McGill School of Architecture
Advanced Construction
Fall 2012

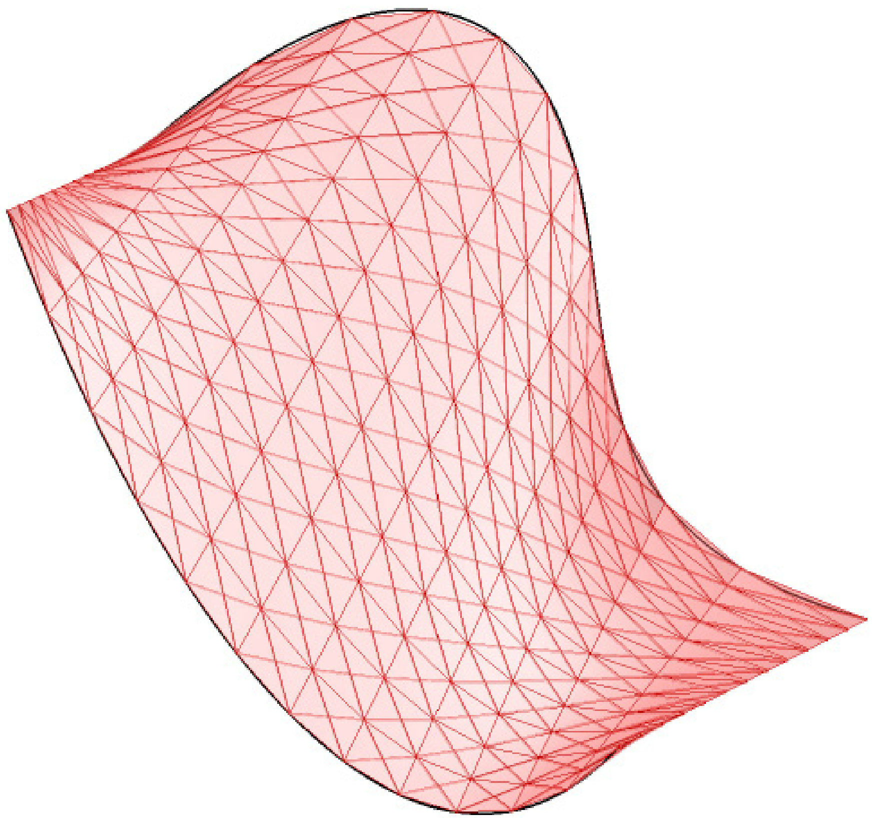
Introduction to Grasshopper for Rhinoceros

Maria Mingallon
Msc (Emtech AA) MEng CEng MICE
maria.mingallon@arup.com

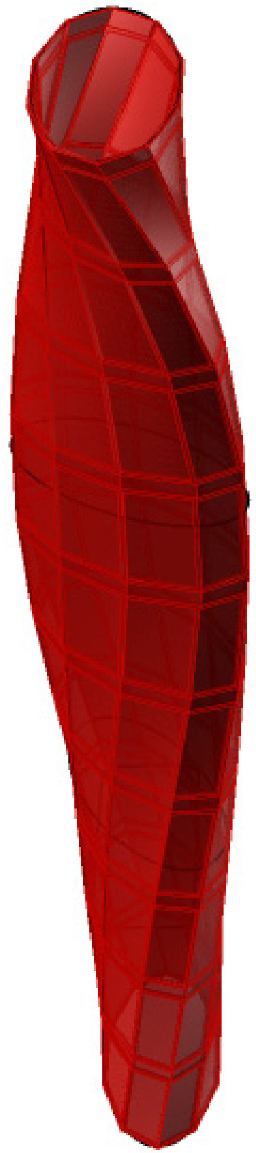
Schedule

FRIDAY (7th Sep)	Introduction Data Storage I: Objects, Scalars, Data Trees ☐ Examples: Spiral Function, Curve paths
FRIDAY (14th Sep)	Data Storage II: Vectors ☐ Unit Vectors ☐ Examples: Shift Data, Attractor scaled boxes Curves (NURBS) ☐ Definition & type of curves ☐ Curve Analysis (Curve Domain) Examples: Curve Domain & reparametrize, Curves in between two curves
FRIDAY (21st Sep)	Surfaces (NURBS) ☐ Definition & Surface Analysis ☐ Surface Domain & Evaluate Surface ☐ Examples: Surface Connect & Panelling of a surface
FRIDAY (28th Sep)	Surfaces (NURBS): ☐ Manipulation of Curves & Surfaces II ☐ Example: Surface Diagrid Uniform and Non-uniform Distribution

What can Grasshopper do for you?



Surface DiaGrid



Paneling a Surface

What can Grasshopper do for you?

ContemPLAY pvailion

1. Data Storage

1.1 Objects

a) **Params:**

- **Import and store data**
- **Used to introduced objects previously created in Rhino Interface**

e.g. point, surface, plane, curve, etc

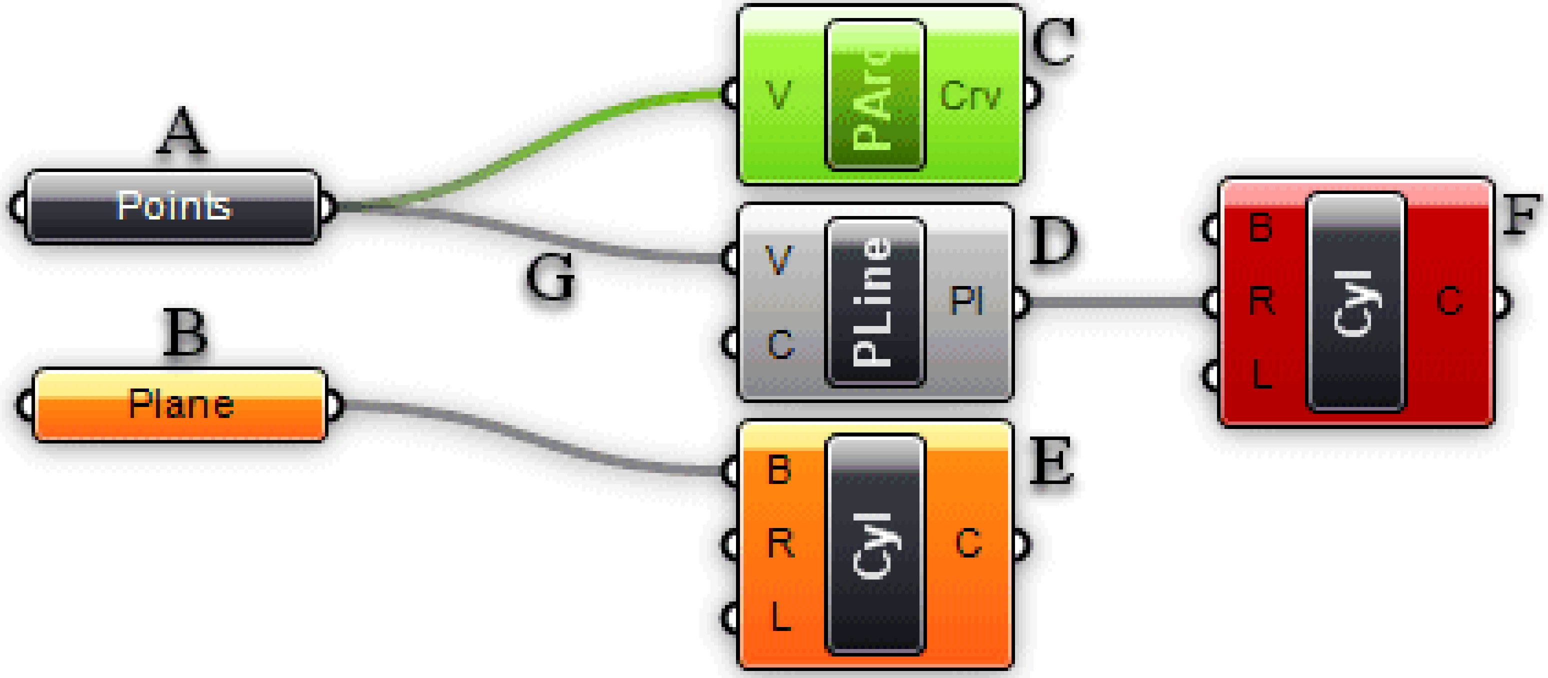
b) **Components:**

- **Generate and store data**
- **Created within Grasshopper**

e.g. logic, scalar, vector, curve, surface, mesh, etc

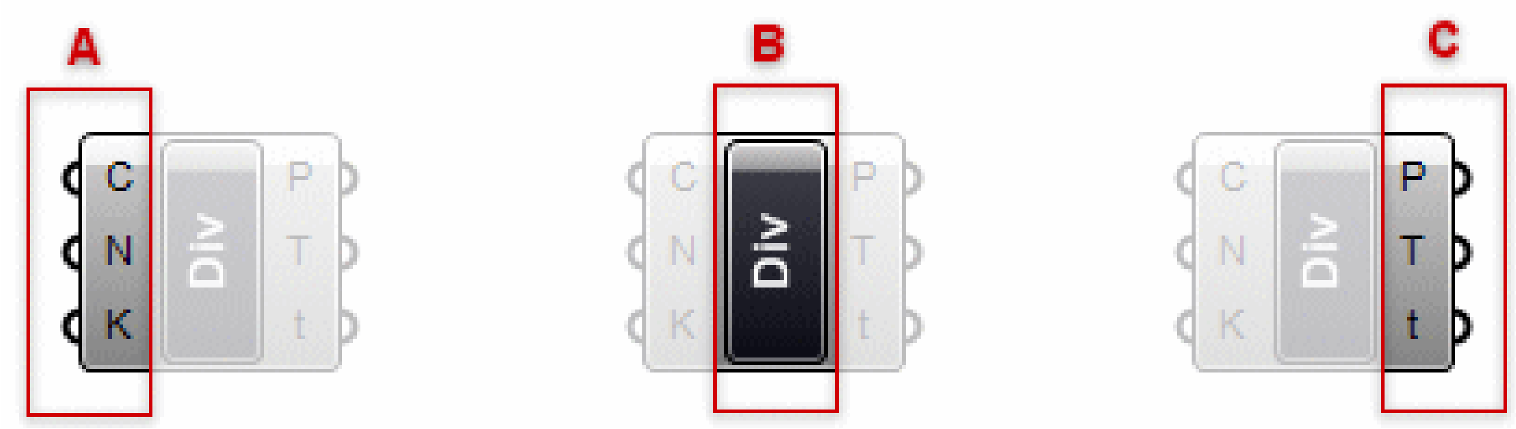
1. Data Storage

1.1 Objects



1. Data Storage

1.1 Objects



A: Input parameters
B: Name of Component
C: Output parameters

1. Data Storage

1.2 Scalars

A) Constants: Used to represent a constant value (e.g. Pi, Golden Ratio, etc).

B) Operators: used for mathematical operations such as Add, Subtract, Multiply, etc. Example: scalar_operators.ghx

C) Intervals: used to divide numeric extremes (or domains) into interval parts. Apart from Intervals, Ranges and Series are included within this group. Example: scalar_intervals.ghx

D) Utility: evaluate two or more numerical values. Examples: conditional_statements.ghx and If_Elsetest.ghx

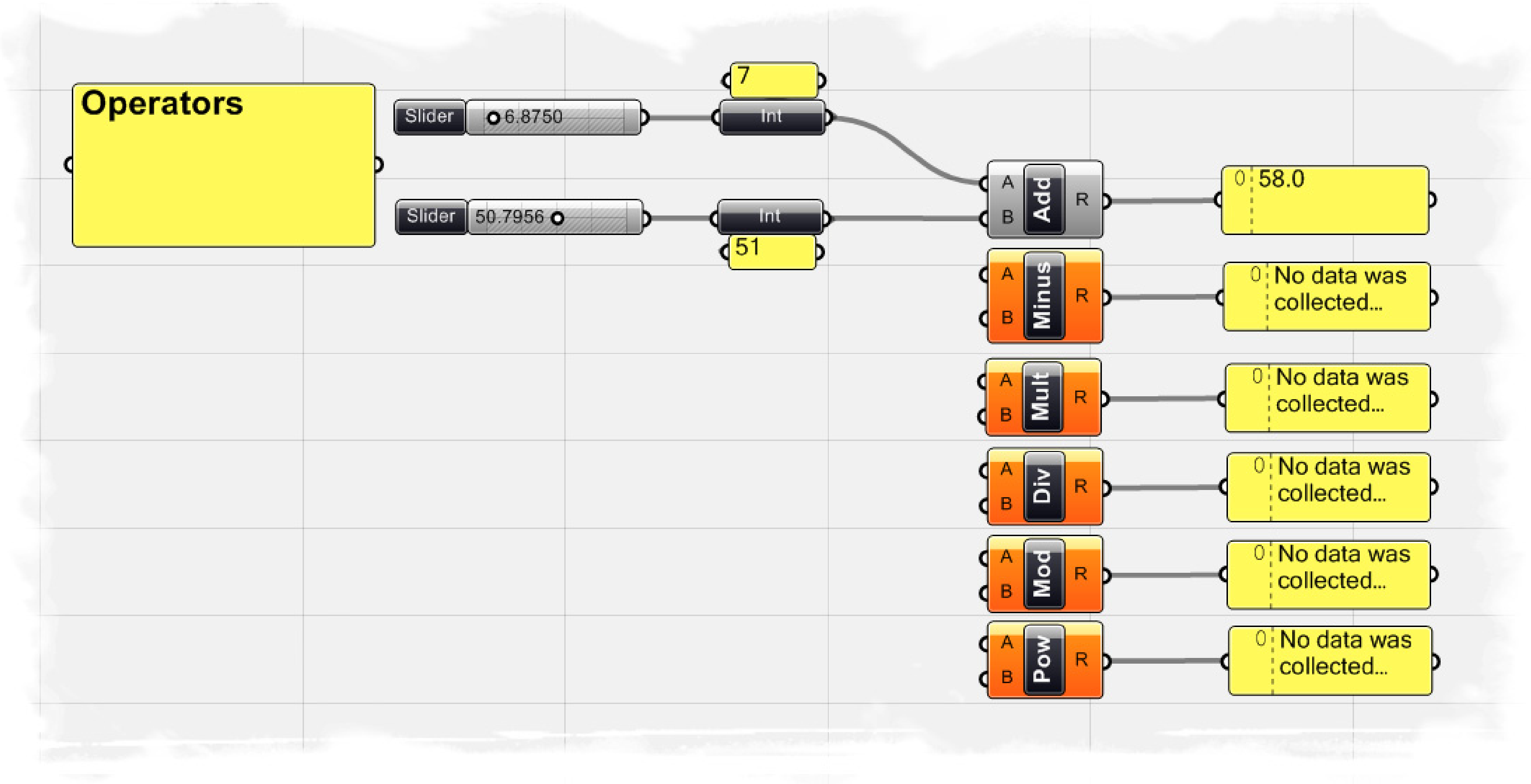
- Conditional Statements: Equality, Similarity, Larger than, Smaller than
- Booleans: 'If/Else' statements in grasshopper

E) Polynomials: used to define a polynomial function. Example: Function_Spiral.ghx

F) Trigonometry: used to define a trigonometric function such as sine, cosine, tangent, etc. Example: Trigonometric_curves.ghx

1. Data Storage

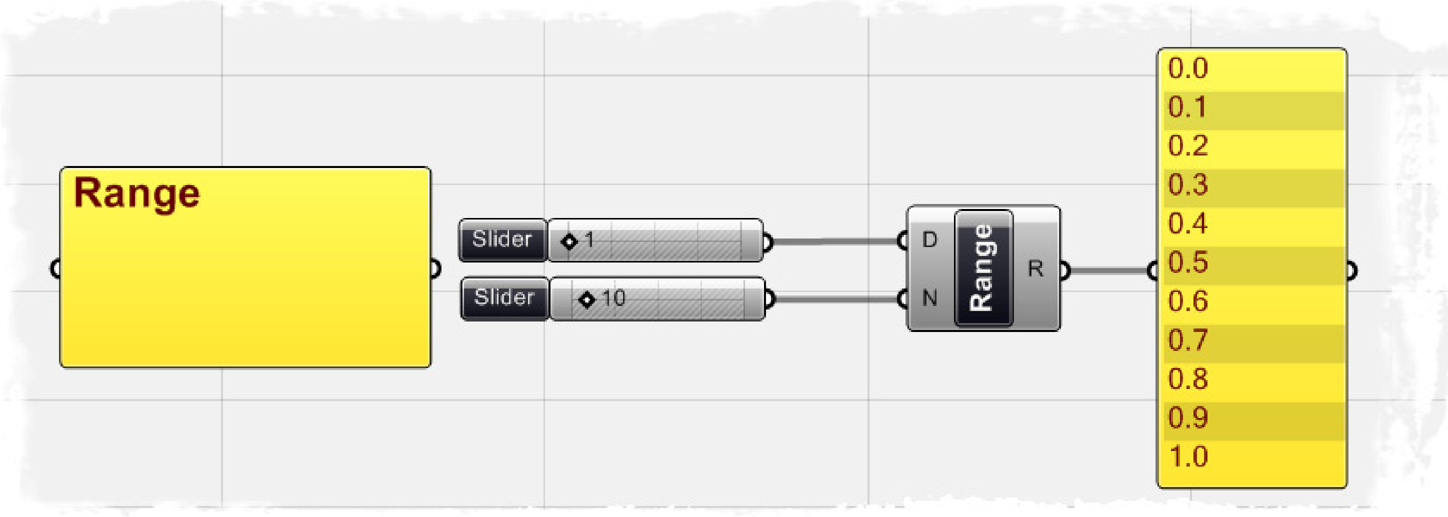
1.2 Scalars : (B) Operators

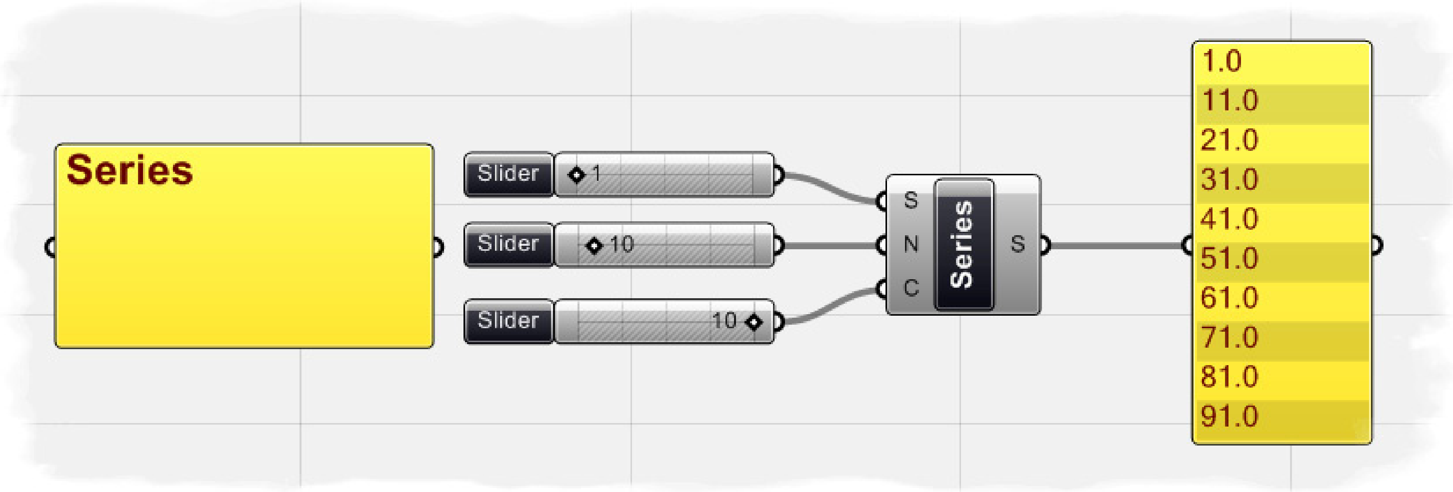


1. Data Storage

1.2 Scalars :

(C) Intervals
Range
Series

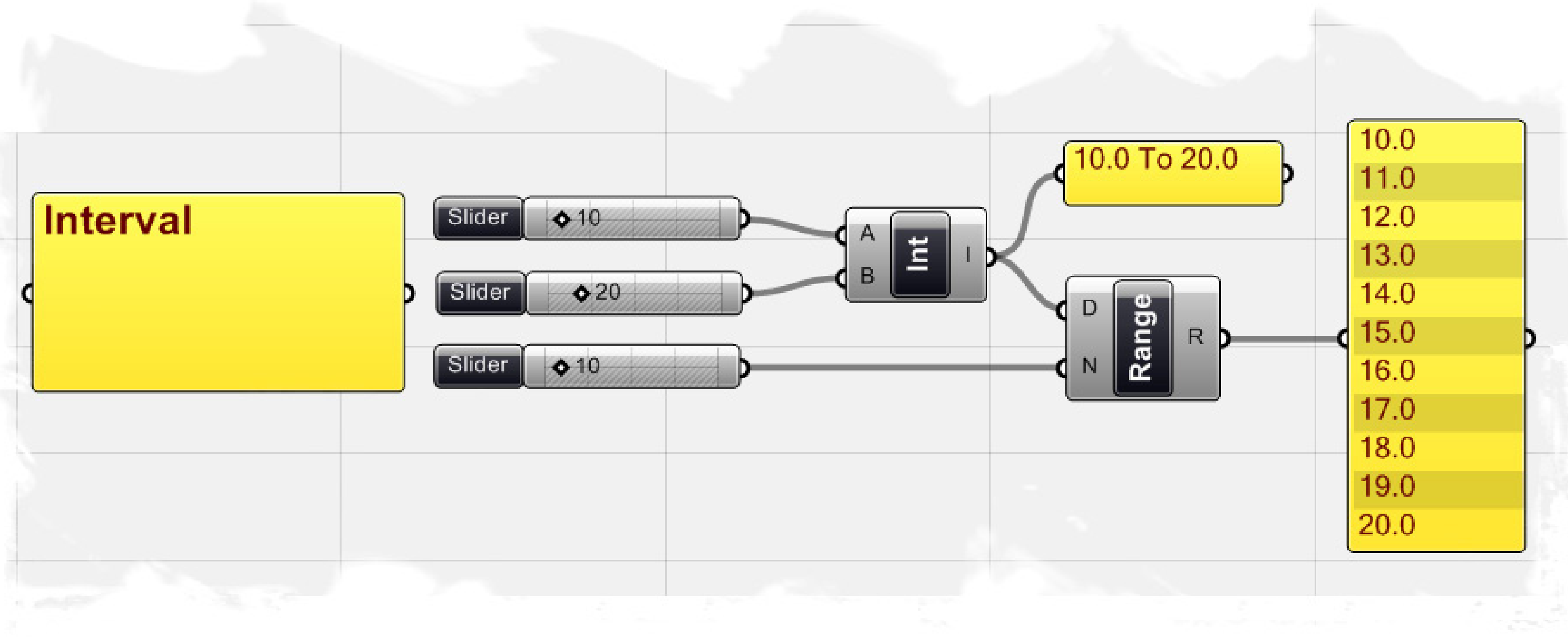




1. Data Storage

1.2 Scalars :

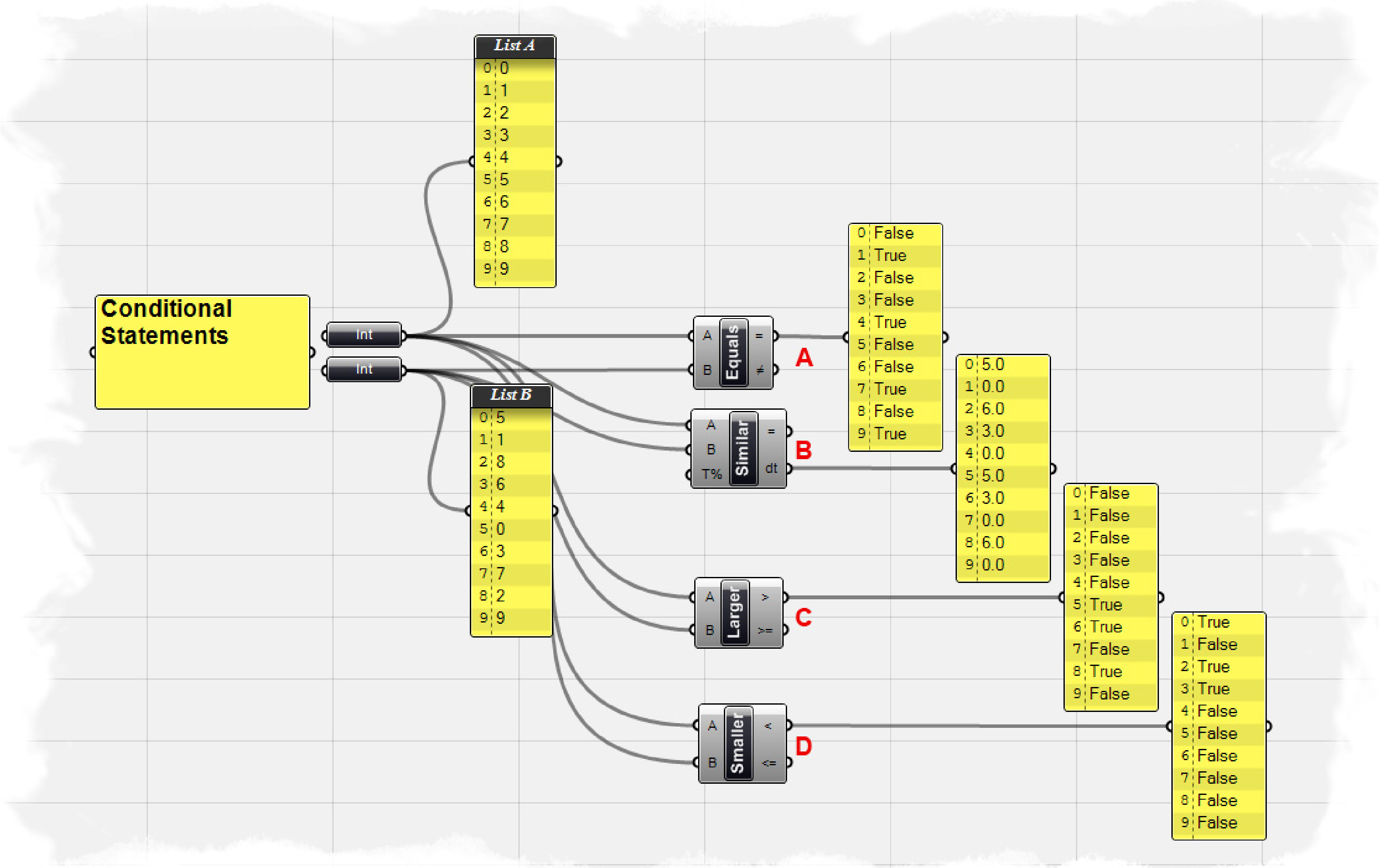
(C)
Intervals
Range
& Series



1. Data Storage

1.2 Scalars:

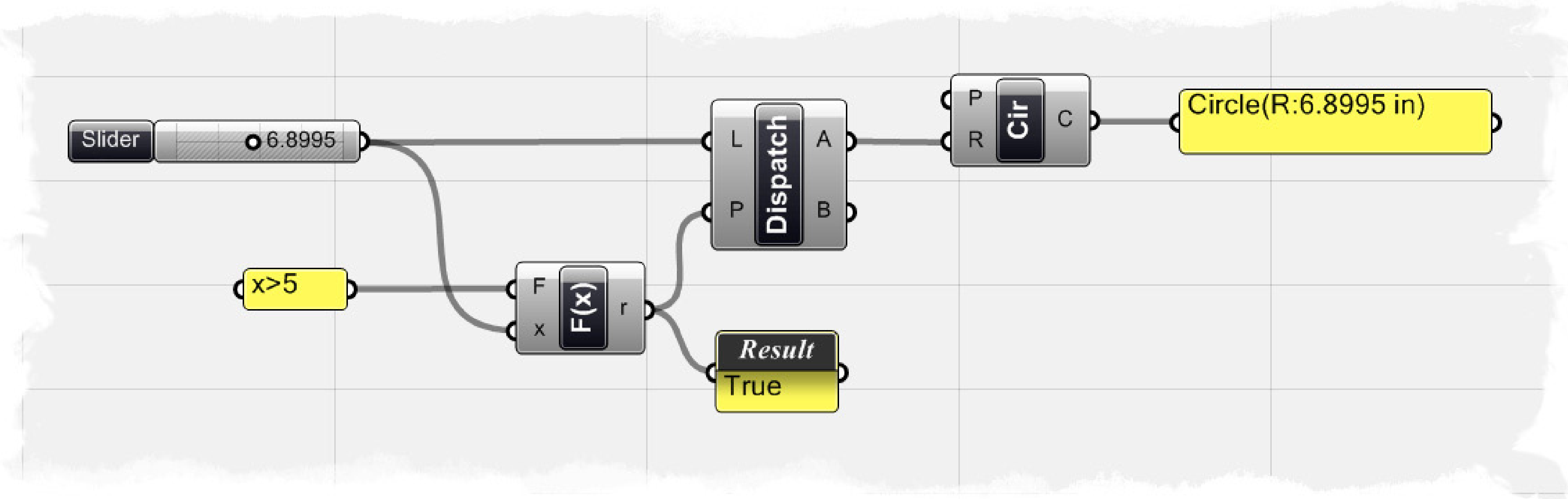
(D) Conditional Statements :



1. Data Storage

1.2 Scalars:

(D) Conditional Statements :

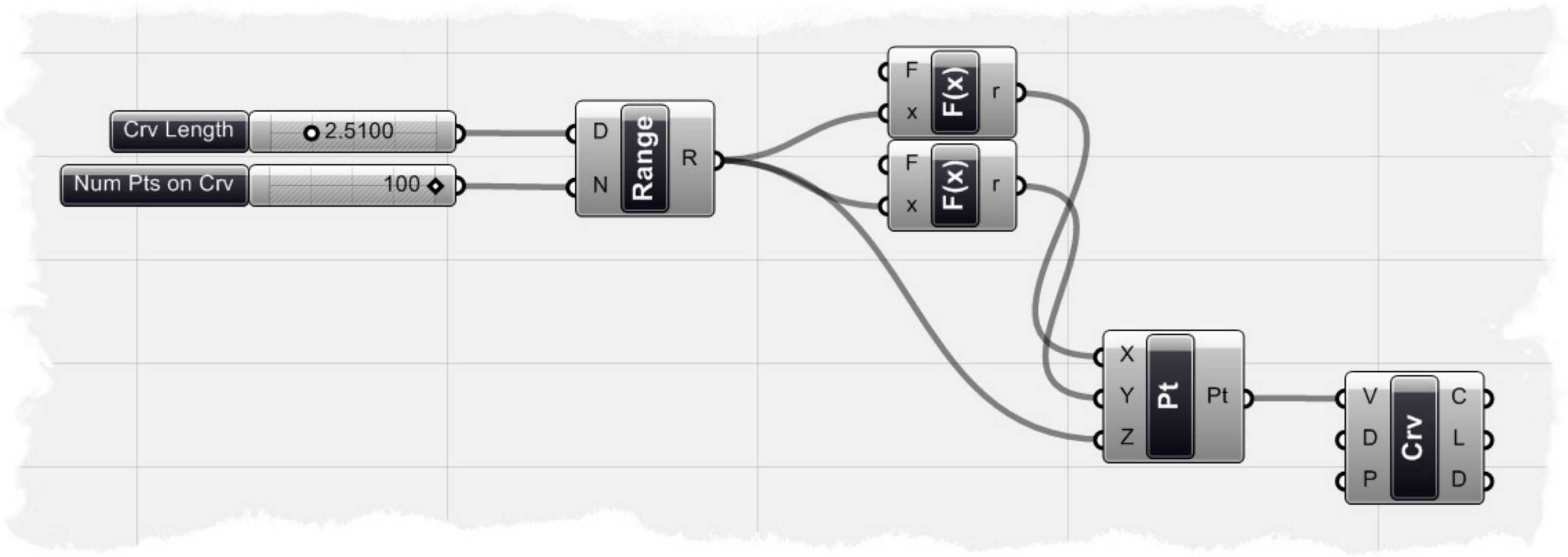


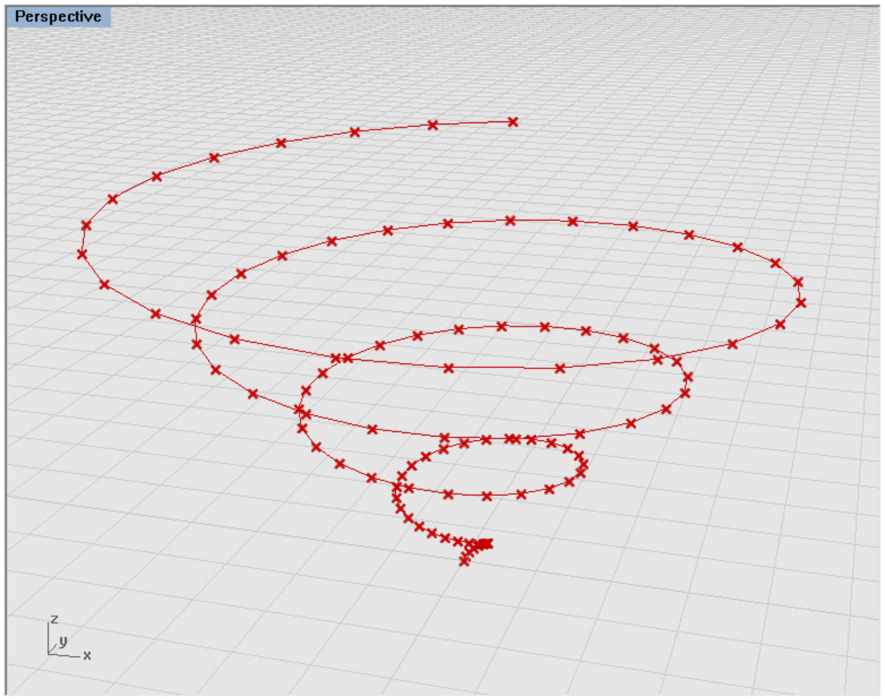
1. Data Storage

1.2 Scalars:

(E) Polynomials:

Spiral Function



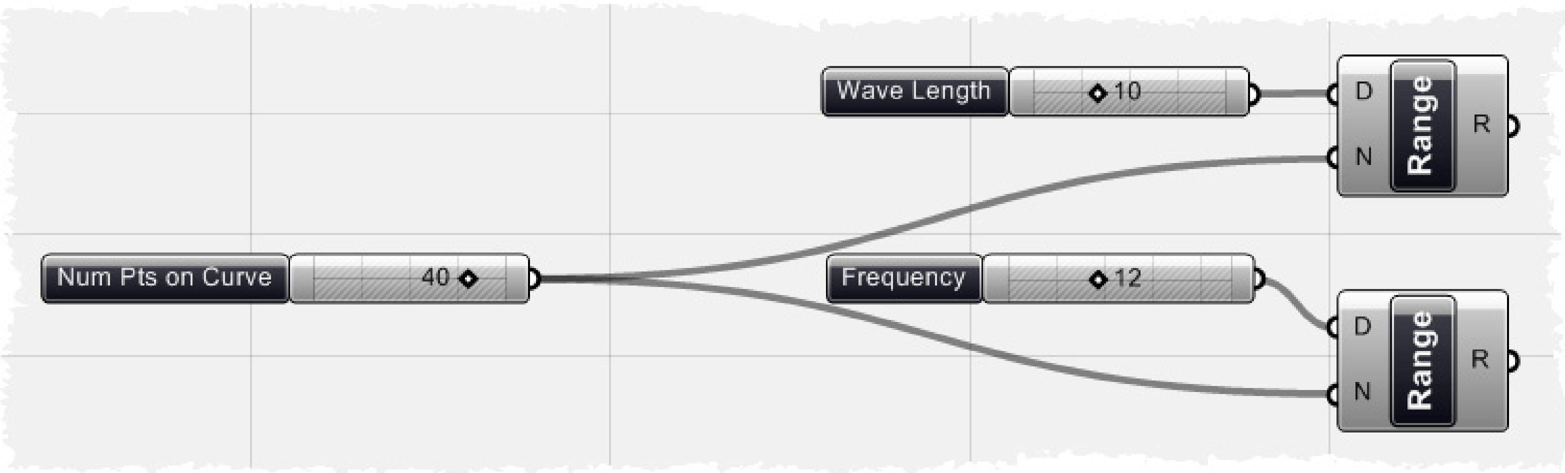


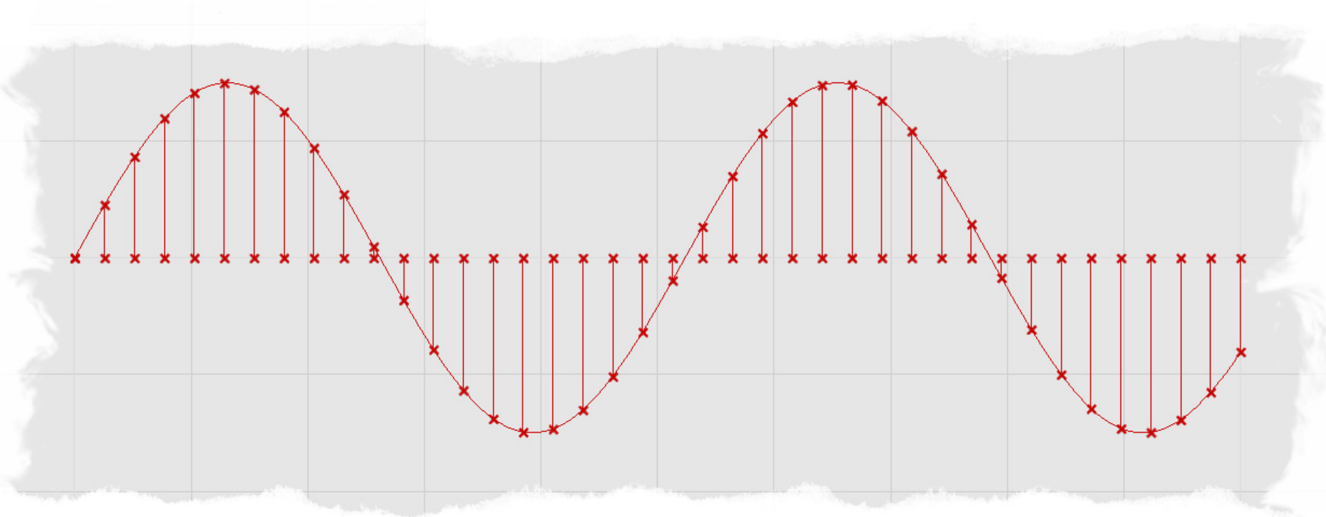
1. Data Storage

1.2 Scalars:

(F)

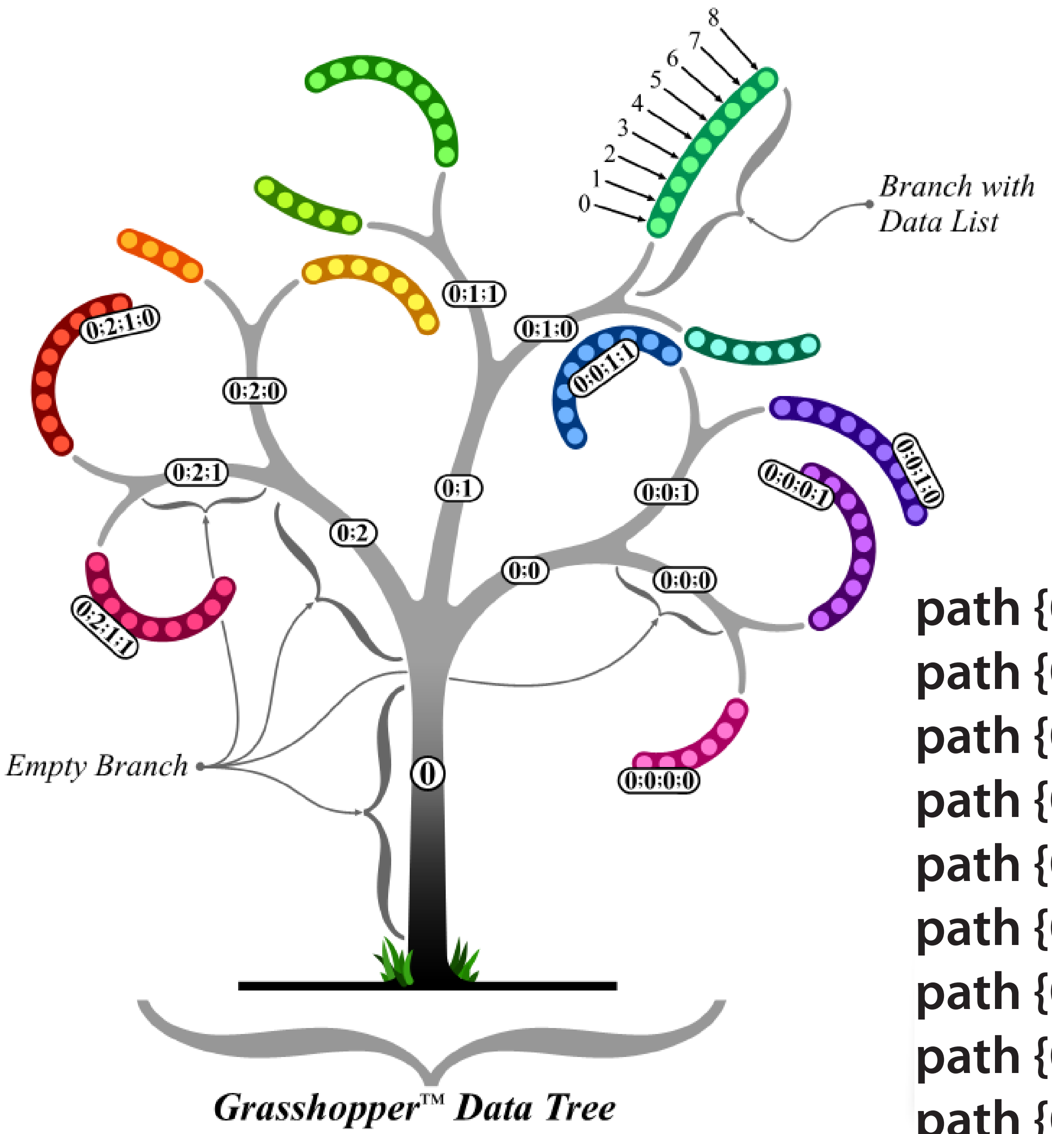
Trigonometric:





1. Data Storage

1.3 Data Trees



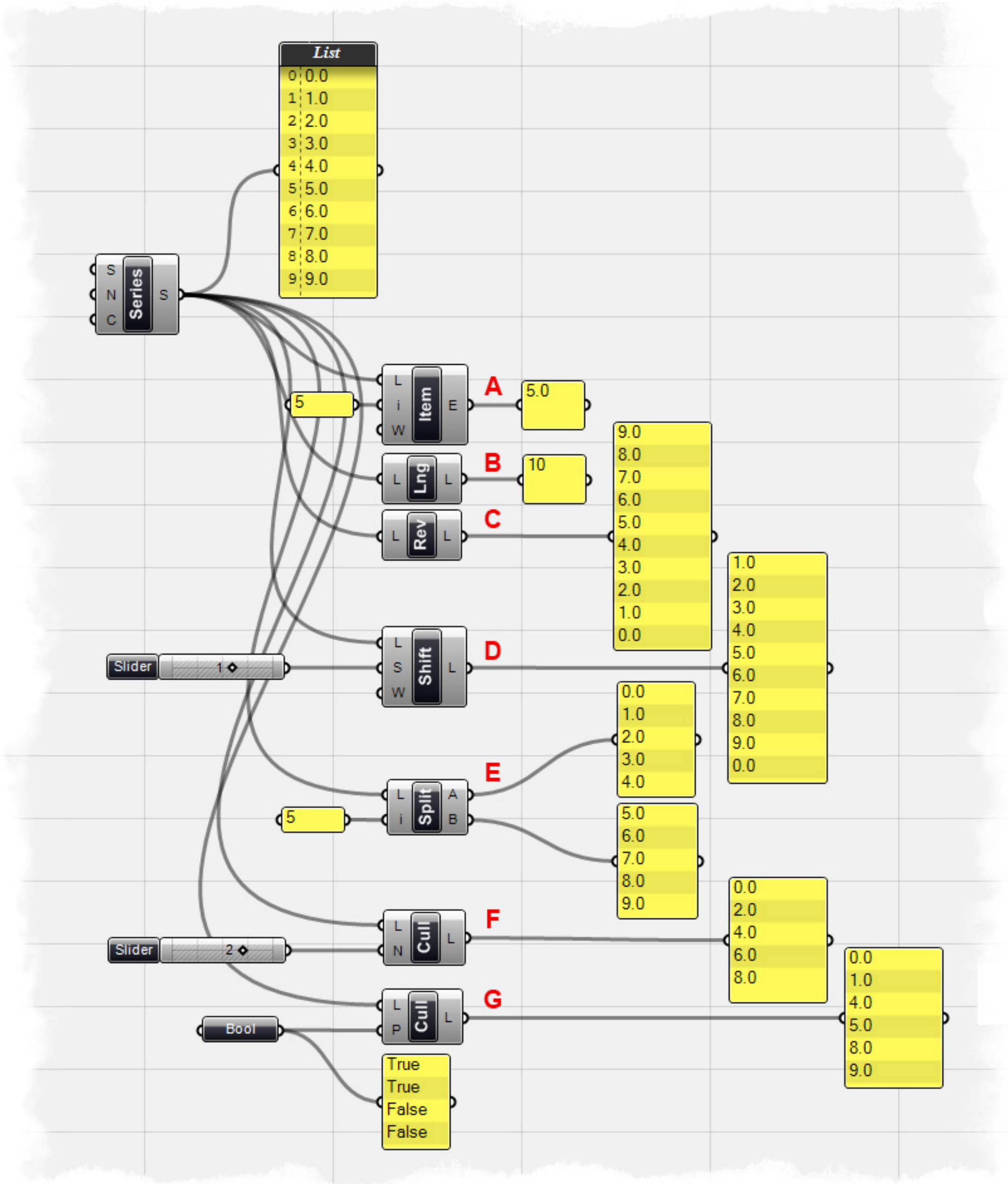
path {0;0;0;0} (N=6)
path {0;0;0;1} (N=9)
path {0;0;1;0} (N=9)
path {0;0;1;1} (N=9)
path {0;1;0;0} (N=6)
path {0;1;0;1} (N=9)
path {0;1;1;0} (N=9)
path {0;1;1;1} (N=5)
path {0;2;0;0} (N=7)
path {0;2;0;1} (N=4)
path {0;2;1;0} (N=9)
path {0;2;1;1} (N=9)

1. Data Storage

1.3 Data Trees

Lists & data management

- List Item
- List length
- List Reverse
- List Shift
- Split list
- Cull nth
- Cull Pattern
- Weaving Data



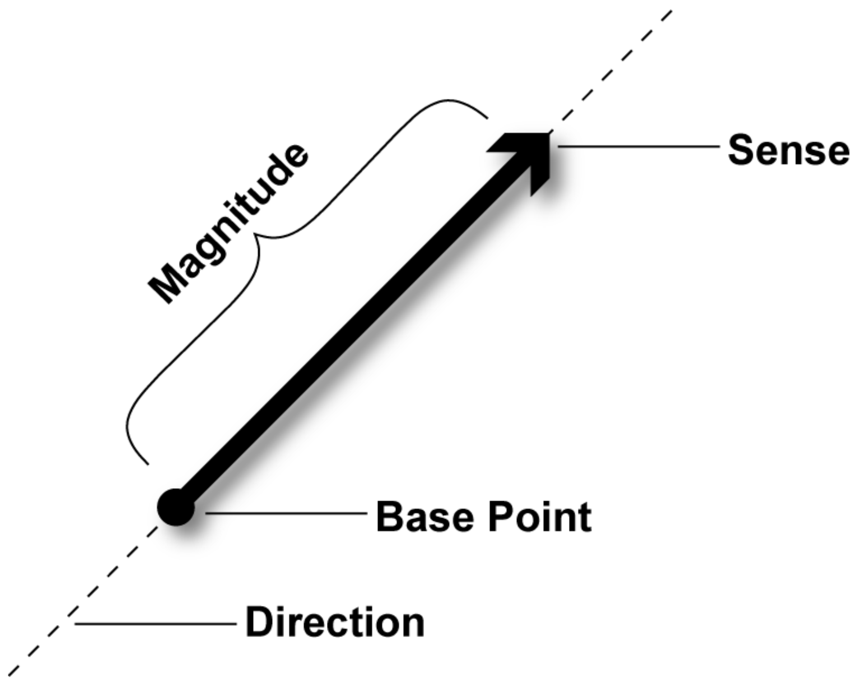
1. Data Storage

1.4 Vectors:

Definition

Vector Definition:

magnitude:	length of the vector; distance between the start and end points (scalar value)
direction:	the line ($-\infty$ to $+\infty$) on which the vector is based
sense:	whether vector is oriented towards $-\infty$ or $+\infty$



1. Data Storage

1.4 Vectors:

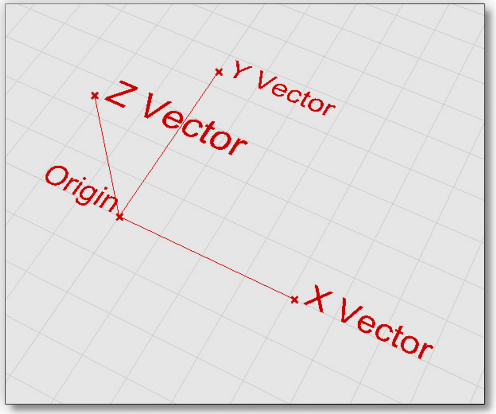
Vector Unit

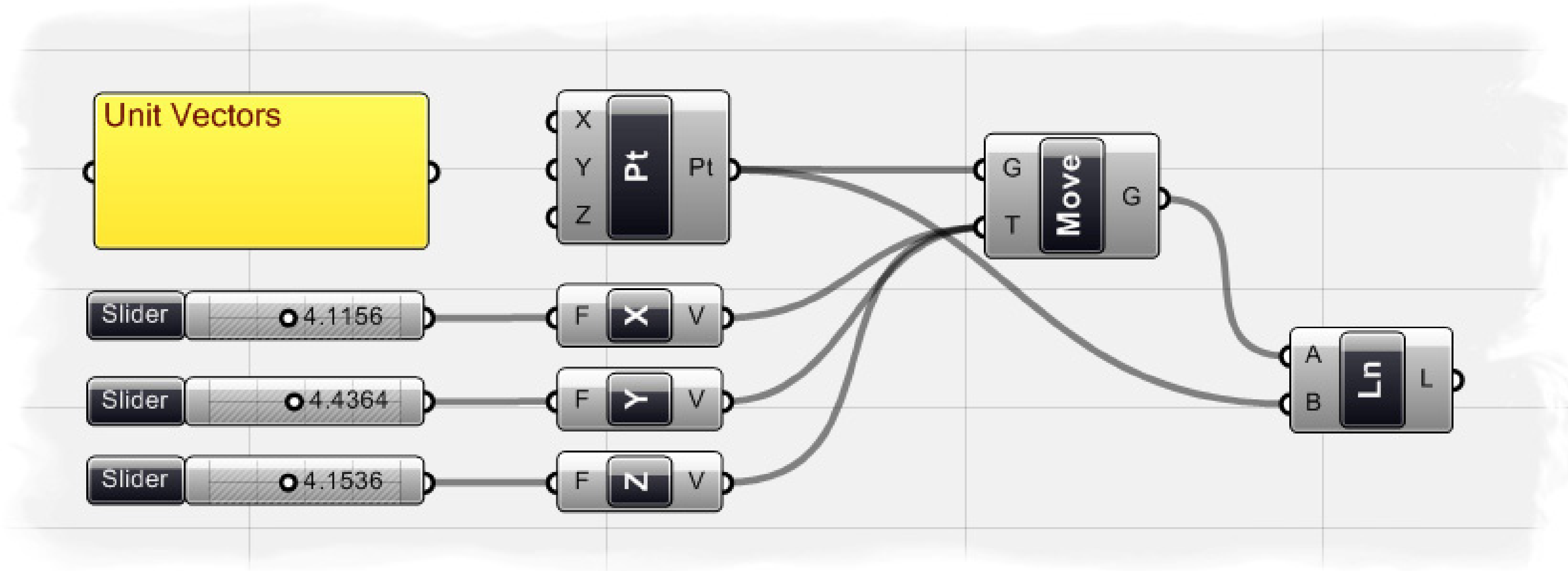
Vector Unit:

magnitude: 1 (the unity)

direction: x, y or z cartesian system axis

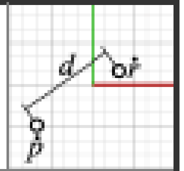
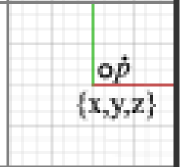
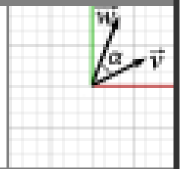
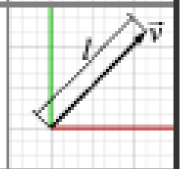
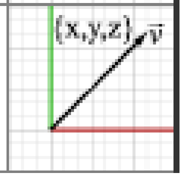
sense: cartesian system agreed senses





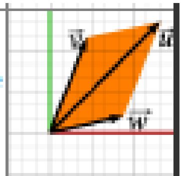
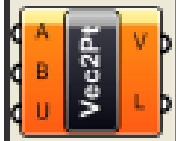
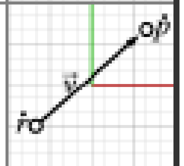
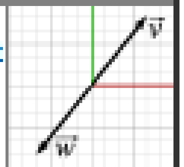
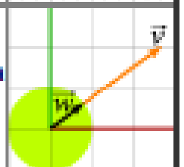
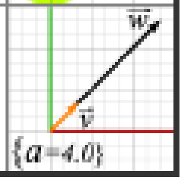
1. Data Storage

1.4 Vectors: Manipulation

Component	Location	Description	Example
	Vector/Point/**Distance**	Compute the Distance between two points (A and B inputs)	
	Vector/Point/**Decompose**	Break down a point into its X, Y, and Z components	
	Vector/Vector/**Angle**	Compute the angle between two vectors Output computed in Radians	
	Vector/Vector/**Length**	Compute the length (amplitude) of a vector	
	Vector/Vector/**Decompose**	Break down a vector into its component parts	

1. Data Storage

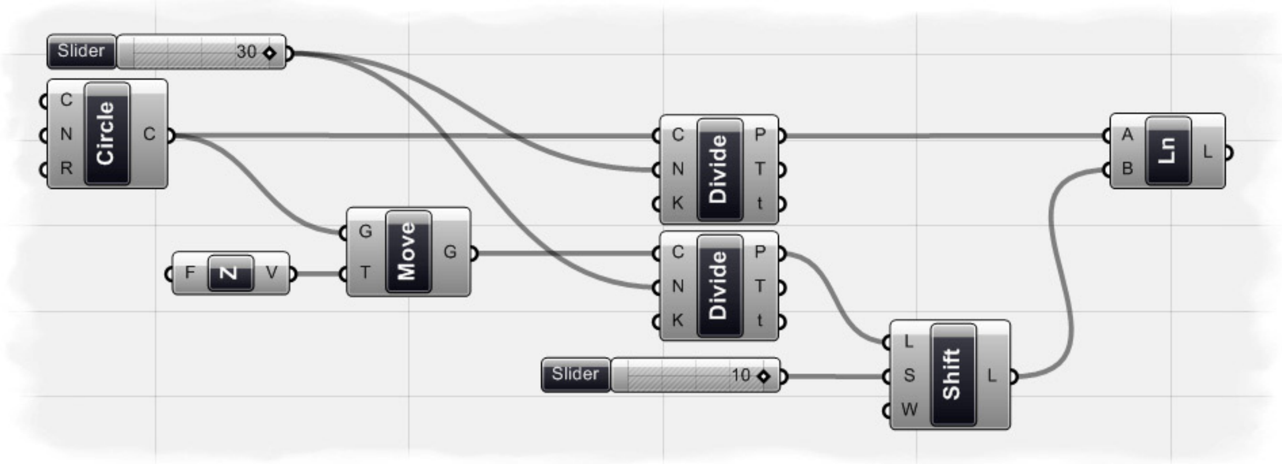
1.4 Vectors: Manipulation

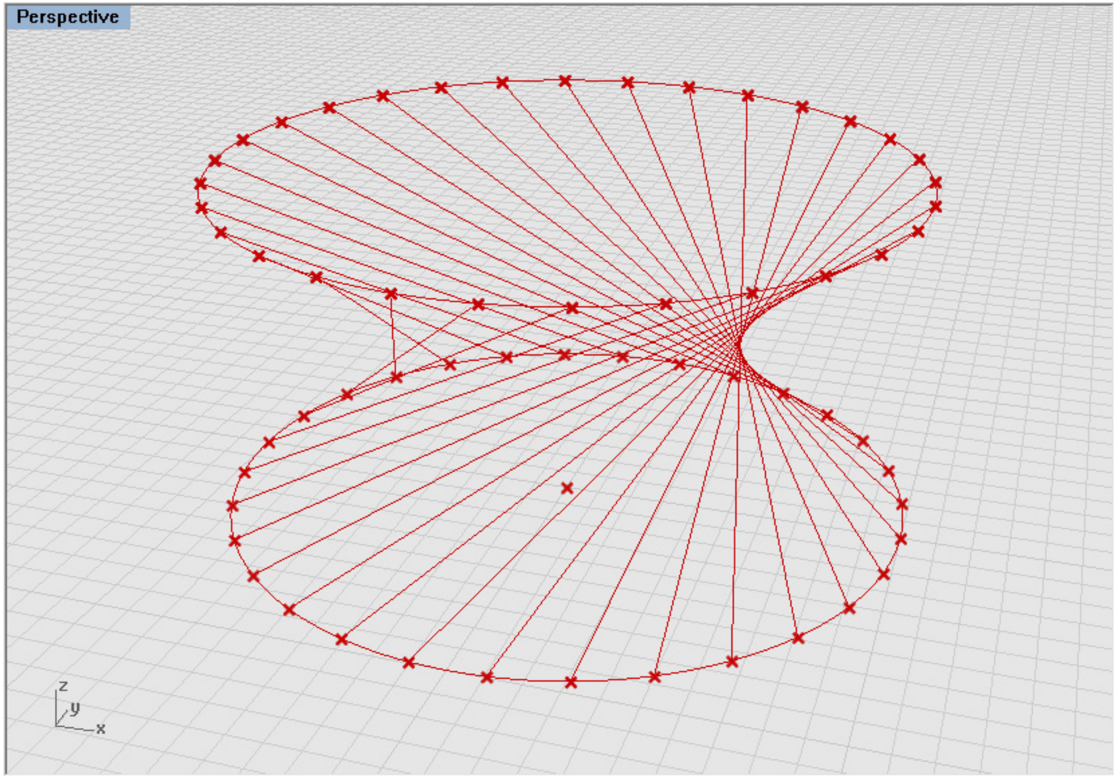
	Vector/Vector/**Summation**	Add the components of vector 1(A input) to the components of vector 2 (B input)	
	Vector/Vector/**Vector2pt**	Creates a vector from two defined points	
	Vector/Vector/**Reverse**	Negate all the components of a vector to invert the direction. The length of the vector is maintained	
	Vector/Vector/**Unit Vector**	Divide all components by the inverse of the length of the vector. The resulting vector has a length of 1.0 and is called the unit vector. Sometimes referred to as 'normalizing'	
	Vector/Vector/**Multiply**	Multiply the components of the vector by a specified factor	

1. Data Storage

1.4 Vectors:

Manipulation of Vectors & Data Trees Example 'Shift Data'



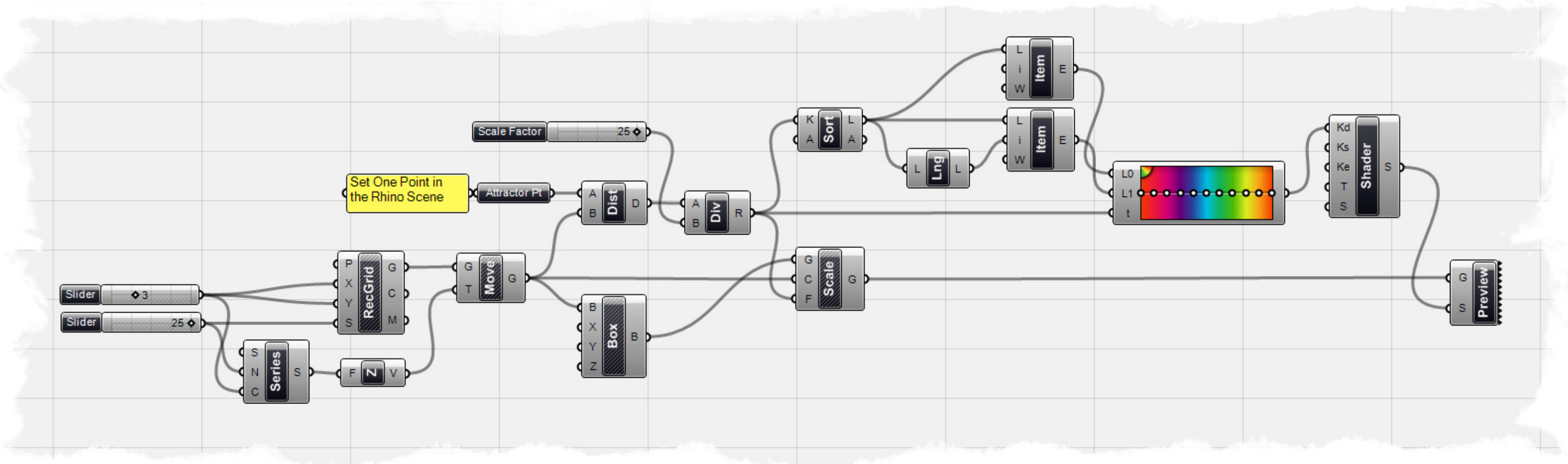


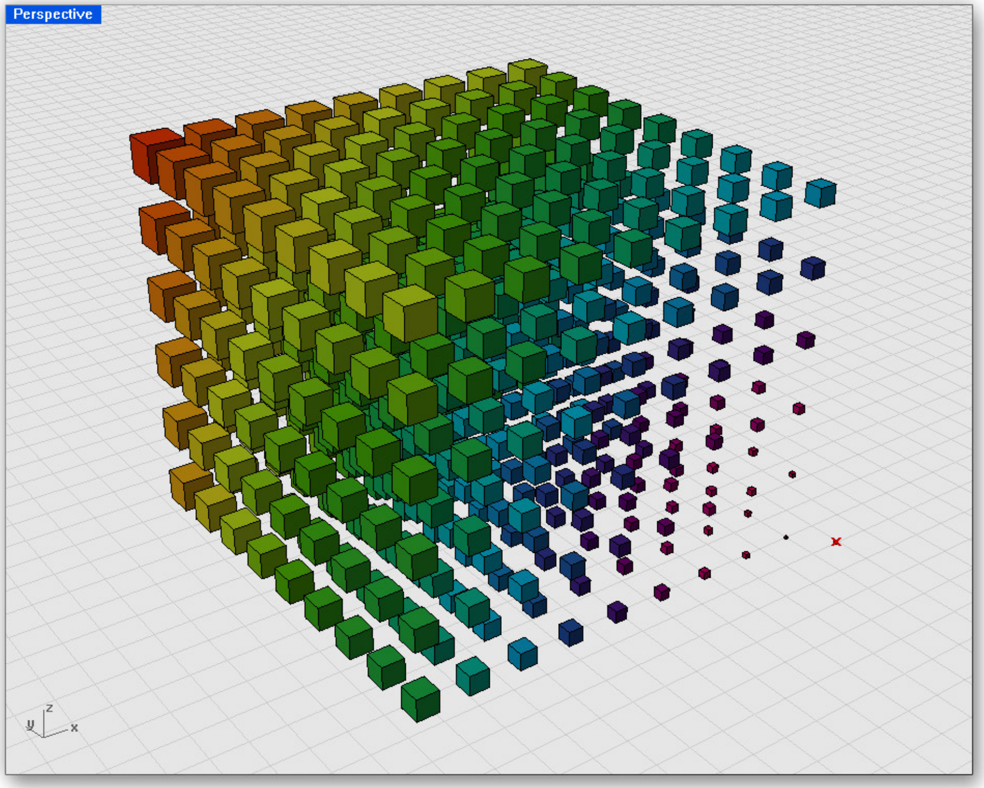
1. Data Storage

1.4 Vectors:

Manipulation of Vectors & Data Trees & Grids of Points.

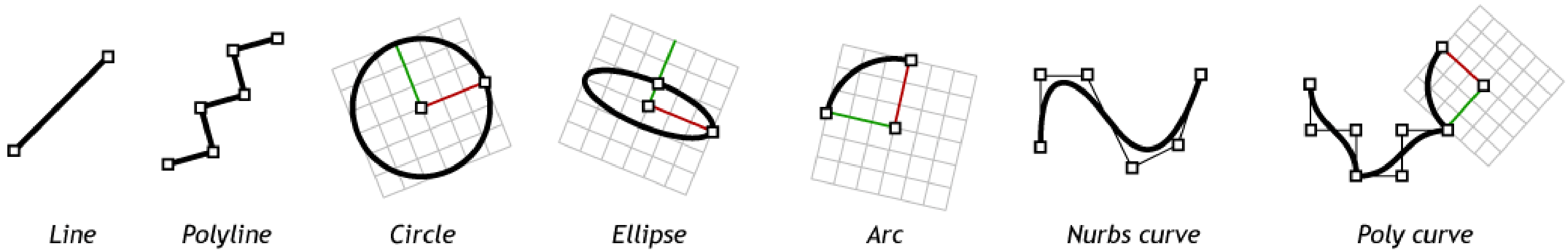
Example 'Scaled Boxes'



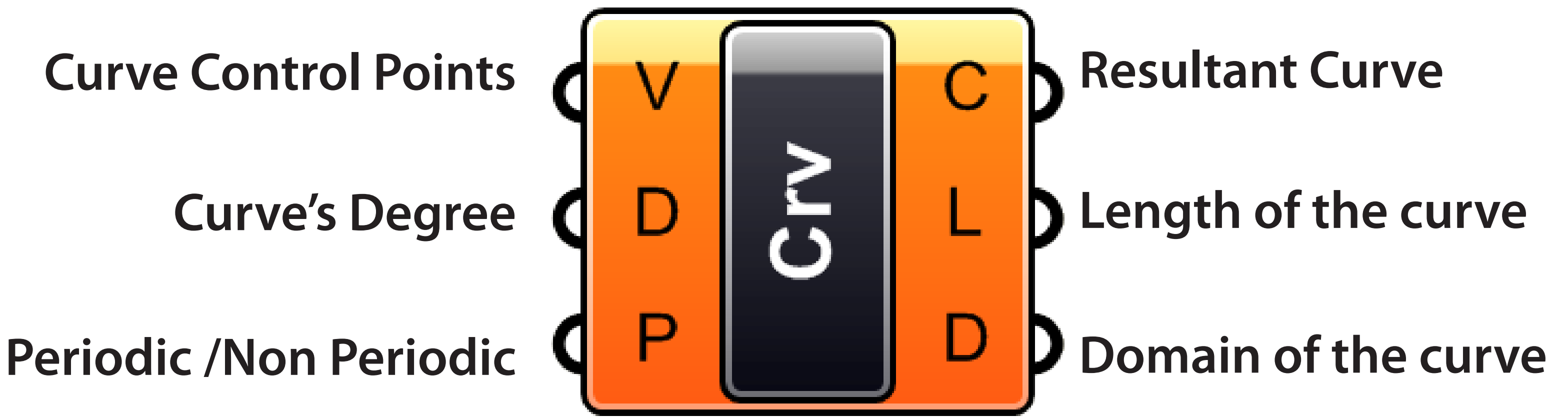


2. Curves

2.1 Definition & Type of Curves



NURBS Curve

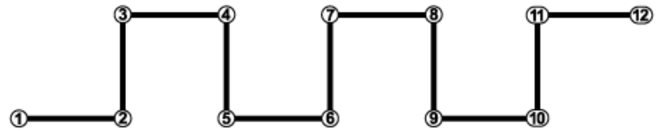


2. Curves

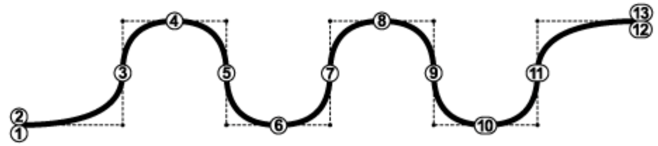
2.1 Definition & Type of Curves

Degree of a NURBS curve

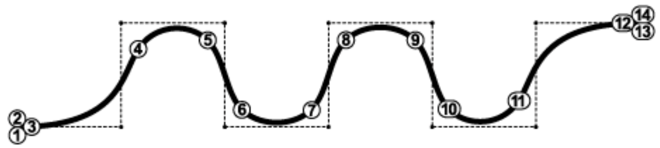
NURBS curve knot vectors as a result of varying degree



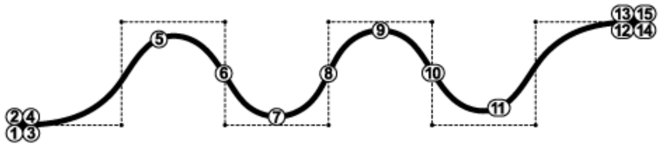
A D^1 nurbs curve behaves the same as a polyline. It follows from the knotcount formula that a D^1 curve has a knot for every control point. Thus, there is a one-to-one relationship.



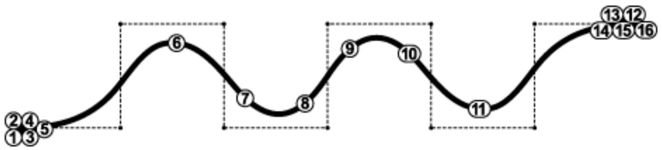
A D^2 nurbs curve is in fact a rare sighting. It always looks like it is over-stressed, but the knots are at least in straightforward locations. The spline intersects with the control polygon halfway each segment. D^2 nurbs curves are typically only used to approximate arcs and circles.



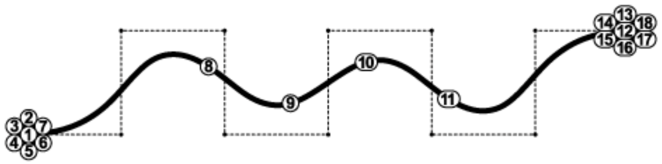
D^3 is the most common type of nurbs curve and -indeed- the default in Rhino. You are probably very familiar with the visual progression of the spline, eventhough the knots appear to be in odd locations.

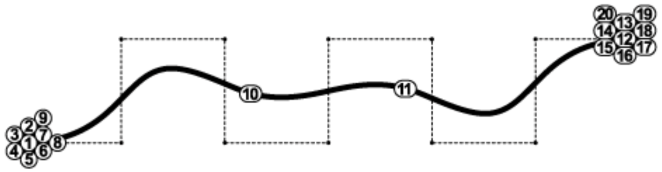


D^4 is technically possible in Rhino, but the math for nurbs curves doesn't work as well with even degrees. Odd numbers are usually preferred.



D^5 is also quite a common degree. Like the D^3 curves it has a natural, but smoother appearance. Because of the higher degree, control points have a larger range of influence.





D^7 and D^9 are pretty much hypothetical degrees. Rhino goes all the way up to D^{11}, but these high-degree-splines bear so little resemblance to the shape of the control polygon that they are unlikely to be of use in typical modeling applications.

2. Curves

2.1 Definition & Type of Curves

Other Types:

Interpolated Curve

Kinky Curve

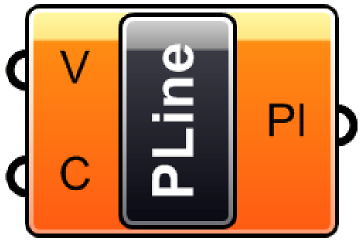
Polyline

PolyArc

Curve through control points

Similar to interpolated; kinnks at each vertex; requires angle (A) of kink

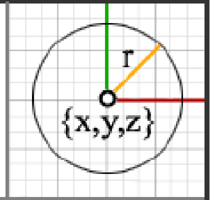
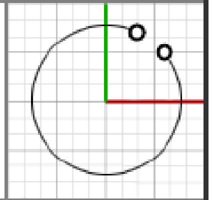
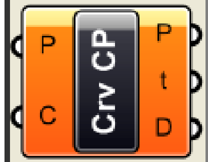
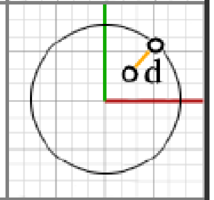
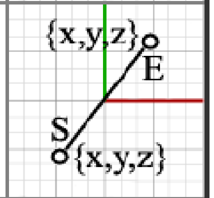
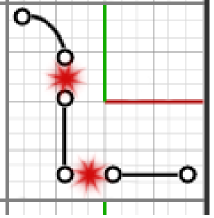
Curve made of straight segments



Curve made of arched segments

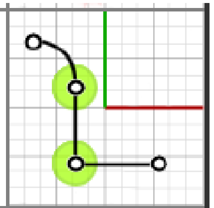
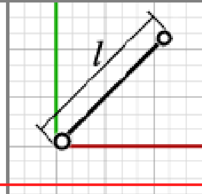
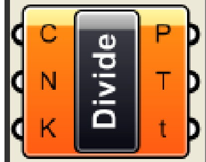
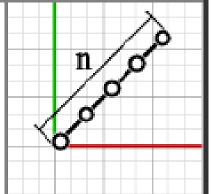
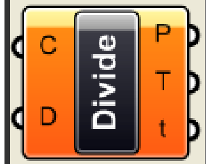
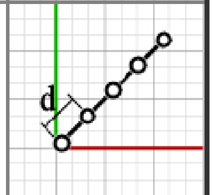
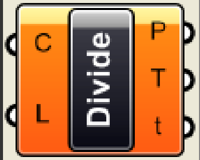
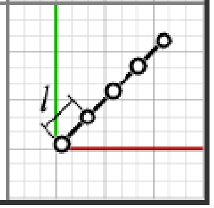
2. Curves

2.2 Curve Analytics

Component	Location	Description	Example
	Curve/Analysis/**Center**	Find the center point and radius of arcs and circles	
	Curve/Analysis/**Closed**	Test if a curve is closed or periodic	
	Curve/Analysis/**Closest Point**	Find the closest point on a curve to any sample point in space	
	Curve/Analysis/**End Points**	Extract the end points of a curve.	
	Curve/Analysis/**Explode**	Decompose a curve into its component parts	

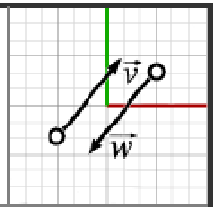
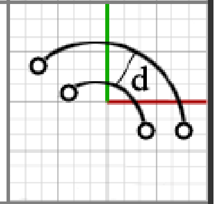
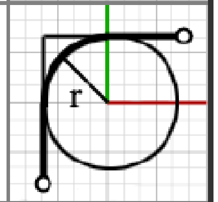
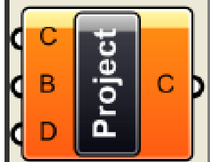
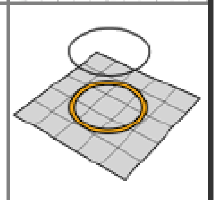
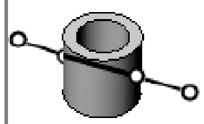
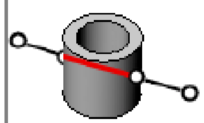
2. Curves

2.2 Curve Analytics

	Curve/Utility/**Join Curves**	Join as many curve segments together as possible	
	Curve/Analysis/**Length**	Measure the length of a curve	
	Curve/Division/**Divide Curve**	Divide a curve into a equal length segments	
	Curve/Division/**Divide Distance**	Divide a curve with a preset distance between points	
	Curve/Division/**Divide Length**	Divide a curve with a segments with a preset length	

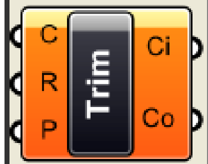
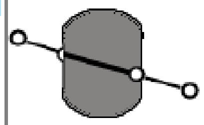
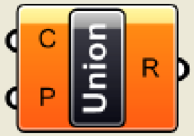
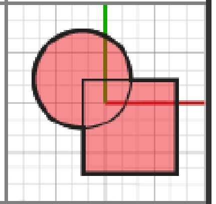
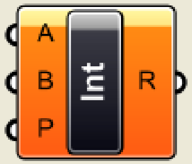
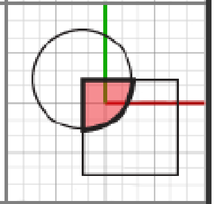
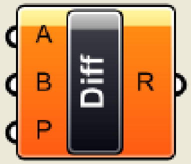
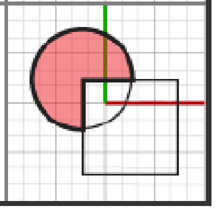
2. Curves

2.2 Curve Analytics

	Curve/Utility/**Flip**	Flip the direction of a curve using an optional guide curve	
	Curve/Utility/**Offset**	Offset a curve with a specified distance	
	Curve/Utility/**Fillet**	Fillets the sharp corners of a curve with an input radius	
	Curve/Utility/**Project**	Project a curve onto a Brep (a Brep is a set of joined surfaces like a polysurface in Rhino)	
	Intersect/Region/**Split with Brep(s)**	Split a curve with one or more Breps	
	Intersect/Region/**Trim with Brep(s)**	Trim a curve with one or more Breps. The **Ci** (Curves Inside) and **Co** (Curves Outside) outputs indicate the direction in which you would like the trim to occur.	

2. Curves

2.2 Curve Analytics

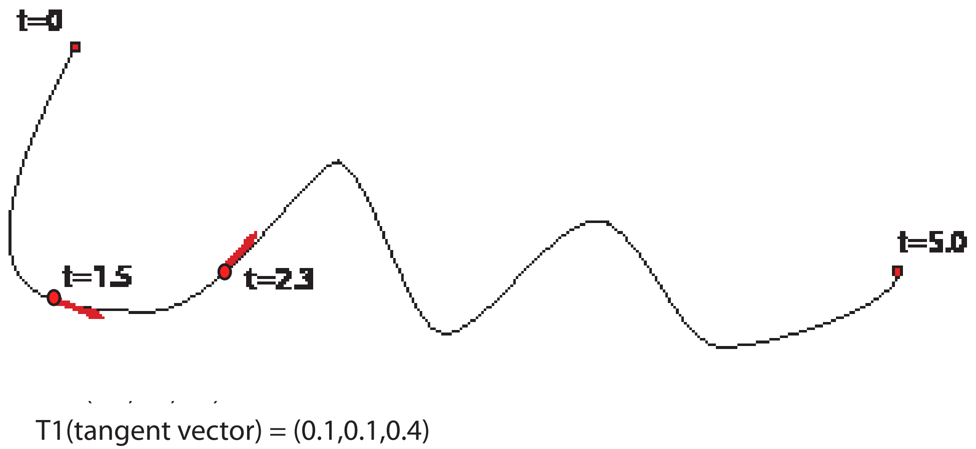
	Intersect/Region/**Trim with Region(s)**	Trim a curve with one or more Regions. The **Ci** (Curves Inside) and **Co** (Curves Outside) outputs indicate the direction in which you would like the trim to occur.	
	Intersect/Boolean/**Region Union**	Finds the outline (or union) of two planar closed curves	
	Intersect/Boolean/**Region Intersection**	Finds the intersection of two planar closed curves	
	Intersect/Boolean/**Region Difference**	Finds the difference between two planar closed curves	

2. Curves

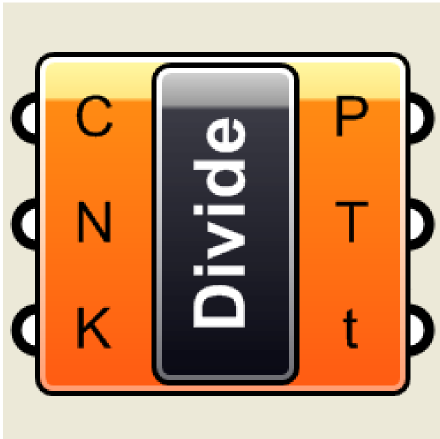
2.2 Curve Analytics

Curve Divide & Curve Domain

Curve Domain



Divide curve by number of Divisons

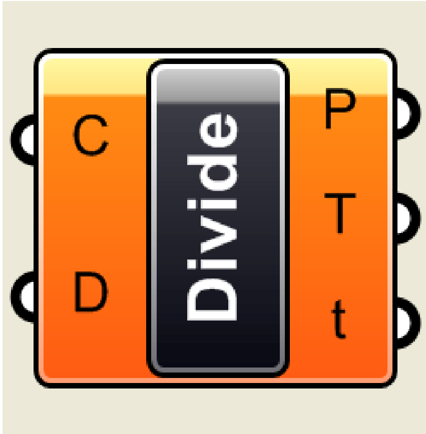


2. Curves

2.2 Curve Analytics

Curve Domain Other Options:

Divide curve by Distance

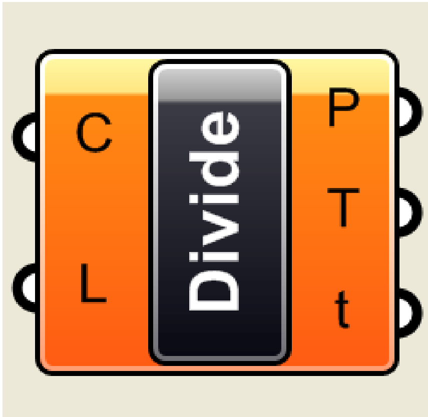


3D Points at Division

Tangent at Divisions

Domain parameter at divisions

Divide curve by length



3D Points at Division

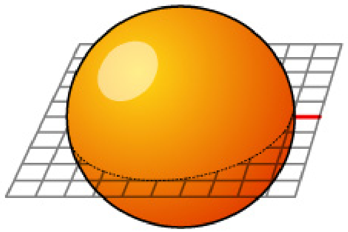
Tangent at Divisions

Domain parameter at divisions

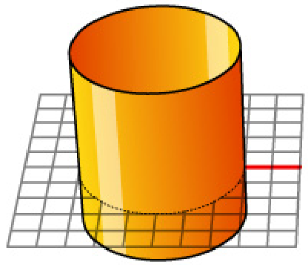
3. Surfaces

3.1 Type of Surfaces

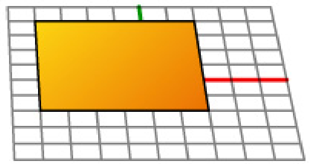
Primitives



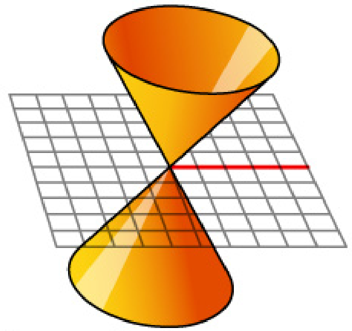
Sphere primitive
{Plane; Radius}



Cylinder primitive
{Plane; Radius; Height}

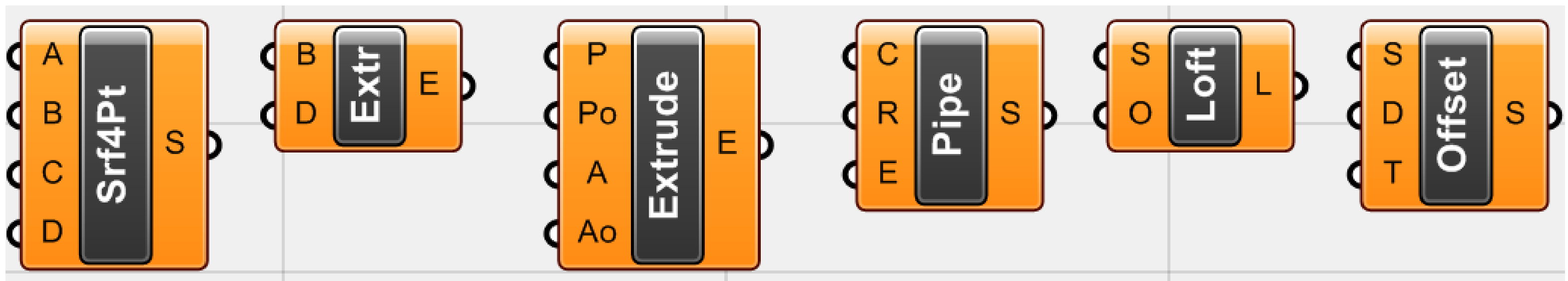


Plane primitive
{Plane; Width; Height}



Cone primitive
{Plane; Radius; Height}

Free-Form



Surface from 4 corner points

Extrude Curves

Extrude linear

Pipe

Loft Curves

Offset Surface

3. Surfaces

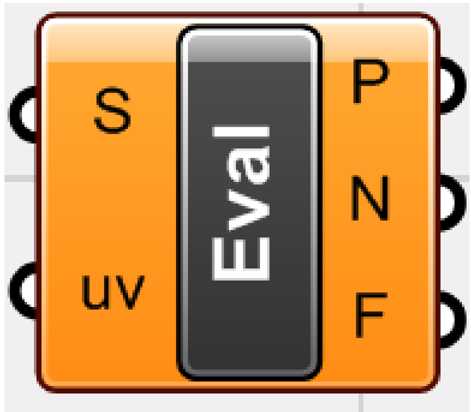
3.1 Surface Analytics:

Surface Domain

Tangent Plane and Normal Vector

Surface

u and v coordinates to evaluate



3D Points at u,v coordinates

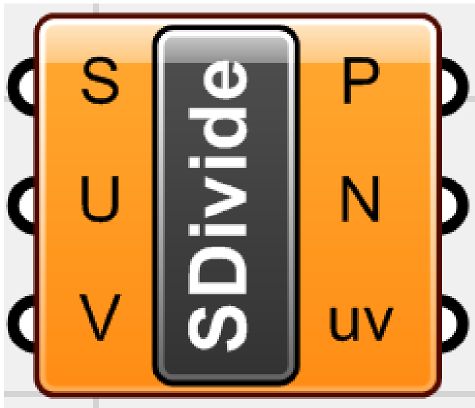
Normal Vectors at u,v

Tangent Plane (Frame)

Surface Divide (to obtain coordinates in u,v directions)

Surface

Divisions in V



3D Points at u,v coordinates

Normal Vectors at u,v

u,v coordinates from division of surface

3. Surfaces

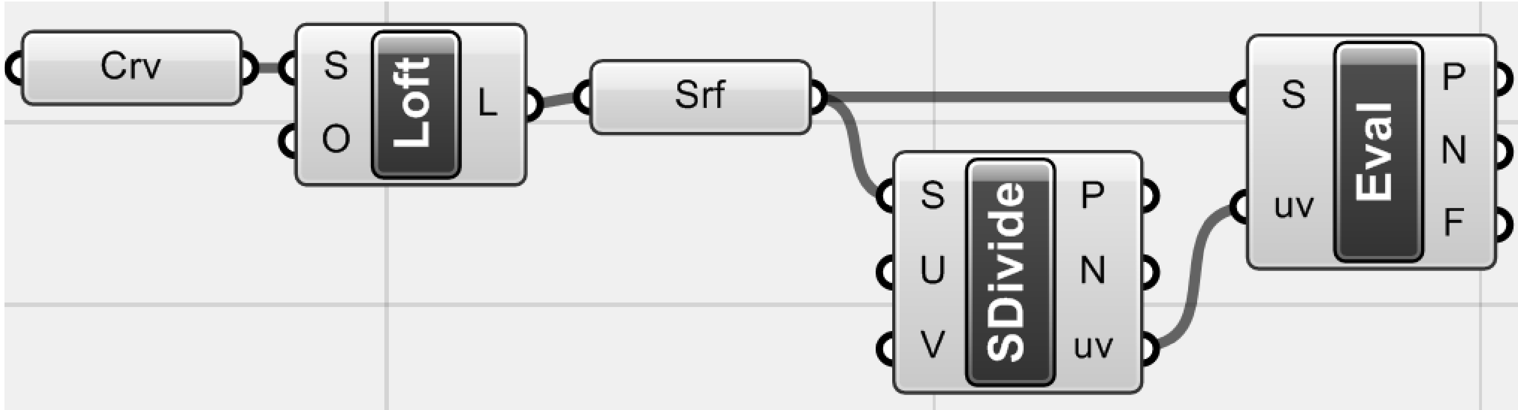
3.1 Surface Analytics:

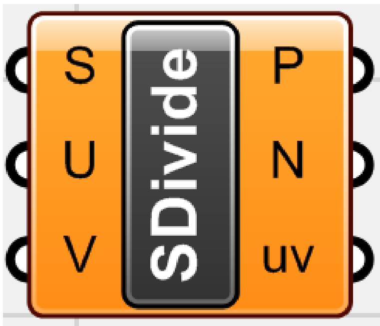
Surface Domain

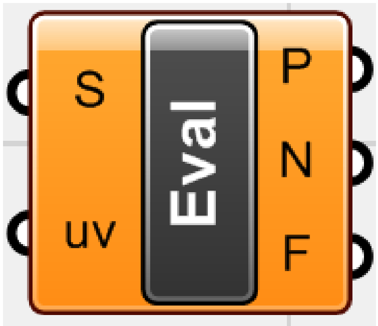
Tangent Plane and Normal Vector

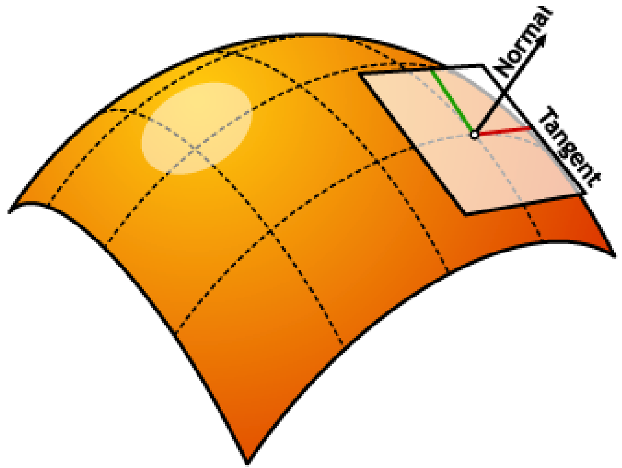
Normal Vectors & Tangent Planes:

Surface Divide + Evaluate Surface





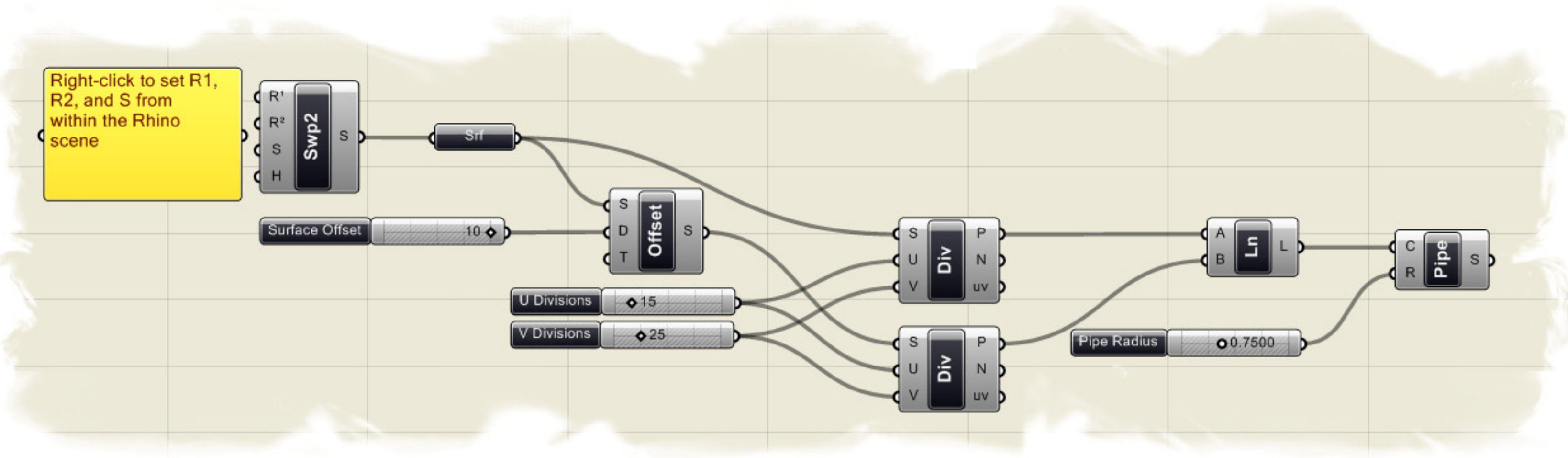


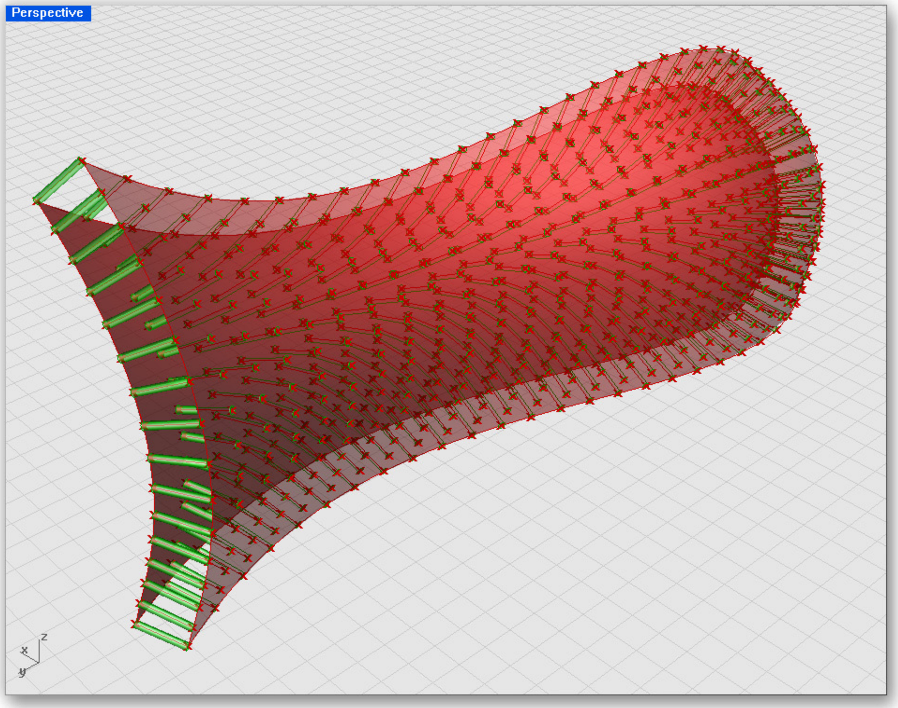


3. Surfaces

3.1 Surface Analytics:

Example Surface Connect

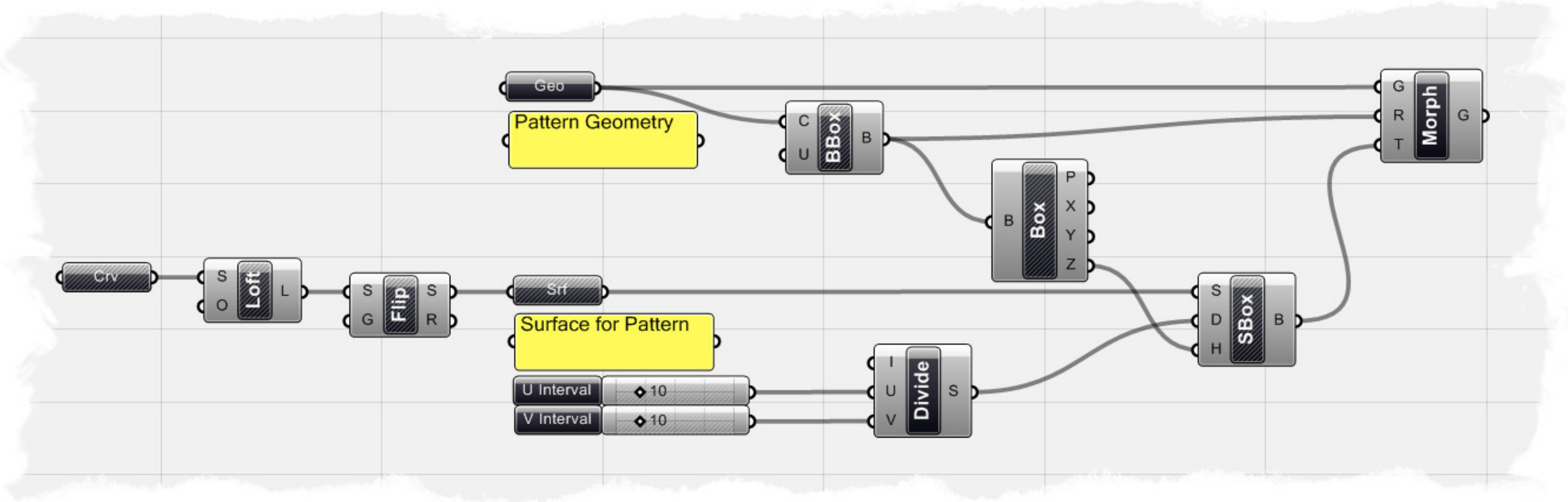


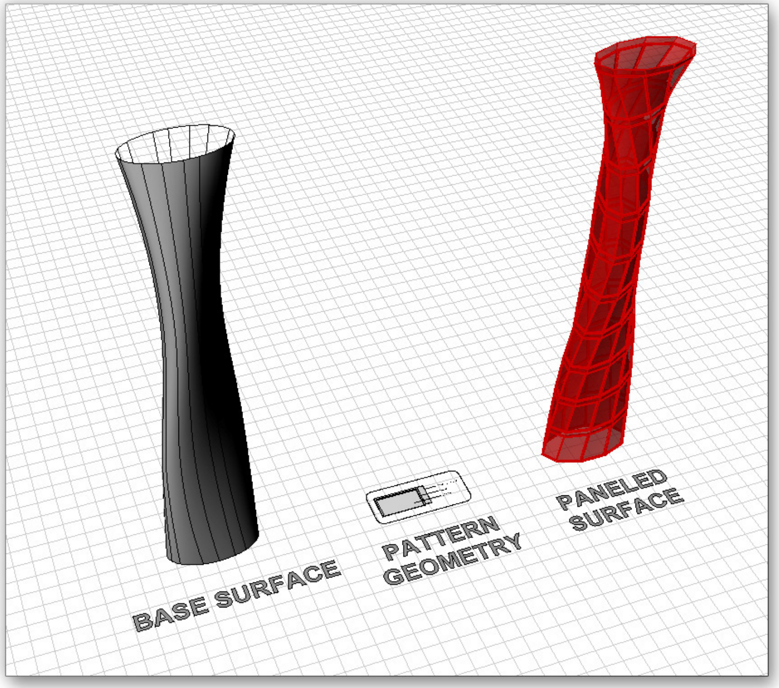


3. Surfaces

3.1 Surface Analytics:

Example Panelisation

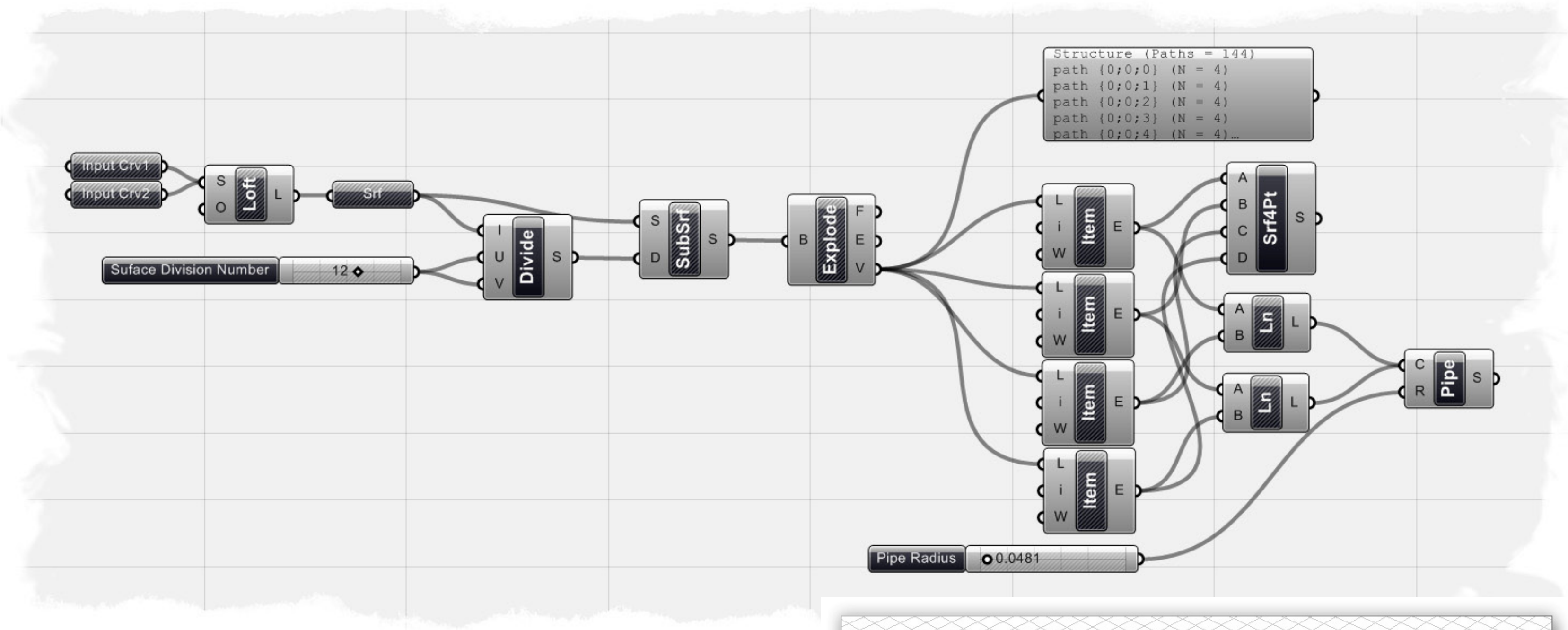


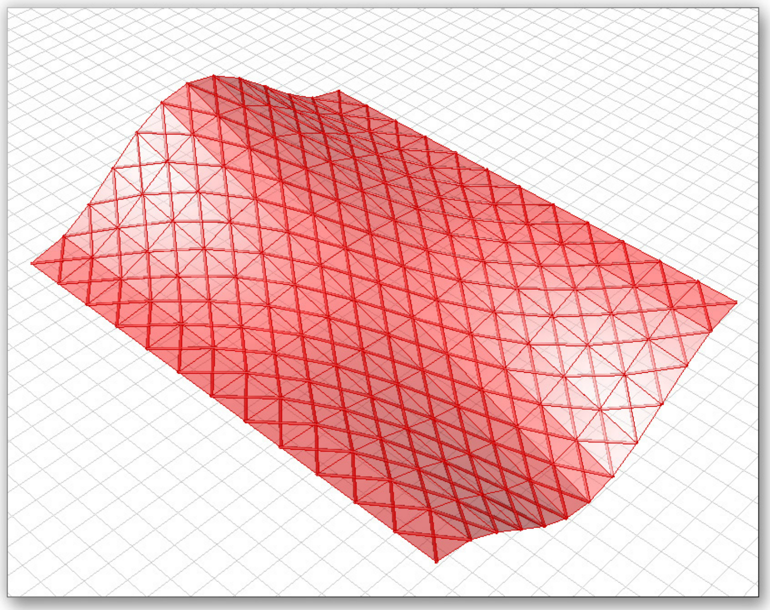


3. Surfaces

3.1 Surface Analytics:

Surface Diagrid (Uniform)



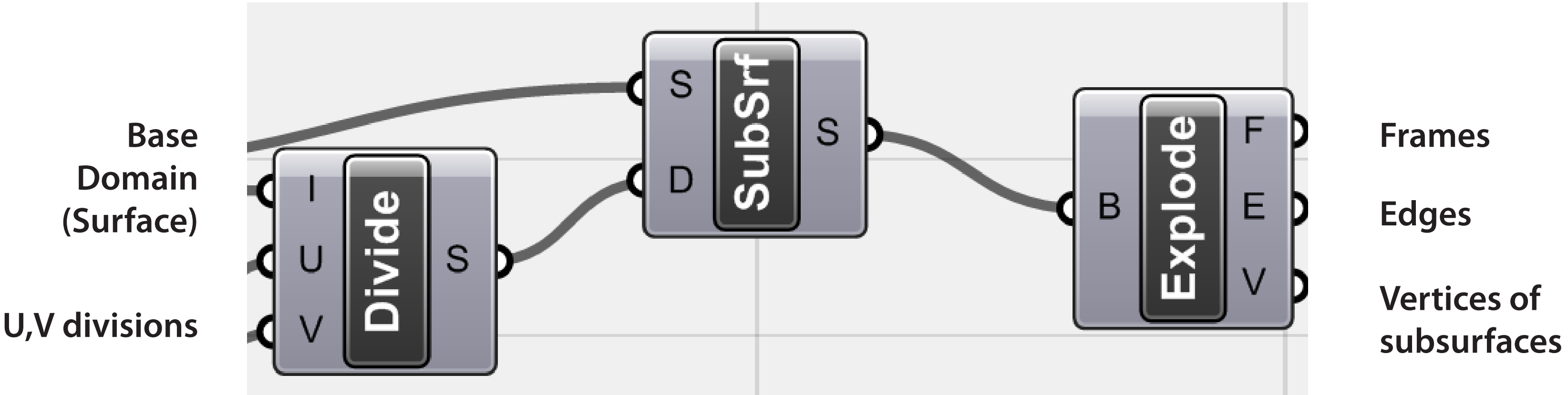


3. Surfaces

3.1 Surface Analytics:

Surface Diagrid (Uniform)

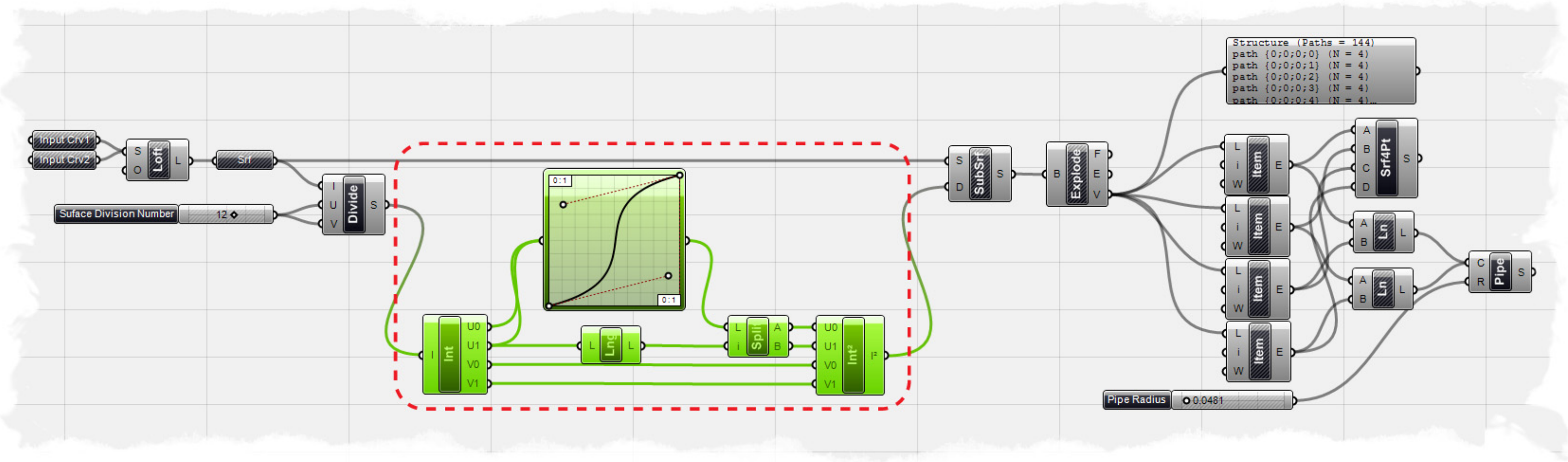
Divide + SubSurface + Explode

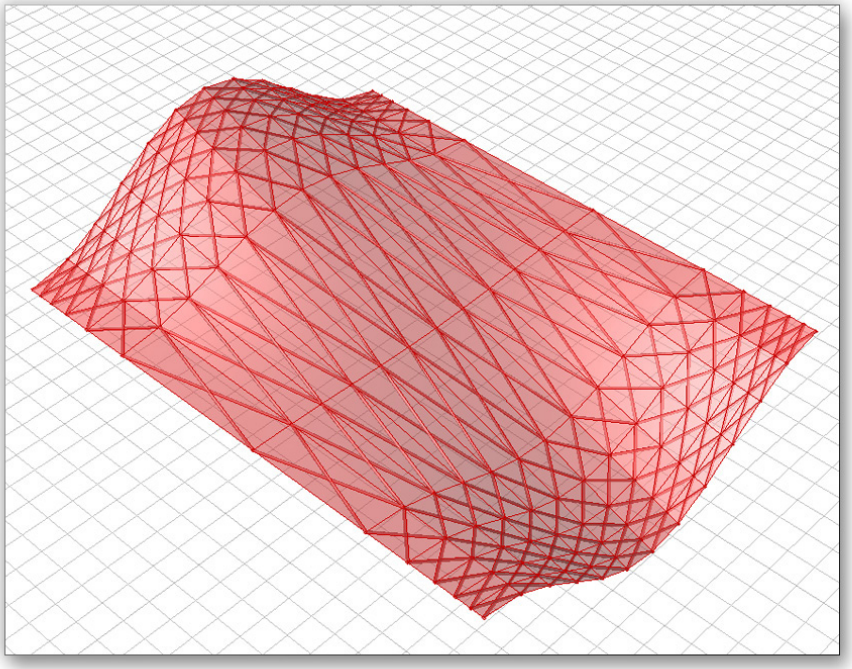


3. Surfaces

3.1 Surface Analytics:

Surface Diagrid (Non-uniform)





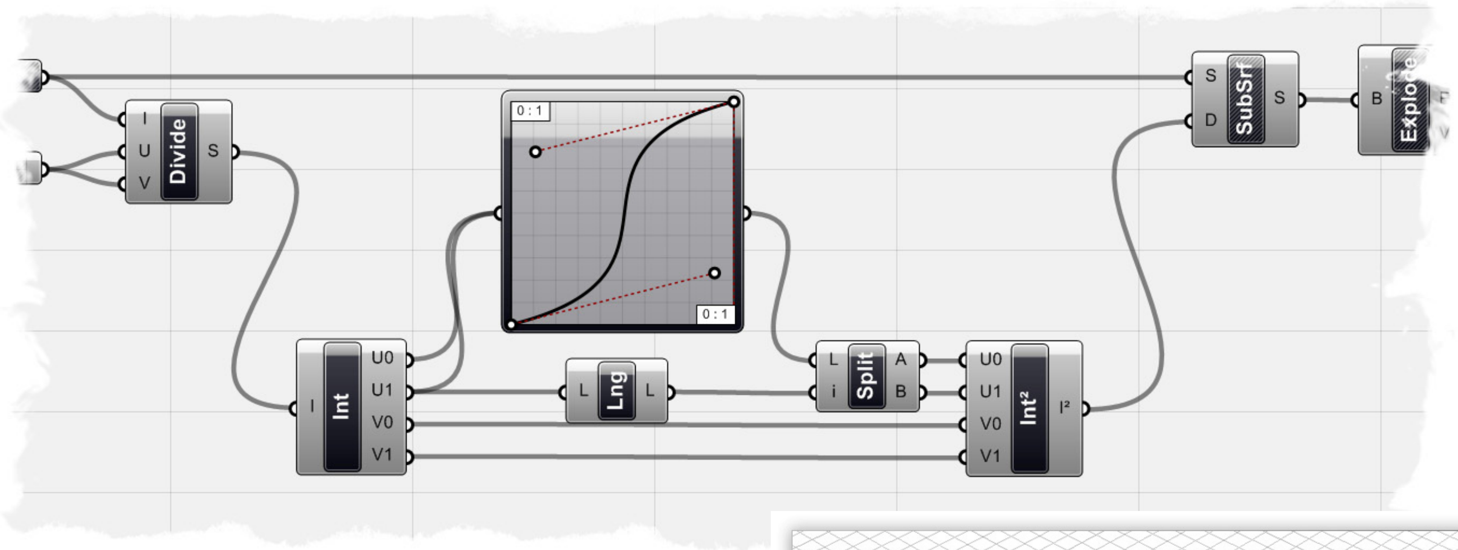
3. Surfaces

3.1 Surface Analytics:

Surface Diagrid (Non-uniform)

Non-Uniform Distortion:

Alter u & v coordinate grid by enlarging or reducing the length of the u,v spaces



U divisions = larger
V divisions = mantained