The computation of traveltimes when assuming locally circular or spherical wavefronts

John C. Bancroft and Xiang Du*

Summary

A method is presented for identifying the source of a locally circular or spherical wavefront given the traveltimes at arbitrary locations. For 2D data, the center of the circular wavefront is computed from three traveltimes recorded at three arbitrary locations. Applications to 3D data requires four traveltimes recorded at four arbitrary locations.

This method is suited for a number of applications such as mapping traveltimes that are computed along sparse raypaths to gridded traveltimes, the monitoring of micro-seismic events caused by fraccing, or to the possible prediction of landslides in geologically unstable areas.

The analytic solutions for both 2D and 3D produce two solutions from which one must be chosen based on neighbouring conditions or by using another known traveltime and its location.

Introduction

Traveltime computations continue to be an integral part of modelling and seismic imaging by providing efficient kinematic information on the location of propagated energy. The traveltimes may be computed analytically using simplifying assumptions such as RMS velocities, or may be estimated within a complex geological structure using raytracing or gridded traveltimes.

Traveltimes on a grid may be computed using a finite difference solution to the Eikonal equation, however that solution is based on a plane wave assumptions (Bancroft 2005a). However the wavefront may be curved and approximated by a circle. Solving for an unknown traveltime, such as a corner of the square, may involve estimating the center of the circle (center of curvature) and then computing the traveltime to the desired location. The traveltime computations assume the velocity to be locally constant within the square, but this velocity is extended outside the square to the location of the center of curvature.

Traveltimes may also be calculated along raypaths, and these traveltime may then be mapped to a grid. In this case the traveltimes along the raypaths do not fall on a regular grid and increase the complexity of the computation. A circular wavefront can also be estimated from these raypath traveltimes, and then the center of curvature used to compute travel times on neighbouring grids points.

Micro-seismic events occur during well fraccing, well injections, or in areas of geological stress. Knowledge of the locations of these events aids in identifying the extent of the fraccing or injection.

Micro-seismic events may also be used to indicate a potential hazard such as an impending landslide or earthquake. These areas of interest are continuously monitored by a number of stationary receivers, and when an event occurs, the location is estimated from the corresponding times of the receivers. A number of algorithms are available for computing the source of the event such as pre-computing the traveltimes from all possible source locations, then using a search technique to find the optimum source location.

It is possible to locate the center of curvature using the differences between the traveltimes (Bancroft 2005b). The difference between two traveltimes defines two paths of a hyperbola with the receiver locations being the focus points. One path of the hyperbola can be chosen based on the relative amplitudes of the two traveltimes. Hyperbolic curves can then be defined for each pair of recording points. The intersection points of these hyperbolae identify potential centers of curvature. The recording points can be at any locations off grid, but the computations become simpler when they are located on regular grid points at the corners of a square.

The method to be described is useful for all these applications as it computes the center of curvature directly using three known times and locations for 2D data and four known times and locations for 3D data.

Theory

An alternate method for computing the center of curvature from recorded traveltimes is based on the tangency of circles for 2D data and spheres for 3D data. We will develop the method for 2D data as it can be visualized and then extend the method to 3D using parallel development.

We start by assuming a source location (x_0, z_0) and three arbitrarily located receiver points (x_1, z_1), (x_2, z_2), and (x_3, z_3) as illustrated in Figure 1. The traveltimes recorded at these locations are t_1, t_2, and t_3 and are clock times, i.e. <u>not</u> the travel times from the source point. We assume the event started at the source location at time t_0. The traveltimes between the source and receiver locations are then defined by $t_{01} = (t_1 - t_0)$, $t_{02} = (t_2 - t_0)$, and $t_{03} = (t_3 -$

Computation of traveltimes for spherical wavefronts

t_0). These traveltimes may be used to define radii r_{01}, r_{02}, and r_{03} or distances from the source to the receiver locations. Rather than draw circles centered at the source point, we draw circles centered at the receiver locations as illustrated in Figure 2. The intersection of these circles define the source location. However, we do not know these radii at the beginning of our problem.

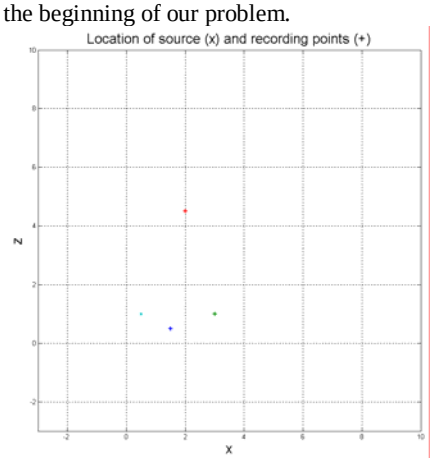

Figure 1 The geometry of a source location (x) and three receiver locations (+).

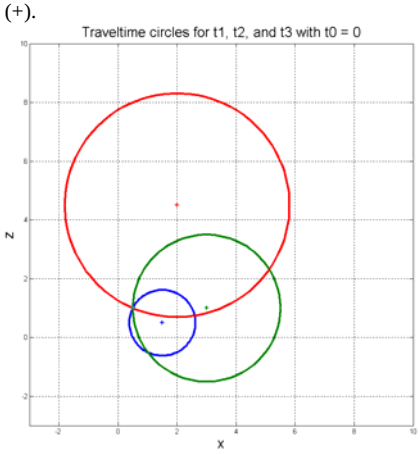

Figure 2 Circles drawn with source-to-receiver radii, centered at the receiver. The intersection of these circles is the source location that we now know, but will be trying to find. Notice that the radii are defined using the source time t_0, which is known at this time but which we will be also trying to find.

We do know the times t_1, t_2, and t_3 and we can draw circles representing their radii as illustrated in Figure 3. The radii of these circles is the sum of the source time t_0 plus the traveltime to the receiver, i.e. $r_1 = v(t_0 + t_{01})$, $r_2 = v(t_0 + t_{02})$, and $r_3 = v(t_0 + t_{03})$. It is now important to note that these three circles pass beyond the source point with the same distance $v*t_0$ as illustrated in Figure 3.

Figure 3 also contains a cyan circle about the source point with radius $v*t_0$. This circle is tangent to the three other circles. The points of tangency are on lines drawn from the receiver location and through the source. At this point, we don't know the location or time t_0 of the source point. What we do know is the radius of the three circles in Figure 3, and that the source location can be found by locating a circle (cyan in Figure 3) that is tangent to these three known circles.

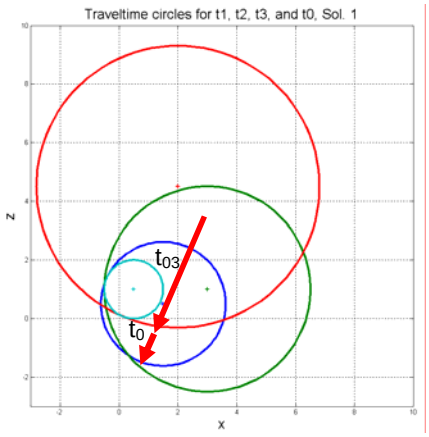

Figure 3 Three circles (red, green and blue) from the receiver locations representing t_0 = 1.0 plus the traveltime from the source to the receiver. A cyan colored circle represents the radius of $v*t_0$ and is tangent to the other circles.

The problem of finding a circle that is tangent to three given circles has been of interest for centuries and the first solution is attributed to Apollonius of Perga who was born in 261 BC. Geometric constructions are possible but complex, illustrating that a solution should "involve nothing more than quadratic equations", (Website 1). Numerous algebraic solutions are also available on the web, and we will follow one based on Website 2.

We start with equations representing the three circles with radii from the source to the three receiver locations, i.e.

$$(x_1 - x_0)^2 + (z_1 - z_0)^2 = v^2(t_1 - t_0)^2 \quad (1)$$

$$(x_2 - x_0)^2 + (z_2 - z_0)^2 = v^2(t_2 - t_0)^2 \quad (2)$$

$$(x_3 - x_0)^2 + (z_3 - z_0)^2 = v^2(t_3 - t_0)^2 . \quad (3)$$

Subtracting equations (2) and (3) from (1) we get

$$2x(x_1 - x_2) + 2z(z_1 - z_2) + (x_2^2 - x_1^2) + (z_2^2 - z_1^2) = 2v^2(t_1 - t_2)t_0 + v^2(t_2^2 - t_1^2) \quad (4)$$

and

$$2x(x_1 - x_3) + 2z(z_1 - z_3) + (x_3^2 - x_1^2) + (z_3^2 - z_1^2) = 2v^2(t_1 - t_3)t_0 + v^2(t_3^2 - t_1^2) . \quad (5)$$

These equations are simplified to

Computation of traveltimes for spherical wavefronts

$$A_1 x_0 + B_1 z_0 + C_1 t_0 = D_1 \quad (6)$$

$$A_2 x_0 + B_2 z_0 + C_2 t_0 = D_2 \quad (7)$$

where

$$A_1 = 2(x_1 - x_2), \quad B_1 = 2(z_1 - z_2), \quad C_1 = -2v^2(t_1 - t_2) \quad (8)$$

$$D_1 = v^2(t_2^2 - t_1^2) - (x_2^2 - x_1^2) - (z_2^2 - z_1^2) \quad (9)$$

and

$$A_2 = 2(x_1 - x_3), \quad B_2 = 2(z_1 - z_3), \quad C_2 = -2v^2(t_1 - t_3) \quad (10)$$

$$D_2 = v^2(t_3^2 - t_1^2) - (x_3^2 - x_1^2) - (z_3^2 - z_1^2) \quad (11)$$

Solving equations (6) and (7) for x_0 and z_0, we get

$$x_0 = \frac{B_2(D_1 - C_1 t_0) - B_1(D_2 - C_2 t_0)}{A_1 B_2 - A_2 B_1} = E_1 t_0 + F_1 \quad (12)$$

$$z_0 = \frac{A_2(D_1 - C_1 t_0) - A_1(D_2 - C_2 t_0)}{A_2 B_1 - A_1 B_2} = E_2 t_0 + F_2 \quad (13)$$

Where

$$E_1 = \frac{B_1 C_2 - B_2 C_1}{A_1 B_2 - A_2 B_1}, \quad F_1 = \frac{B_2 D_1 - B_1 D_2}{A_1 B_2 - A_2 B_1} \quad (14)$$

$$E_2 = \frac{A_1 C_2 - A_2 C_1}{A_2 B_1 - A_1 B_2}, \quad F_2 = \frac{A_2 D_1 - A_1 D_2}{A_2 B_1 - A_1 B_2} \quad (15)$$

Now substitute equations (12) and (13) into equation (1) to get

$$(E_1 t + F_1 - x_1)^2 + (E_2 t_0 + F_2 - z_1)^2 = v^2(t_0 - t_1)^2, \quad (16)$$

where t_0 is the only unknown. Rewriting this equation in quadratic form for t_0 we get

$$\begin{aligned} &t_0^2(v^2 - E_1^2 - E_2^2) + \\ &t_0[-2v^2 t_1 + 2(x_1 - F_1)E_1 + 2(z_1 - F_2)E_2] + \\ &v^2 t_1^2 - (F_1 - x_1)^2 - (F_2 - z_1)^2 = 0 \end{aligned} \quad (17)$$

This equation simplifies to

$$t_0^2 + G t_0 + H = 0, \quad (18)$$

with solutions

$$t_0 = \frac{-G \pm \sqrt{G^2 - 4H}}{2}, \quad (19)$$

where

$$G = \frac{-2v^2 t_1 + 2(x_1 - F_1)E_1 + 2(z_1 - F_2)E_2}{v^2 - E_1^2 - E_2^2} \quad (20)$$

and

$$H = \frac{v^2 t_1^2 - (F_1 - x_1)^2 - (F_2 - z_1)^2}{v^2 - E_1^2 - E_2^2}. \quad (21)$$

Once we have t_0 from equation (19) we then use equations (12) and (13) to solve for x_0 and z_0. MATLAB code for this solution is included in the appendix.

Examples

To test this algorithm we show two cases where we start by defining the source and receiver locations. We then define a value for t_0, and compute t_1, t_2, and t_3. Using only the receiver times and their locations, we compute the two solutions for x_0, z_0, and t_0. For the two examples given, Solution 1 uses the plus sign in equation (19) while Solution 2 uses the minus sign solution. The geometry for the first example is illustrated in Figures 1 through 3. Figure 4 shows the geometry of the alternate solution. Note in the alternate solution for t_0 the circle is tangent to the outside of the circles. The input and results for both cases are included in the appendix.

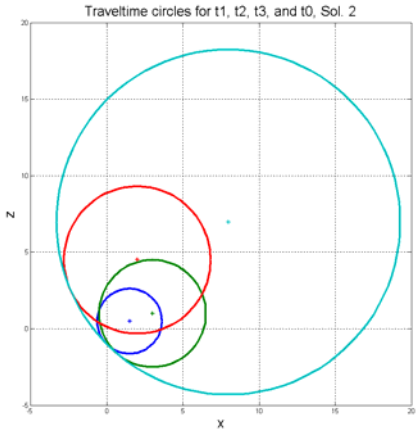

Figure 4 Solution 2 for Example 1 where the t_0 circles is exterior to the three known circles.

Example 2 is illustrated below in Figures 5. Receiver x2 is shifted to the right, enough to change the correct solution to Solution 2 that defines the inner and desired solution.

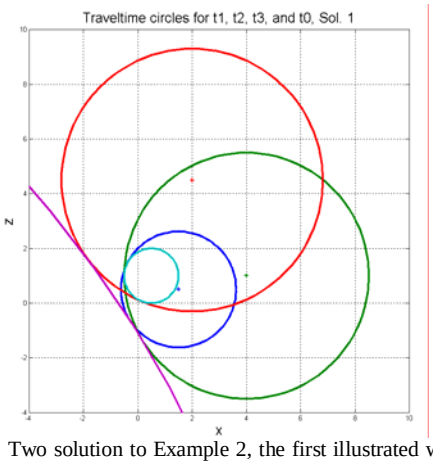

Figure 5 Two solution to Example 2, the first illustrated with the purple curve and the second with the desired cyan circle.

Computation of traveltimes for spherical wavefronts

Solution for 3D data

The 3D solution has four unknowns, x_0, y_0, z_0 and t_0. and we therefore require four independent equations, or four times at four receiver locations. Similar to the 2D case, we start with equations that define spheres from the receiver locations that pass through the source location.

$$(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2 = v^2(t_1 - t_0)^2 \quad (22)$$

$$(x_2 - x_0)^2 + (y_2 - y_0)^2 + (z_2 - z_0)^2 = v^2(t_2 - t_0)^2 \quad (23)$$

$$(x_3 - x_0)^2 + (y_3 - y_0)^2 + (z_3 - z_0)^2 = v^2(t_3 - t_0)^2 \quad (24)$$

$$(x_4 - x_0)^2 + (y_4 - y_0)^2 + (z_4 - z_0)^2 = v^2(t_4 - t_0)^2 \quad (25)$$

Using the same procedure outlined for the 2D case, and tedious algebra we end up with a solution similar to equation (19) that has two possible solutions for t_0, and similar equations that compute x_0, y_0, and z_0.

Comments

In the above solutions, the time t_0 is quite arbitrary and can be virtually any value. Note that the radius of the circles in Figure 3 are dependent on this value and can be very large if t_0 is very large, and could create inaccuracies in the solutions. Since t_0 is arbitrary, so also are the times t_1, t_2, and t_3 in the sense that we could subtract a constant time from each of the values. Subtracting the minimum time from each recorded time will leave one time at zero and the remaining positive. This will have the effect of minimizing the radius of the time circles, with one reduced to a point. The solution now becomes one of fitting a circle through one point and tangent to two circles that should be more efficient than fitting a circle to three other circles.

There are conditions in which a solution is not possible. For example, if all the points are co-linear, then no exact solution is possible. A similar situation exists in the 3D case if all the receivers are in the same plane.

Conclusions

A method was presented that locates the center of curvature for a curved wavefront when given traveltimes at known receiver locations. The receivers may be arbitrarily located.

Applications are suitable for locating micro-seismic events or for mapping raypath traveltimes to a grid.

References

Bancroft, J. C., 2005a, Circular wavefront assumptions for gridded traveltime computations, CREWES Research Report, Vol. 17

Bancroft, J. C., 2005b, The computation of gridded traveltimes when assuming circular wavefronts, 82nd Ann Internat. Mtg, Soc. Expl. Geophys., Expanded Abstracts

Website 1,
htpp://www.mathpages.com/home/kmath113.htm

Website 2,
hrpp://mathworld.wolfram.com/ApolloniusProblem.html

Acknowledgements

The authors thank the CREWES sponsors for supporting this work.

Appendix

MATLAB code for 2D solution

```
A1=2*(x1-x2);        B1=2*(z1-z2);        C1=-2*v*v*(t1-t2);
D1=v*v*(t2*t2-t1*t1)-(x2*x2-x1*x1)-(z2*z2-z1*z1);
A2=2*(x1-x3);        B2=2*(z1-z3);        C2=-2*v*v*(t1-t3);
D2=v*v*(t3*t3-t1*t1)-(x3*x3-x1*x1)-(z3*z3-z1*z1);

E1=(B1*C2-B2*C1)/(A1*B2-A2*B1);
F1=(B2*D1-B1*D2)/(A1*B2-A2*B1);

E2=(A1*C2-A2*C1)/(A2*B1-A1*B2);
F2=(A2*D1-A1*D2)/(A2*B1-A1*B2);

G=(-2*v*v*t1+2*(x1-F1)*E1+2*(z1-F2)*E2)/(v*v-E1*E1-E2*E2);
H=(v*v*t1*t1-(F1-x1)*(F1-x1)-(F2-z1)*(F2-z1))/(v*v-E1*E1-
E2*E2);

% _____________________% Solution 1_____________________
t01=(-G-sqrt((G*G-4*H)))/2.0;
x01=(B2*(D1-C1*t01)-B1*(D2-C2*t01))/(A1*B2-A2*B1);
z01=(A2*(D1-C1*t01)-A1*(D2-C2*t01))/(A2*B1-A1*B2);
% _____________________% Solution 1_____________________

% _____________________% Solution 2_____________________
t02=(-G+sqrt((G*G-4*H)))/2.0;
x02=(B2*(D1-C1*t02)-B1*(D2-C2*t02))/(A1*B2-A2*B1);
z02=(A2*(D1-C1*t02)-A1*(D2-C2*t02))/(A2*B1-A1*B2);

% _____________________% Solution 2_____________________
```

Example Case 1

Input

```
v = 1.0;  % Velocity
x0 = 0.5;  z0 = 1.0; t0 = 1.0;
x1 = 1.5;  z1 = 0.5;
x2 = 3.0;  z2 = 1.0;
x3 = 2.0;  z3 = 4.5;
```

Output

```
Defined values t0=1   x0 =0.5   z0 =1
Solution 1     t01=1  x01=0.5   z01=1
Solution 2 t02=11.27 x02=7.975 z02=6.97
```

Example Case 2

Input …

```
x2 = 4.0;  z2 = 1.0;
```

Output

```
Defined values t0 =1   x0 =0.5   z0 =1
Solution 1 t01=-41.45 x01=-35.13 z01=-23.0
Solution 2     t02=1   x02=0.5   z02=1
```