Wave experiment

Diego Domenzain

Constants and units

$\varepsilon_o = 8.85 \cdot 10^{-12}$	F/m	vacuum permittivity
$\mu_o = 4\pi \cdot 10^{-7}$	H/m	vacuum permeability
$c = \frac{1}{\sqrt{\varepsilon_o \mu_o}} = 299792458$	m/s	light speed in vacuum

Table 1: Constants.

ns	$\rightarrow$	s	$\cdot 10^{-9}$
MHz	$\rightarrow$	Hz	$\cdot 10^{6}$
GHz	$\rightarrow$	Hz	$\cdot 10^{9}$
MHz	$\rightarrow$	GHz	$\cdot 10^{-3}$

Table 2: Unit conversions.

Before going to the field

- Choose f_o and maximum target permittivity ε_{max}.
- Compute λ_o,

$$\lambda_o = \frac{c}{f_o \sqrt{\varepsilon_{max}}}, \tag{1}$$

$$\approx \frac{300}{f_{o,MHz}\sqrt{\varepsilon_{max}}} \quad [1/m]. \tag{2}$$

- Choose $\Delta sr \geq \lambda_o$ and compute $\Delta r \leq \lambda_o/2$. For example,

$$\Delta sr = \lambda_o + \text{odometer-wheel} \qquad \text{and} \qquad \Delta r = \lambda_o/4.$$

f_o MHz	λ_o m	$\lambda_o/4$ m
50	1	0.2
100	0.5	0.1
200	0.25	0.05
250	0.2	0.05
500	0.1	0.02

Table 3: Approximate wavelengths (and spacings) for $\varepsilon_{max} = 30$. You can do your own by computing $\lambda_o = 300/f_{o,MHz}/\sqrt{\varepsilon_{max}}$.

After going to the field

Save data and time-shift: `ss2gerjoii_w.m`

1. Loop through done survey lines and save all of the data to `.mat` in disk, and store r and t as r_{all} and t_{all} in memory.
2. Loop over all saved lines and rewrite them,
   (a) get r and t for line,
   (b) correct for time-shift: $t \leftarrow t - t_{all}(t = t_o)$, $d^o \leftarrow d^o$ corrected,
   (c) transform $\{r, t, \Delta t, \Delta r, \Delta sr, f_o\}$ to m, ns and GHz,
   (d) use Δs, Δsr and r_{all} to create source-receivers tuples $\{s, r\}$ in real coordinates.
3. Store all $\{s, r\}$ to disk.

Quick-see and compute $\Delta t, \Delta x$: `datavis_w.m`

1. Choose data d^o together with $\{r, t, \Delta t, \Delta r, \Delta sr, f_o\}$.
2. Dewow.
3. See d^o in the frequency domain and filter out unwanted signal with a bandpass filter $[f_{min},\ f_{max}]$.
4. Find first arrival event time t_{fa} before which all recordings should be silent. For example, $t_{fa} = \Delta sr/c$.
5. Choose up to which receivers the data looks good.
6. Choose v_{min} and compute $\varepsilon_{max} = (c/v_{min})^2$ to see if v_{min} makes sense.

7. Compute Δx using the *wavelength condition*,

$$\Delta x = \frac{v_{min}}{n_\lambda \cdot f_{max}}. \tag{3}$$

8. Choose $\varepsilon_{min} \geq 1$ and compute $\Delta t_\bullet$ with the *cfl condition*,

$$v_{max} = \frac{c}{\sqrt{\varepsilon_{min}}}, \tag{4}$$

$$\Delta t_\bullet = c_f \cdot \frac{1}{v_{max}\sqrt{1/\Delta x^2 + 1/\Delta z^2}}. \tag{5}$$

   Most likely, $\Delta t_\bullet < \Delta t$ because the gpr system computes Δt to satisfy the *Nyquist condition* on the frequency it can measure ($\approx 1.2\,GHz$).

Get the data ready for the fwi scheme: `data2fwi_w.m`

1. Choose data d^o together with $\{r, t, \Delta t, \Delta r, \Delta sr, f_o\}$.

2. Dewow d^o.

3. Bandpass filter d^o with $[f_{min},\ f_{max}]$.

4. Remove unwanted receivers.

5. Perform 2.5d→2d conversion.

6. Mute unwanted events. For example, everything before t_{fa}.

7. Interpolate d^o on time vector $t_\bullet = t(1) : \Delta t_\bullet : t(n_t)$ and set,

$$t \leftarrow t_\bullet \tag{6}$$

$$\Delta t \leftarrow \Delta t_\bullet. \tag{7}$$

   The *spline* interpolation works best.

8. Estimate source wavelet.

9. Save d^o to disk with its respective $\{s, r\}$ on binned coordinates in the new discretized vectors x and z.

10. Loop until all data has been preprocessed.

Get parameters ready for fwi scheme: `param2fwi_w.m`

- Complete structures,
  - `parame_`
  - `geome_`
  - `finite_`

  with the space and time parameters computed in `data2fwi_w.m`.

Complete `gerjoii_` structure for inversion: `w_load.m`

For one `radargram_` structure,

- Load source location $[i_z, i_x]$ a 1×2 vector.
- Load receiver locations $[i_x, i_z]$ an $n_r \times 2$ matrix.
- Modify source and receivers to account for PML and air.
- Build M_w.
- Load d^o (of size $n_t \times n_r$) and $\mathrm{std}(d^o)$ (of size $n_t n_r \times 1$).
- Load source wavelet.

The source wavelet is updated and saved to disk in the `wvlets_` structure after each iteration.

Inversion routine

1. Begin while loop.
2. Compute update,
   - Begin for loop through experiments,
   - load source-receivers, d^o, $\mathrm{std}(d^o)$ and wavelet,
   - build M_w,
   - forward run,
   - evaluate objective function,
   - compute gradient
   - compute step-size,

- compute and store update,
- estimate source wavelet and store,
- end for loop.

Stack all updates into one update.

3. Update.

4. End while loop.

Extracting more information

1. Build source-receiver matrix of size $n_s \times n_r$ on binned coordinates in discretized vectors x and z.

2. Interferometry.

3. Velocity semblance.

4. Source estimation.