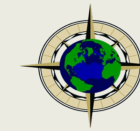


Analog geophysical data [from nuclear tests]: Which digitization software should I use? Can I leverage the power of AI for forensic monitoring?

Robert Walker¹, Lillian Soto-Cordero²

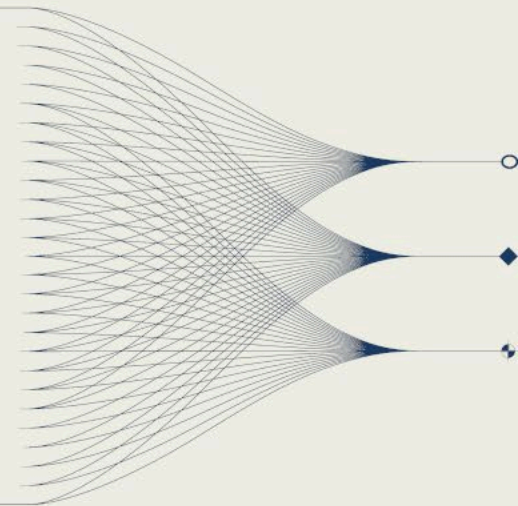
Kegman, Inc¹, Air Force Technical Applications Center²



KEGMAN
Technology that Inspires - Service that Exceeds

INTRODUCTION AND MAIN RESULTS

Digitization of historic seismograms is an imperative need for the Nuclear Explosion Monitoring community since most recorded observations of nuclear tests are in analog media. To assess the effectiveness of available digitization software, we identified 4 potential algorithms (out of 33 reviewed references) for further testing and developed a Python toolkit for the generation of synthetic analog helicorder records.

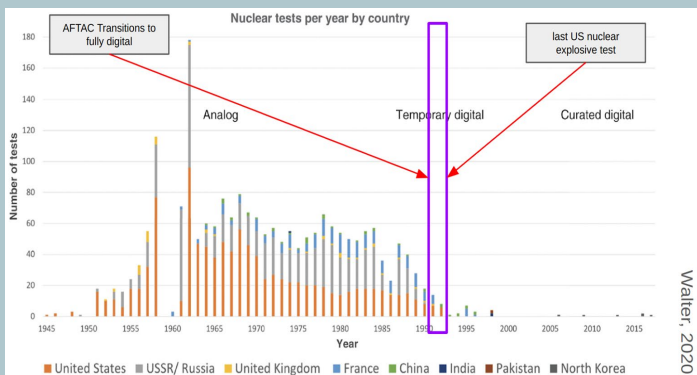


Relevance

Digitization of historic analog seismograms from nuclear explosions is an imperative need.

Current digitization options require careful user oversight and are both resource and time intensive.

US NDC analog data are very diverse (varied formats over 50 years).



Most nuclear tests were recorded only in analog form.

We seek to...

- Assess the performance and reliability of digitization software for extraction waveforms
- Produce digitization testing datasets for benchmarking
- Develop metrics to assess software performance
- Test and evaluate existing software

Digitization Code Review

***We sought to find and evaluate available options to digitize scanned analog records.
We sought codes that would streamline the process as much as possible.***

To be considered, a tools needed to...

- Automate the digitization process
- Be publicly available
- Have open source code
- **Produce full digital waveforms & preserve full frequency content**

- **12 programs reviewed**
- **33 references consulted**

Comprehensive evaluation of:

- **Originators**
 - Who sponsored / wrote it.
- **Source Availability/Accessibility**
 - Where can we find the code?
- **Last Update**
 - How recent was the latest update?
- **Support**
 - Who can we reach out to for help?
- **Language Used**
 - Programming language used.
 - Dependencies needed
- **Methodology**
 - How did they go about addressing the problem?
- **Use Cases**
 - What sort of records has this been used on (plus the corresponding references).
- **Advantages**
- **Disadvantages**
- **Assembly Required?**
 - Can it be easily setup, and will it then function, without [extensive] technical expertise?

We identified four (4) promising prospects; our analysis focuses on the 3 openly available codes:

Package	Originators	Approach	Open Source	Openly Available?
DigitSeis	Harvard	Computer Vision	No	Yes
SKATE	Retriever Technologies	Image processing / CV	Yes	Yes
Teseo	INGV	Semi automated tracing	Yes	Yes
JMA	Japanese Meteorological Agency	Machine Learning?	No	No

Synthetic Trace Generator

We needed to produce numerous and varied examples of challenging analog seismograms to test the capability of our codes with. To accomplish this, we developed a toolkit to present digital array data in a fashion similar to historic analog seismograms.

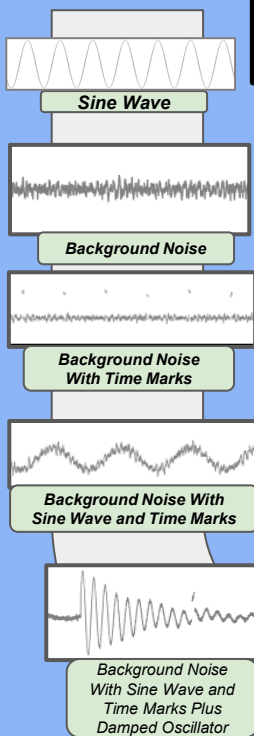
Python sandbox for synthetic analog seismogram generation

Current Features

- Generation of a signal of interest through numpy / scipy signal processing
- Plotting of dayplots with user defined wrap lengths
- WWSSN style time markers
- Soviet style time markers
- Randomized event generation
- Digital twin (SAC format) alongside analog image result
- Modeling of scanning detail effect (plot to specific DPI)
- Accurate thinning with amplitude
- Drop in plotting of digitally recorded (present day) seismic data

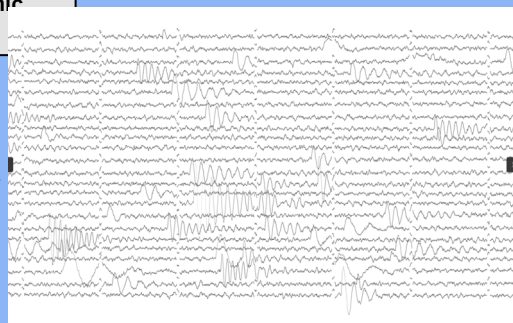
In Development

- US NDC time markers
- Finite arm length modeling
- More realistic representation of seismic waveforms
- Representation of WWSSN transfer functions (ground motion to recorded data)



Initially we used Damped Oscillators as a stand in for seismic events of interest.

Background Noise With Sine Wave and Time Marks Plus Damped Oscillator - Four Hour Duration



Making Legacy Datasets from Modern Data

P2.4-066

In order to effectively evaluate digitization codes, (and to leverage AI/ML) we need large datasets of scanned seismograms with known digital waveform recordings and instrument response. These do not exist for historic data. We leverage our Synthetic Generation and seek to include realistic waveforms produced with a minimum of computational cost. We accomplish this by presenting data recorded on modern digital seismometers in a format reminiscent of historical instruments.

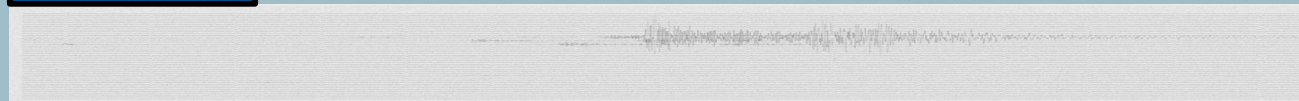
To reproduce the past, we turned to the present....

Below: example of "synthetic analog" data created from a digitally-recorded event
Ridgecrest Earthquake 6 July 2019 0000Z - 1200Z
M_w 7.1, Recorded at IU.ANMO.00.BHZ

Digital displacement, Response Removed



Simulated WWSSN SP



Simulated WWSSN LP



Conversion to reproduction 'legacy' data may be achieved through removing modern instrument response and applying historic (e.g. WWSSN) transfer functions to resultant ground motion data
Production of large quantities of "legacy presenting" seismogram images allows for ample training and test data for machine learning algorithms.

Testing Rubric

To compare code effectiveness, we followed the lead of prior US NDC procedures (Kemerait et. al., 1981) outlined in "A Study of the Hand Digitizing Process for Short Period Seismic Data" to grade the performance of our evaluated code. This breaks down to:

Time Domain

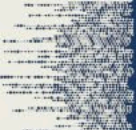
- Time series overlay
- Waveform cross correlation

Frequency Domain

- Spectra overlay
- Cross correlation of Power Spectra

To grade the performance of our effort, we must consider....

- Robustness of waveform conversion to digital time series – can we handle complexity and if so how much?
- Fidelity of result – Overall
- What are our frequency bands of interest and are we preserving them?



Test Waveforms

P2.4-066

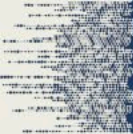
Our test waveforms were synthetically generated and plotted in image (TIFF and PNG) format. Included in the plot were WWSSN SP style time markings and a representation of thinning with trace amplitude. Generated images were then convolved with noise and a treated with a Gaussian blurring function to present a more realistic pixel intensity distribution (replicating an actual scanned image.)



12 Minute noise record with WWSSN SP style time marks

Software Performance Assessment

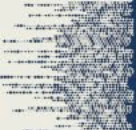
	Teseo	DigitSeis	SKATE
Process Duration			
Time it takes to complete digitization of a 12 minute trace	60 Min	10-15 Min	~ 2 min <i>(PRELIMINARY - output is fractured segments)</i>
Setup & Stability			
Software Setup	Straightforward docker container	Straightforward -- Windows/Mac executable worked upon execution	Required code modification. No documentation for local installation
Number of crashes per run	3-5	1	N/A
Tasks where digitization process crashes.	onset of impulsive, high amplitude waves	n/a	Difficulty delineating "Region of Interest" (region to scan) (Pipeline processing will throw error and stop)
Tasks where digitization process slows down.	- Encountering faint traces <i>(auto algorithm dependent on a color weighted mean requiring significant image tuning)</i>	- Onset of impulsive, high amplitude wave. - Faint traces return fragmented segments - Manual adjustment of traces after auto detection	N/A Faint traces return fragmented segments



Software Performance Assessment


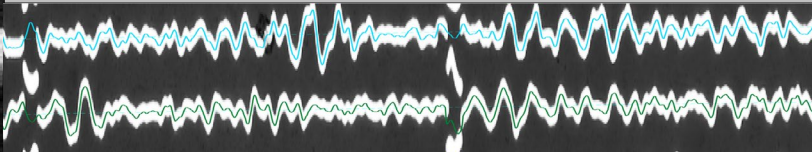
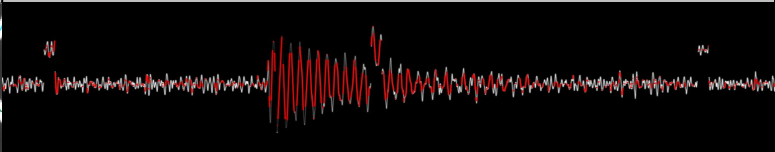
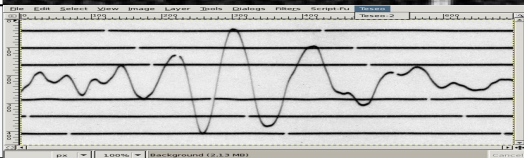
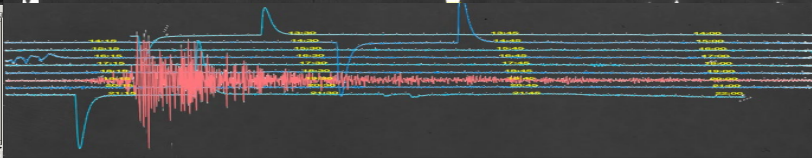
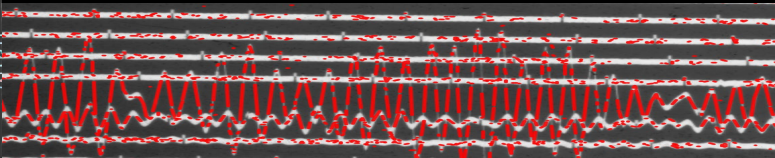
P2.4-066

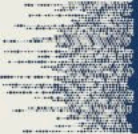
	Teseo	DigitSeis	SKATE
Performance			
How is impulsive phase onset handled?	Line-following algorithm breaks down / code crashes / manual intervention required	Manual intervention	N/A
Is the code able to automatically digitize peak amplitudes?	No	No	No
How are faint traces handled?	Manual	Manual	Surprisingly good
How are scanning artifacts handled?	Ignored / code is a line-following algorithm	Identified and manually removed.	Manual removal
How are timemarks in the target signal handled?	First sample of time mark is noted Provision for time mark path loading available	Detects deflected time marks (WWSSN style) and folds into trace	Identification is by size and deflection from meanlines (Time marks are not currently used by code)
Produces continuous digital time series?	Yes	Yes	No
Full-frequency content recovery (spectrogram results)	Mostly	Mostly	N/A



Software Performance Assessment

P2.4-066

	Teseo	DigitSeis	SKATE
Requirements & Limitations			
Time marks needed to run?	no	Yes (limited to WWSSN)	No
Image requirements	grayscale; dependent on a color weighted mean	inverted (white trace / black background)	Needs padded image delineating region of interest
Additional Areas for Optimization			
Outputs extracted signal in common format (mseed, sac, etc)?	Yes (sac)	Yes (sac)	No
User settings saved?	No, reverts to default values	No, reverts to default values	N/A
Waveform Examples			
Digitization of our synthetics			
Example digitization from developers			

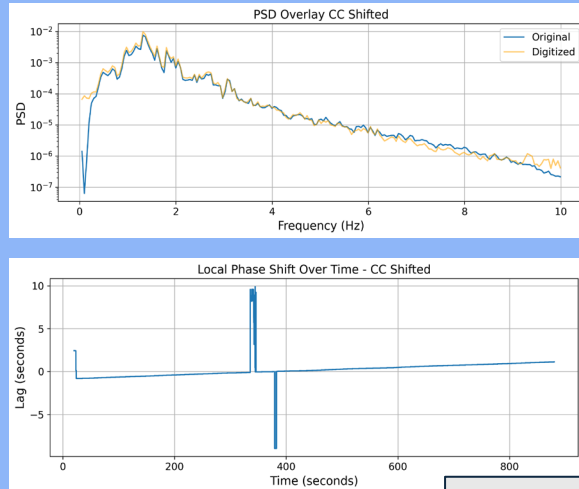
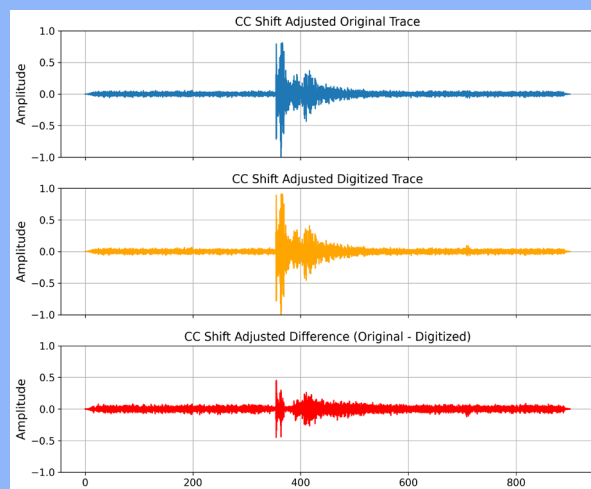


Example Result – 2017 DPRK Test

We present a test example in the form of a 15 minute noise trace with a sample from the 3 September 2017 DPRK test spliced in. Digitized traces were shifted to maximize cross correlation, filtered to show 0.1 – 10 Hz, and normalized for comparison.

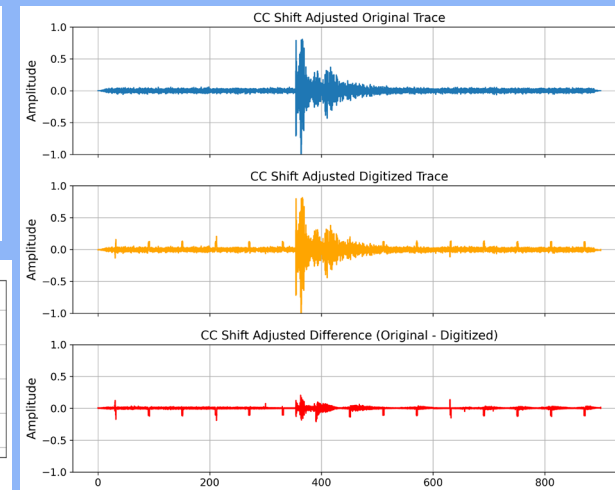
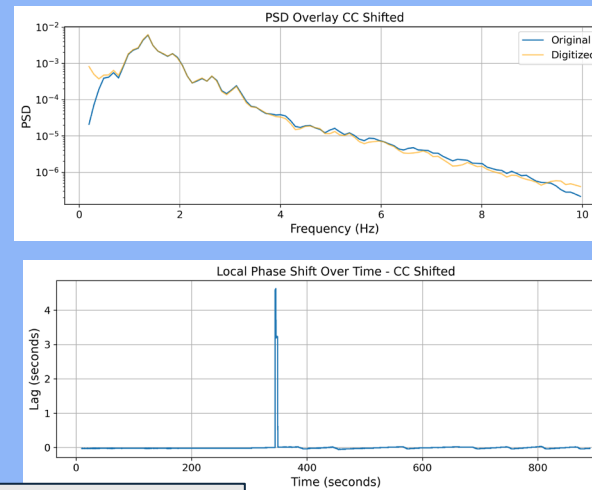
DigitSeis

- *Cross correlation adjustment added – shift +1.1626 seconds*
- *Progressive phase shift along trace duration – zero crossing shifted to first break of impulsive event (point of maximum correlation)*
- *Handled time marks well*
- *Good recovery of frequency band of interest (0.1 – 10 Hz)*
- *Digitized amplitudes exceeded original trace for event.*



Teseo

- *Small cross correlation adjustment added – shift -0.0100 seconds*
- *Good phase representation throughout waveform*
- *Time marks not corrected and clearly visible on digitized trace.*
- *Good fidelity for frequency band of interest (0.1 – 10 Hz)*



Local phase shift over the duration of the digitized trace

Conclusion

DigitSeis

- IS NOT open source
- Provides the greatest level of automation (but manual intervention still needed)
- Faster performance - BUT limited to WWSSN-style records

Produces full digital waveforms

SKATE

- IS open source
- Local installation is not straightforward
- Seems to perform faster but the output is fractured in numerous segments
- Output is CSV of the complete image

DOES NOT
produce full digital waveforms

Teseo

- IS open source
- Is the closest to hand digitization based on the time and the amount of human intervention required
- Took the longest time

Produces full digital waveforms