The Compute Module

Does arithmetic on the point-by-point Data component of a field, and outputs the modified field

The 3 (in this case) inputs

The output expression, in this case, a 3-vector with a newly-created Z value
The Compute Module

Does arithmetic on the point-by-point Data component of a field, and outputs the modified field. But, what if you want to do arithmetic on a different component?

The *Mark* module renames the Data component to something temporary, and renames a component you select to “Data”. *Compute* then acts on this component. The *Unmark* module changes the component names back to what they were originally.
Sequencer outputs a series of integers. You set the minimum and maximum using Edit→Configuration.
In this case, *Compute* turns an integer into a scalar to be used to animate an isovalue.
A good Sequencer Strategy:
Run the sequence from 1-100 (or 0-100).

Then, base the Compute quantity on these "Percent Units".
In this case, *Compute* turns an integer into a scalar to be used to animate the isovalue.
The Sequencer Module: Setting a Scalar Isovalue

Cutting plane position = [0.0, 0.0, 4.3], Isovalue = 56.0
In this case, Compute turns an integer into a 3-element vector to be used to animate the position of the cutting plane.
The Sequencer Module: Setting a Vector to act as a Plane Location
In this case, **Compute** turns an integer into a rotation angle in *degrees.*
Why Does the Rotation Occur around the Edge of the Cube, not about its Center?

Rotation and Scaling always occur about the origin. To change this to the center of the volume, translate the volume to the origin, perform the rotation or scale, and then translate it back.

Translate by [-15,-15,-15]

Translate by [15,15,15]
Writing Out a MIFF Animation File

convert -quality 100 sample2.miff sample2.gif