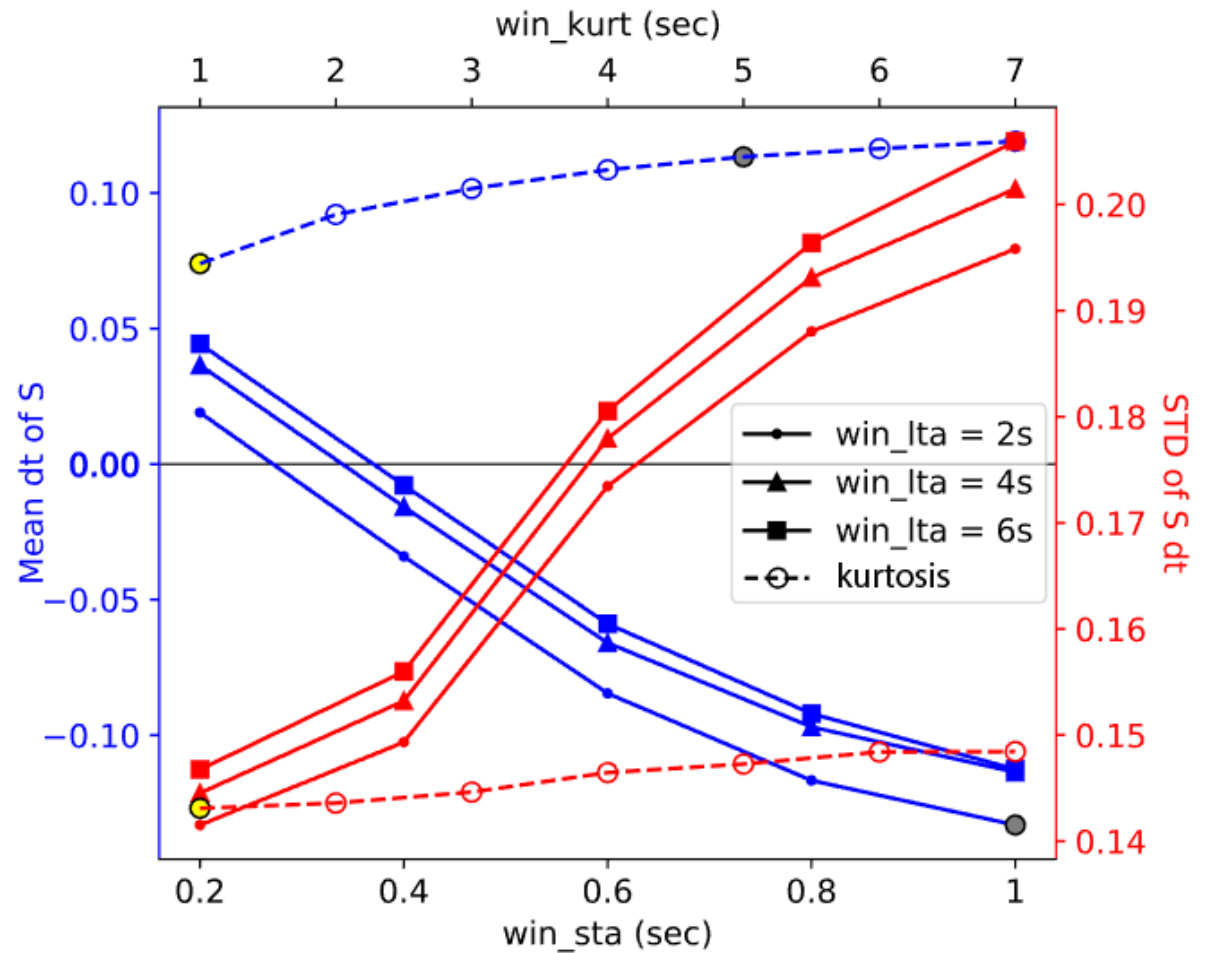
Characteristics of kurtosis:
 sensitive to amplitude change
 insensitive to frequency and
 phase change
 insensitive to window length

Test on Ridgecrest: new insights on STA/LTA & Kurtosis



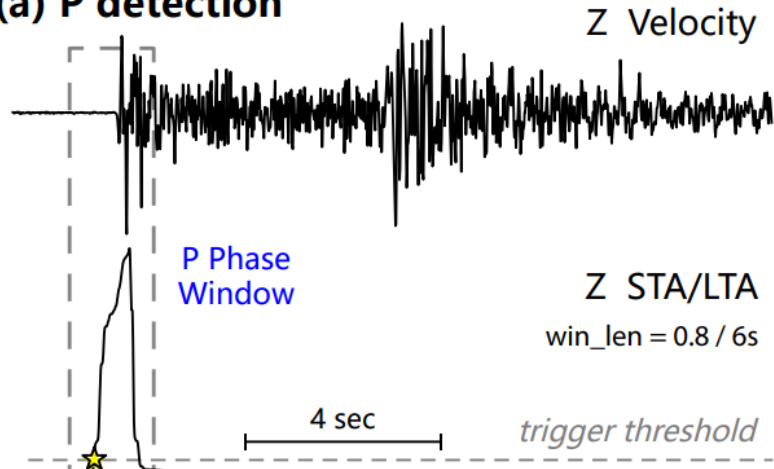
Zhou et al., SRL 2021

1. STA/LTA is **suitable** for high-SNR P wave

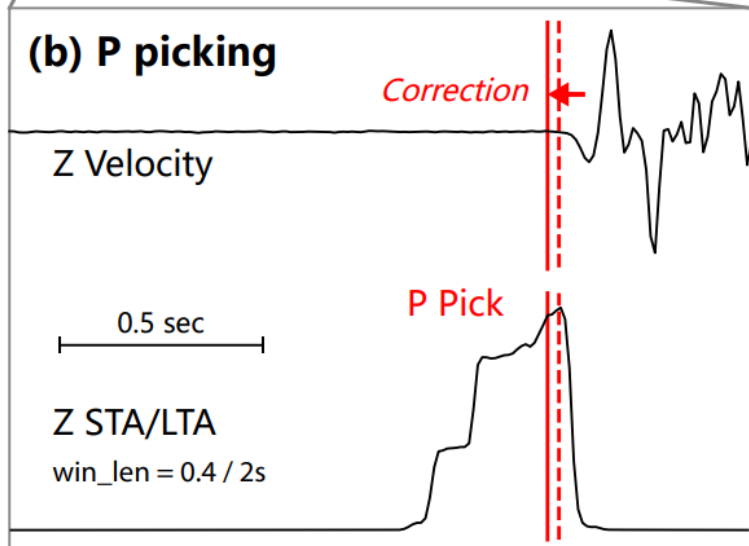


2. STA/LTA & Kurtosis provide a constraint for S
3. Kurtosis is **stable** for low-SNR S wave

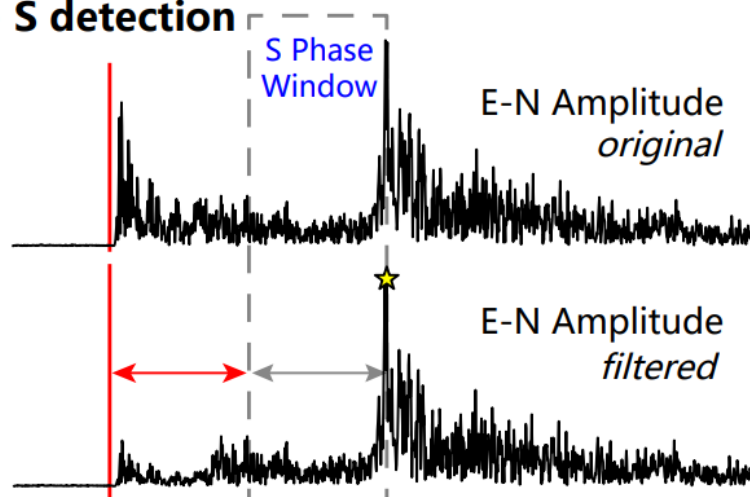
(a) P detection



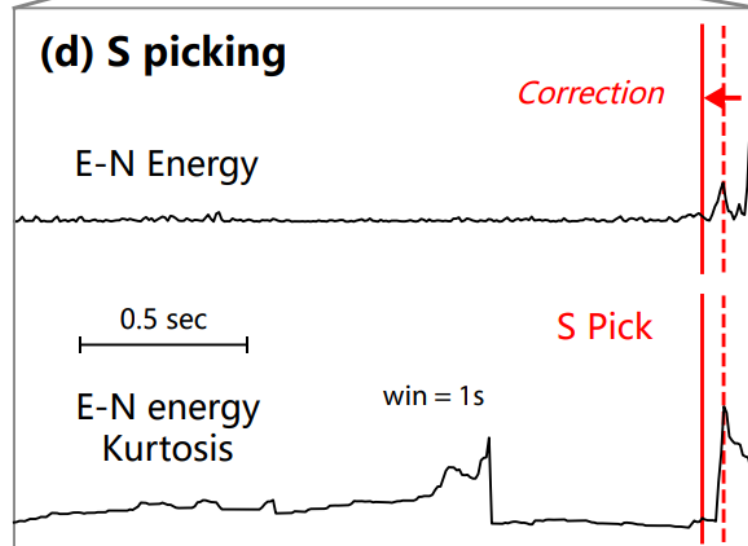
(b) P picking



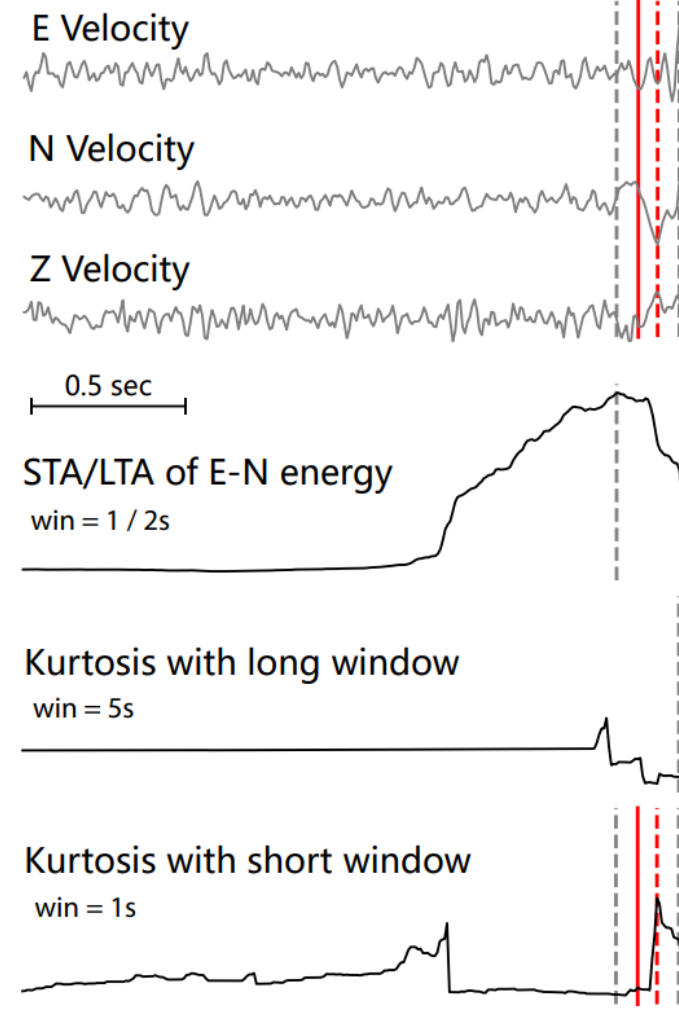
(c) S detection

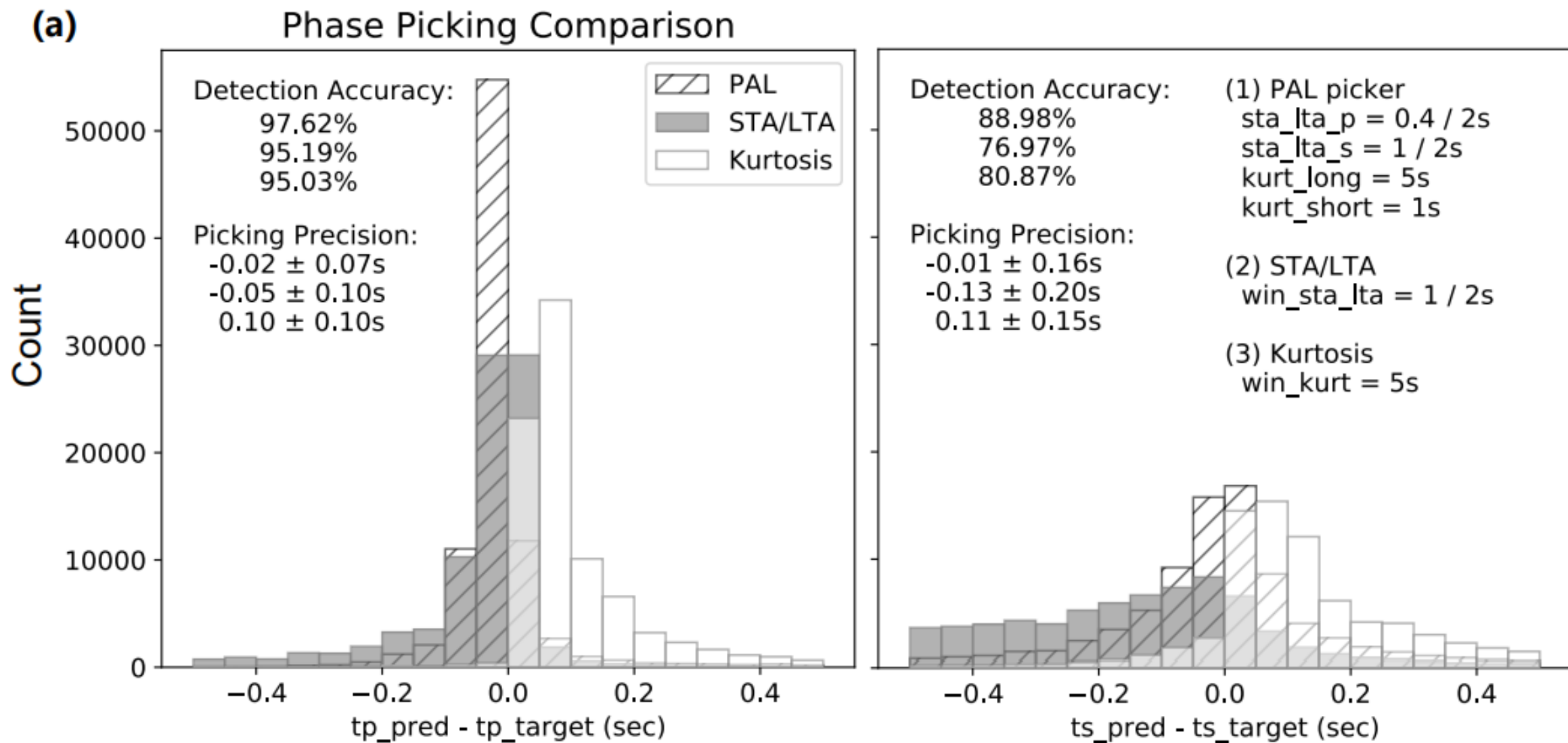


(d) S picking

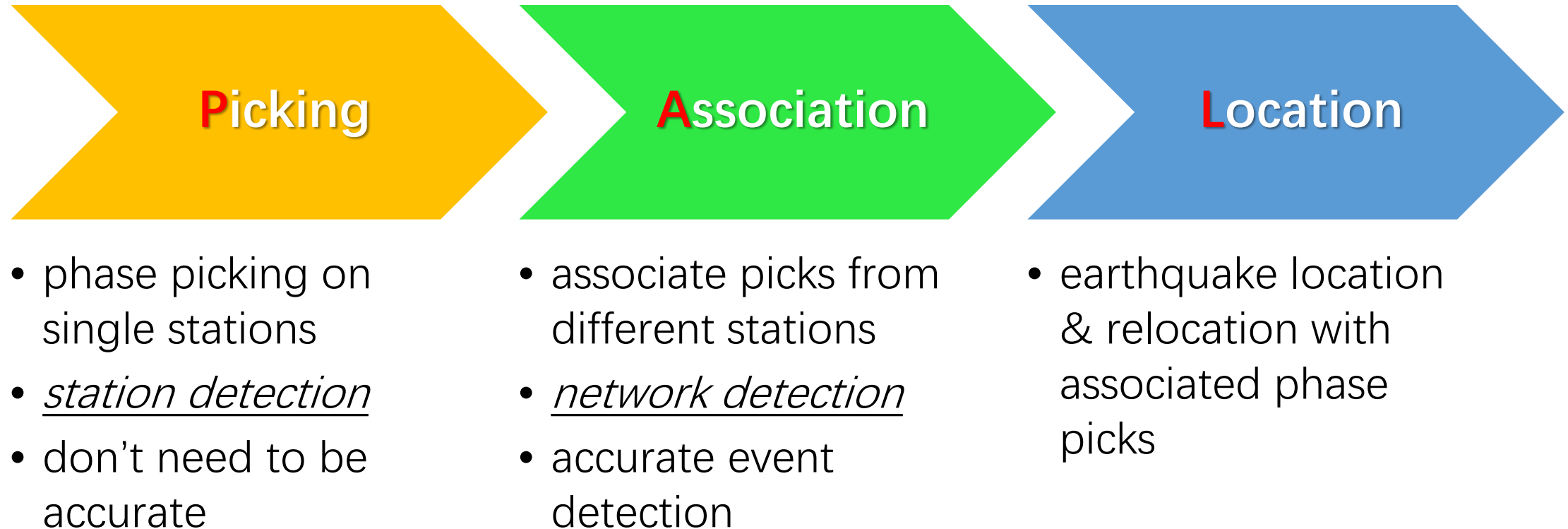


(e) S picking (in detail)



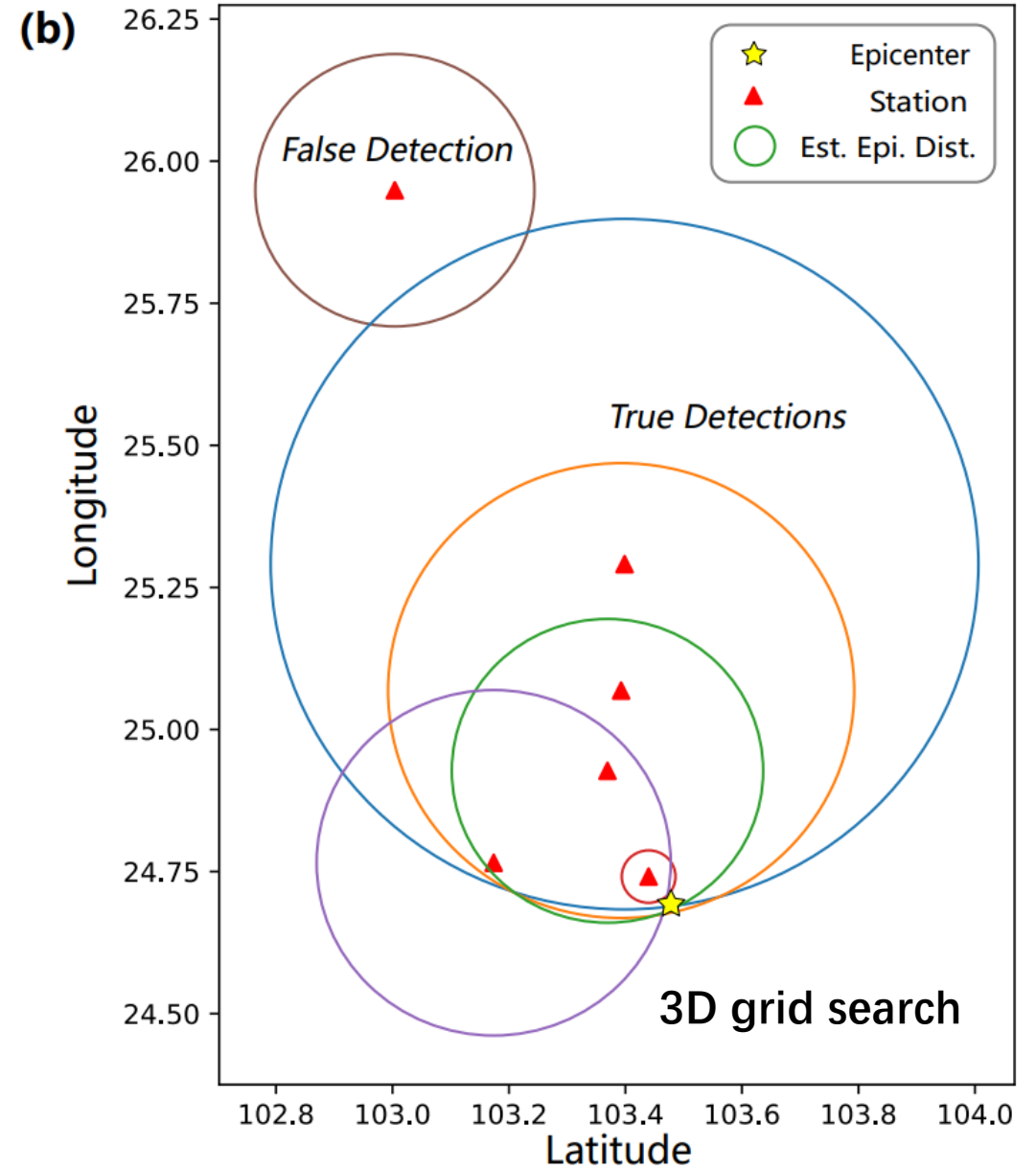
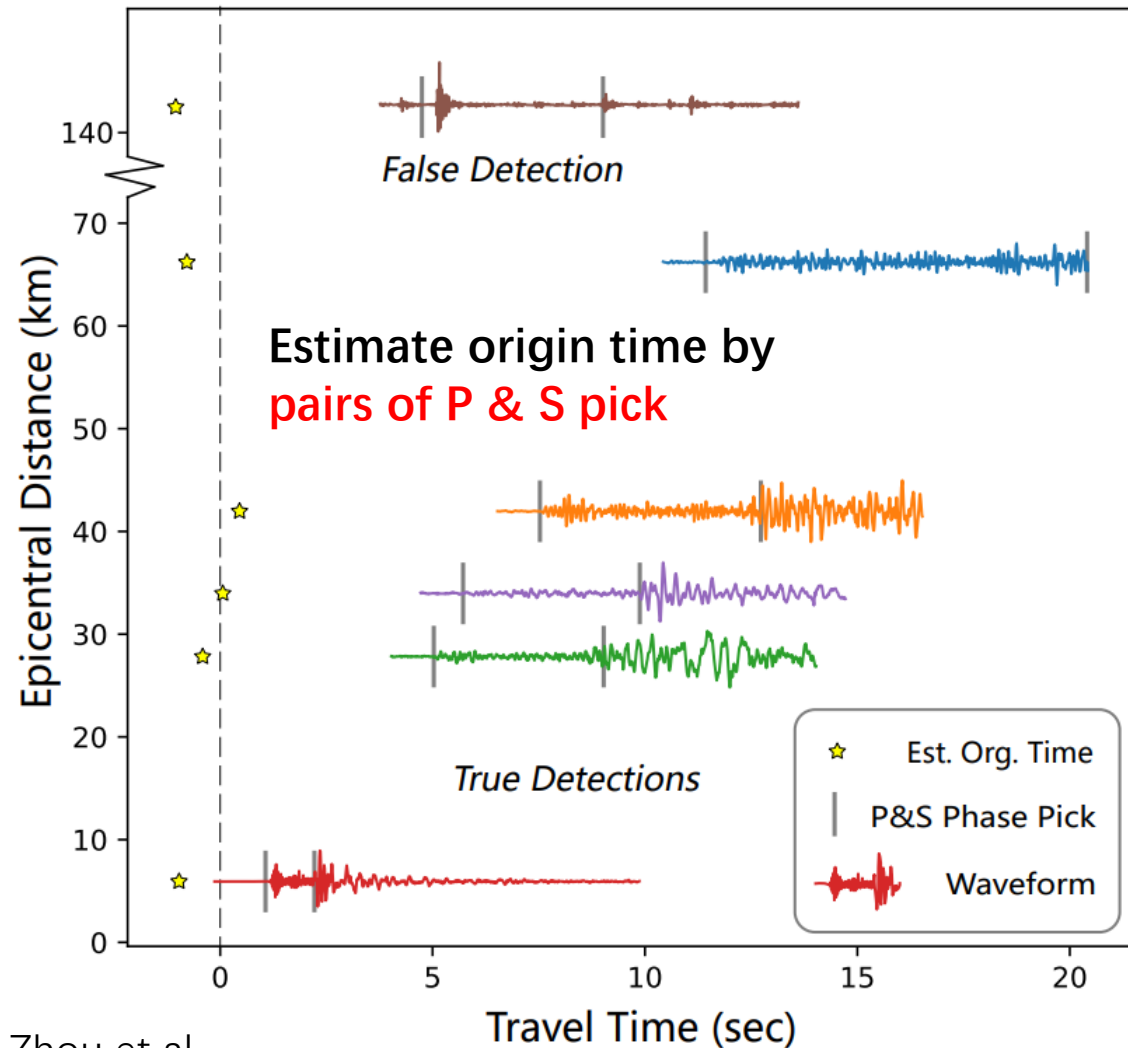


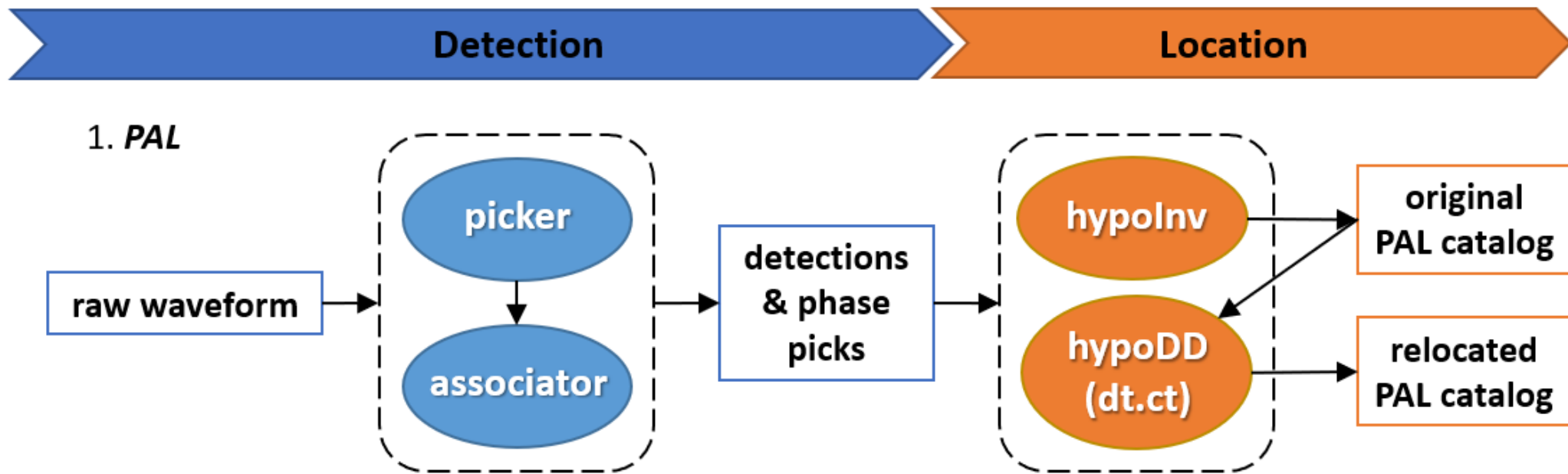
Picking-based Workflow



Waiting for tests on associators!

Event 20160913103607.17

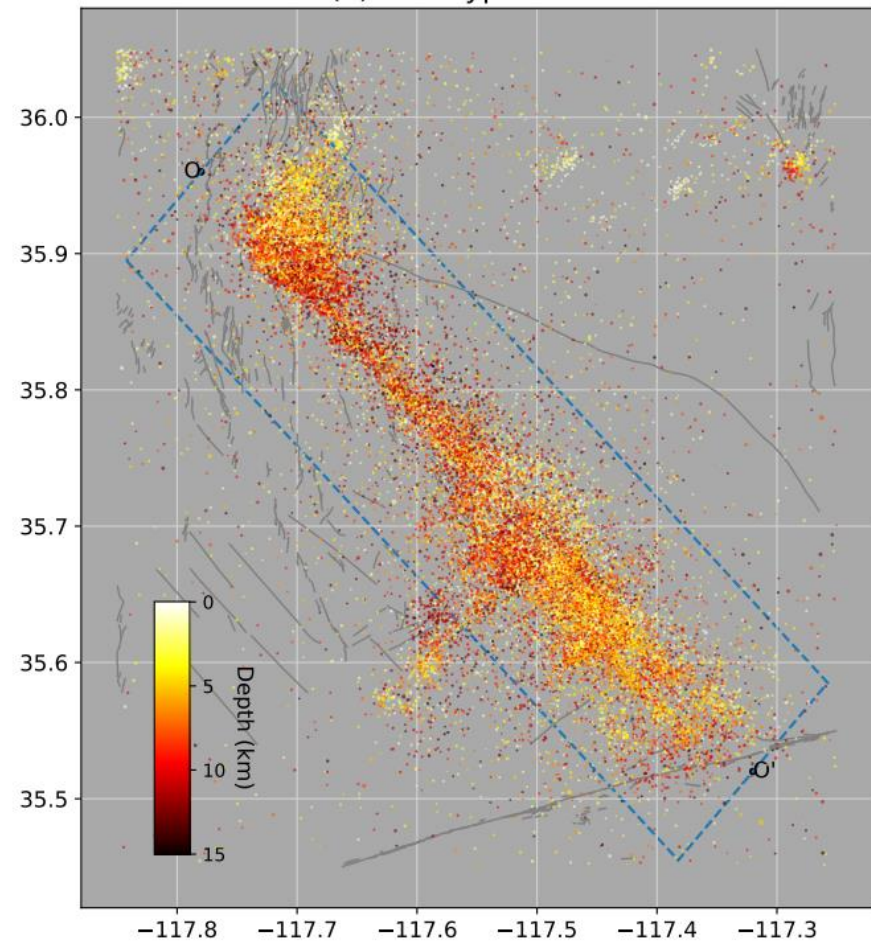




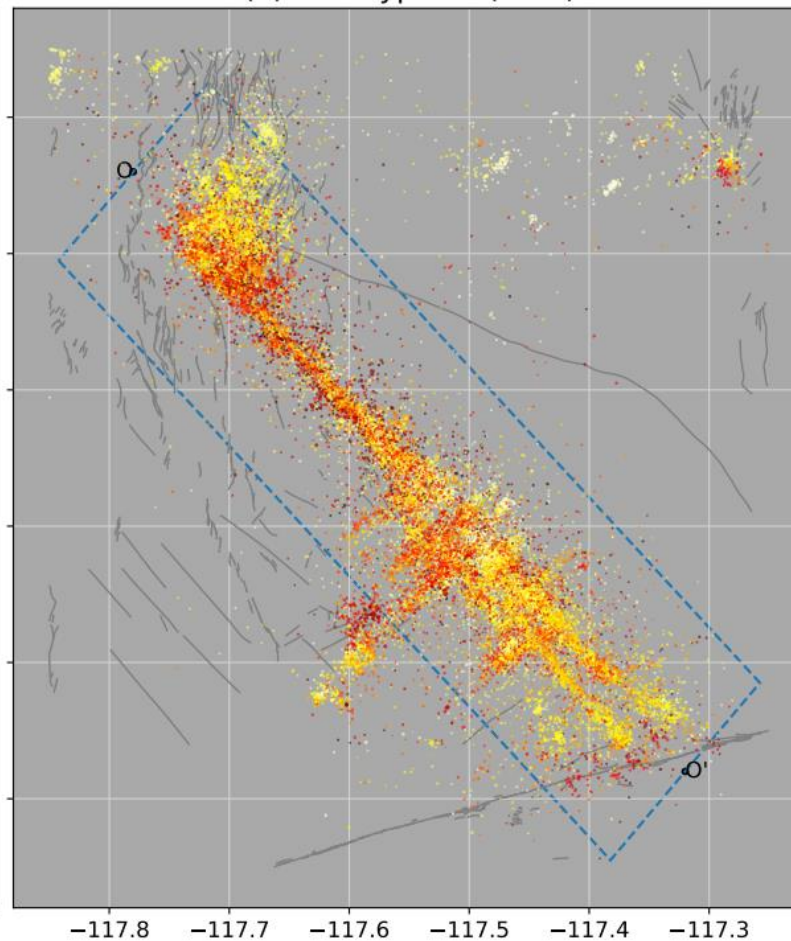
<https://github.com/YijianZhou/PAL>

Zhou et al., SRL 2021

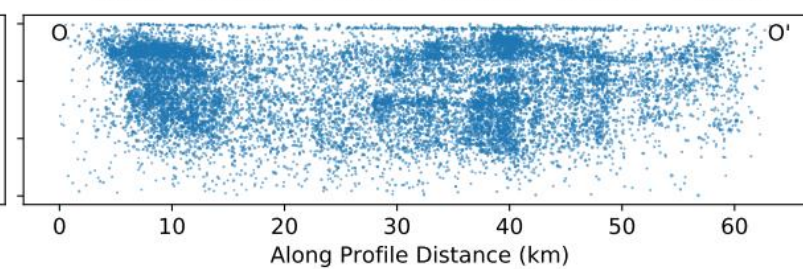
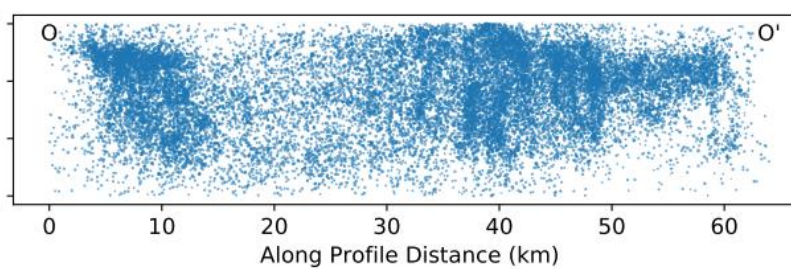
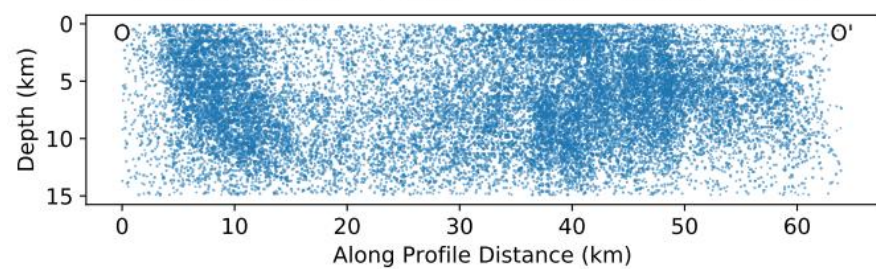
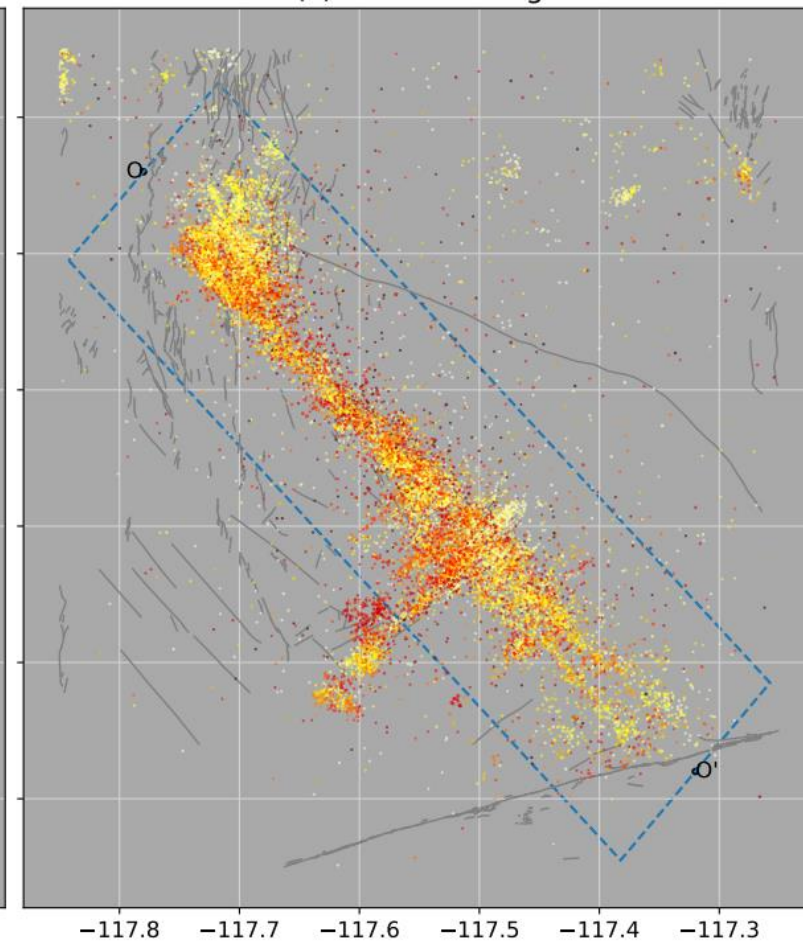
(a) PAL HypoInverse



(b) PAL HypoDD (dt.ct)



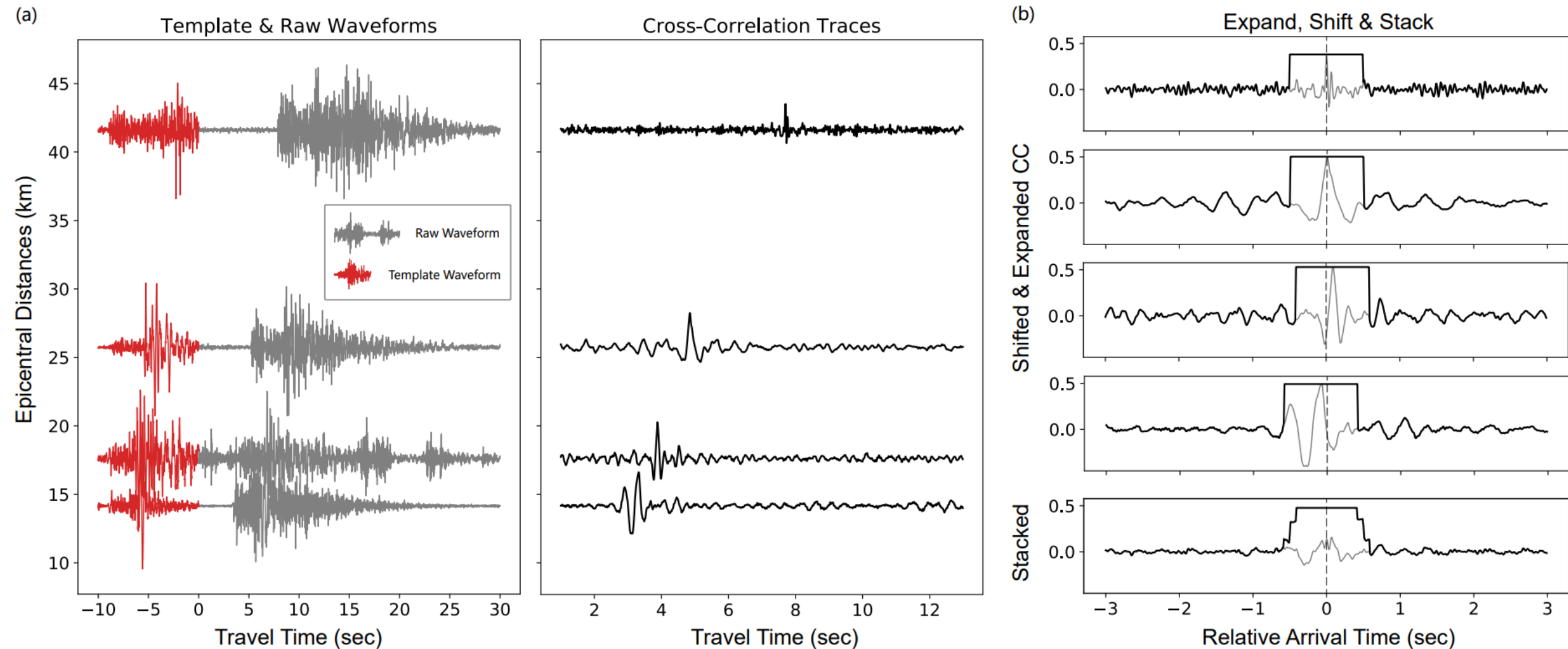
(c) SCSN Catalog



Outline

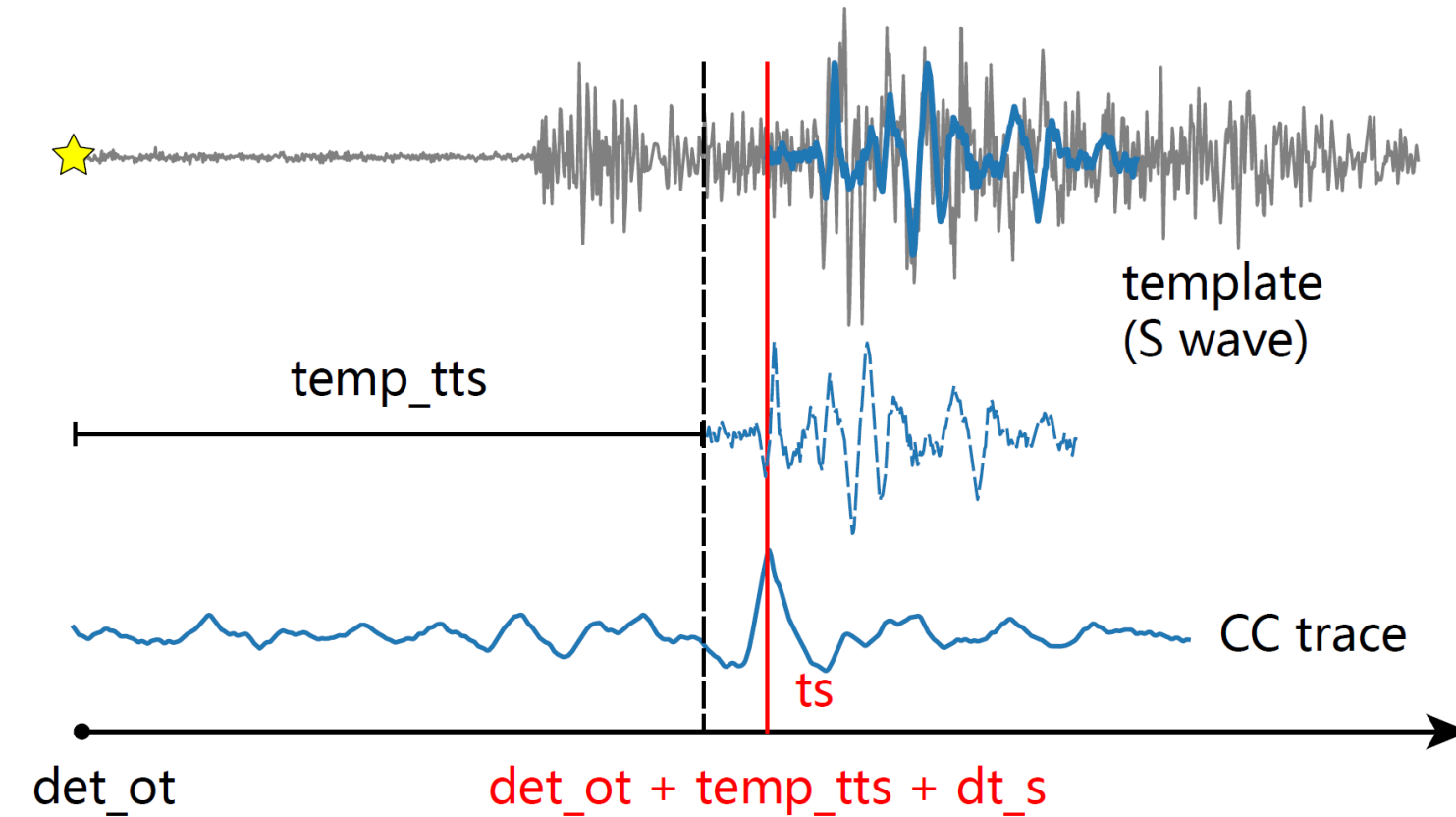
- PAL: phase picking, association, and location
- **MESS: a new matched filter detector**
- Apply PALM to build high-resolution catalog

MESS: Matched filter + Expand + Shift + Stack



Zhou et al., SRL 2021

CC Measurement of Differential Travel Time



template event - $temp$
newly detected event - det

0. Detect on det_ot
on stacked CC trace
1. Pick ts by CC
 $tts = ts - det_ot$
2. Travel time difference
 $dt_s = tts - temp_tts$

Note 1. error in det_ot does not matters.

For multi-stations:

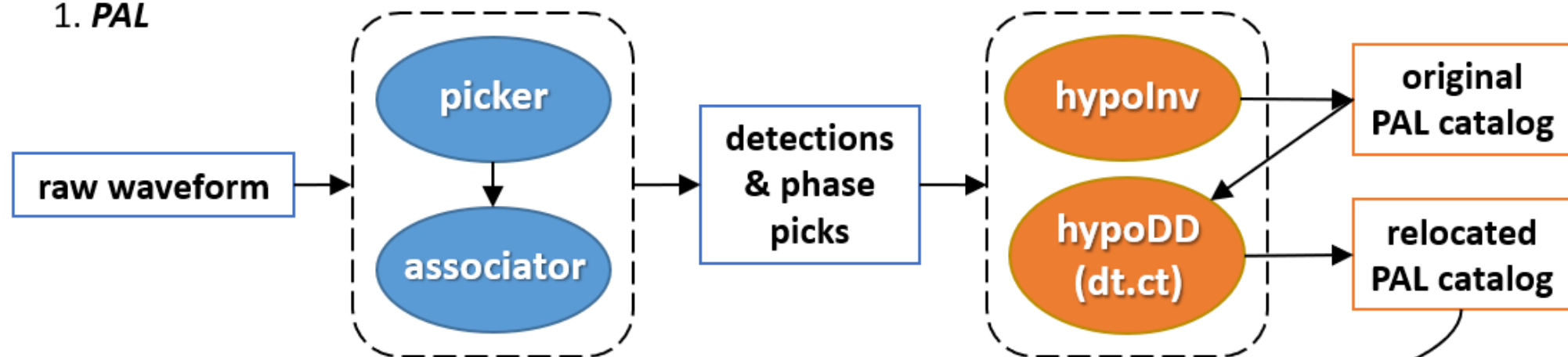
$[dt_s] = [ts - det_ot] - temp_tts$,
where det_ot is to be relocated

Note 2. picking error does not effect the dt measurement.

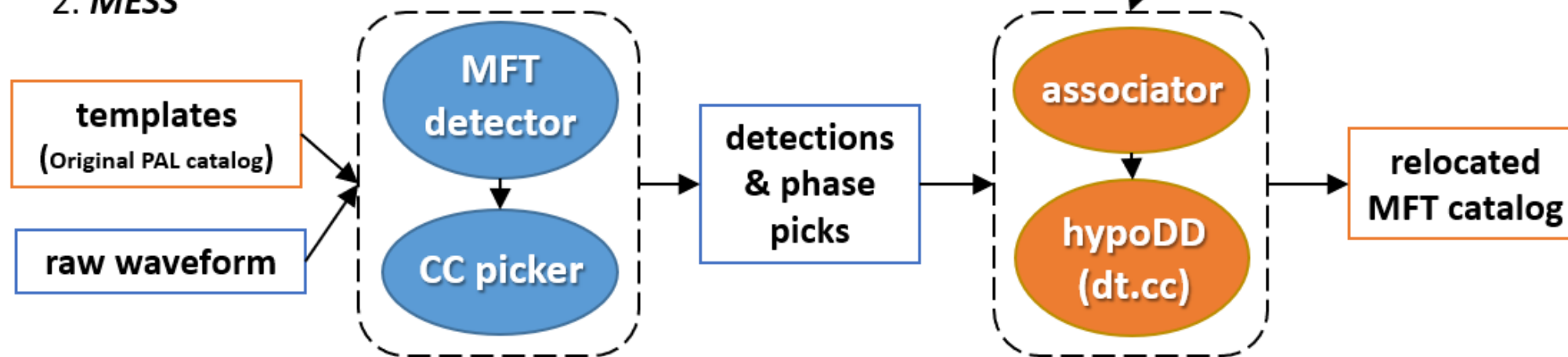
ts and $temp_tts$ share the same error



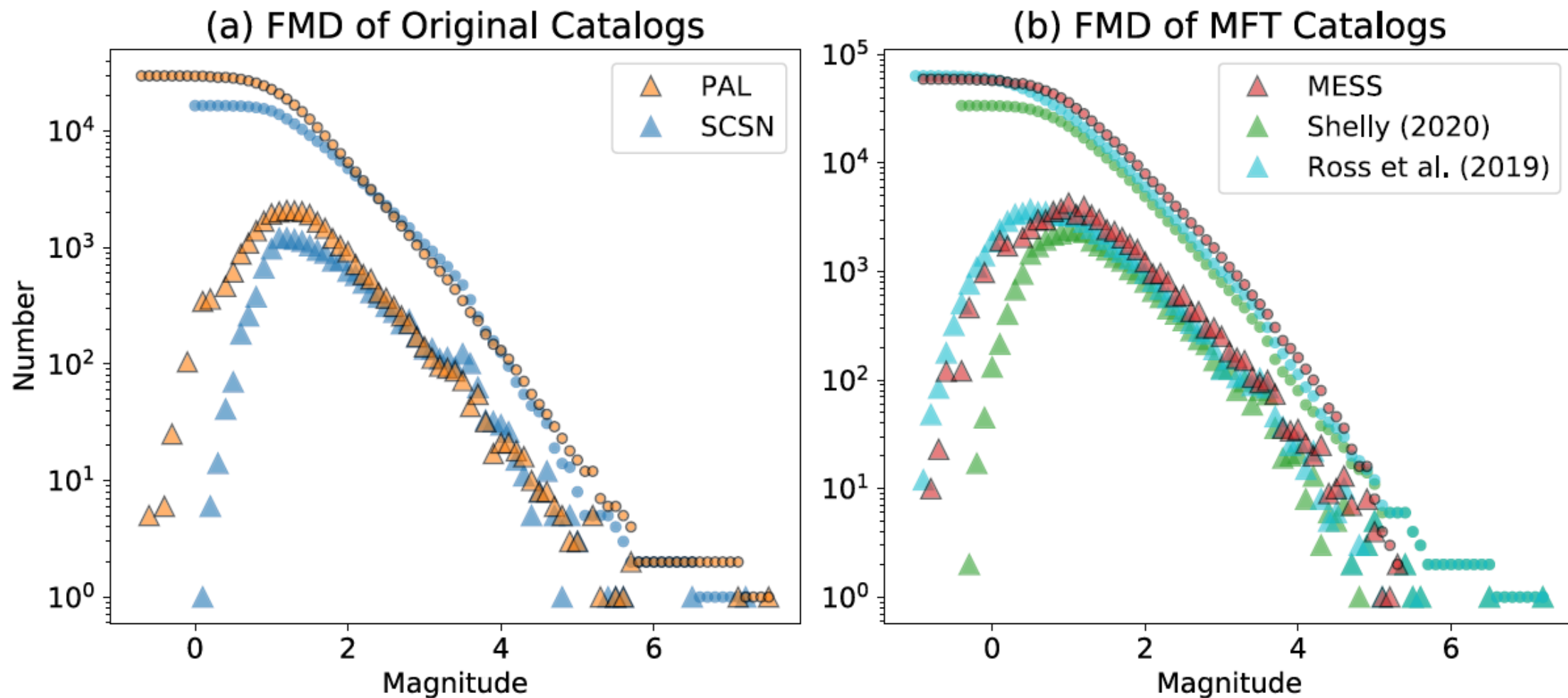
1. PAL



2. MESS

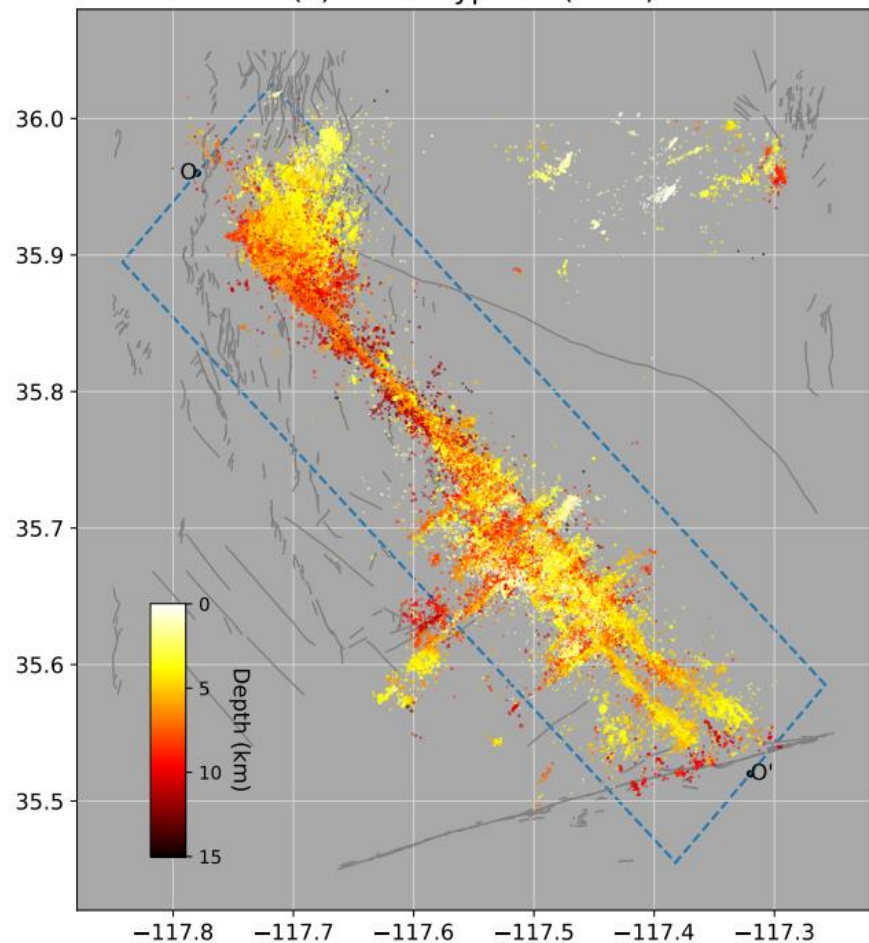


Test on 2019 Ridgecrest Aftershock

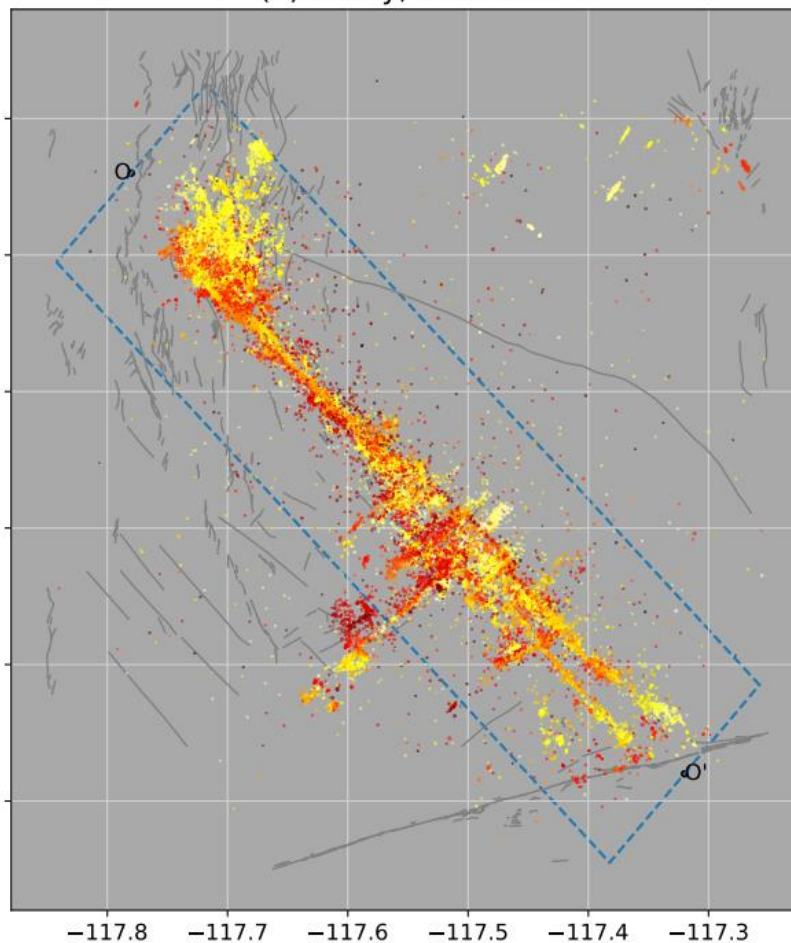


Zhou et al., SRL 2021

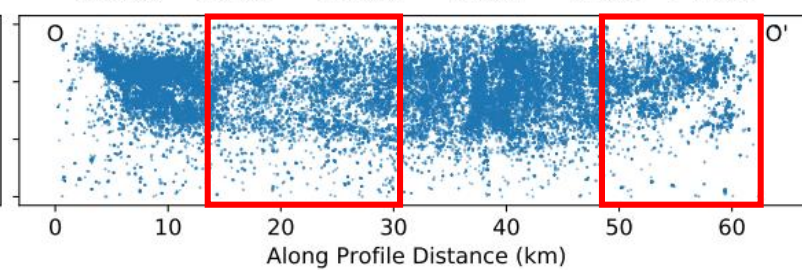
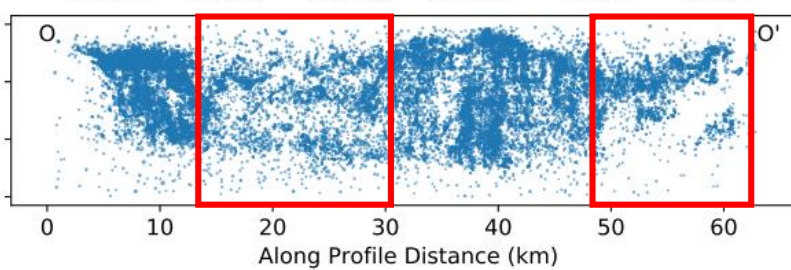
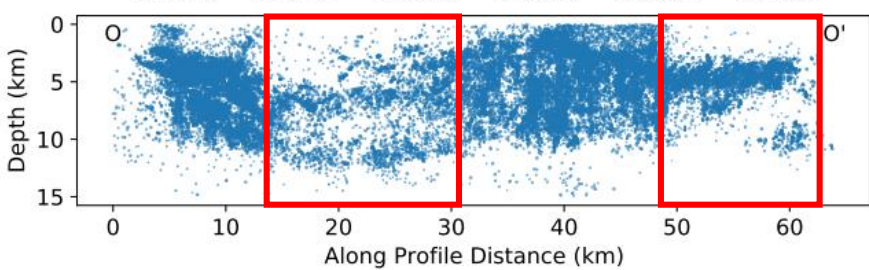
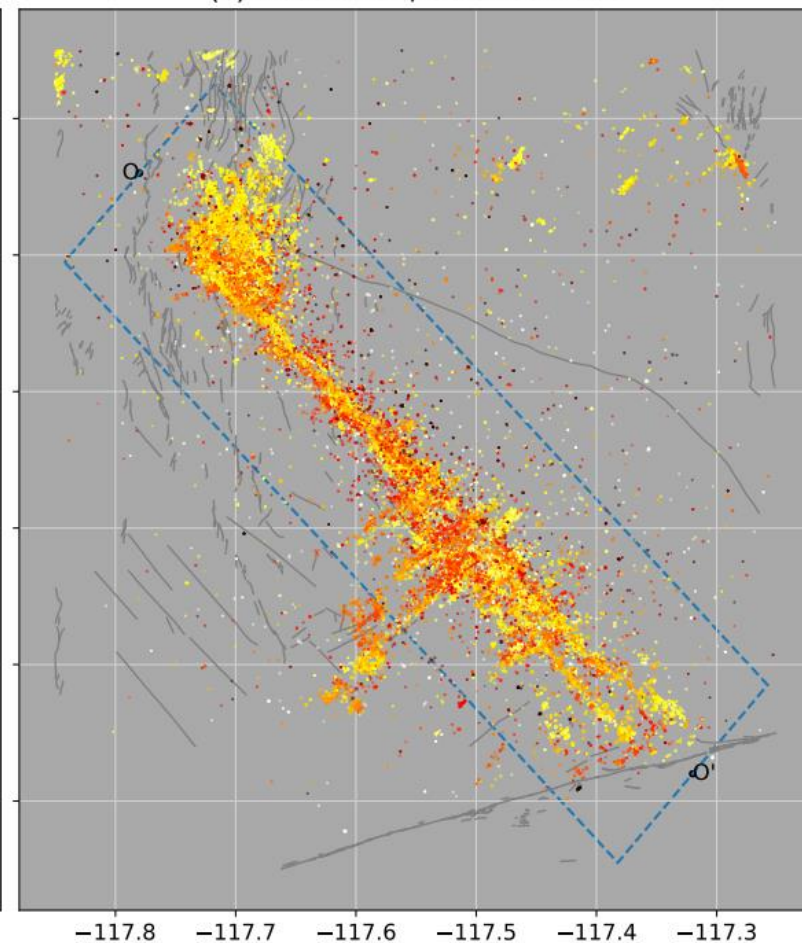
(a) MESS HypoDD (dt.cc)



(b) Shelly, SRL 2020



(c) Ross et al., Science 2019

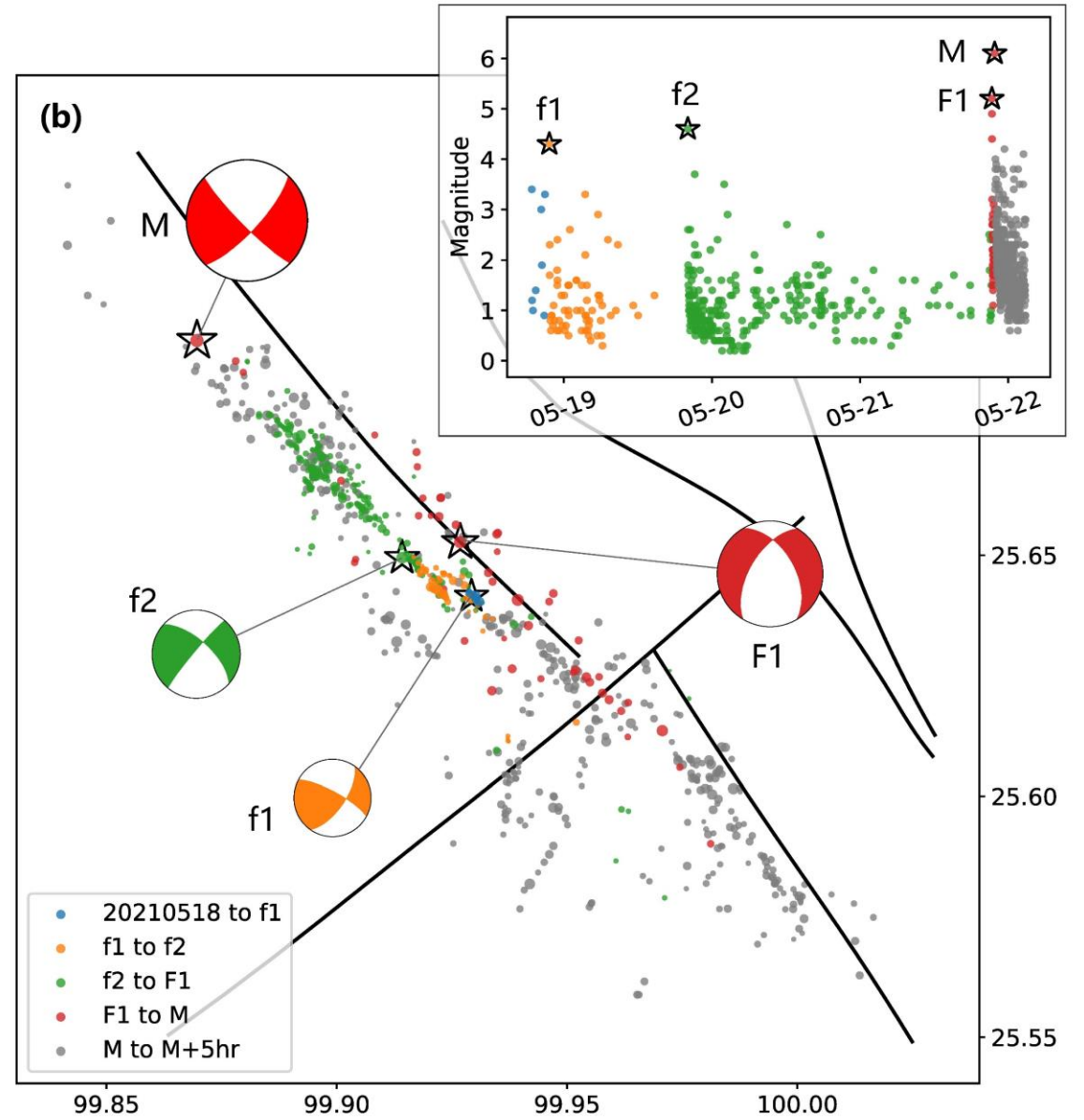
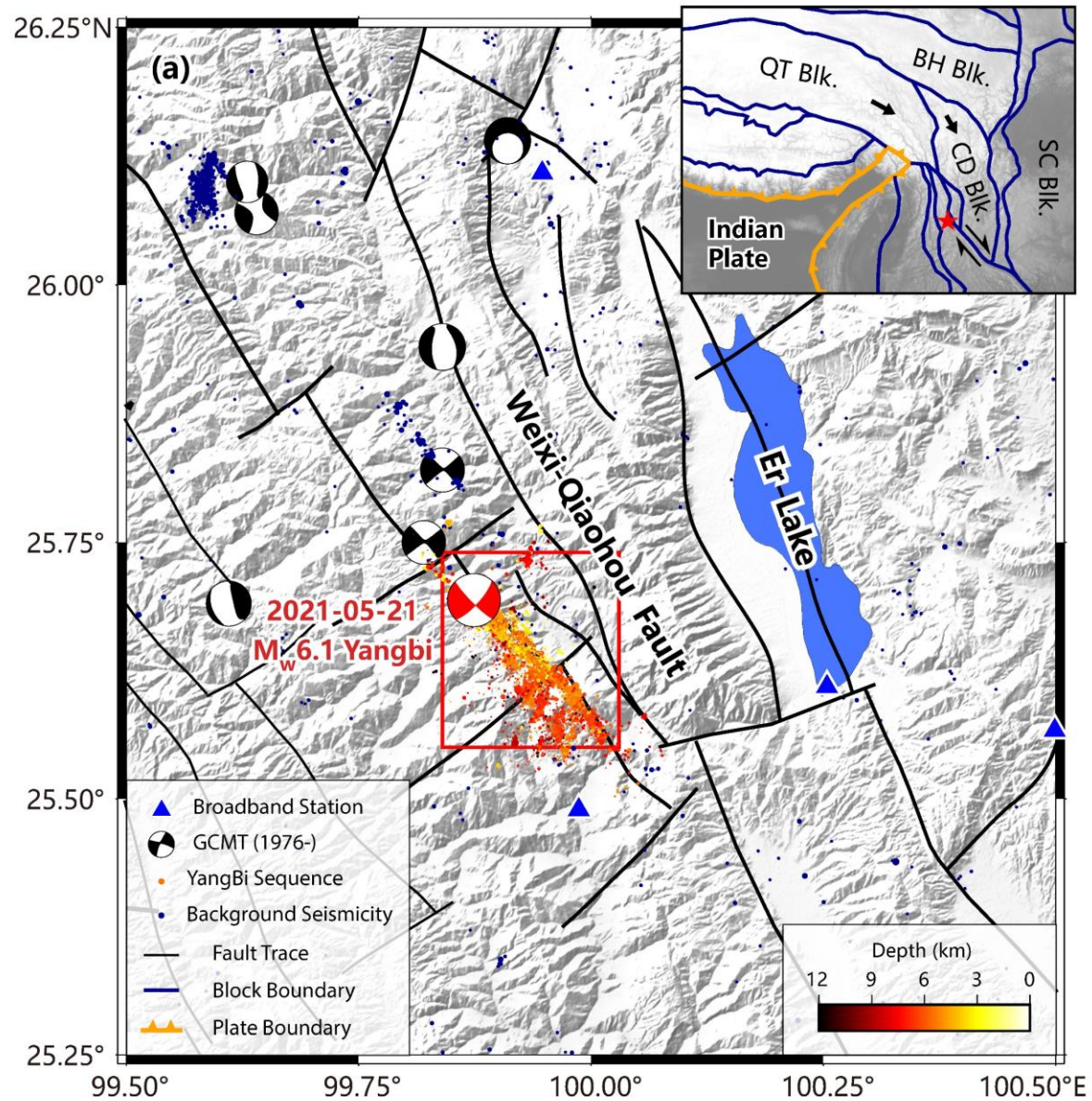


Repeater Detection

- Repeaters indicate aseismic fault slip (Uchida & Burgmann, 2019)
- This helper implement the repeater detection method in Zhou et al. (2022), which utilize both waveform similarity and location:
 - Waveform similarity is measured by CC of long window, covering both P & S waves
 - The location separation is constraint by $\Delta(S-P)$, which is measure by CC of short windows separately for P & S.
 - A strict detection of repeating earthquakes would adopt parameters like (1) average $CC > 0.9$, and (2) $\Delta(S-P) \leq 0.01s$ for at least 3 stations, under a proper frequency band (Uchida, 2019)

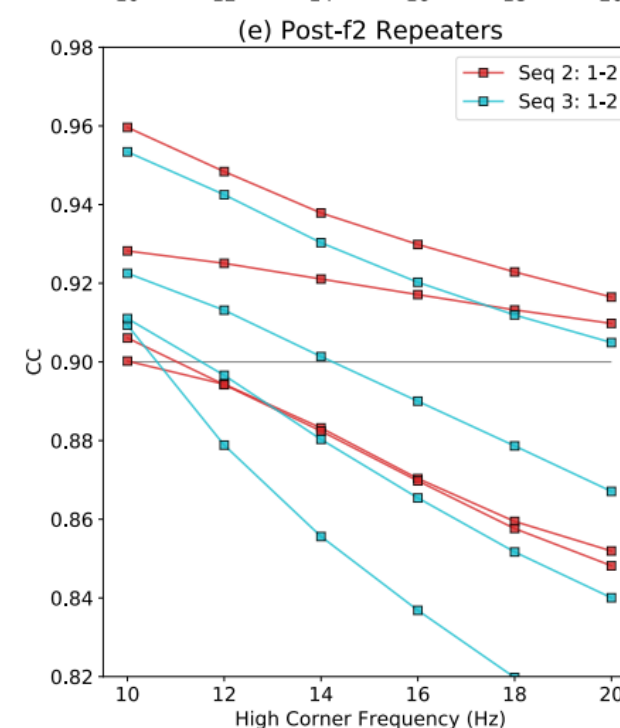
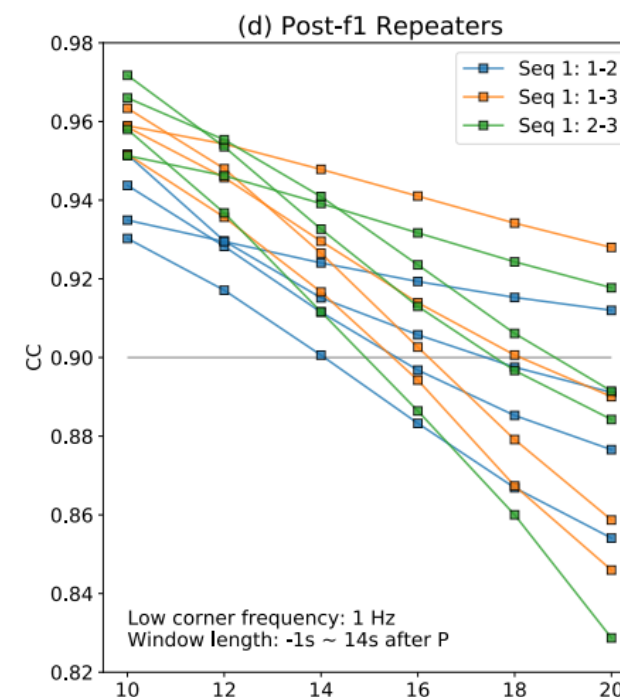
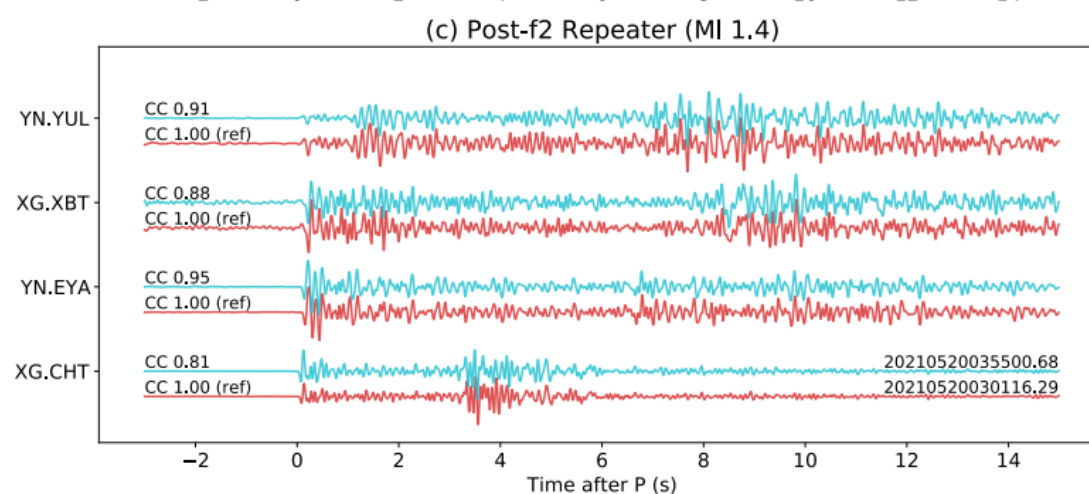
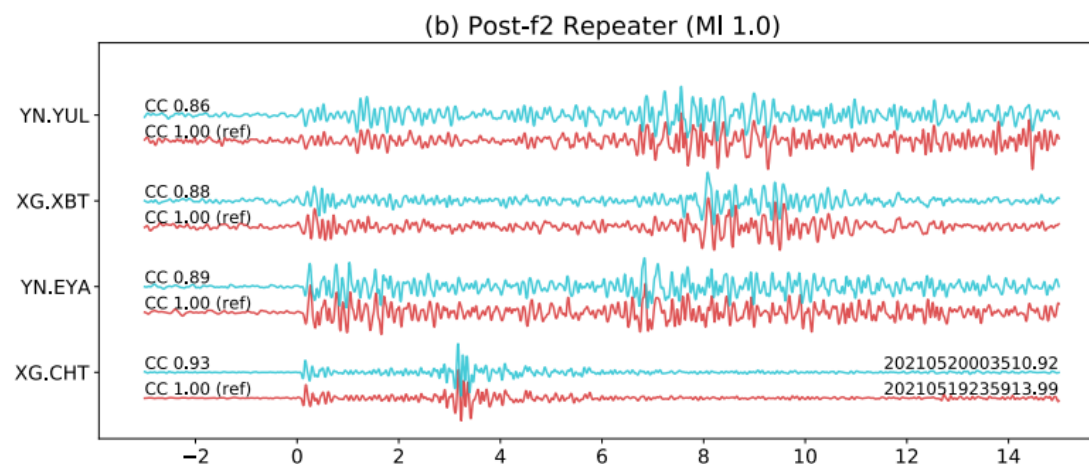
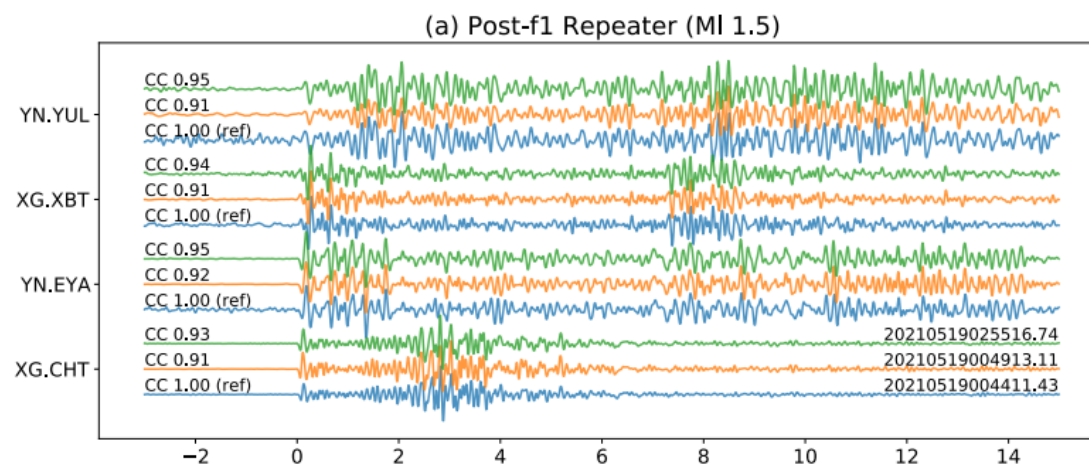
Input	Operation	Output	Notes
fpha_pal & fsta	run MESS	eg_mess.pha eg_mess_cc.ctlg	i.e. run PALM
eg_mess.pha	<i>run_clustering.py</i>	eg_rep-org_full.pha	find repeater candidates in MESS detections
eg_rep-org_full.pha	run MESS	eg_mess-rep.pha	detect with high CC threshold, e.g. 0.8
eg_mess-rep.pha	<i>run_clustering.py</i>	eg_rep.clust eg_rep.pha	using strict criteria, e.g. cc>0.9, dt_sp≤0.01s

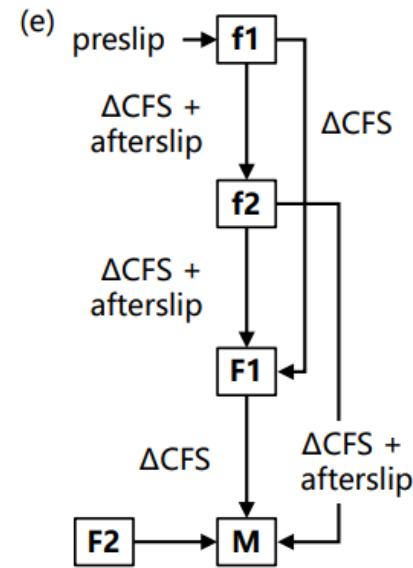
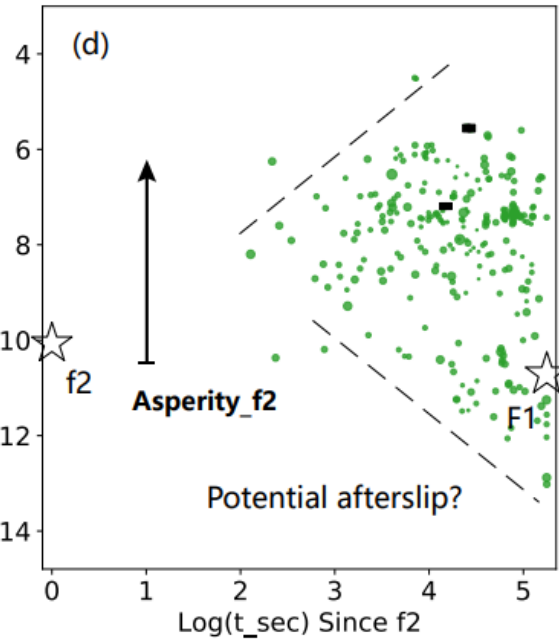
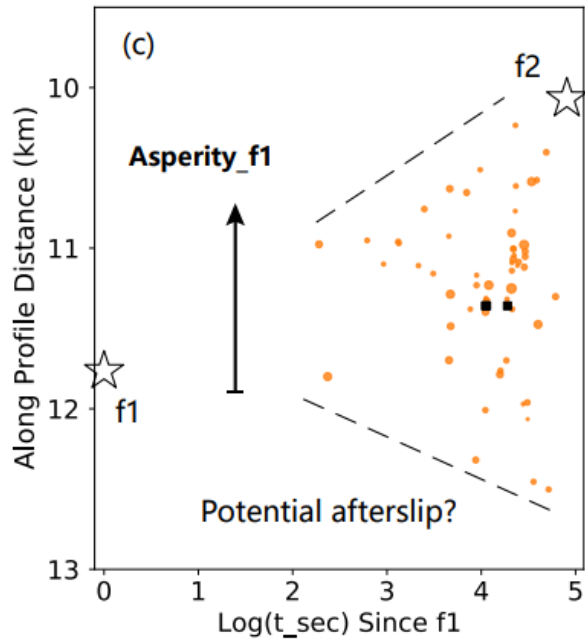
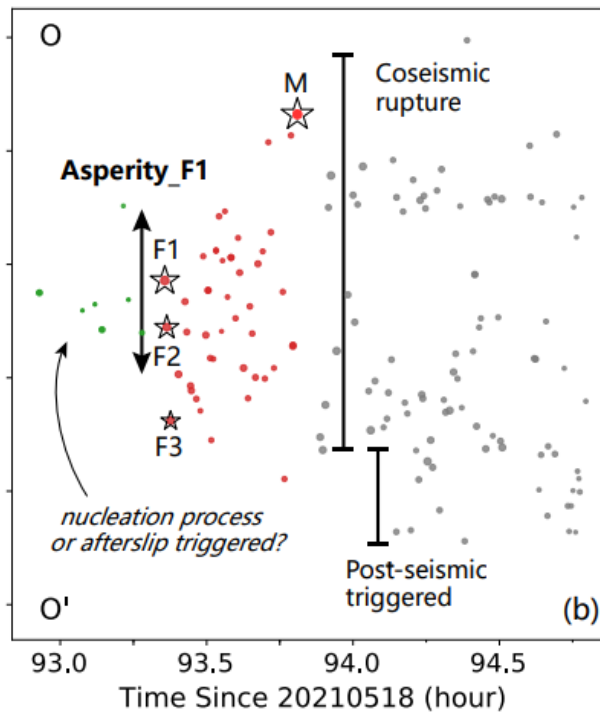
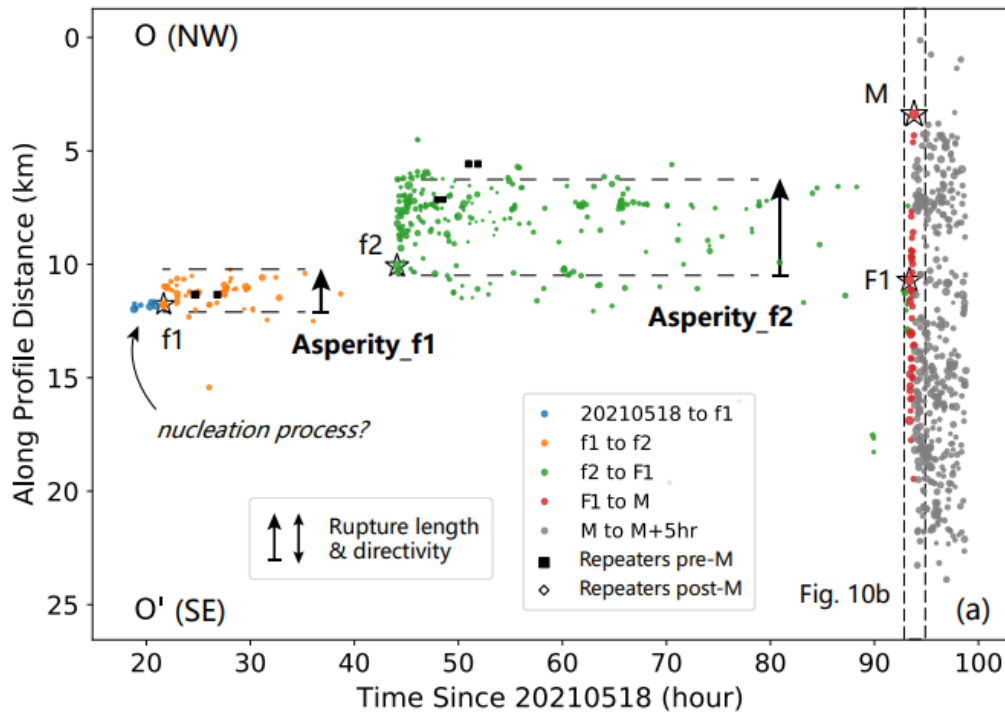
<https://github.com/YijianZhou/Manual-Analysis-Helper/tree/main/repeater>



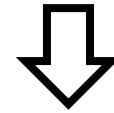
Repeating aftershocks in the 2021 Yangbi foreshock sequence

Zhou et al.,
JGR 2022
(under review)





Repeating
aftershocks
in the 2021
Yangbi
foreshock
sequence



afterslip-
driven
cascade
sequence

Zhou et al.,
JGR 2022
(under review)

Outline

- PAL: phase picking, association, and location
- MESS: a new matched filter detector
- **Apply PALM to build high-resolution catalog**
 - hardware & software
 - running PAL
 - running MESS

Two Complete Architectures for Automatic Earthquake Detection and Location: Part 1. Basic Principles of PALM & CEPR

地震学会青工委系列学术讲座

Two Complete Architectures for Automatic Earthquake Detection and Location: Part 1. Basic Principles of PALM & CEPR



剑

加州大学河滨分校

时间：2021年10月29日（星期五）9:00

地震学会青工委系列学术讲座



1 Two Complete Architectures fo...

9375播放



2 基于深度学习的区域地震目录...

2549播放



3 玛多地震地表变形特征: 异常...

1840播放

Hardware Requirements

- CPU
 - number of CPU threads > number of processes allocated
 - e.g. 4C8T CPU can allocate up to 8 parallel PAL processes
- GPU
 - Nvidia RTX series, suggest >4G GPU memory
 - e.g. RTX 3060 12G
- Memory & Hard drive
 - baseline >8G, suggest >16G memory
 - SSD is great, HDD also works




Hard Drive	
1~10 Tb	SATA



100M/s 

Memory	
100 Gb	PCIe




15G/s



15G/s 

Graphics Memory	
10 Gb	GDDR


100G/s



Software Requirements

- Anaconda
 - <https://www.anaconda.com/products/individual>
 - Numpy, Scipy etc. are included
- Obspy
 - <https://github.com/obspy/obspy/wiki/Installation-via-Anaconda>
 - processing seismic data
- Pytorch
 - <https://pytorch.org/>
 - deep learning architecture, GPU computation

“Install” PALM

- Setup software environment
 - suggest installing Anaconda & Obspy & Pytorch sequentially
- Download code from Github (latest release v2.7)
 - suggest saving at /home/user/software
- Install HypoInverse & HypoDD
 - suggest saving the binary file at /home/user/bin
- *Go through the example dataset*

Download Example Data

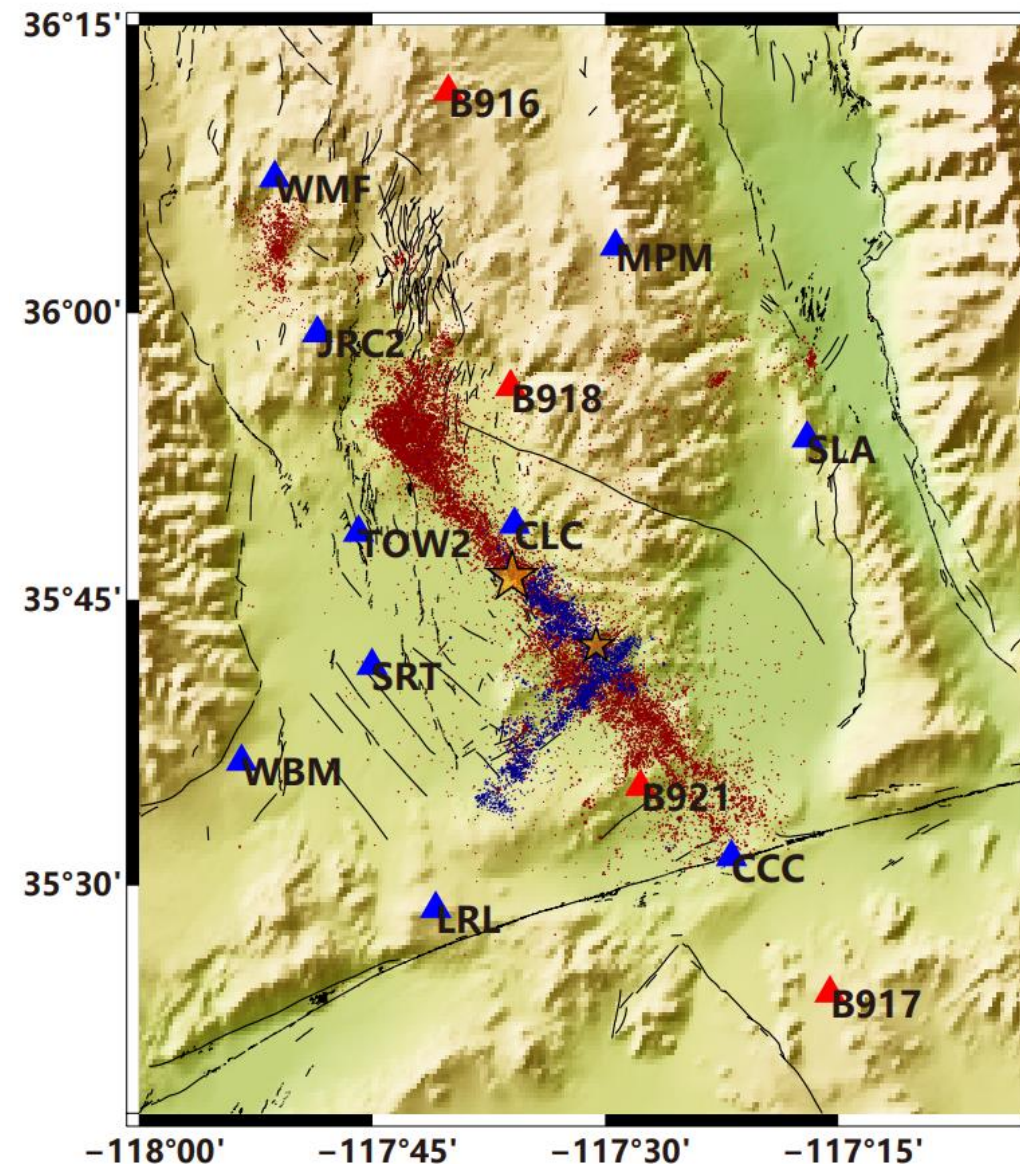
master ▾ PAL / example_pal_workdir / down_stp_data_eg.py / <> Jump to ▾

 YijianZhou update example

1 contributor

52 lines (49 sloc) | 1.74 KB














```
1 """ Download Example SCSN data by STP
2 STP can be downloaded from https://scedc.caltech.edu/data/stp/index.html
3 """
4 import os, shutil, glob
5 from obspy import UTCDateTime
6 import subprocess
7 import multiprocessing as mp
8
9 # i/o files
10 num_workers = 10
11 fsta = 'input/example_pal.sta'
12 time_range = '20190704-20190707'
13 out_root = '/data/Example_data'
14 if not os.path.exists(out_root): os.makedirs(out_root)
15 start_time, end_time = [UTCDateTime(date) for date in time_range.split('-')]
16 num_days = (end_time.date - start_time.date).days
```

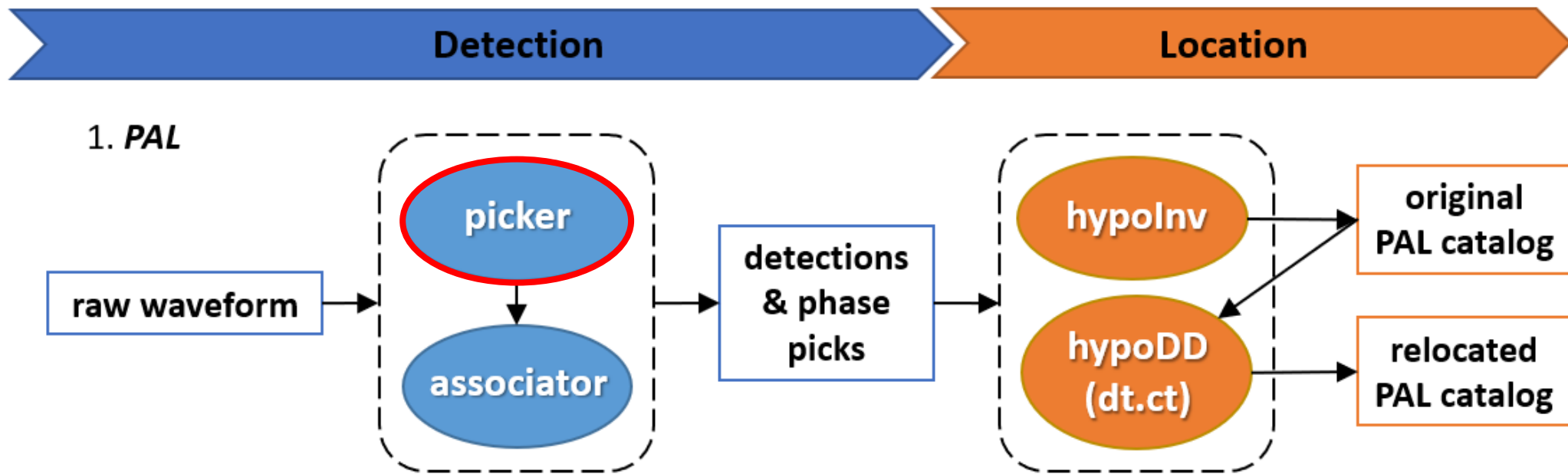


Outline

- Hardware & software
- **Running PAL**
- Running MESS

<https://github.com/YijianZhou/PAL/releases/tag/v2.7>

 YijianZhou	Update mk_sta.py	a8c20a2 yesterday	🕒 311 commits
 doc	add workflow		2 months ago
 env	Update pal.yml		13 months ago
 example_pal_workdir	update fsta format		3 months ago
 hypodd	Update mk_sta.py		yesterday
 hypoinverse	Update mk_sta.py		yesterday
 README.md	Update README.md		2 months ago
 associator_pal.py	Update associator_pal.py		last month
 config.py	Update config.py		10 months ago
 data_pipeline.py	Update data_pipeline.py		last month
 picker_pal.py	Update picker_pal.py		19 days ago
 run_assoc.py	3D assoc		16 months ago
 run_pick_assoc.py	3D assoc		16 months ago



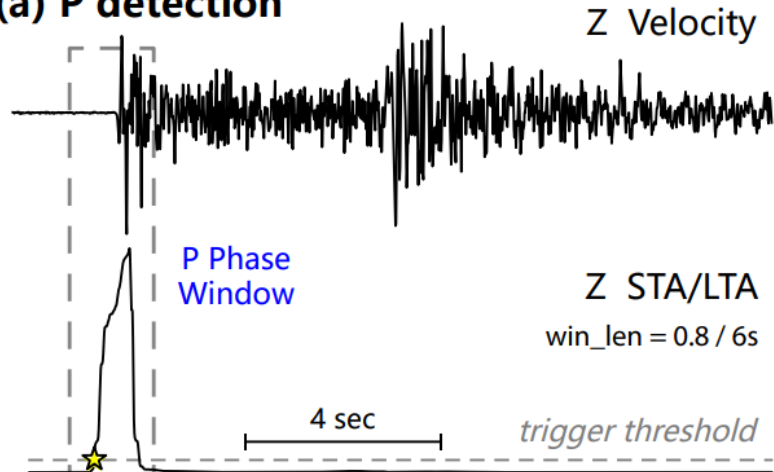
<https://github.com/YijianZhou/PAL>

Zhou et al., SRL 2021

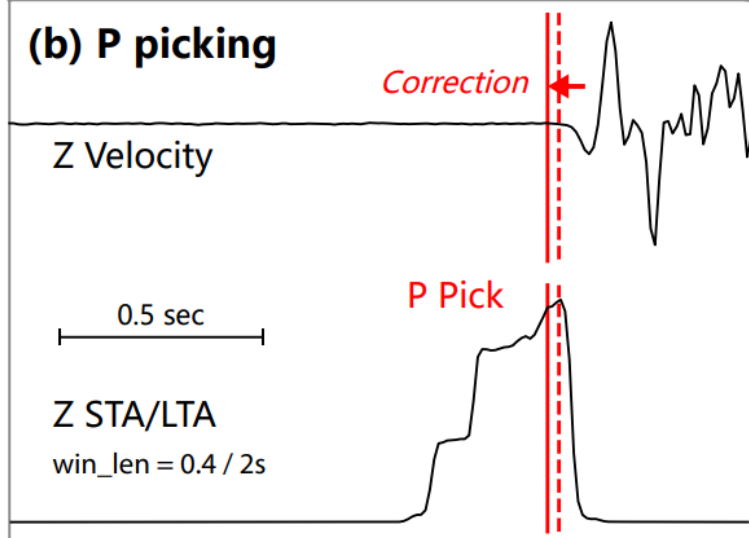
PAL Config 1: Picking

```
# 1. picker params
self.win_sta      = [0.8, 0.4, 1.] # win for STA: det, p, s
self.win_lta      = [6., 2., 2.]  # win for LTA: det, p, s
self.win_kurt      = [5., 1.]      # win for kurtosis: long & short
self.trig_thres    = 12.           # threshold to trig picker (by energy)
self.p_win         = [.5, 1.]      # search win for P
self.s_win         = 10.           # search win for S
self.pca_win       = 1.            # win_len for PCA filter
self.pca_range     = [0., 2.]      # time range to apply PCA filter
self.fd_thres      = 2.5           # min value of dominant frequency
self.amp_win       = [1., 4.]      # time win to get S amplitude
self.det_gap       = 5.            # time gap between detections
self.to_prep       = True          # whether to preprocess the raw data
self.freq_band     = [2, 40]       # frequency band
```

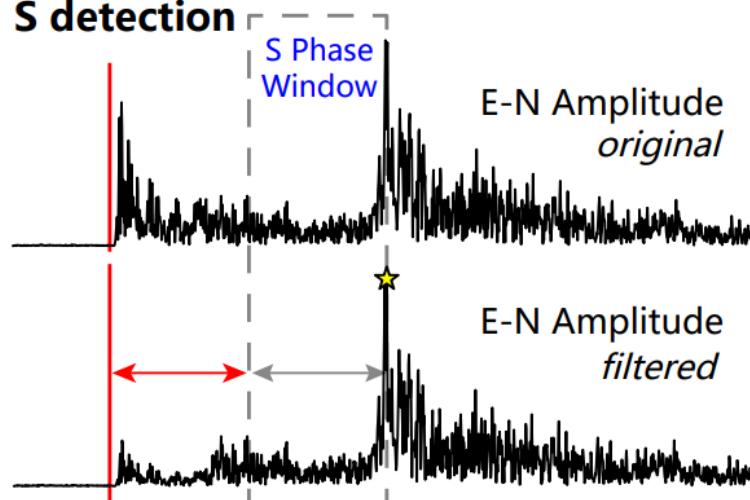
(a) P detection



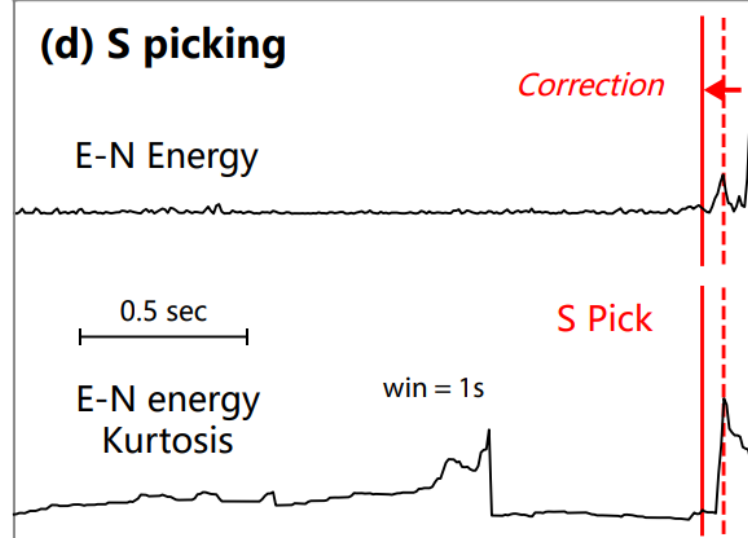
(b) P picking



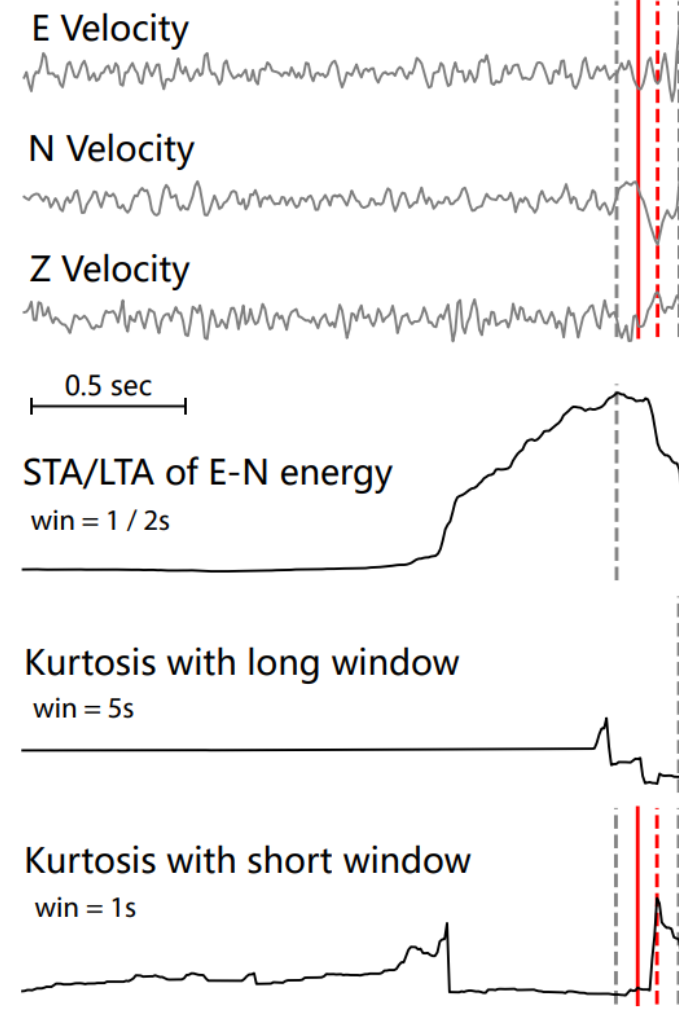
(c) S detection



(d) S picking



(e) S picking (in detail)



PAL Config 1: Picking

```
# 1. picker params
```

```
self.win_sta = [0.8, 0.4, 1.]
```

```
self.win_lta = [6., 2., 2.]
```

```
self.win_kurt = [5., 1.]
```

```
self.trig_thres = 12.
```

```
self.p_win = [.5, 1.]
```

```
self.s_win = 10.
```

```
self.pca_win = 1.
```

```
self.pca_range = [0., 2.]
```

```
self.fd_thres = 2.5
```

```
self.amp_win = [1., 4.]
```

```
self.det_gap = 5.
```

```
self.to_prep = True
```

```
self.freq_band = [2, 40]
```

suitable for most cases

```
# win for STA: det, p, s
```

```
# win for LTA: det, p, s
```

```
# win for kurtosis: long & short
```

```
# threshold to trig picker (by energy)
```

```
# search win for P
```

```
# search win for S
```

```
# win_len for PCA filter
```

```
# time range to apply PCA filter
```

```
# min value of dominant frequency
```

```
# time win to get S amplitude
```

```
# time gap between detections
```

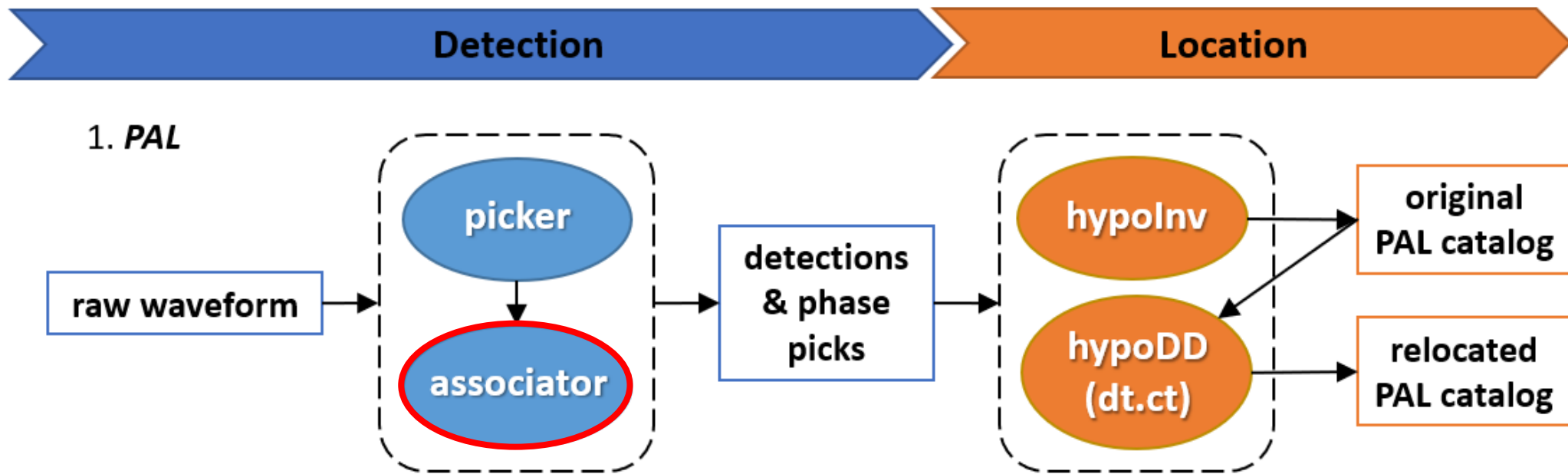
```
# whether to preprocess the raw data
```

```
# frequency band
```

PAL Config 1: Picking

```
# 1. picker params
self.win_sta      = [0.8, 0.4, 1.] # win for STA: det, p, s
self.win_lta      = [6., 2., 2.]  # win for LTA: det, p, s
self.win_kurt     = [5., 1.]      # win for kurtosis: long & short
self.trig_thres  = 12.            # threshold to trig picker (by energy)
self.p_win        = [.5, 1.]      # search win for P
self.s_win      = 10.            # search win for S
self.pca_win      = 1.            # win_len for PCA filter
self.pca_range    = [0., 2.]      # time range to apply PCA filter
self.fd_thres     = 2.5           # min value of dominant frequency
self.amp_win    = [1., 4.]       # time win to get S amplitude
self.det_gap      = 5.            # time gap between detections
self.to_prep      = True          # whether to preprocess the raw data
self.freq_band = [2, 40]        # frequency band
```

Set picking params according to the *station distribution*
(average inter-distance etc.) & noise level

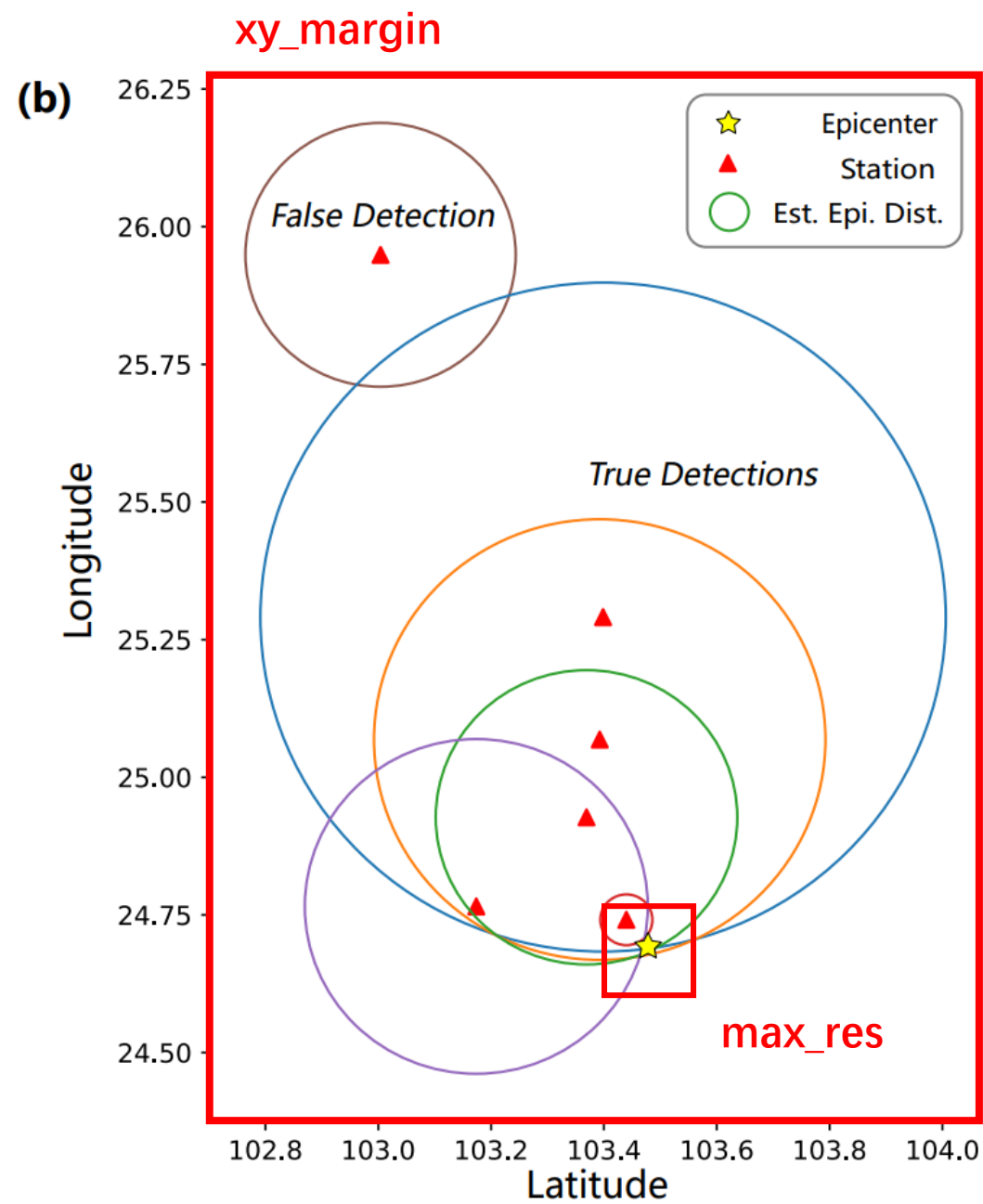
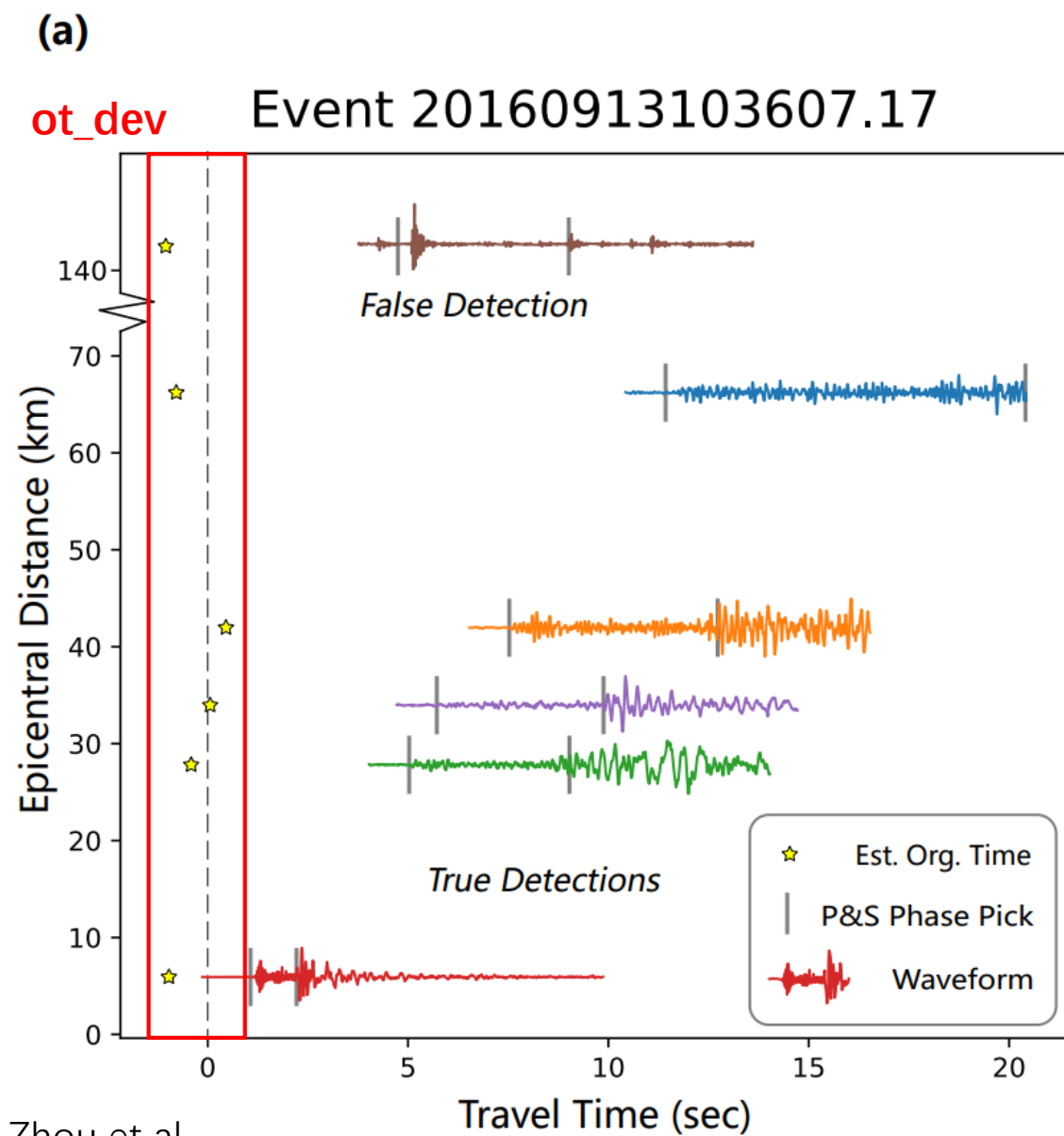


<https://github.com/YijianZhou/PAL>

Zhou et al., SRL 2021

PAL Config 2: Association

```
# 2. assoc params
self.min_sta      = 4           # min num of sta to assoc
self.ot_dev       = 2.         # max time deviation for ot assoc
self.max_res      = 1.5        # max P res for loc assoc
self.xy_margin    = 0.1        # xy (lateral) range inferred from sta loc
self.xy_grid      = 0.02       # xy (lateral) grid size (in degree)
self.z_grids      = [5]        # z (dep) grids (in km)
self.vp           = 5.9        # averaged P velocity
```

PAL Config 2: Association

```
# 2. assoc params
self.min_sta      = 4           # min num of sta to assoc
self.ot_dev       = 2.         # max time deviation for ot assoc
self.max_res      = 1.5        # max P res for loc assoc
self.xy_margin    = 0.1        # xy (lateral) range inferred from sta loc
self.xy_grid      = 0.02       # xy (lateral) grid size (in degree)
self.z_grids      = [5]        # z (dep) grids (in km)
self.vp           = 5.9        # averaged P velocity
```

**Set assoc params according to
the station distribution
(average inter-distance etc.)**

PAL Data_pipeline

1. Directory structure:
data_root/yyyymmdd/net.sta.date.chn.sac
2. Require 3-channel, naming as ENZ/123 etc.
3. Modify *get_data_dict* if necessary
e.g. if include single-channel stations

```
# get data path dict
def get_data_dict(date, data_dir):
    # get data paths
    data_dict = {}
    date_code = '{:0>4}{:0>2}{:0>2}'.format(date.year, date.month, date.day)
    st_paths = sorted(glob.glob(os.path.join(data_dir, date_code, '*')))
    for st_path in st_paths:
        fname = os.path.basename(st_path)
        net_sta = '.'.join(fname.split('.')[0:2])
        if net_sta in data_dict: data_dict[net_sta].append(st_path)
        else: data_dict[net_sta] = [st_path]
    # drop bad sta
    todel = [net_sta for net_sta in data_dict if len(data_dict[net_sta])!=3]
    for net_sta in todel: data_dict.pop(net_sta)
    return data_dict
```

```

# get station loc & gain dict
def get_sta_dict(sta_file):
    sta_dict = {}
    f=open(sta_file); lines=f.readlines
    for line in lines:
        codes = line.split(',')
        net_sta = codes[0]
        lat, lon, ele = [float(code) for code in codes[1:4]]
        # format 1: same gain for 3-chn & time invariant
        if len(codes[4:])==1: gain = float(codes[4])
        # format 2: different gain for 3-chn & time invariant
        elif len(codes[4:])==3: gain = [float(code) for code in codes[4:]]
        # format 3: different gain for 3-chn & time variant
        elif len(codes[4:])==5:
            gain = [float(code) for code in codes[4:7]]
            gain += [UTCDateTime(code) for code in codes[7:9]]
            gain = [gain]
        else: print('false sta_file format!'); continue
        if net_sta not in sta_dict:
            sta_dict[net_sta] = [lat,lon,ele,gain]
        else:
            sta_dict[net_sta][-1].append(gain[0]) # if format 3
    return sta_dict

```

Format of station file (3 formats):

net.sta, sta_lat, sta_lon, sta_ele, gain

net.sta, sta_lat, sta_lon, sta_ele, gain_e, n, z

net.sta, sta_lat, sta_lon, sta_ele, gain_e, n, z, t0, t1

Running PAL

- Pick + Assoc: *parallel_pick_assoc.py*
- Assoc only: *parallel_assoc.py*

```
pal_dir = '/home/zhouyj/software/PAL'
shutil.copyfile('config_eg.py', os.path.join(pal_dir, 'config.py'))
data_dir = '/data/Example_data'
time_range = '20190704-20190707'
sta_file = 'input/example_pal.sta'
num_workers = 3
out_root = 'output/eg'
out_pick_dir = 'output/eg/picks'
```

1. Parallel by time, the minimum unit is 1 day
2. Set date for *time_range*, but they are used as time (*end date + 1*)
3. Run *python parallel_pick_assoc.py*
4. In the output dir: *cat phase_* > pal.pha*

PAL Output

- PAL output picks
 - fpath: out_root/picks/yyyy-mm-dd.pick
 - format: net.sta, ot, tp, ts, s_amp, p_snr, freq_dom
- PAL output phase file
 - fpath: out_root/phase_yyyymmdd-yyymmdd.dat
 - event line: ot, lat, lon, dep, mag
 - phase line: net.sta, tp, ts, s_amp, p_snr

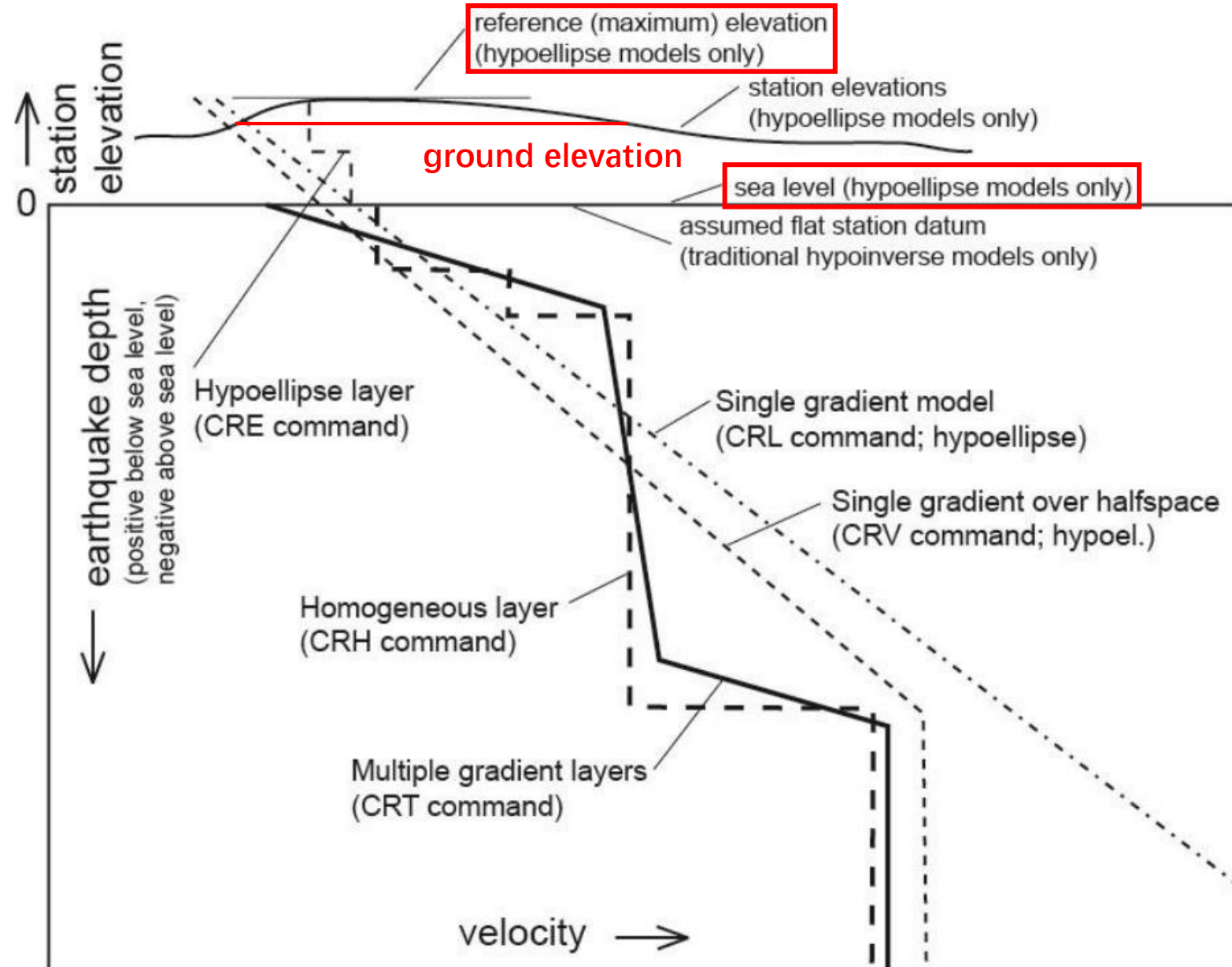
Pick-based Location

- HypoInverse
 - Software: <https://www.usgs.gov/software/hypoinverse-earthquake-location>
 - Document: <https://doi.org/10.3133/ofr02171>
- HypoDD (dt.ct)
 - Software: <https://www.ldeo.columbia.edu/~felixw/hypoDD.html>
 - Document: https://www.ldeo.columbia.edu/~felixw/papers/Waldhauser_OFR2001.pdf

Run HypoInverse

- Station file
 - same as PAL, just copy into input/
- Velocity model
 - in CRE format, which supports station elevation
 - set *ref_ele* & *grd_ele*, make necessary correction
- Location parameters
 - weighting by distance
 - weighting by residual

Crustal model types



```

self.ctrlg_code = 'eg_pal_hyp'
# 1. mk_sta: format station file
self.fsta = 'input/example_pal.sta'
self.lat_code = 'N'
self.lon_code = 'W'
# 2. mk_pha: format phase file
self.fpha = 'input/eg_pal.pha'
# 3. sum2csv: format output files
self.ref_ele = 2.5 # ref ele for CRE mod (max sta ele)
self.grd_ele = 1.5 # typical station elevation
# 4. run_hyp
self.ztr_rng = np.arange(0,20,1)
self.p_wht = 0 # weight code
self.s_wht = 1
self.rms_wht = '4 0.3 1 3'
self.dist_init = '1 50 1 2'
self.dist_wht = '4 20 1 3'
self.wht_code = '1 0.6 0.3 0.2'
self.fhyp_temp = 'temp_hyp/temp_vp-pos.hyp'
self.pmod = 'input/example_p.cre'
self.smod = [None, 'input/example_s.cre'][0]
self.pos = 1.73 # provide smod or pos

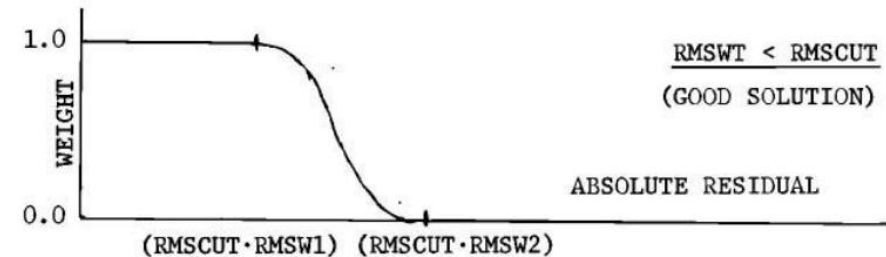
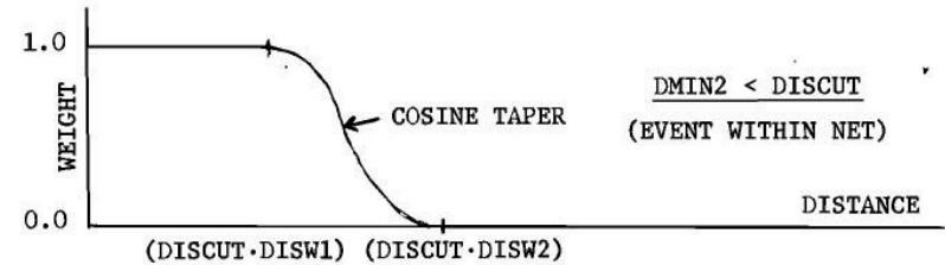
```

Write CRE Velocity Model

HK Model with V_P/V_S Ratio of 1.73		
Depth to Top of Layer (km)	CRE Interfaces	P -Velocity (km/sec)
0.0	0.0	5.5
5.5	6.5	6.3
16.0	17.0	6.7
32.0	33.0	7.8

Weighting Scheme

- Distance weighting
 - $< \text{min_dist}$: full weight
 - $> \text{max_dist}$: zero weight
 - $\text{min_dist} \sim \text{max_dist}$: cos taper
- Residual weighting
 - $< \text{min_res}$: full weight
 - $> \text{max_res}$: zero weight
 - $\text{min_res} \sim \text{max_res}$: cos taper



Klein, 2014

```

self.ctrlg_code = 'eg_pal_hyp'
# 1. mk_sta: format station file
self.fsta = 'input/example_pal.sta'
self.lat_code = 'N'
self.lon_code = 'W'
# 2. mk_pha: format phase file
self.fpha = 'input/eg_pal.pha'
# 3. sum2csv: format output files
self.ref_ele = 2.5 # ref ele for CRE mod (max sta ele)
self.grd_ele = 1.5 # typical station elevation
# 4. run_hyp
self.ztr_rng = np.arange(0,20,1)
self.p_wht = 0 # weight code
self.s_wht = 1
self.rms_wht = '4 0.3 1 3'
self.dist_init = '1 50 1 2'
self.dist_wht = '4 20 1 3'
self.wht_code = '1 0.6 0.3 0.2'
self.fhyp_temp = 'temp_hyp/temp_vp-pos.hyp'
self.pmod = 'input/example_p.cre'
self.smod = [None, 'input/example_s.cre'][0]
self.pos = 1.73 # provide smod or pos

```

HypolInverse Output

- Catalog: *eg_pal_hyp.ctlg*
 - format: ot, lat, lon, dep, mag
- Phase: *eg_pal_hyp.pha*
 - event line: ot, lat, lon, dep, mag
 - phase line: net.sta, tp, ts, s_amp, p_snr
- Phase with evid: *eg_pal_hyp_full.pha*
 - event line: ot, lat, lon, dep, mag, evid
 - phase line: net.sta, tp, ts, s_amp, p_snr
- Quality control files: *.sum, _good.csv, _bad.csv*

Run HypoDD

- Station file
 - same as PAL, just *cp* into *input/*
- Phase file
 - _full.pha, output of HypoInverse
 - event line: ot, lat, lon, dep, mag, *evid*
 - phase line: net.sta, tp, ts
- Location parameters
 - ph2dt
 - hypoDD

```
self.hypo_root = '/home/zhouyj/bin'  
self.ctlg_code = 'eg_pal_ct'  
self.fsta = 'input/example_pal.sta'  
self.fpha = 'input/eg_pal_hyp_full.pha'  
self.dep_corr = 5 # avoid air quake  
self.ot_range = '20190704-20190707'  
self.lat_range = [35.45, 36.05]  
self.lon_range = [-117.8, -117.25]  
self.num_grids = [1, 1] # x, y (lon, lat)  
self.xy_pad = [0.06, 0.05] # degree  
self.num_workers = 5
```

hypoDD -- A Program to Compute Double-Difference Hypocenter Locations

(*hypoDD* version 1.0 - 03/2001)

by

Felix Waldhauser

U.S. Geol. Survey
345 Middlefield Rd, MS977
Menlo Park, CA 94025
felix@andreas.wr.usgs.gov

**Please read
this document!**

















HypoDD Output

- Catalog: *eg_pal_ct.ctlg*
 - format: ot, lat, lon, dep, mag
- Phase: *eg_pal_ct.pha*
 - event line: ot, lat, lon, dep, mag
 - phase line: net.sta, tp, ts, s_amp, p_snr
- Phase with evid: *eg_pal_ct_full.pha*
 - event line: ot, lat, lon, dep, mag, evid
 - phase line: net.sta, tp, ts, s_amp, p_snr
- Quality control files: *.ph2dt & .hypoDD (screen output)*

Outline

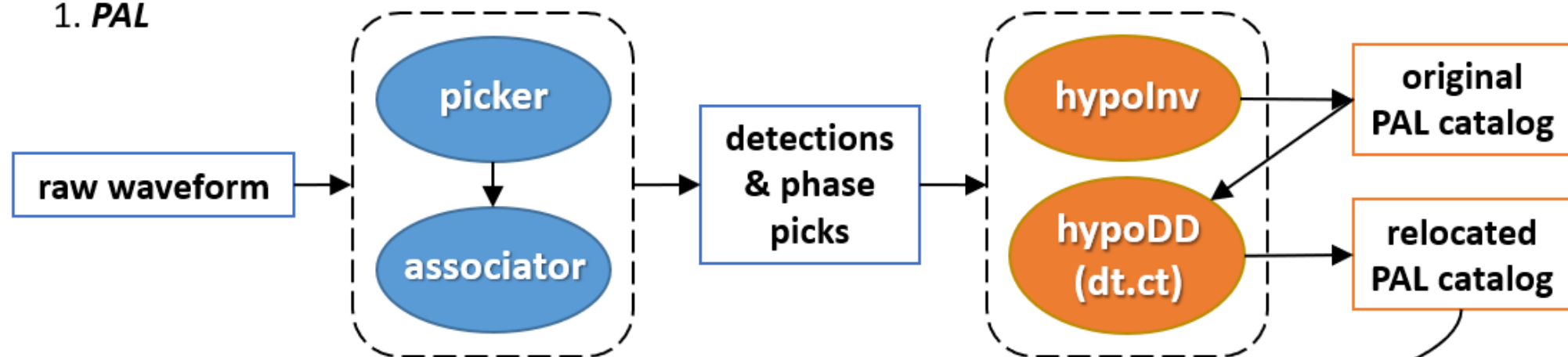
- Hardware & software
- Running PAL
- **Running MESS**

<https://github.com/YijianZhou/MESS/releases/tag/v2.7>

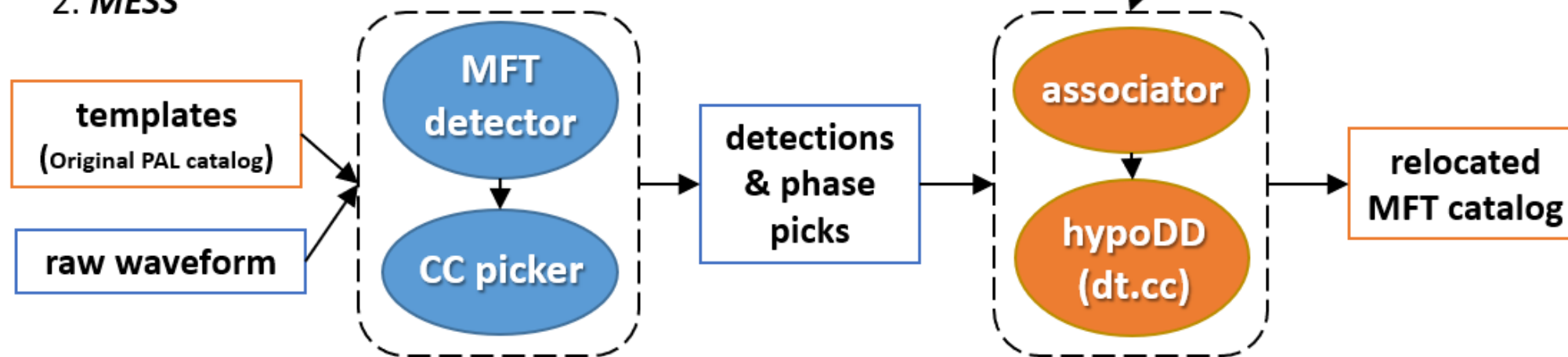
 YijianZhou Update cut_template_intense.py	2974294 7 hours ago	 236 commits
 doc	add workflow	2 months ago
 env	Update mess.yml	13 months ago
 example_mess_workdir	update cut template	15 days ago
 hypodd	rename fsta	7 hours ago
 README.md	add workflow	2 months ago
 config.py	update MFT det & template_cut	2 months ago
 cut_template_intense.py	Update cut_template_intense.py	7 hours ago
 cut_template_long.py	update cut template	15 days ago
 dataset.py	Update dataset.py	19 days ago
 dataset_gpu.py	Update dataset_gpu.py	19 days ago
 mess_lib.py	fix bug: sta_dict	20 days ago
 mess_lib_gpu.py	update MFT det & template_cut	2 months ago
 run_mess.py	update example	10 months ago
 run_mess_gpu.py	update example	10 months ago



1. PAL



2. MESS

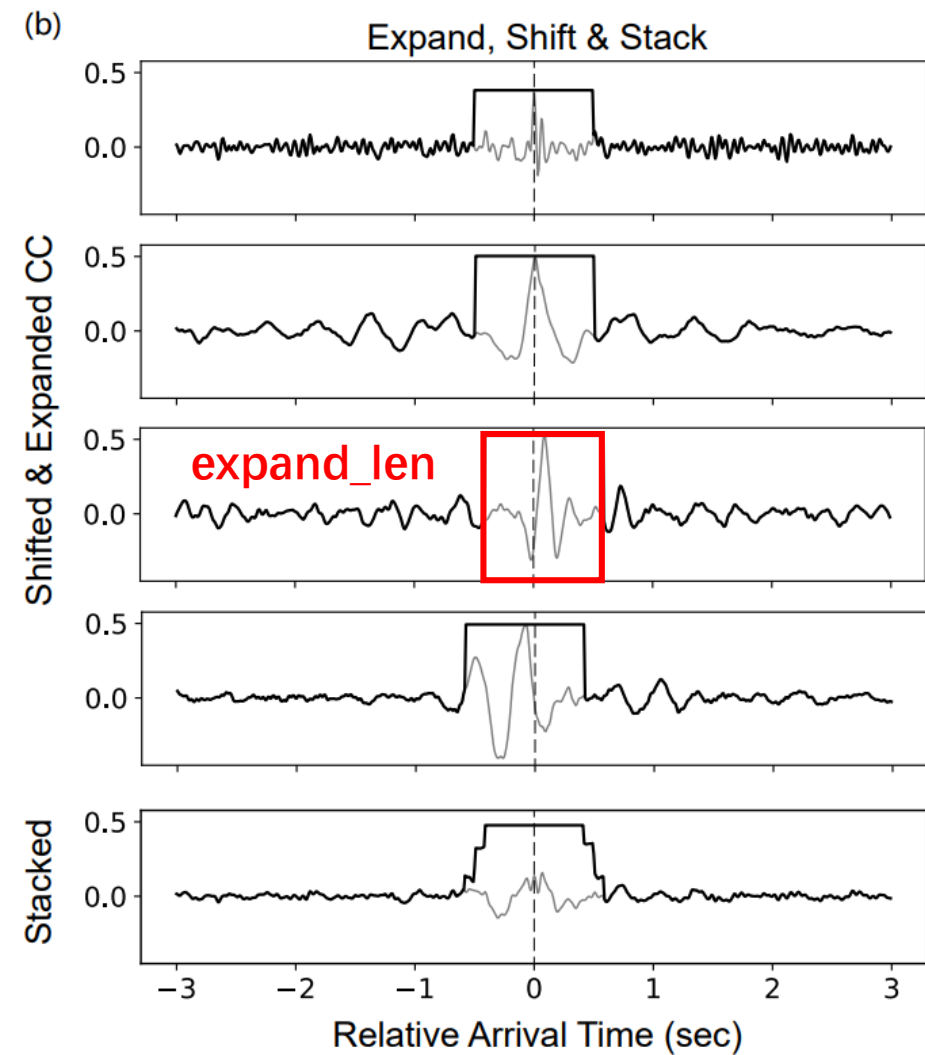
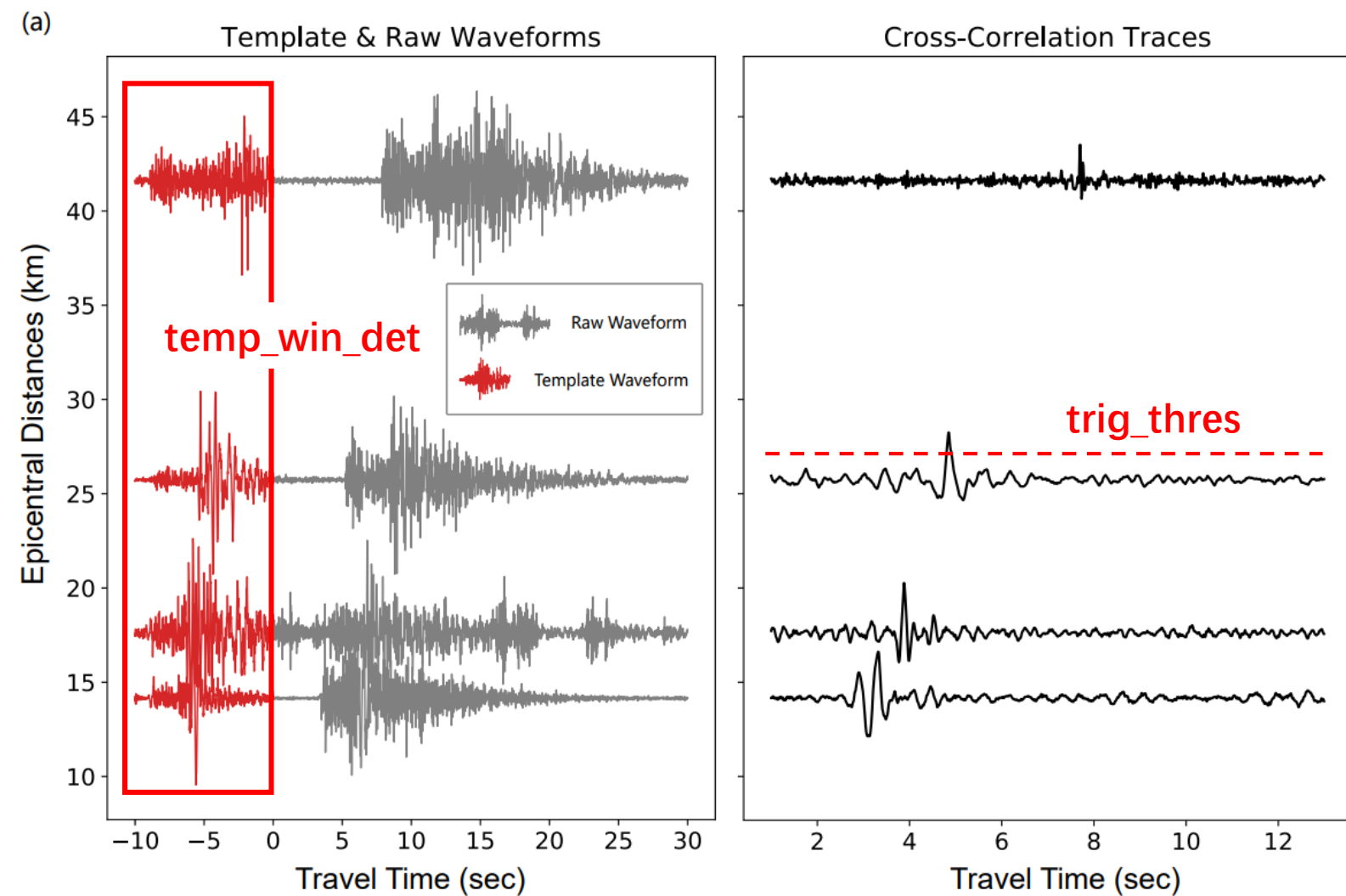


```

# MFT params
self.win_len = [10,20]
self.min_sta = 4
self.max_sta = 15
self.temp_win_det = [1.,9.]
self.temp_win_p = [0.5,1.5]
self.temp_win_s = [0.5,2.5]
self.trig_thres = 0.3
self.expand_len = 1.
self.det_gap = 5.
self.pick_win_p = [1.0,1.0]
self.pick_win_s = [1.6,1.6]
self.chn_p = [2]
self.chn_s = [0,1]
self.amp_win = [1,4]
# data process
self.samp_rate = 50
self.freq_band = [2.,40.]
self.num_workers = 10
self.get_data_dict = dp.get_data_dict
self.get_sta_dict = dp.get_sta_dict

# cut template length
# min sta num for template event
# max sta num for template event
# temp win for detection, pre & post P
# temp win for p pick, pre & post P
# temp win for s pick, pre & post S
# cc thres for det & peak expansion
# win len for cc peak expansion
# gap sec for detection
# search win for P pick
# search win for S pick
# chn for P pick
# chn for S pick
# win for amp measurement

```



Zhou et al., SRL 2021

```

# MFT params
self.win_len = [10,20]
self.min_sta = 4
self.max_sta = 15
self.temp_win_det = [1.,9.]
self.temp_win_p = [0.5,1.5]
self.temp_win_s = [0.5,2.5]
self.trig_thres = 0.3
self.expand_len = 1.
self.det_gap = 5.
self.pick_win_p = [1.0,1.0]
self.pick_win_s = [1.6,1.6]
self.chn_p = [2]
self.chn_s = [0,1]
self.amp_win = [1,4]
# data process
self.samp_rate = 50
self.freq_band = [2.,40.]
self.num_workers = 10
self.get_data_dict = dp.get_data_dict
self.get_sta_dict = dp.get_sta_dict

# cut template length
# min sta num for template event
# max sta num for template event
# temp win for detection, pre & post P
# temp win for p pick, pre & post P
# temp win for s pick, pre & post S
# cc thres for det & peak expansion
# win len for cc peak expansion
# gap sec for detection
# search win for P pick
# search win for S pick
# chn for P pick
# chn for S pick
# win for amp measurement

```

**Set MESS params based on the
station distribution (average
inter-distance etc.)**

Running MESS

- Make template phase
 - PAL det phase: *eg_pal.pha* → evid & event name
 - HypoInverse phase: *eg_pal_hyp.pha* → location
- Cut templates
 - *cut_templates_intense.py*: if many events in one day
 - *cut_templates_long.py*: if few events in one day
- Run MESS
 - use GPU version to enjoy ×40 acceleration
 - use CPU version if you have >100 cores cluster

1. Make Template Phase: *pha2temp.py*

- Inputs
 - Detection phase: *eg_pal.pha*, providing evid & event name
 - Located phase: *eg_pal_hyp_full.pha*
- Outputs
 - Template phase: *evid_name, ot, lat, lon, dep, mag*

```
# i/o paths
fpha_det = 'input/eg_pal.pha'
fpha_loc = 'input/eg_pal_hyp_full.pha'
fout = open('input/eg_pal.temp', 'w')
# selection criteria
ot_range = '20190704-20190707'
ot_range = [UTCDateTime(code) for code in ot_range.split('-')]
lat_range = [35.5, 36.]
lon_range = [-117.8, -117.3]
```

2. Cut Templates

- Cut intense sequence: *cut_templates_intense.py*
 - read 1-sta 1-day's data & preprocess
 - slice all phases on that station-date pair
- Cut long-term data: *cut_templates_long.py*
 - for all events,
 - read event window (~30s) & preprocess

**~100,000 events / ~1,000,000
phases, finish with 20min**

```
# i/o paths
mess_dir = '/home/zhouyj/software/MESS'
data_dir = '/data/Example_data'
out_root = 'output/Example_templates'
temp pha = 'input/eg_pal.temp'
```


3. Running MESS

- Can start multiple processes based on the GPU memory
 - $n_sta * 3 \text{ chn} * 4 \text{ bit} * 86400 \text{ sec} * \text{samp_rate}$
 - \rightarrow if 50 Hz, $100 \text{ Mb} * n_sta$
- Can start multiple processes based on the CPU threads

```
# i/o paths
gpu_idx = '0'
mess_dir = '/home/zhouyj/software/MESS'
data_dir = '/data/Example_data'
time_range = '20190704-20190707'
sta_file = 'input/example_pal.sta'
temp_root = 'output/Example_templates'
temp_pha = 'input/eg_pal.temp'
```

```
template 0_20190704161342.98
15 detections, 10 stations, 0.8s
det_ot 2019-07-04T15:35:29.700000Z, det_cc 0.30
CI.TOW2 | dt_p -0.02s, dt_s -0.02s | cc_p 0.716, cc_s 0.744
CI.SRT  | dt_p -0.02s, dt_s 0.00 s | cc_p 0.395, cc_s 0.410
CI.CCC  | dt_p -0.02s, dt_s -0.02s | cc_p 0.378, cc_s 0.567
CI.MPM  | dt_p 0.00 s, dt_s 0.00 s | cc_p 0.849, cc_s 0.429
CI.JRC2 | dt_p 0.00 s, dt_s -0.02s | cc_p 0.334, cc_s 0.669
CI.CLC  | dt_p -0.02s, dt_s -0.02s | cc_p 0.558, cc_s 0.599
CI.SLA  | dt_p -0.56s, dt_s 0.00 s | cc_p 0.370, cc_s 0.382
CI.LRL  | dt_p 0.40 s, dt_s -0.02s | cc_p 0.265, cc_s 0.280
CI.WBM  | dt_p -0.22s, dt_s -0.90s | cc_p 0.255, cc_s 0.208
```

HypoDD Relocation

- Relocate templates (*pal_ct_full.pha*)
- Relocate MESS detections: *python run_hypoDD.py*
 - associate detections from different templates → *dt.cc*
 - run hypoDD (relocate with *dt.cc*)

```
# 2. mk_dt
self.temp_pha = 'input/eg_pal_ct_full.pha' # reloc template phase file
self.det_pha = 'input/eg_mess.pha'         # mess output phase file
self.ot_dev = 2.                           # ot diff for det assoc
self.cc_thres = [0.3, 0.4]                 # CC thres for det & pick
self.dt_thres = [0.6, 1.]                  # max dt_p & dt_s
self.nbr_thres = [2, 30]                   # min & max num of neighbor event
self.min_sta = 4

# 3. reloc2csv
self.lat_range = [35.4, 36.1]
self.lon_range = [-117.85, -117.25]
self.xy_pad = [0.046, 0.037]               # degree
self.num_grids = [1, 1]                   # x,y (lon, lat)
```

MESS Output

- “Phase” output: *eg_mess.pha*
 - fpath: out_root/phase_yyyymmdd-yyyymddd.dat
 - event line: evid_name, ot, lat, lon, dep, cc
 - phase line: net.sta, tp, ts, dt_p, dt_s, s_amp, cc_p, cc_s
- HypoDD output catalog: *eg_mess_cc.ctlg*
 - format: ot, lat, lon, dep, mag

Summary: A Simplified Version

- Prepare data
 - consistent directory structure → whether modify *data_pipeline.py*
 - plot station location → reselection of station & tune PALM parameters
- Run PALM
 - rewrite file & directories
 - inspect detection and location results

Summary: In Detail

- Plot station distribution & data continuity
 - proper distribution? continuous? → may need select stations
 - *s_win* in PAL & *temp_win_det* in MESS
- Run PAL
 - number of detected events? as expected?
 - ratio of bad-located events with hypoInverse?
 - plot location distribution, FMD, M-T
- Run MESS
 - number of detected events? as expected?
 - ratio of dropped events in hypoDD?
 - plot location distribution, FMD, M-T

Remaining Topics

- Complex network configuration
- Tuning detection parameters
- Tuning location parameters
- Combination of different modules
- Special datasets (e.g. large-N array, OBS, DAS etc.)
-

References

- **Zhou, Y.**, H. Yue, L. Fang, S. Zhou, L. Zhao, & A. Ghosh (2021). An Earthquake Detection and Location Architecture for Continuous Seismograms: Phase Picking, Association, Location, and Matched Filter (PALM). *Seismological Research Letters*. doi: <https://doi.org/10.1785/0220210111>
- **Zhou, Y.**, A. Ghosh, L. Fang, H. Yue, S. Zhou, & Y. Su (2021). A High-Resolution Seismic Catalog for the 2021 $M_s6.4/M_w6.1$ YangBi Earthquake Sequence, Yunnan, China: Application of AI picker and Matched Filter. *Earthquake Science*; doi: [10.29382/eqs-2021-0031](https://doi.org/10.29382/eqs-2021-0031)
- **Zhou, Y.**, H. Yue, S. Zhou, L. Fang, Y. Zhou, L. Xu, Z. Liu, T. Wang, L. Zhao, & A. Ghosh (2022). Microseismicity along Xiaojiang Fault Zone (Southeastern Tibetan Plateau) and the Characterization of Interseismic Fault Behavior. *Tectonophysics*, 833: 229364. doi: <https://doi.org/10.1016/j.tecto.2022.229364>
- **Zhou, Y.**, C. Ren, A. Ghosh, H. Meng, L. Fang, H. Yue, S. Zhou, & Y. Su (2022, under review). Seismological Characterization of the 2021 Yangbi Foreshock-Mainshock Sequence, Yunnan, China: More than a Triggered Cascade. *Journal of Geophysical Research: Solid Earth*

Open-Source Software

- PAL <https://github.com/YijianZhou/PAL>
- MESS <https://github.com/YijianZhou/MESS>
- Repeater detection <https://github.com/YijianZhou/Manual-Analysis-Helper>
- Data preparation <https://github.com/YijianZhou/Seismic-Data-Preparation>
- Data visualization <https://github.com/YijianZhou/Seismicity-Visualization>

Contact Us!



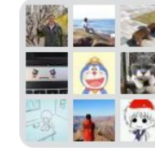
**Include but not
limited to:**
Report bugs
Assistance in usage
Cooperation
Technical discussion
... ..

周一剑 Yijian ZHOU

Email: yijian.zhou@email.ucr.edu

WeChat: zhouyj_observer

Github: <https://github.com/YijianZhou>



PALM用户交流服务群



Scan my name card,
and I will invite you
into the group



Valid until 11/4 and will update upon joining group