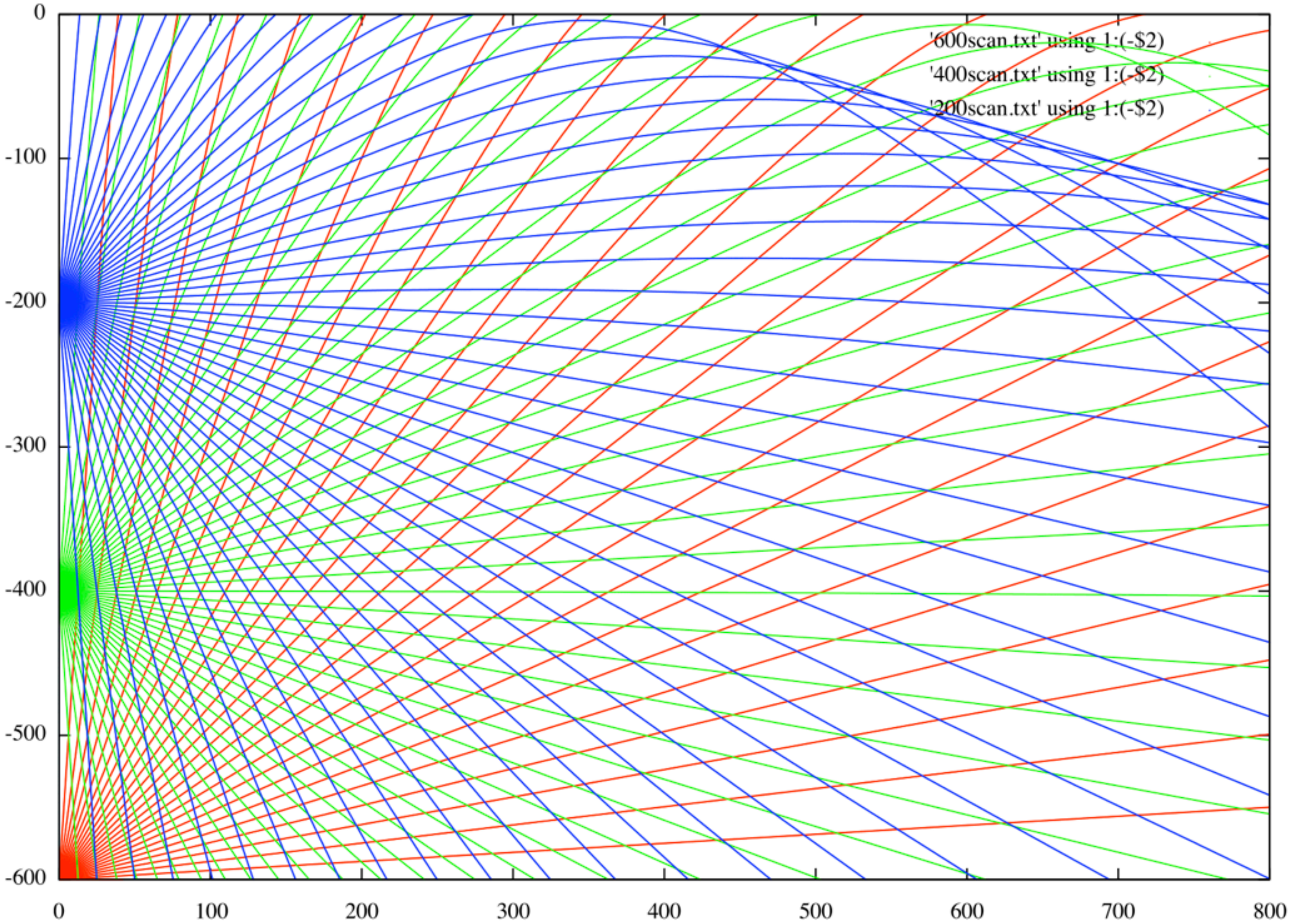
Some Ideas in Data Analysis

Chris Weaver

ARA Meeting March 17, 2010

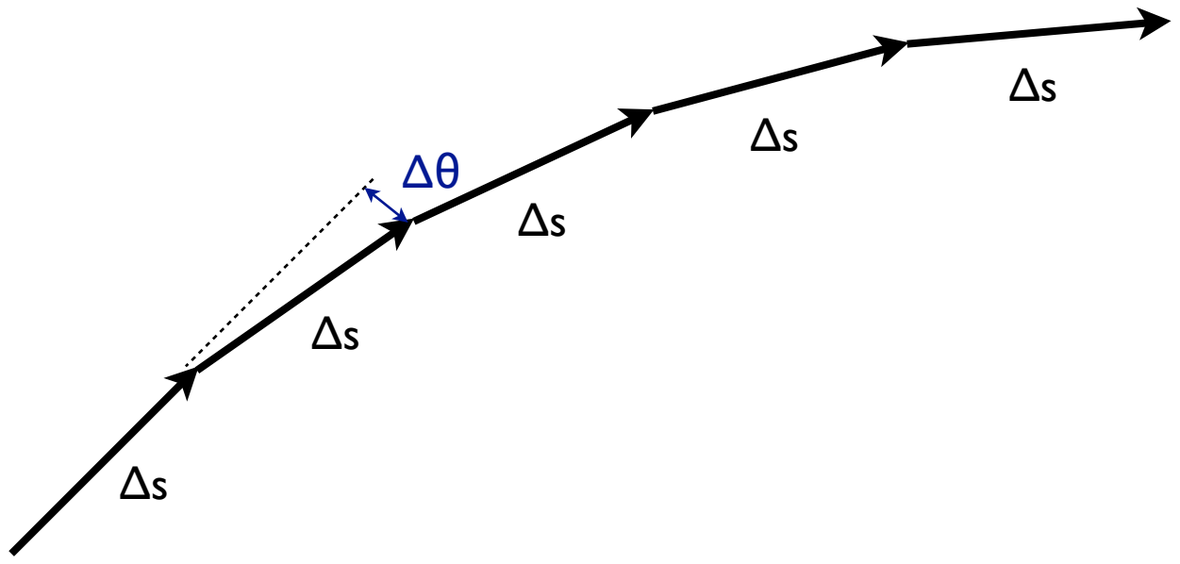
Ray Tracing


'600scan.txt' using 1:(-$2)
'400scan.txt' using 1:(-$2)
'200scan.txt' using 1:(-$2)
0
-100
-200
-300
-400
-500
-600
0
100
200
300
400
500
600
700
800

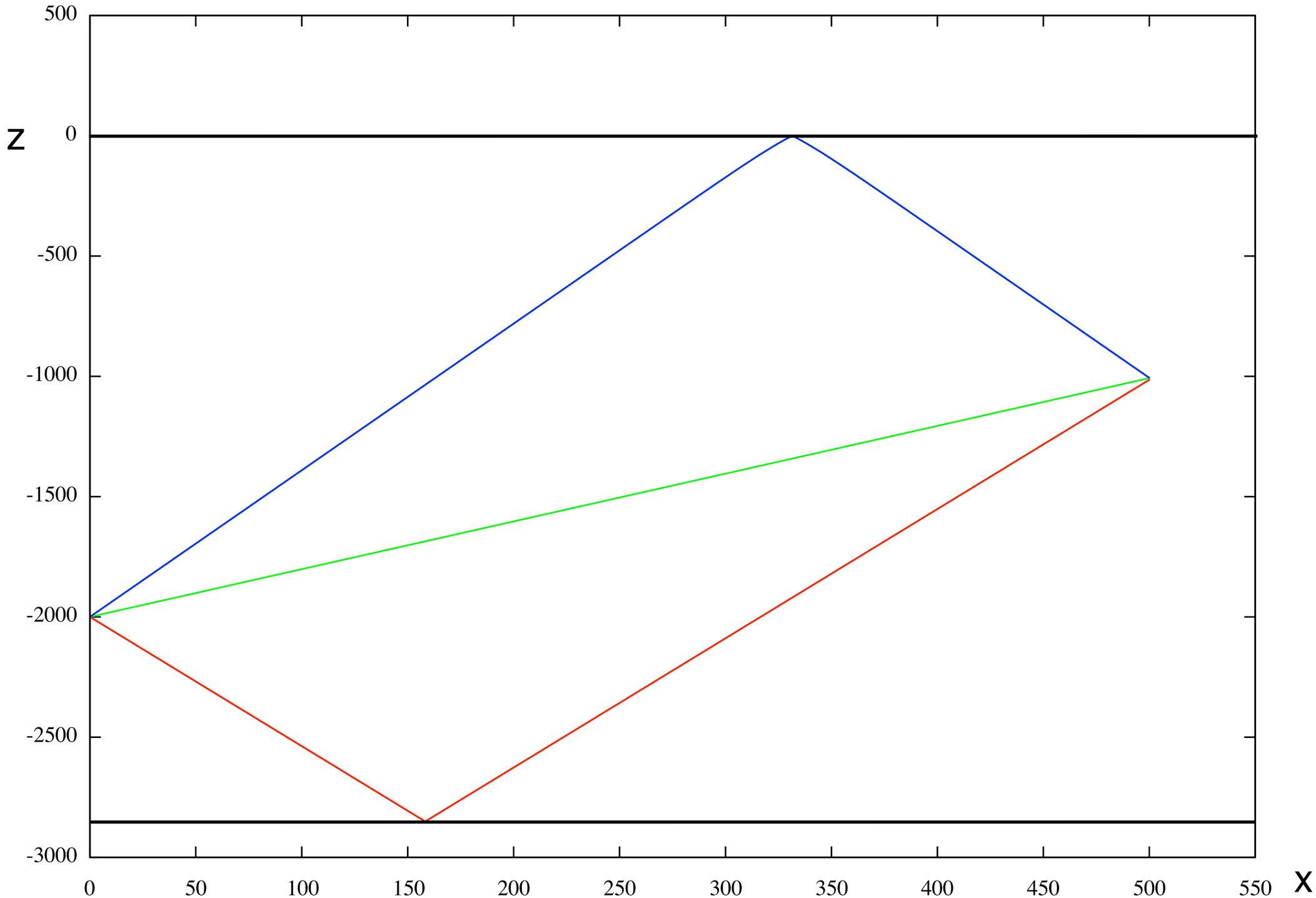
Basic Method

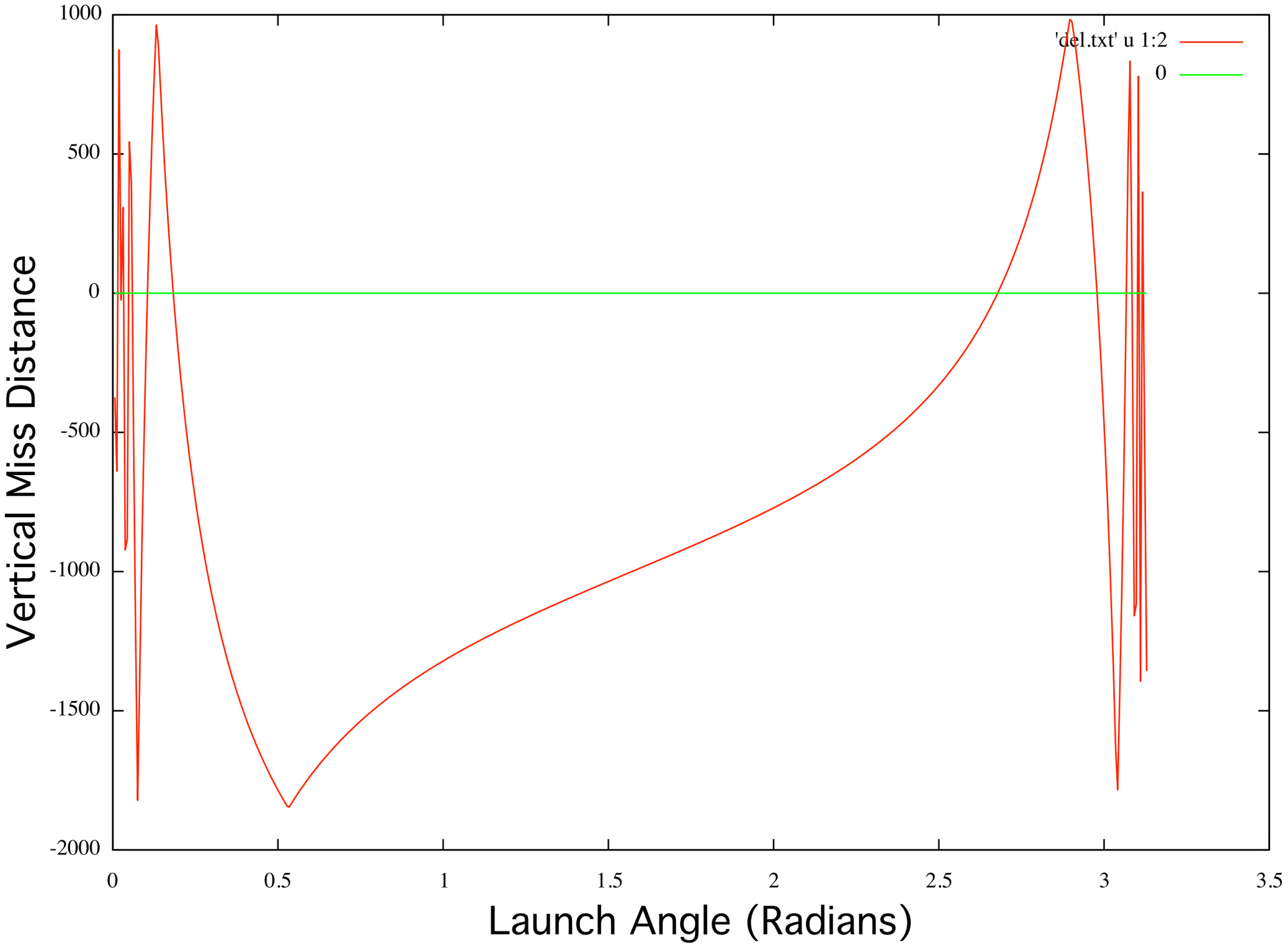
- Really simple stepwise process
- Takes a while for longer paths

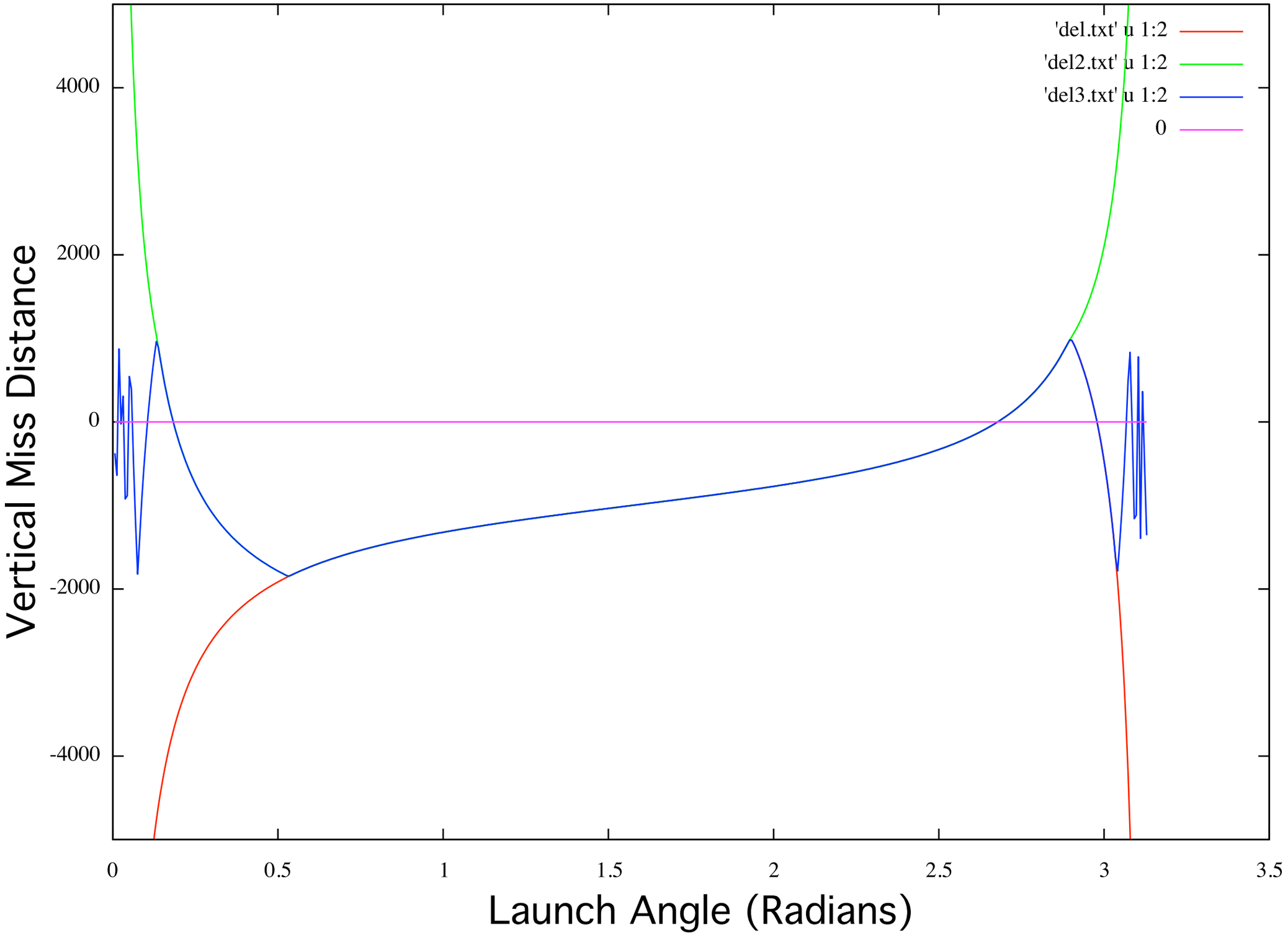
$$\Delta\theta = \frac{1}{n(z)} \sin(\theta) \frac{dn}{dz} \Delta s$$



Finding Reflected Solutions




'del.txt' u 1:2
0
Vertical Miss Distance
Launch Angle (Radians)
1000
500
0
-500
-1000
-1500
-2000
0
0.5
1
1.5
2
2.5
3
3.5


'del.txt' u 1:2
'del2.txt' u 1:2
'del3.txt' u 1:2
0
Vertical Miss Distance
4000
2000
0
-2000
-4000
0
0.5
1
1.5
2
2.5
3
3.5
Launch Angle (Radians)

Using Tables for Speed

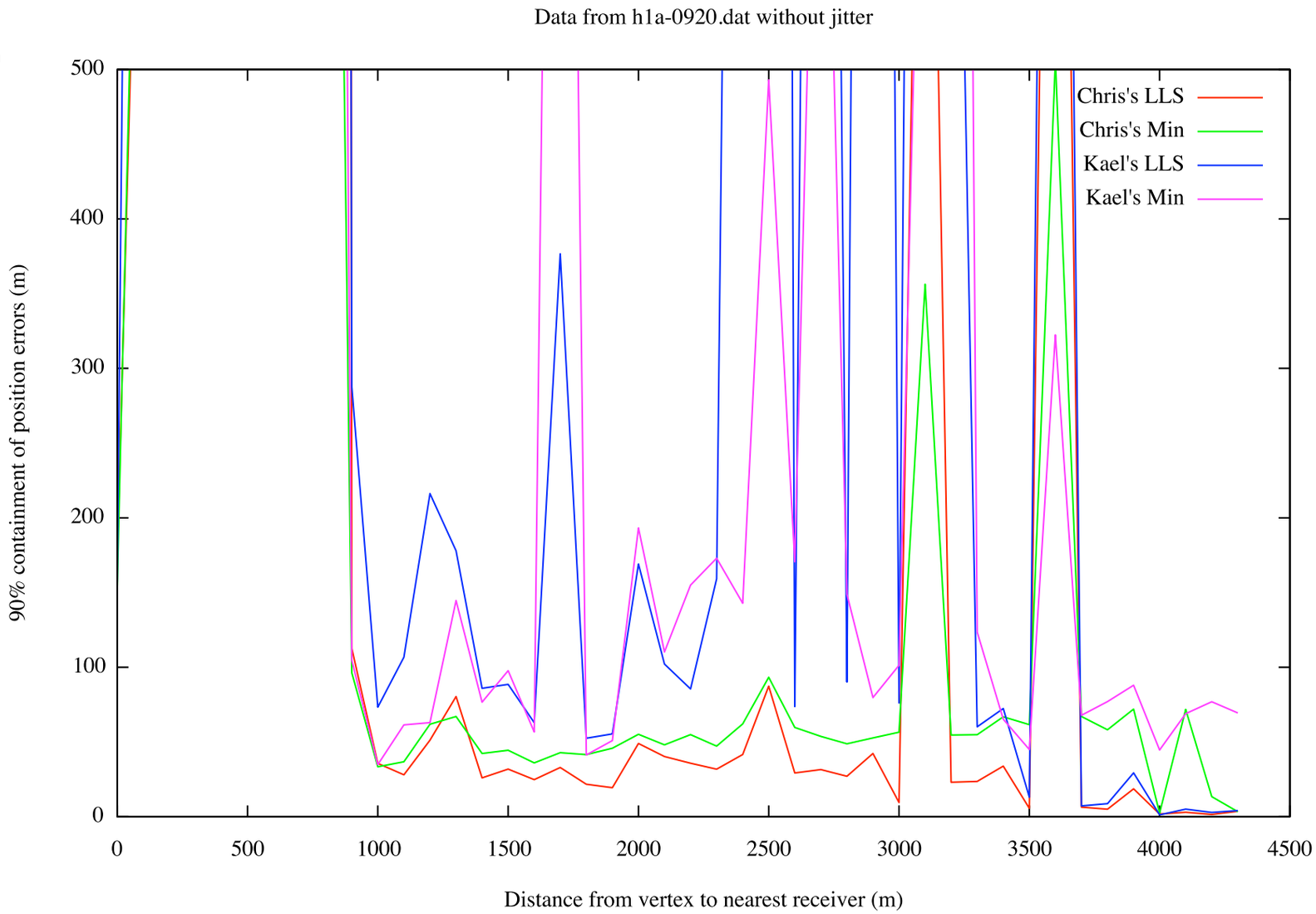
- Table must have three indices:
  - Receiver Depth
  - Transmitter Depth
  - Horizontal Distance between Transmitter and Receiver
- Don’t want to to have to tabulate very finely, so interpolate between table entries
- Trouble when adjacent entries have different numbers or types of solutions
- Even so, and without loading tables into RAM, still works out around 25 times faster

Vertex Reconstruction Ideas

- Kael's linear least squares
- Full-blown minimization (for full effect, use with ray tracing)
- Grid testing
- Zenith-Ring-Cone. . . Thing

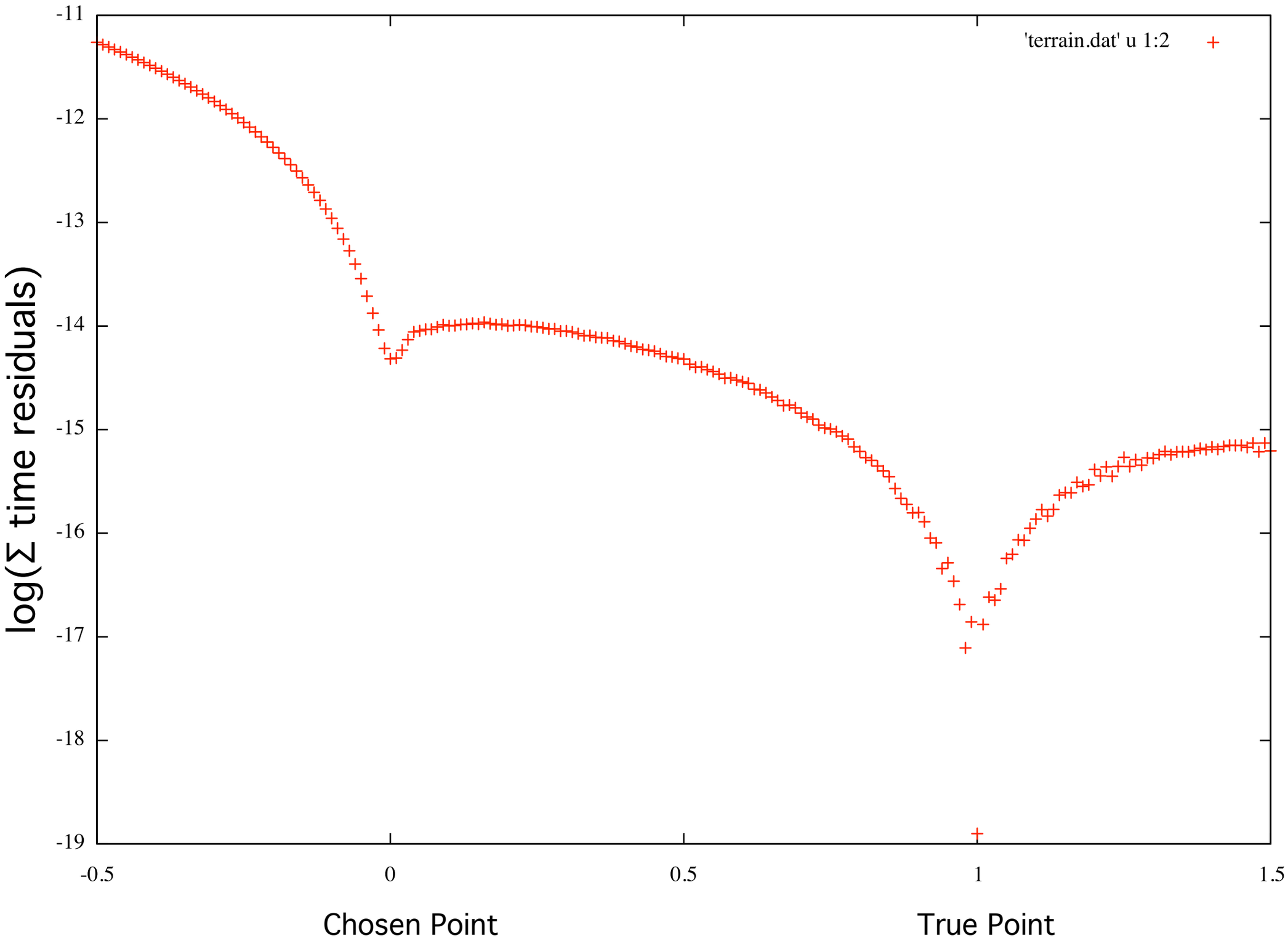
Linear Least-Squares

- Based on a pair of simplifying assumptions:
  - Rays are straight
  - Velocity is constant
- I have never been able to make this work well



Full Minimization

- Ask a general minimizer to minimize the sum of squared time residuals for all hits
- Can be very accurate, can be be very slow, can get stuck in strange places
- Doesn't always play well with tabulated ray-traces; may decide to investigate points that fall far outside of tables


'terrain.dat' u 1:2
log(Σ time residuals)
-11
-12
-13
-14
-15
-16
-17
-18
-19
-0.5
0
0.5
1
1.5
Chosen Point
True Point

Grid Testing

- A brute-force way to get into the ballpark of the correct position
- For each point in some grid, for each recorded hit, find the time at which the vertex would have had to occur
- The best vertex guesses are the ones with the smallest spread in required times
- Seems capable of giving pretty good time estimates, as long as a moderately good position is found

Zenith-Ring-Cone Intersection

- For any pair of hits on the same string, can compute an incident zenith angle
- For any pair of zenith angles, can follow ray-traces back and find intersection point
- This defines a ring (since azimuth unknown)
- Rays are nearly parallel, so exact position of each ring is rather uncertain, but the collection of rings for a given string seem to come quite close to forming a nice cone

Zenith-Ring-Cone Intersection continued

- Given the set of cones for all strings with hits, can look for point nearest to all cones
- Depends on having plenty of hits, on separate strings
- Appears to degrade rapidly as time jitter increases
- Seems to give pretty good depth estimates, complements the grid search