Two point raytracing for reflection off a 3D plane

Stewart A. Levin

ABSTRACT

I present a simple, elegant approach to calculating two-point rays reflecting off a 3D dipping plane and investigate extensions to converted wave reflection and offset-vector map demigration.

INTRODUCTION

For SEP-147, I calculated the response of various classic seismic algorithms on a reflection off of a plane in 3D. After wrestling with spatial geometry in old textbooks, I derived the following result from scratch using elegant, coordinate-free vector notation.

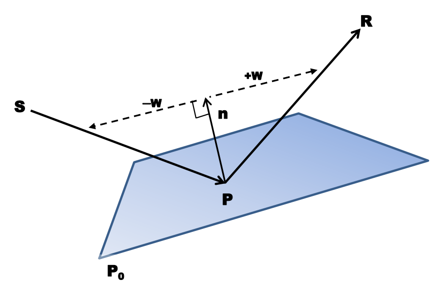


Figure 1: Diagram of planar reflector and the points and vectors I use for calculating the reflected raypath. [**NR**]

Given a source location S, a receiver location R, and a plane $\mathbf{n}\cdot(P-P_0)=0$, where $\mathbf{n}$ is a unit normal, to find the reflection point P, drop a perpendicular $\mathbf{w}$ from $\mathbf{n}$ to the line connecting P to R. Snell's Law says that running $\mathbf{w}$ in the other direction connects to the line between P and S. So for some scalars α and β we have

$$\begin{array}{rcl} (R-P) & = & \alpha(\mathbf{n}+\mathbf{w}) \\ (S-P) & = & \beta(\mathbf{n}-\mathbf{w}) \\ \mathbf{n}\cdot\mathbf{w} & = & 0 \\ \mathbf{n}\cdot(P-P_0) & = & 0 \end{array} .$$

Dotting $\mathbf{n}$ onto the first two equations gives

$$\begin{array}{rcccl} \mathbf{n}\cdot(R-P) & = & \mathbf{n}\cdot(R-P_0) & = & \alpha \\ \mathbf{n}\cdot(S-P) & = & \mathbf{n}\cdot(S-P_0) & = & \beta \end{array} ,$$

and subtracting the first two equations produces

$$(R-S) = (\alpha-\beta)\mathbf{n} + (\alpha+\beta)\mathbf{w},$$

which can be solved directly for $\mathbf{w}$ now that we have α and β. Given this $\mathbf{w}$, the first equation immediately yields

$$P = R - \alpha(\mathbf{n}+\mathbf{w}),$$

the desired reflection point. This can also be described in terms of the midpoint M of the source and receiver as

$$P = M - \tfrac{1}{2}(\alpha+\beta)\mathbf{n} - \tfrac{1}{2}(\alpha-\beta)\mathbf{w} \quad .$$

CONVERTED WAVE REFLECTION

The same approach applies to P-to-S or S-to-P reflection as well with one important difference—the angle of reflection differs from the angle of incidence. Now

$$\begin{array}{rcc} (R-P) & = & \alpha(\mathbf{n}+\mathbf{w}) \\ (S-P) & = & \beta(\mathbf{n}-\zeta\mathbf{w}) \\ \mathbf{n}\cdot\mathbf{w} & = & 0 \\ \mathbf{n}\cdot(P-P_0) & = & 0 \end{array}$$

for some scalar ζ. To determined ζ let v_s and v_r be the velocities of the source and receiver paths respectively and θ_s and θ_r be the corresponding angles of incidence and reflection. Then Snell's Law says

$$\frac{\sin\theta_s}{v_s} = \frac{\sin\theta_r}{v_r} \quad .$$

By our definition of $\mathbf{w}$, we also have the identities

$$\begin{array}{rcl} |\mathbf{w}| & = & \tan\theta_r \\ |\zeta\mathbf{w}| & = & \tan\theta_s \end{array}$$

which, using the identity,

$$\sin\theta = \frac{\tan\theta}{\sqrt{1+\tan^2\theta}} \ ,$$

gives the relation for ζ

$$\frac{1}{\zeta^2} = \left(\frac{v_r}{v_s}\right)^2 + \left(\left(\frac{v_r}{v_s}\right)^2 - 1\right)|\mathbf{w}|^2$$

which, combined with

$$(R-S) = (\alpha-\beta)\mathbf{n} + (\alpha+\zeta\beta)\mathbf{w} \ ,$$

produces a fourth order equation for ζ.

The fourth order equation can be solved directly using algebraic formulas. Lanczos (1956) provides a clean, efficient numerical approximation, reproduced in Appendix A, that is about 10 times faster than using a general purpose numerical root finder. (Appendix B shows how to make it free of floating point divisions.)

An interesting alternative to direct solution is to apply Newton iterations to the shooting method wherein source ray parameters are repeatedly adjusted to return very near to the target receiver. This approach applies to multiple layers and multiple reflections, not just a single interface. In Appendix C, I demonstrate *global* convergence of that method when applied to forward ray tracing through a stack of horizontal layers.

OFFSET-VECTOR MAP DEMIGRATION

Another application of the coordinate neutral approach for 3D reflection point calculation that arose at SEP recently is offset-vector map demigration. For this, the aim is to model where a point, P, on a planar subsurface reflector will appear in a constant-offset, constant-azimuth survey.

For this calculation, there is one fixed coordinate, the depth axis, with the sources and receivers on the surface, described by an arbitrary point Q_0 with (downward) normal $\mathbf{z}$. We are further given the source-to-receiver offset vector $2h\mathbf{x}$ and the reflector inward normal $\mathbf{n}$ from the point P on the reflector.

We know the ellipsoid of specular reflection has its major axis through the source and receiver, and that the inward normal bisects the reflection angle between the source and receiver. Therefore the normal line through the reflection point intersects the source-receiver axis somewhere between the source and receiver. Let Q be the point on the surface where the normal ray would reach. Then we may write

$$Q = P + \gamma\mathbf{n}$$

for some scalar γ. As before we calculate

$$\begin{aligned} \mathbf{z}\cdot(Q - Q_0) &= \mathbf{z}\cdot(P - Q_0) + \gamma\mathbf{z}\cdot\mathbf{n} \\ 0 &= \mathbf{z}\cdot(P - Q_0) + \gamma\mathbf{z}\cdot\mathbf{n} \\ \gamma &= \frac{\mathbf{z}\cdot(Q_0 - P)}{\mathbf{z}\cdot\mathbf{n}} \end{aligned}$$

and the horizontal distance of Q from the vertical plane through P as

$$\mathbf{x}\cdot[(P - \{\mathbf{z}\cdot(P - Q_0)\}\,\mathbf{z}) - (P + \gamma\mathbf{n})] = \gamma\,\mathbf{x}\cdot\mathbf{n}$$

thereby fixing the source-receiver axis and the relative location of Q. What still remains is to ascertain the source-receiver midpoint relative to P. This we can determine by means of tedious algebra, the way I did it, or by a succinct bit of trigonometry provide by Daniel Kane (pers. comm.) of the Stanford Department of Mathematics.

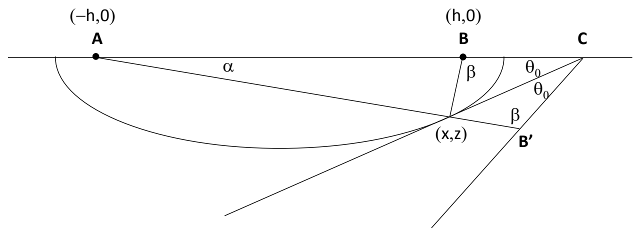


Figure 2: Diagram used to obtaining a quadratic relation for calculating x from z, h, and θ_0. [**NR**]

Due to symmetry, we may rotate the reflection point around the source-receiver axis until it is directly below that axis. This does not change the unknown distance to the source-receiver midpoint, but does reduce the computation to one on a planar ellipse. Let x_0 and z_0 denote the respective horizontal and vertical distances from the source-receiver midpoint to the reflection point. The dip angle θ_0 is implicitly determined by $\sin\theta_0 = -\mathbf{n}\cdot\mathbf{x}$ and $\cos\theta_0 = \sqrt{1-\sin^2\theta_0}$. Using this dip angle, z_0 may be written as $\gamma\cos\theta_0$. Referring to Fig. 2, Fermat's principle of extremal traveltime tells us that reflecting a focus of the ellipse around the tangent produces an image point on the straight line connecting the reflection point and the other focus. Hence we know that $AB'C$ forms a triangle. Denoting the three angles α, β, and θ_0 as illustrated in the figure, we have

$$\alpha + \beta + 2\theta_0 = \pi$$

whence

$$\tan 2\theta_0 = -\frac{\tan\alpha + \tan\beta}{1-\tan\alpha\tan\beta} \quad .$$

But

$$\begin{aligned}\tan\alpha &= \frac{z_0}{x_0+h}\\ \tan\beta &= \frac{z_0}{x_0-h}\end{aligned}$$

hence

$$\tan 2\theta_0 = -\frac{z_0(x_0-h)+z_0(x_0+h)}{x_0^2-h^2-z_0^2} = \frac{-2x_0z_0}{x_0^2-z_0^2-h^2}$$

and so we have the quadratic relation

$$x_0^2 + 2x_0z_0\cot 2\theta_0 - (z_0^2+h^2) = 0 \quad .$$

Solving the quadratic equation we get

$$\begin{aligned}x_0 &= -z_0\cot 2\theta_0 + \sqrt{z_0^2\cot^2 2\theta_0 + z_0^2 + h^2}\\ &= -z_0\cot 2\theta_0 + \sqrt{z_0^2\csc^2 2\theta_0 + h^2}\\ &= -\frac{z_0^2(\csc^2 2\theta_0 - \cot^2 2\theta_0)+h^2}{-z_0\cot 2\theta_0 - \sqrt{z_0^2\csc^2 2\theta_0 + h^2}}\\ &= \frac{\sin 2\theta_0(z_0^2+h^2)}{z_0\cos 2\theta_0 + \sqrt{z_0^2 + h^2\sin^2 2\theta_0}}\end{aligned}$$

in a form that does not exhibit a numerical singularity at $\theta_0 = 0$.

The relation $\alpha + \beta + 2\theta_0 = \pi$ is actually a special case of the more general proposition attributed to Bošković (Boscovich) (1754):

From any point H outside an ellipse with foci F and f, with F being no farther from H than f, draw two tangents, touching the ellipse at P and p respectively. Then the interior angle PHp is half the difference of the interior angles PFp and Pfp.

A translation of his original Latin demonstration appears in Appendix D.

So, in summary, only the dot products $\mathbf{z} \cdot \mathbf{n}$ and $\mathbf{x} \cdot \mathbf{n}$ are needed to find the demigration location of point P.

Reflection gradient

If we are interested in map *migration*, the information we have is not the reflector normal, but the normal to the arrival time surface. To calculate this slope, we can conflate distance and time by choosing an arbitrary temporal unit, say a *glorp* equated to $1/V$ seconds. This makes a traveltime of 1 glorp correspond to 1 meter of travel distance.

I make life simpler by observing that the traveltime gradient has the same azimuth as the reflector's dip azimuth. This must be so because translating the source-receiver pair along strike does not change the reflection arrival time. I note that this does *not* say that the reflection point moves along the dip azimuth when the surface arrival point moves along the dip azimuth.

The next twist is that instead of translating the source-receiver pair along the dip azimuth, I'll translate the reflector plane along its normal direction. This implies that derivatives with respect to the reflector normal direction need to be scaled by the sine of the reflector dip, i.e. $\sin\theta = \sqrt{1 - (\mathbf{n} \cdot \mathbf{z})^2}$, as the surface intercept of the reflector moves a distance inversely related to the sine of the dip. Fortunately, even the zero dip case, where the reflector does not intersect the surface, is handled properly because the sine is zero in that case.

If we translate the initial reflection point P by $-\epsilon\mathbf{n}$, where my convention for θ implies $\epsilon \geq 0$ corresponds to a positive time slope, we obtain a point on the displaced reflection plane, though generally not the new reflection point $\hat{P}$. The relation of $\hat{P}$ to P can be ascertained as before by dotting with $\mathbf{n}$:

$$\begin{array}{rclcl} \mathbf{n} \cdot (R - \hat{P}) & = & \mathbf{n} \cdot ((R - P) + \epsilon\mathbf{n})) & = & \alpha + \epsilon \\ \mathbf{n} \cdot (S - \hat{P}) & = & \mathbf{n} \cdot ((S - P) + \epsilon\mathbf{n})) & = & \beta + \epsilon \end{array} .$$

Continuing as before,

$$\begin{array}{rcrcr} R - S & = & (\alpha - \beta)\mathbf{n} & + & (\alpha + \beta)\mathbf{w} \\ R - S & = & ((\alpha + \epsilon) - (\beta + \epsilon))\mathbf{n} & + & ((\alpha + \epsilon) + (\beta + \epsilon))\hat{\mathbf{w}} \end{array} ,$$

yielding

$$\hat{\mathbf{w}} = \left(1 - \frac{2\epsilon}{\alpha+\beta+2\epsilon}\right)\mathbf{w}$$

which says that $\mathbf{w}$ does not rotate.

To compute changes in lengths (traveltimes), we have the relations

$$\begin{aligned} R - P &= \alpha(\mathbf{n}+\mathbf{w}) \\ R - \hat{P} &= (\alpha+\epsilon)(\mathbf{n}+\hat{\mathbf{w}}) = -(\alpha+\epsilon)(\mathbf{w}-\hat{\mathbf{w}}) + (\alpha+\epsilon)(\mathbf{n}+\mathbf{w}) \end{aligned}$$

and

$$\begin{aligned} S - P &= \beta(\mathbf{n}-\mathbf{w}) \\ S - \hat{P} &= (\beta+\epsilon)(\mathbf{n}-\hat{\mathbf{w}}) = (\beta+\epsilon)(\mathbf{w}-\hat{\mathbf{w}}) + (\beta+\epsilon)(\mathbf{n}-\mathbf{w}) \end{aligned} ,$$

whence

$$\begin{aligned} \hat{P} - P &= -(\alpha+\epsilon)(\hat{\mathbf{w}}-\mathbf{w}) - \epsilon(\mathbf{n}+\mathbf{w}) \\ &= (\beta+\epsilon)(\hat{\mathbf{w}}-\mathbf{w}) - \epsilon(\mathbf{n}-\mathbf{w}) \end{aligned} .$$

Taking first differences, we have

$$\frac{\hat{\mathbf{w}}-\mathbf{w}}{\epsilon} = -\frac{2}{\alpha+\beta+2\epsilon}\mathbf{w}$$

and

$$\begin{aligned} \frac{\hat{P}-P}{\epsilon} &= -(\alpha+\epsilon)\frac{\hat{\mathbf{w}}-\mathbf{w}}{\epsilon} - (\mathbf{n}+\mathbf{w}) \\ &= (\beta+\epsilon)\frac{\hat{\mathbf{w}}-\mathbf{w}}{\epsilon} - (\mathbf{n}-\mathbf{w}) \end{aligned}$$

whence

$$\begin{aligned} \frac{dP}{d\epsilon} &= \left(-1 + \frac{2\alpha}{\alpha+\beta}\right)\mathbf{w} - \mathbf{n} \\ &= \left(1 - \frac{2\beta}{\alpha+\beta}\right)\mathbf{w} - \mathbf{n} \end{aligned}$$

or, averaging the two,

$$= \frac{\alpha-\beta}{\alpha+\beta}\mathbf{w} - \mathbf{n} \quad .$$

With these in hand, we may differentiate the traveltime

$$T = T_R + T_S = |P-R| + |P-S|$$

to get

$$\begin{aligned} \frac{dT}{d\epsilon} &= \left(\frac{P-R}{|P-R|} + \frac{P-S}{|P-S|}\right)\cdot\frac{dP}{d\epsilon} \\ &= -\left(\frac{\alpha(\mathbf{n}+\mathbf{w})}{T_R} + \frac{\beta(\mathbf{n}-\mathbf{w})}{T_S}\right)\cdot\left(\frac{\alpha-\beta}{\alpha+\beta}\mathbf{w} - \mathbf{n}\right) \\ &= \frac{\alpha}{T_R}\left(\frac{(1-|\mathbf{w}|^2)\alpha + (1+|\mathbf{w}|^2)\beta}{\alpha+\beta}\right) + \\ &\quad \frac{\beta}{T_S}\left(\frac{(1+|\mathbf{w}|^2)\alpha + (1-|\mathbf{w}|^2)\beta}{\alpha+\beta}\right) \end{aligned}$$

which, as remarked earlier, is then multiplied by $\sin\theta$ to obtain the surface time slope.

This last expression has a simple geometric meaning. As

$$\frac{P-R}{|P-R|} \text{ and } \frac{P-S}{|P-S|}$$

are unit vectors pointing towards the reflection point from the receiver and source respectively, their sum is necessarily parallel to their angle bisector, the normal. In particular, they sum to $-2\cos\xi\,\mathbf{n}$ where ξ is the angle of incidence or reflection. Dotting this with $dP/d\epsilon$ and multiplying by $\sin\theta$ we have that the time slope is simply $2\cos\xi\sin\theta$. Changing units from glorps back to seconds, this agrees with the well-known zero-offset result $2\sin\theta/V$.

A Postscript

One of the references I allude to in the introduction was the classic posthumous publication of Slotnick (1959). In that tome, I found the proposition, a consequence of Apollonius' Theorem (see, e.g., Godfrey and Siddons (1908) pages 20–21), that for a fixed source location and with receivers placed diagonally opposite each other at equal distances from the source, the sum of the squares of the two source to receiver traveltimes is *independent* of source-receiver azimuth. This result, analogous on the face of it to the updip-downdip refraction shooting method, appears to have been used fairly routinely to estimate moveout velocities before the advent of the common midpoint gather but is no longer taught to students or industry professionals. I think it, or some modern recasting of it, may well provide uplift to both academia and industry seismic processing and analysis.

DISCUSSION AND CONCLUSIONS

As we have seen, while not a panacea, the power of vector notation really shines once we leave the Euclidean plane and begin to work in 3D. It can allow us to reduce a problem to its algebraic or geometric essentials and to subsequently cleanly code the solution using any Cartesian coordinate system. In addition, the interests of academic scholarship have brought me new insights into historical thinking about seismic acquisition, processing, and imaging that offer tantalizing hints how more recent approaches may benefit from those "old school" ideas. Stay tuned!

APPENDIX A

Lanczos solutions for cubics and quartics

From Lanczos (1956), pages 6–8, 19–22:

3. Cubic equations. Equations of third and fourth order are still solvable by algebraic formulas. However, the numerical computations required by the formulas are usually so involved and time-absorbing that we prefer less cumbersome methods which give the roots *in approximation* only but still close enough for later refinement.

The solution of a cubic equation (with real coefficients) is particularly convenient since one of the roots must be real. After finding this root, the other two roots follow immediately by solving a quadratic equation.

A general cubic equation can be written in the form

$$f(\xi) = \xi^3 + a\xi^2 + b\xi - c = 0 \quad .$$

The factor of ξ^3 can always be normalized to 1 since we can divide through by the highest coefficient. Moreover, the absolute term can always be made negative because, if it is originally positive, we put $\xi_1 = -\xi$ and operate with ξ_1.

Now it is convenient to introduce a new scale factor which will normalize the absolute term to -1. We put

$$x = \alpha\xi, a_1 = \alpha a, b_1 = \alpha^2 b, c_1 = \alpha^3 c$$

and write the new equation

$$f(x) = x^3 + a_1 x^2 + b_1 x - c_1 = 0$$

If we choose

$$\alpha = 1/\sqrt[3]{c}$$

we obtain

$$c_1 = 1.$$

Now, since $f(0)$ is negative and $f(\infty)$ is positive, we know that there must be at least one root between $x = 0$ and $x = \infty$. We put $x = 1$ and evaluate $f(1)$. If $f(1)$ is positive, the root must be between 0 and 1; if $f(1)$ is negative, the root must be between 1 and ∞. Moreover, since

$$x_1 \cdot x_2 \cdot x_3 = 1$$

we know in advance that we cannot have *three* roots between 0 and 1, or 1 and ∞. Hence if $f(1) > 0$, we know that there must be one and *only one* real root in the interval $[0, 1]$, while if $f(1) < 0$, we know that there must be one and only one real root in the interval $[1, \infty]$. The latter interval can be changed to the interval $[1, 0]$ by the transformation

$$\bar{x} = \frac{1}{x}$$

which simply means that the coefficients of the equation change their sequence:

$$-c_1\bar{x}^3 + b_1\bar{x}^2 + a_1\bar{x} + 1 = 0$$

Hence we have reduced our problem to the new problem: find the real root of a cubic equation in the range $[0, 1]$. We solve this problem in good approximation by taking advantage of the remarkable properties of the Chebyshev polynomials (cf. VII, 9) which enable us to reduce a higher power to lower powers with a small error. In particular, the third Chebyshev polynomial

$$T_3^*(x) = 32x^3 - 48x^2 + 18x - 1$$

normalized to the range $[0, 1]$ gives

$$x^3 = \frac{48x^2 - 18x + 1}{32} = 1.5x^2 - 0.5625x + 0.03125$$

with a maximum error of $\pm\frac{1}{32}$. The original cubic is thus reducible to a quadratic with an error not exceeding 3%.

We now solve this quadratic, retaining only the root between 0 and 1.

⋮

11. Equations of fourth order. Algebraic equations of fourth order with generally complex roots occur frequently in the stability analysis of airplanes and in problems involving servomechanisms. The historical method of solving algebraic equations of fourth order (also called biquadratic or quartic equations) involves the following steps. By a transformation of the form $x+\alpha$ the coefficient of the cubic term is annihilated. Then an auxiliary cubic equation is solved. The roots of the original equation are constructed with the help of the three roots of the auxiliary cubic. Numerically this method is lengthy and cumbersome. The following modification of the traditional procedure yields the four roots of an arbitrary quartic equation with real coefficients on the basis of a quick and numerically convenient scheme.

Every equation of the form

$$x^4 + c_1x^3 + c_2x^2 + c_3x + c_4 = 0$$

can be rewritten as follows:

$$(x^2 + \alpha x + \beta)^2 = (ax + b)^2 \quad .$$

If the original c_i are real, the new coefficients are also real. Hence the original equation becomes solvable in the form of the quadratic equation

$$x^2 + \alpha x + \beta \pm (ax + b) = 0$$

which has four (generally complex) roots, obtainable by the standard formula. The new coefficients can be determined as follows. We evaluate in succession the following numerical constants:

$$\alpha = \frac{c_1}{2}, A = c_2 - \alpha^2, B = c_3 - \alpha A$$

and form the cubic equation

$$\xi^3 + (2A - \alpha^2)\xi^2 + (A^2 + 2B\alpha - 4c_4)\xi - B^2 = 0$$

Since the left side is negative at $\xi = 0$, a positive real root must exist. We determine this root according to the method of § 3. In order to avoid later corrections, it is advisable to add at this point Newton's correction (cf. § 5), obtaining ξ with great accuracy. The coefficients of the reduced equation are then determined as follows:

$$\begin{aligned} \alpha &= \tfrac{1}{2}c_1, \quad \beta = \tfrac{1}{2}(A+\xi) \\ a &= \sqrt{\xi}, \quad b = \frac{a}{2}\left(\alpha - \frac{B}{\xi}\right) \end{aligned} \quad .$$

APPENDIX B

Division-free reciprocal cube roots

Sometime back in the '90s, square roots started to be implemented as $z \times z^{-1/2}$ where the reciprocal square root was implemented using one or two iterations of Newton's method. As the Newton formula for the reciprocal square root could be written with only multiplications and additions, this was several times faster than computer division. Indeed, division was often replaced by squaring the reciprocal square root.

For the Lanczos root-finding methods in the previous appendix, a reciprocal *cube* root is needed. Fortunately, this, too, can be obtained using Newton's method in a division-free manner as follows:

Let

$$f(x) = \frac{1}{x^3} - z$$

be the function whose root we want to find. Taking its derivative,

$$f'(x) = \frac{-3}{x^4} \quad ,$$

produces the Newton step

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n(\tfrac{4}{3} - \tfrac{1}{3}z\,x_n^3) \quad .$$

The remaining issue is choosing an appropriate first guess, x_0, of the root in order to start the iteration. For this I again look to the fast reciprocal square root for guidance. McEniry (2007) reproduces a classic code (without the profane comment) containing a "magic number" from which half the integer representation of the floating point input is subtracted to produce an integer representation of the initial guess. This starting point was good enough that a single Newton iteration resulted in a worst case relative error of less than 0.175%. Mimicing McEniry's development yields the following code for a reciprocal cube root:

```
float InvCubeRoot ( float x ) {
   const float onethird = 0.333333333333;
   const float fourthirds = 1.333333333333;
   float thirdx = x * onethird;
   union {
     int ix;
     float fx;
   } z;

   z.fx = x;
   z.ix = 0x54a21d2a - z.ix/3; /* magic */
   x = z.fx;
   x = x * ( fourthirds - thirdx * x*x*x ); /* max relerr < 2.34E-3 */
   x = x * ( fourthirds - thirdx * x*x*x ); /* max relerr < 1.09E-5 */
   return x;
}
```

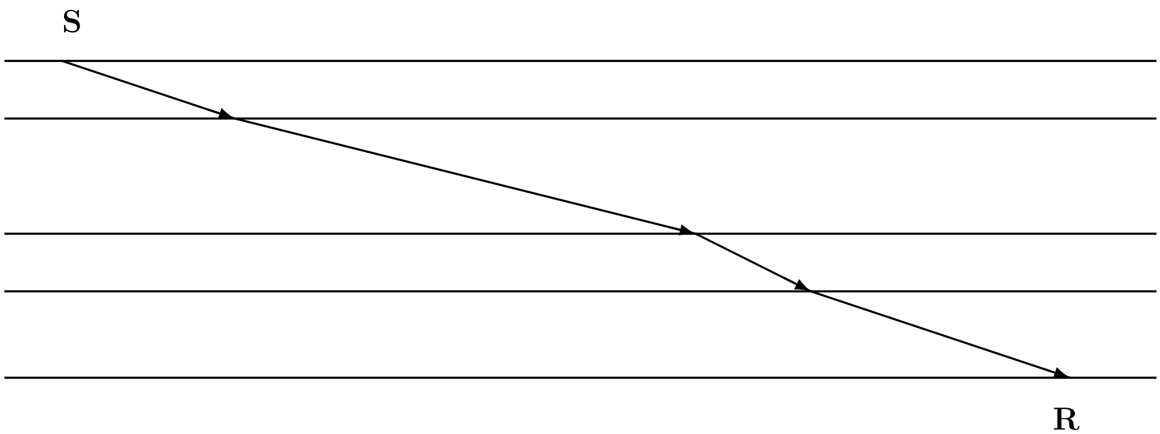
There is still one hitch—the "magic" line is not division free. Fortunately, the hacker and compiler community has worked out division-free integer division. For division by 3, this is accomplished by multiplying the numerator by the binary expansion 0.0101010101 01... of $\frac{1}{3}$ in fixed point arithmetic just like we were all taught in elementary school. For 32 bit numerators, we multiply by the hexadecimal constant 55555556 and shift the (64 bit) result down by 32 binary places. Therefore the "magic" line of code becomes

```
z.ix = 0x54a21d2a -
       (int) ((z.ix * (int64_t) 0x55555556)>>32); /* magic */
```

where the tail end 6 instead of 5 in the multiplier handles the cases where the integer is not an exact multiple of 3.

Performing timing tests in C with random numbers, this algorithm ran 20 times faster than calling *powf*(*x*,-1.0*f*/3.0*f*) from the C math runtime library and about 10 times faster than my best previous effort to calculate a fast reciprocal cube root.

APPENDIX C



Globally convergent Newton's method for ray shooting

Quite some time ago, Bob Keyes at Mobil mentioned that Newton's method applied to shooting rays to solve the two-point problem in horizontally layered media is globally convergent, assuming, of course, that there is a solution. Specifically, there must be a solution if an initial guess at the ray parameter overshoots the target.

Formally, let the ray parameter p be in the open interval $(0, 1/v_{max})$. Starting from the origin, Snell's law $pv = \sin\theta$ says that

$$x = \int_0^z \tan\theta \, dz = \int_0^z \frac{pv}{(1-p^2v^2)^{1/2}} \, dz$$

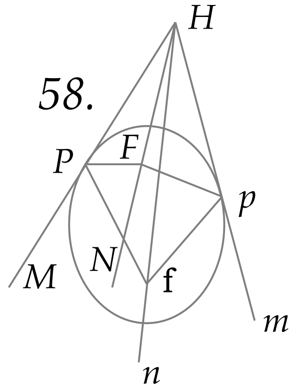
gives the horizontal displacement of the ray from the origin when it reaches depth z. Taking two derivatives of this formula with respect to p, we have

$$\frac{dx}{dp} = \int_0^z \frac{v}{(1-p^2v^2)^{3/2}} \, dz \ , \quad \frac{d^2x}{dp^2} = \int_0^z \frac{3pv^3}{(1-p^2v^2)^{5/2}} \, dz \ .$$

At a glance one sees that the second derivative is a quantity guaranteed to be positive in $(0, 1/v_{max})$. By Thorlund-Petersen (2004), Newton's method applied to finding the p for a ray that reaches a given x at given depth z is therefore globally convergent. (Technically, we do need to ensure that the Newton update doesn't overshoot the range $(0, 1/v_{max})$.)

APPENDIX D

Translation of the geometric proof in Bošković (Boscovich) (1754).



186. At Ellipsi in fig. 58 ductis HFN, Hfn, bini FPH, FpH æquales erunt binis fPM, fpm, sive quatuor internis, & oppofitis PfH, PHf, pfH, pHf, nimirum toti PHp, & toti Pfp. Angulus autem PFp æqualis binis PFN, pFN, sive quatuor internis FPH, FHP, FpH, FHp, vel binis illis FPH, FpH cum angulo PHp, adeoque angulo PHp bis, & toti Pfp semel. Quare angulo Pfp dempto a PFp, remanet angulus PHp bis.

186. In the ellipse in fig. 58, draw HFN and Hfn. Then FPH and FpH are equal to fPM and fpm respectively and so the four internal opposite angles PfH, PHf, pFH, and pfH evidently sum to PHp with Pfp.† Now angle PFp is the sum of PFN and pFN and so the sum of internal angles FPH, FHP, FpH, and FHp, hence [the sum of] FPH, FpH and angle PHp. [From above,] this is precisely equal to PHp twice combined with Pfp once. Therefore subtracting Pfp from PFp leaves twice the angle PHp.

†The external angle is the sum of the two opposite internal angles in a triangle.

REFERENCES

Bošković (Boscovich), R. J., 1754, Elementorum universæ matheseos 3, Continens sectionum conicarum elementa nova quadam methodo concinnata et dissertationem de transformatione locorum geometricorum ubi de continuitatis legem ac de quibusdam infiniti mysteriis: Amideus.

Godfrey, C. and A. W. Siddons, 1908, Modern geometry: Cambridge University Press.

Lanczos, C., 1956, Applied analysis: Prentice Hall.

McEniry, C., 2007, The mathematics behind the fast inverse square root function code.

Slotnick, M. M., 1959, Lessons in seismic computing, a memorial to the author: Society of Exploration Geophysicists.

Thorlund-Petersen, L., 2004, Global convergence of Newton's method: Mathematical Methods of Operations Research, **59**, 91–110.