1.5D tau-p internal multiple prediction in Seismic Unix

Andrew Iverson*, Kris Innanen, and Daniel Trad
andrew.iverson1@ucalgary.ca

Abstract

The Inverse scattering series is implemented in 1.5 dimensions in the tau-p domain for internal multiple prediction. The computational time for the chosen model is reduced by a factor of approximately 120 in Seismic Unix with parallel processing in comparison to the MATLAB implementation. It is also shown how artifacts from the prediction in 1.5D tau-p can be minimized through a time domain tau-p transform and as shown by Sun and Innanen (2015) a spatial cosine taper.

Inverse Scattering Series

$$b_3(p_g,\omega) = \int_{-\infty}^{\infty} d\tau_1 e^{i\omega\tau_1} b_1(p_g,\tau_1) \int_{-\infty}^{\tau_1-\varepsilon} d\tau_2 e^{-i\omega\tau_2} b_1(p_g,\tau_2) \int_{\tau_2+\varepsilon}^{\infty} d\tau_3 e^{i\omega\tau_3} b_1(p_g,\tau_3)$$

b_3 is the interbed multiple prediction, b_1 is the prepared input data, τ_1, τ_2 and τ_3 are chosen to satisfy lower-higher-lower criteria and ε is the search limiting parameter

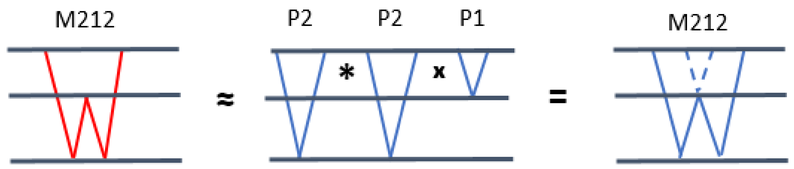


FIG. 1. Schematic displaying how a multiple can be replicated with a combination of primaries through a convolution (*) and correlation (x), or through multiplication in the Fourier domain

Code Comparison

Ex. 1.

MATLAB

```
intPos = b1_p_tau(:,ii).*Ipos;
intNeg = b1_p_tau(:,ii).*Ineg;
```

Seismic Unix

```
float complex intPos[nt];
float complex intNeg[nt];
for(int it=0; it<nt; ++it) {
  intPos[it] = data[ip][it]*Ipos[it][iw];
  intNeg[it] = data[ip][it]*Ineg[it][iw];
}
```

Ex. 2.

MATLAB

```
for kk = tBeg:tEnd
    INNER = dt*sum(intPos(kk+epsilon:tEnd));
    pred_p_tau(jj,1) = pred_p_tau(jj,1) + intNeg(kk)*INNER*INNER;
    clear INNER
end
```

Seismic Unix

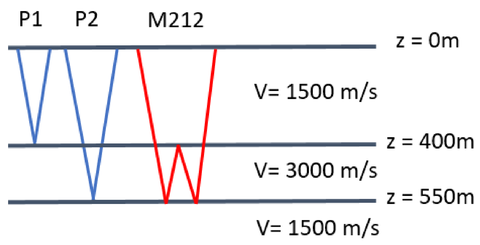
```
for( int it=0; it<nt; ++it) {
  float complex sintPos = 0;
  for( int j=it+e; j<nt; ++j) {
    sintPos = sintPos + intPos[j];
  }
  float complex Inner = dt*sintPos;
  b3[ip][iw] = b3[ip][iw] + intNeg[it]*Inner*Inner;
}
```

FIG. 2. Two examples from internal multiple prediction code in MATLAB and Seismic Unix

Though there are syntax differences the bulk of the algorithm is largely equivalent. Both algorithms operate by solving for all possible subevents for each frequency. This is then repeated for all slowness values (p) for the given dataset

Shot Record

FIG. 3. Geologic model displaying velocities and depths used with the two primaries and first order internal multiple



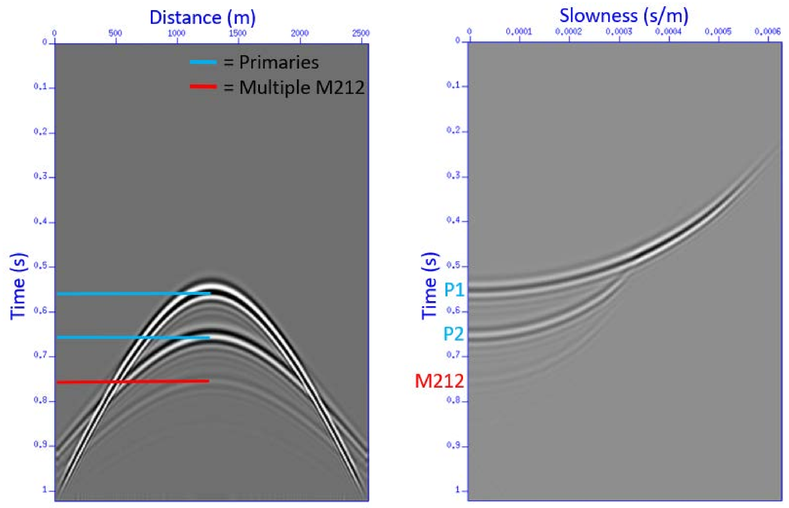


FIG. 4. (Left) Seismic data in SU (Right) Prepared data in tau-p domain in SU

MATLAB Prediction

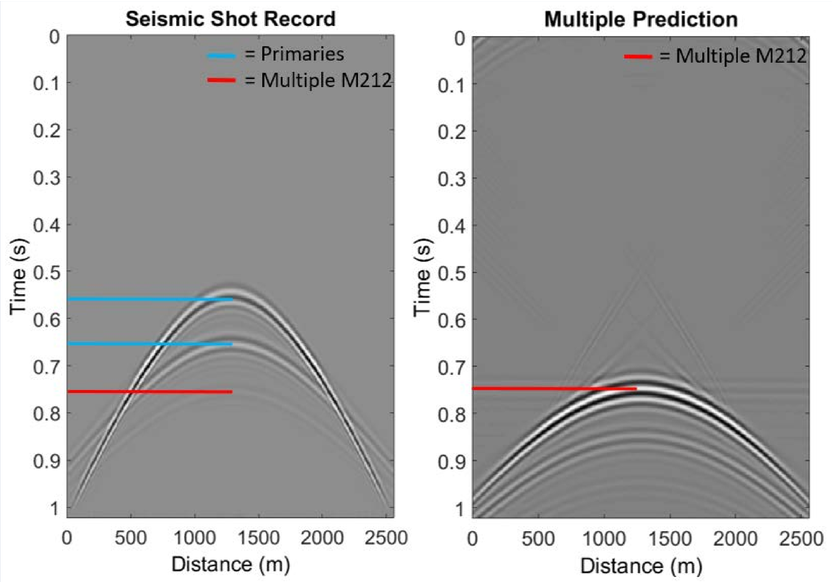


FIG. 5. (Left) Synthetic seismic shot record (Right) 1.5D tau-p multiple prediction in MATLAB

Accurately predicted the internal multiples present in the data set. The computational time was approximately 10 minutes. This data set was 512x256 samples. Numerically this was only computed on approximately 128 slowness values due to the v(z) medium being identical for both positive and negative spatial dimensions.

Seismic Unix Prediction

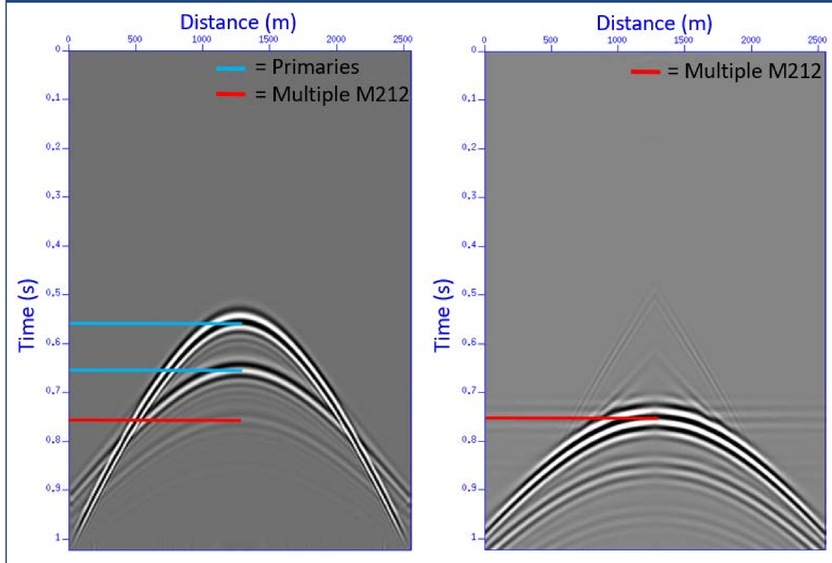


FIG. 6. (Left) Synthetic seismic shot record (Right) 1.5D tau-p multiple prediction in Seismic Unix

The prediction implemented in SU took approximately 21 seconds to complete. Parallel processing was also applied to the algorithm, with the use of 16 threads the computation time was further reduced to approximately 5 seconds.

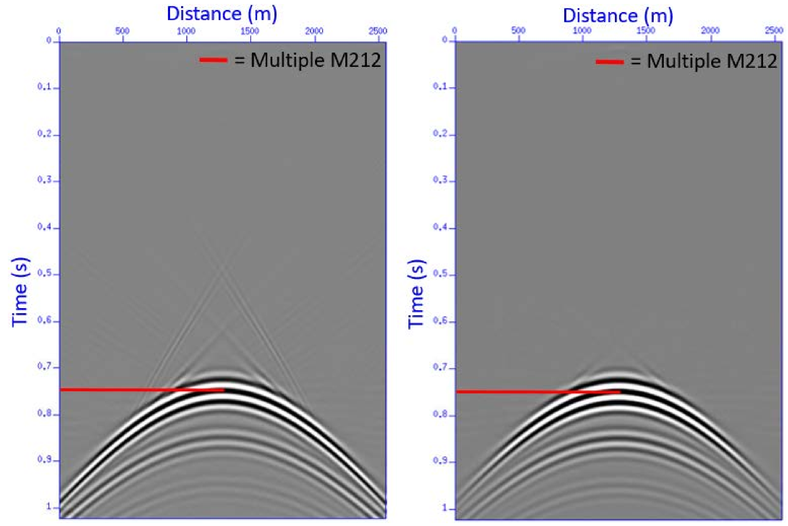


FIG. 7. (Left) Prediction using the time domain tau-p transform (Right) Prediction using an aggressive cosine taper and time domain tau-p transform

With the use of the time domain tau-p transform and a harsh cosine taper the result is an accurate prediction of internal multiples mostly free of artifacts.

Acknowledgments

The authors would like to thank the sponsors of CREWES for the support of this work. This work was also funded by NSERC through the grant CRDPJ 461179-13.