Quality Control of Heterogeneous IMS Stations #336

Presenter: Kenneth A. Macpherson
Staff of Wilson Alaska Technical Center

This work was supported by the Nuclear Arms Control Technology (NACT) Program at Defense Threat Reduction Agency (DTRA)

DISTRIBUTION STATEMENT: Cleared for Release
Motivation: Detecting infrasonic or seismic events such as nuclear detonations is heavily dependent on data quality (DQ), and Station Operators are responsible for monitoring waveform data for completeness, transient noise, spikes, and changing or anomalous instrument response (Marty, 2010)

- The Wilson Alaska Technical Center (WATC) has long used an array of DQ tools to monitor state of health and completeness, but this project augments existing DQ procedures by adding a metric-based waveform data quality control (QC) scheme

Challenges: The diverse station-types monitored by WATC, which include infrasound and seismic arrays, broadband seismic, short-period seismic, and meteorological channels, are challenging for existing QC tools such as IRIS MUSTANG/ISPAQ (Caset, et al., 2018)

Approach: The new WATC QC system leverages existing tools, but has been built-out and customized to accommodate diverse channel types. We make heavy use of ObsPy to customize the system (Beyreuther et al., 2010)
Metric-based QC: Doctrine and Terminology

**Metrics**

A measurement (or algorithm to generate a measurement) that is indicative of some aspect of data quality

- facts
- observations
- raw data

`dead_channel_lin = 1.8`

**Thresholds**

A metric or combination of metrics combined with a cut-off value

- organize metrics
- identify and alert to potential data defects
- reduce analyst workload

`num_gap/percent_availability > .12`

`dead_channel_lin < 2.0`

**Analysis**

A human must investigate potential issues flagged by thresholds

- review issues
- inspect waveforms
- develop an interpretation
- provide actionable intelligence

`Dead sensor at KDAK`
Core Software

- **ISPAQ**: IRIS System for Portable Assessment of Quality
  - Python/R code for station operators to generate MUSTANG metrics locally
  - 40 metrics available
  - ISPAQ 3.0 writes metrics to a SQLite database

- **QuARG**: Quality Assurance Report Generator
  - Uses thresholds to identify potential issues and alert the analyst
  - Generates QC report
  - Employs a ticketing system

- **ObsPy**: Python toolbox for seismology
  - Flexible through pull requests
  - Used to compute metrics for channels not currently handled by ISPAQ
WATC Metric-based Waveform QC System

Seiscomp3 FDSN server
- waveform data

**ISPAQ 3.0**
1. Computes simple metrics:
   - percent_availability, num_gaps, num_spikes, sample_rms, sample_unique, sample_mean
2. Computes spectral metrics:
   - pct_above_nlnm, pct_below_nlnm, dead_channel_lin
3. Generates plots of PSDs and PDFs

**auto_ispaq** runs ISPAQ and WATC ISPAQ Tools at the end of the UTC day

rasters of noise spectra

**infrasound_noise**:
1. Computes pct_above_nm, pct_below_nm, dead_channel_lin, dead_channel_gsn, array_rms
2. Writes to "standard" ispaq output

**plot_summary**:
- computes station quality indices and generates a weekly summary report

**ispaq2influxdb**:
- write metrics to a database for display

**sqlite**
- daily metrics

**QuARG**
- Helps the analyst:
  1. identify issues
  2. create problem tickets
  3. generate reports

**Analyst Reviewed Report (weekly)**
1. displayed on internal website
2. files distributed as needed

**metadata**

**Grafana**
- quasi-real time display of metrics

**influxdb**
Noise Spectra for Infrasound

- ISPAQ and ObsPy use the Mcnamara and Buland (2004) method of computing power spectral densities (PSD)
- This method differentiates seismic data to acceleration in the response removal stage, which is not required for infrasound
- We modified the ObsPy Signal library to correctly handle PSD/PDFs for infrasound
- Also added ambient acoustic noise models from Brown (2012)

```python
# Create a psd object for infrasound channel
tr = st[0]
ppsd = PPSD(tr.stats, metadata = inv,
special_handling = 'infrasound',
db_bins = (-100, 40, 1.))
ppsd.add(st)
ppsd.plot(xaxis_frequency = True,
period_lim = (0.01, 10),
cmap=pqlx)
```

New feature!
We also added MUSTANG-like spectral metrics to the ObsPy PPSD object:
- `get_pct_above_nm()`: percentage of spectra above IDC high noise model
- `get_pct_below_nm()`: percentage of spectra below IDC low noise model
- `get_dead_channel_lin()`: residual from fitting a line to spectra
- `get_dead_channel_gsn()`: spectra more than 5 dB below low model near the microbarom

```python
In [8]: ppsd.get_pct_above_nm()
Out[8]: 26.786688488816147
In [9]: ppsd.get_pct_below_nm()
Out[9]: 0.0
In [10]: ppsd.get_dead_channel_lin()
Out[10]: 3.6031948727904397
In [11]: ppsd.get_dead_channel_gsn()
Out[11]: 0
```
Custom Metrics

- The system is extensible and allows adding custom metrics
- **array_rms**: metric designed to detect transients on single element of an infrasound array
  - Grab 24 hours of data for all elements
  - Find the standard deviation of each element for the day
  - Find the mean of the standard deviation for all other elements
  - The metric is the ratio of the element standard deviation to the array average
  - If ratio is large, there is likely transient noise on a single element

For an $n$ element array, the $array_{rms}$ for the $i$th element is given by

$$\text{array}_{rms,i} = \frac{\sigma_i}{\bar{\sigma}[1,n] \setminus i}$$

Where the numerator is the standard deviation of the $i$th element and the denominator is the mean standard deviation of the array excluding the $i$th element

![Graph showing spike in array_rms indicating a transient](image)
Defect Detection Examples

Ex. 1 Metadata problem on broadband
Consistently high noise
Spectra mostly above model
Corrected response

Ex. 2 Dead Weather Sensor
Small number of unique values each day
Digitizer noise
Waveform transients
cause: Site visit!

Ex. 3 Transient on infrasound
Spike in array_rms
Short duration noise
WATC uses QuARG to generate a detailed analyst-reviewed QC weekly report and also distributes a single-page summary overview.

WATC has developed a metric-based station quality index ($Q_s$) to rate stations. A channel that is defect-free has a score of “1”, while a score of “0” would indicate an outage, digitizer noise, etc.

$$Q_s = \left( 1 - \frac{1}{n} \sum_{i=1}^{n} \text{dead-channel}_{i,\text{gsn}} \right) \times \left( 1 - \frac{1}{100} \sum_{i=1}^{n} \text{pct}_{i,\text{above-nm}} \right) \times \ldots$$

$$\left( 1 - \frac{1}{n} \sum_{i=1}^{n} \text{num}_{i,\text{spikes}} \right) \times \left( 1 - \frac{1}{\text{max num spikes}} \right) \times \left( \frac{1}{100} \sum_{i=1}^{n} \text{percent}_{i,\text{availability}} \right)$$

where $n$ is the number of days.

Summary plots provide insight into $Q_s$ score. $Q_s$ useful for summarizing station and network performance.
Conclusions

- We developed a **metric-based waveform QC scheme**, augmenting existing monitoring
- The system **accommodates diverse channel types** including seismic, infrasound, and environmental
- The system is **extensible** through the use of custom metrics
- The system **generates network intelligence** including completeness, transient detection, and spectral characteristics
- The system **communicates network intelligence** to engineers and data consumers in the form of weekly detailed and summary reports

Select References:


Robert Casey, Mary E. Templeton, Gillian Sharer, Laura Keyson, Bruce R. Weertman, Tim Ahern; Assuring the Quality of IRIS Data with MUSTANG. Seismological Research Letters 2018;; 89 (2A): 630–639.
