

Refraction propagation algorithm

Marc Wathelet, ISTerre, France

January 11, 2013

Abstract

Library QGpCoreWave implements a refraction forward computation based on a stratified ground structure with tilted layers. Shots must be located on the ground surface. Class TiltLayeredModel encapsulates the model parameters and the computation of arrival times. This document develops the complete formulation of the problem and discusses the details of the algorithm. The formulation considers one layer a time independently of property of the above layers. There is no limit on the number of layers.

1 Incidence angles of critical ray paths

By convention, layer $i - 1$ is delimited on top by interface $i - 1$ and at bottom by interface i (see figure 1). Interface 0 is the ground surface. Layer 0 is the most superficial layer.

We first define a local thickness (h_{i-1}) and a local shift (s_i). The local thickness is measured along a perpendicular to the above interface $i - 1$. The angle between the current interface i and the thickness direction is $\pi/2 - (\alpha_i - \alpha_{i-1})$. Angles are counted anti-clockwise from the horizontal axis. By the sine law in a non-right triangle,

$$\frac{\sin(\pi/2 - (\alpha_i - \alpha_{i-1}))}{d_i - d_{i-1}} = \frac{\sin(\alpha_{i-1})}{s_i} = \frac{\sin(\pi/2 + \alpha_i)}{h_i}$$

$$s_i = -\frac{(d_i - d_{i-1})\sin(\alpha_{i-1})}{\cos(\alpha_i - \alpha_{i-1})} \quad (1)$$

$$h_{i-1} = \frac{(d_i - d_{i-1})\cos(\alpha_i)}{\cos(\alpha_i - \alpha_{i-1})} \quad (2)$$

d_i is the vertical depth of layer i at the reference profile (vertical dotted line in Figure 1). The reference profile is the left one. h_i is always positive because α_i and the difference $(\alpha_i - \alpha_{i-1})$ are less than $\frac{\pi}{2}$. s_i is a signed quantity, positive when α_{i-1} is negative like in Figure 1. This is the reason for the presence of the minus sign in Equation 1.

If $slow_i$ is less than $slow_{i-1}$ a refraction at critical angle is possible. With Snell's law, the critical angle is defined by

$$\gamma_{i,i-1} = \arcsin\left(\frac{slow_i}{slow_{i-1}}\right) \quad (3)$$

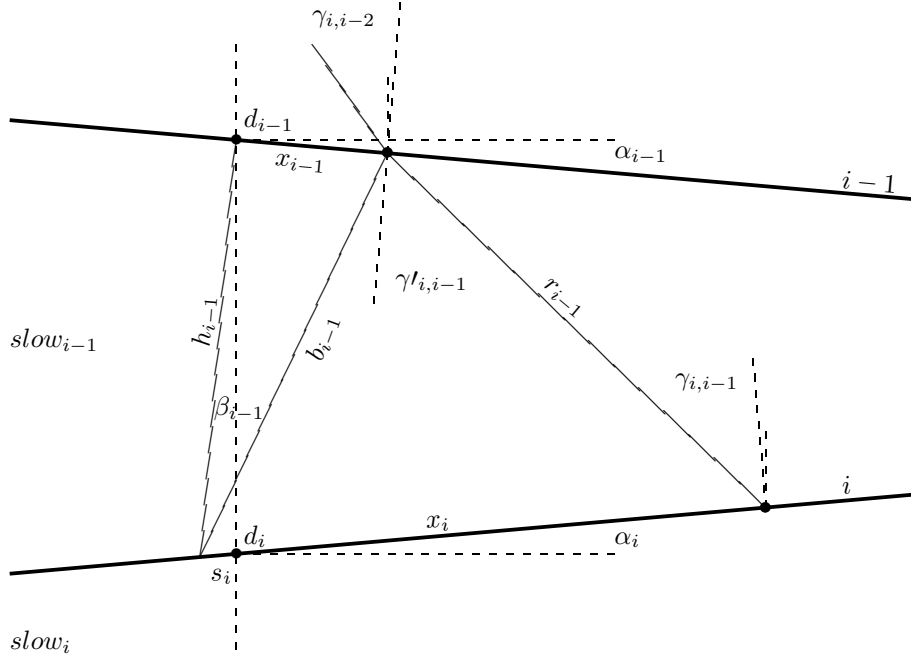


Figure 1: Layer geometry and ray propagation

$\gamma_{i,j}$ is the incidence angle in layer j for a total internal reflection at interface i , $j < i$. $\gamma_{i,i-2\dots i,0}$ are the incidence angles in upper layers for a total internal reflection at the interface i .

$$\gamma'_{i,i-1} = \gamma_{i,i-1} - (\alpha_i - \alpha_{i-1}) \quad (4)$$

By Snell's law at interface $i - 1$,

$$\frac{\sin(\gamma_{i,i-2})}{\sin(\gamma'_{i,i-1})} = \frac{slow_{i-1}}{slow_{i-2}}$$

$$a_{i-2} = \sin(\gamma'_{i,i-1}) \frac{slow_{i-1}}{slow_{i-2}} \quad (5)$$

$$\gamma_{i,i-2} = asin(a_{i-2}) \quad (6)$$

If a_{i-2} is greater or equal to 1, the critical angle in layer i is never lit. $\gamma_{i,j} \forall j \in [0, i-1]$ are discarded.

The same formulation can be achieved for a ray propagating downwards from right to left (the opposite of the ray displayed in figure 1). The sign of $\gamma_{i,i-1}$ is simply changed. Hence, two sets of γ angles are generated: γ_l and γ_r .

These computations are implemented by function,

```
void TiltLayeredModel::begin()
void TiltLayeredModel::end()
```

2 Travel time across a layer

The time to cross the layer is

$$t = \text{slow}_{i-1} r_{i-1} \quad (7)$$

By the sine law in a non-right triangle,

$$\begin{aligned} \frac{\sin(\pi/2 - \gamma_{j,i-1})}{b_i} &= \frac{\sin(\pi/2 - (\alpha_{i-1} - \alpha_i + \beta_{i-1}))}{r_{i-1}} \\ &= \frac{\sin(\alpha_{i-1} - \alpha_i + \beta_{i-1} + \gamma_{j,i-1})}{x_i + s_i} \end{aligned}$$

$$r_{i-1} = b_i \frac{\cos(\alpha_{i-1} - \alpha_i + \beta_{i-1})}{\cos(\gamma_{j,i-1})} \quad (8)$$

$$\beta_{i-1} = \arccos\left(\frac{h_{i-1}}{d_{i-1}}\right) \quad (9)$$

$$d_{i-1} = \sqrt{x_{i-1}^2 + h_{i-1}^2} \quad (10)$$

$$x_i = b_i \frac{\sin(\alpha_{i-1} - \alpha_i + \beta_{i-1} + \gamma_{j,i-1})}{\cos(\gamma_{j,i-1})} - s_i \quad (11)$$

These computations are implemented by function,

```
double TiltLayeredModel::propagate(Direction dir,
                                     int iLayer,
                                     int iDeepestLayer,
                                     double& x) const
```