

# Turning Ray Prestack Time Migration with Raytraced Offset-Dependent Velocities

Kevin J. Hellman, *Geotrace*

## Summary

Conventional prestack time migration uses RMS velocities to map input data to output samples. Raytraced traveltimes may be converted to “velocities” at a suite of offset values, and used in place of traditional RMS velocities. This provides an implementation of curved ray migration that is not only more accurate than the 4<sup>th</sup> or 6<sup>th</sup> order approximations, but is also equivalent in computational time to the conventional straight ray (2<sup>nd</sup> order) approach to time migration.

## Introduction

The mainstream use of supercomputers and large scale clusters has enabled the introduction of very accurate and computationally challenging migration algorithms in the 3D depth domain. However, 3D prestack time migration has remained an industry staple, with Kirchhoff methods at the forefront. Longer acquisition offsets – 10 kilometers or more – have stretched the assumptions of conventional Kirchhoff time migration about as far as they can go.

Time migration makes the implicit assumption that the velocities are one dimensional, and that these velocities can be globally approximated by their RMS value at each output image point. With this assumption, the traveltime no longer depends directly on the surface locations of the trace, but only on the offset between the image location and the trace source/receiver position, along with the vertical time to the image point. This results in the familiar double square root migration equation

$$t_{input} = \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_s^2}{V_t^2}} + \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_r^2}{V_t^2}}, \quad (1)$$

where  $d_s$  and  $d_r$  are the image offsets to the trace source and receiver, respectively,  $t$  is the two-way traveltime for the output image point, and  $V_t$  is the RMS velocity to the image point.(Fig. 1). A brute force approach that has been used to compensate for the errors due to the violation of these assumptions has been to apply Kirchhoff depth migration with a layered velocity model, then stretch the results back to time, either during or after the migration. This method would imply a migration equation of the form

$$t_{input} = \sqrt{\left(\frac{1}{2}t\right)^2 + \tau_t(d_s)} + \sqrt{\left(\frac{1}{2}t\right)^2 + \tau_t(d_r)}, \quad (2)$$

where  $\tau_t(d)$  represents the traveltime from the image point to the shot/receiver location. This is a more expensive alternative to time migration, due to the increased I/O and interpolation of traveltime tables that are usually involved in a depth migration.

New approaches to curved ray migration have included 4<sup>th</sup> and 6<sup>th</sup> order approximations to the moveout equation (e.g. Wang, et.al., 2002). Now the migration equation becomes

$$t_{input} = \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_s^2}{V_t^2} + c_4 d_s^4 + c_6 d_s^6} + \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_r^2}{V_t^2} + c_4 d_r^4 + c_6 d_r^6} \quad (3)$$

which requires considerably more computation. Since the extra computations are in the kernel of the migration, these curved ray methods have become even more time-consuming and expensive.

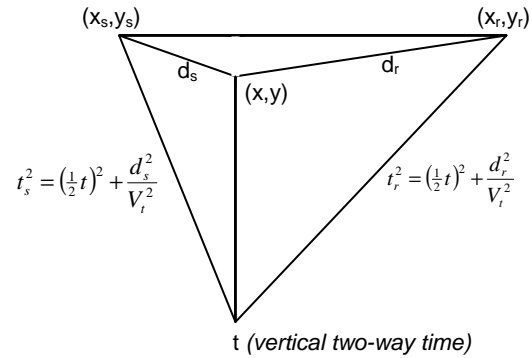


Fig. 1. Prestack migration geometry.

Comparing (1) and (2), it is apparent that it is the RMS velocity approximation that introduces error into the migration. The “bent rays” that would have come through the one dimensional velocity model are replaced with a “straight ray” that has come through a constant pseudo-velocity, which is based on the RMS average of an implied stack of interval velocity layers (Fig. 2). For larger offsets and imaging angles, this velocity is too slow, resulting in overcorrection of events, and the familiar “hockey sticking” of the farther offsets (Levin, 2003). Equation (3) attempts to correct the error by adding terms to the truncated Taylor

## Turning Ray Prestack Time Migration

series fit that is equation (1). This becomes complicated because the  $c_4$  and  $c_6$  terms also depend on the implied stack of interval velocities (Fig.2), and in principle must be calculated for every CDP location and output time. An alternative approach is to raytrace the traveltimes, but cast the results in terms of velocities, rather than times, to enable an efficient implementation of a turning ray migration.

### Migration velocities from raytraced times

The raytracing for time migration is particularly simple, due to the one dimensional nature of the velocity field at each image location. Of course, this one dimensional velocity will typically vary with spatial location. After stretching the velocities from time to depth, the traveltimes  $\tau_i(d)$  are determined through this *depth* velocity model for a two dimensional array of depths and image offsets. The depth axis is subsequently stretched back to vertical two-way time, and the raytraced time values may then be expressed as “pseudo-velocities” via the simple distance/time equation

$$V_t^2(d) = \frac{d^2 + z_r^2}{\tau_i^2(d)}. \quad (4)$$

The migration equation now becomes

$$t_{input} = \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_s^2}{V_t^2(d_s)}} + \sqrt{\left(\frac{1}{2}t\right)^2 + \frac{d_r^2}{V_t^2(d_r)}}, \quad (5)$$

which can be used directly in a conventional migration program, with no significant computational penalty. The only modification is the addition of an offset dimension to the velocity traces, forming a “gather” within the usual velocity volume (Fig. 3). These velocities are designed to be those which will result in a  $t_{input}$  that would have been obtained by explicit raytracing, without the error from the RMS approximation. Although these are still treated as “straight rays”, they produce a true turning ray result.

The generation of the  $V_t(d)$  volume is carried out as a separate step, prior to the migration. A layered interval velocity model is constructed from the original RMS velocity trace. A simple Dix inversion can result in unstable interval velocities, so a constrained inversion to interval velocities is used instead (Harlan, 1999). A fan of rays is then propagated from layer to layer via Snell’s law (Fig. 4), and traveltimes are interpolated to the equispaced image offsets. Since the velocity model is so simple, the fan can be quite fine, and the resulting times and pseudo-

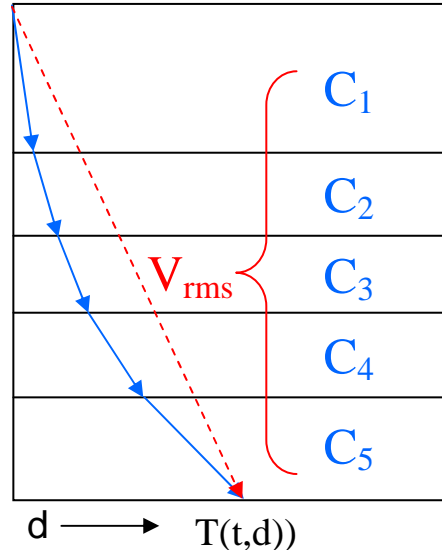


Fig. 2. Error in RMS velocity approximation.

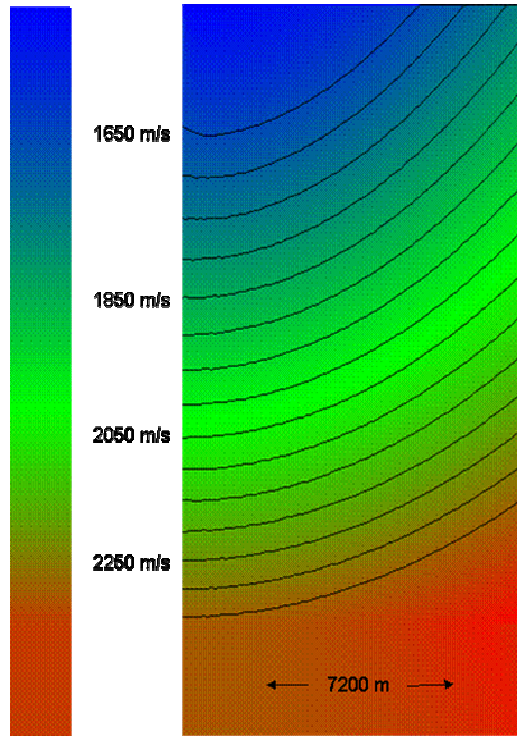


Fig. 3. One dimensional velocity model for an image location (left) and the two dimensional offset dependent velocity “gather” (right) generated by raytracing and converting the traveltimes to velocity.

## Turning Ray Prestack Time Migration

velocities are easily spaced to adequately cover the range of image offsets. A choice may be made when the rays turn critical: they can generate upward traveling rays, which results in a “minimum time” migration, or one can use the horizontally traveling rays to create a “minimum path” migration. Each location in the velocity model (usually sparse compared to the image locations) produces a gather of velocity traces, determined according to equation (4).

Although the derivation of the migration equation (5) is in terms of velocity, the actual implementation would interpolate and apply these as slownesses. This is no different than how one would handle traveltimes in a depth migration. In fact, it makes sense to store traveltimes as “pseudo-slownesses” in order to facilitate interpolations and eliminate the division from the inner loop of the migration. The turning ray time migration approach presented here is merely an adaptation of this “pseudo-slowness” representation.

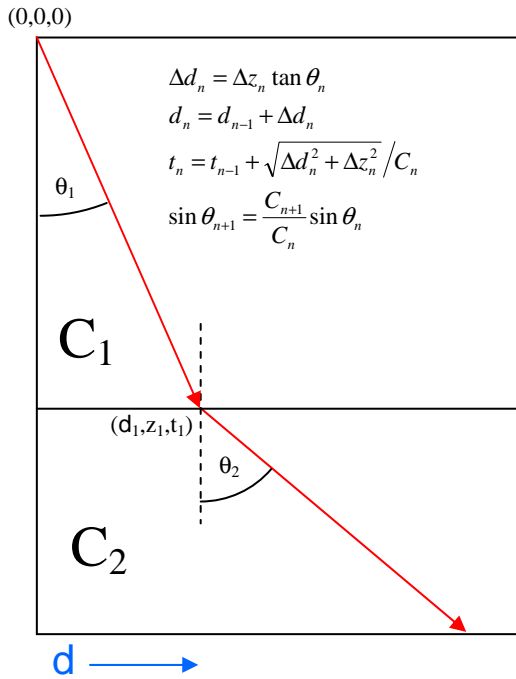


Fig. 4. Illustration of the simple raytracing required to populate the offset-dependent velocity gathers.

### Examples

Figure 5 shows a synthetic common midpoint gather, generated from a  $v_0 + kz$  model, with spikes at the exact traveltimes, and followed by a bandpass filter. This gather has then been migrated with both a conventional prestack time migration, and with the proposed raytraced method.

The RMS velocity function, and the offset-dependent velocity gather for this model are the same as those shown in Fig. 3. The gathers are clearly flattened better at the far offsets.

Figure 6 shows a real data example from a land survey around a salt dome. The steeply dipping flanks are better imaged with the proposed turning ray migration.

### Conclusions

The conventional “straight ray” migration has been highly criticized in recent years, but it is the choice of the velocity used in the conventional calculation that is responsible for the migration errors. By converting raytraced traveltimes into offset-dependent velocities, a simple adaptation of straight ray migration becomes equivalent to a “turning ray” migration. This represents the most accurate implementation of the PSTM algorithm; anything better would necessarily be a depth migration.

### Acknowledgements

I would like to thank Geotrace for allowing me to publish this work.

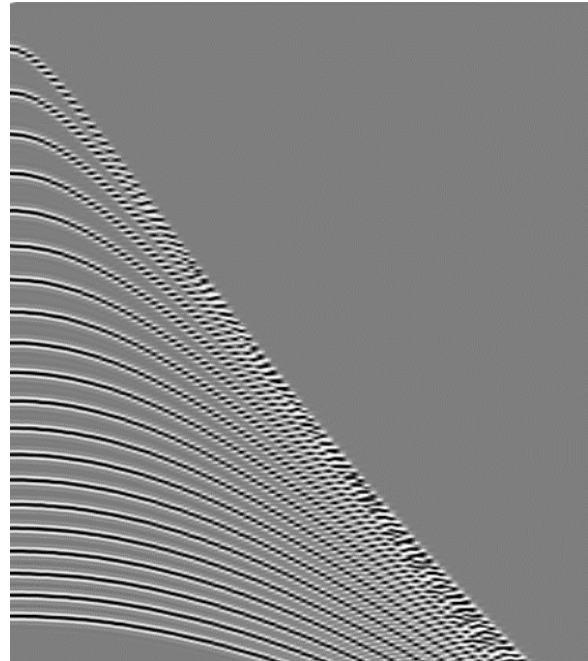


Fig. 5a. Gather of 96 traces with a far offset 7200m, through an interval velocity of  $V(z) = 1500 + 0.5 \cdot z$ .

## Turning Ray Prestack Time Migration

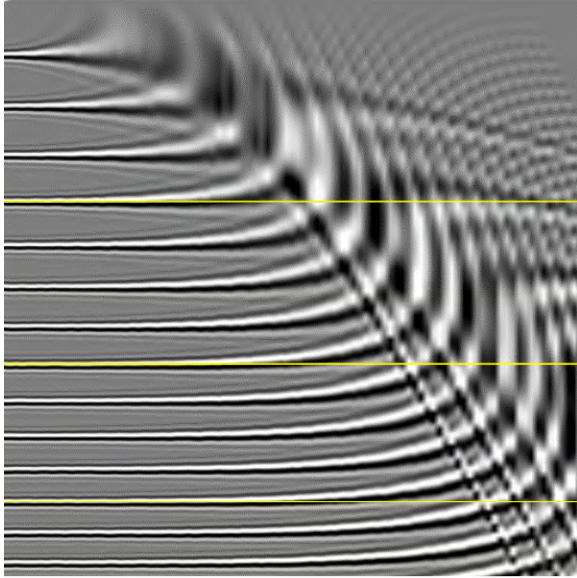


Fig 5b. Conventional straight-ray migration with RMS velocity.

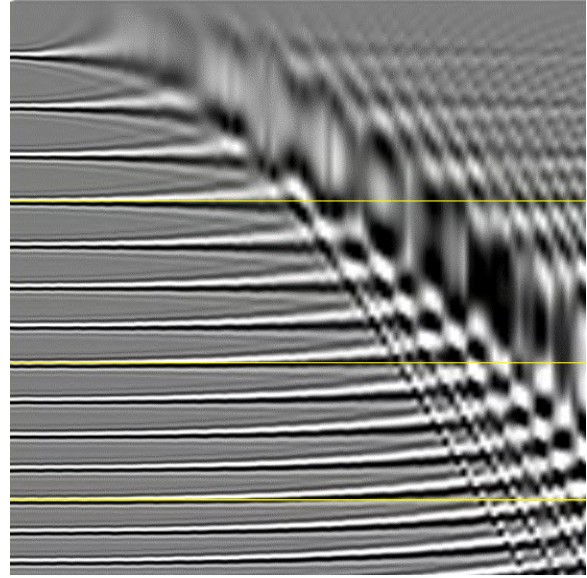


Fig 5c. Migration with raytraced offset-dependent velocities.

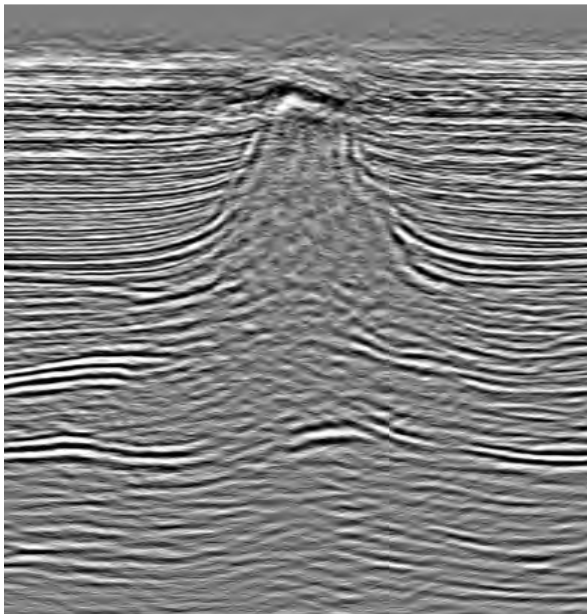


Fig 6a. Prestack time migration using conventional straight ray migration and RMS velocities.

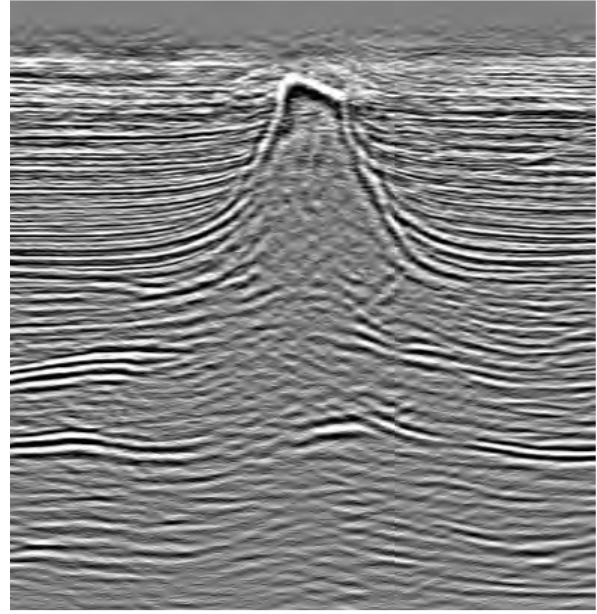


Fig 6b. Prestack time migration using proposed method and raytraced velocities.