

An abstract, stylized representation of a DNA double helix. The structure is composed of numerous glowing blue and white helical strands that spiral outwards from a central point, creating a star-like or snowflake-like pattern. The strands are rendered with a sense of depth and movement, giving the impression of a complex, three-dimensional molecular structure. The background is a dark, deep blue, which makes the glowing strands stand out prominently.

# AdvArray Modifier

Documentation and Reference

Tim Catalano

## Table of Contents

Introduction .....	4
Supported Types .....	4
How array's work .....	4
Object Space or World Space? .....	4
Basic Parameters   Hotkey 1 .....	5
Count .....	5
Offset .....	5
Rotation .....	5
Scale .....	5
Transform Mode .....	5
Center   Hotkey 2 .....	6
Pre-Transform   Hotkey 3 .....	7
Gizmos .....	8
1: Gizmo .....	8
2: Center .....	8
3: Pre Transform .....	8
4: Path Spacing .....	8
5: Path Offset .....	8
Oscillate .....	9
Waveform .....	9
Frequency Range .....	9
Amplitude Alignment .....	10
Amplitude, Frequency, and Phase .....	11
Position, Rotation, and Scale .....	11
Random .....	12
Seed .....	12
Alignment .....	12
Offset, Rotation, and Scale .....	12
Path .....	13
Spacing and Offset   Hotkeys 4 & 5 .....	13
Distance is Percentage .....	13
Follow Path .....	13

Move To .....	14
Loop .....	14
Normalized.....	15
Cache Path Position .....	15
Banking and Smoothness.....	15
Material IDs.....	16
Iterate .....	16
Additive .....	16
Loop .....	16
Random.....	17
Minimum and Maximum .....	17
Advanced .....	18
Transform Order .....	18
Separate Transforms.....	19
Skip Last .....	19
Defer Processing .....	20
Process Poly as Mesh .....	21
Iteration UVs -> 1U.....	22
Sub-Selection Processing .....	22
Explode.....	22
Reset All .....	22
Additional Information.....	23
Reset Rollout Buttons .....	23
Align Tool .....	23
Conclusion.....	24

## Introduction

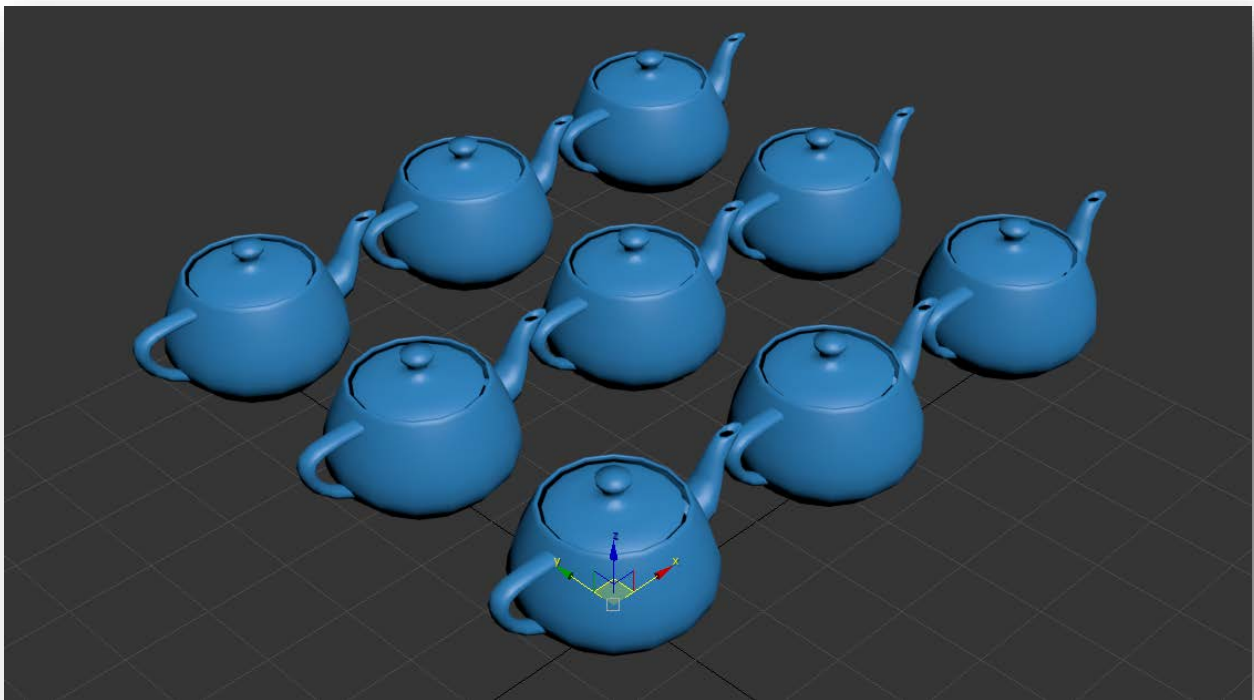
The AdvArray modifier is an advanced parametric array type modifier for 3ds Max. Being that 3ds Max does not ship with any sort of parametric array modifier, I've programmed this modifier to fill that void as well as add a number of advanced array features not found in 3ds Max by default, and even some that cannot be found in other third party modifiers.

## Supported Types

The AdvArray modifier supports Shapes, Polys, and Mesh's natively, but also supports any object type that can be converted to a mesh. The modifier can also be instanced across multiple object types. Very useful for many creative workflows.

## How array's work

An array takes a source object and creates duplicated copies of the geometry and moves, rotates, and scales the duplicated geometry successively each iteration based on the parameters given. This can be useful for creating rows or circles of objects for instance. Being that AdvArray is a parametric modifier, it can also be stacked, allowing for multiple dimensions of duplication.



## Object Space or World Space?

AdvArray comes in two variations, the OSM or Object Space Modifier version, and the WSM or World Space Modifier version. Which to choose? Generally most will want to use the OSM version so that you can stack additional modifiers above the array. The WSM version is primarily for enabling proper placement on a path when the modifier is being instanced to various nodes. This is due to the way that the 3ds Max modifier stack works. For additional information see the Path [Move To](#) Parameter.

## Basic Parameters | Hotkey 1

The basic parameters are the primary driving mechanism of the AdvArray modifier. These are the parameters that are most commonly used. These parameters are used in conjunction with the rest of the rollouts in the modifier.

### Count

The count parameter can be considered the single most important parameter in the array. It sets the number of duplicates that will be created during the array operation. Be careful though, this will multiply the number of polygons in your object by the Count.

### Offset

The offset drives the translation of each duplicate. The parameters are shown in XYZ order. For each iteration in the array, the geometry will be offset by this amount from the previous iteration in the array.

### Rotation

The rotation drives the rotation of each duplicate in the array. The spinners are displayed in XYZ order. For each iteration in the array, the duplicated geometry will be rotated the amount entered relative to the previous iteration in the array.

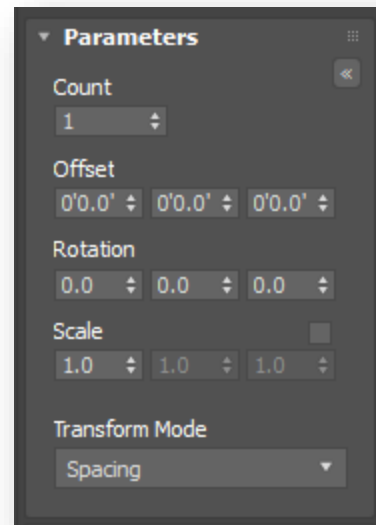
### Scale

As with the offset and rotation parameters, the scale drives the scale of each iteration in the array. The spinners are displayed in XYZ order. But by default the Y and the Z dimensions are disabled. This causes the scale to be applied uniformly across all the dimensions. If you enable the checkbox beside the scale, it will enable non-uniform scaling of each iteration. This can be useful for example when making each iteration taller or shorter while still keeping the length and width uniform.

### Transform Mode

The transform mode affects how each of the dimensions are translated by the modifier. The following list describes how each mode affects the array. The mode specified here also affects other parameters in other panels of the modifier.

- **Spacing:** In this mode, each iteration is transformed by the parameter values set.
- **Object Bounds:** The offset parameters are translated as percentages of the bounding box of the input geometry. Using a value of 1.0 in the offset will line the duplicates side to side.
  - *Affected parameters:* Offset, Center Offset, Pre-Transform Offset, Random Offset, and Oscillate Amplitude.
- **Final Transform:** Use this mode to set the transform of the final iteration in the array. The offset, rotation, and scale parameters are divided by the count and applied incrementally. This is so you can set the final transform from the source to the final iteration and just allow the number of duplicates to fill in the space. This is especially useful for 360 degree radial arrays. See also the [Skip Last](#) parameter on the [Advanced](#) rollout.
  - *Affected parameters:* Offset, Rotation, Scale, and Path Spacing.

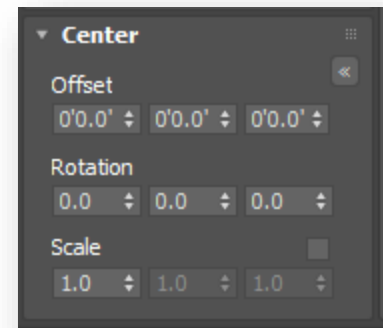




## Center | Hotkey 2

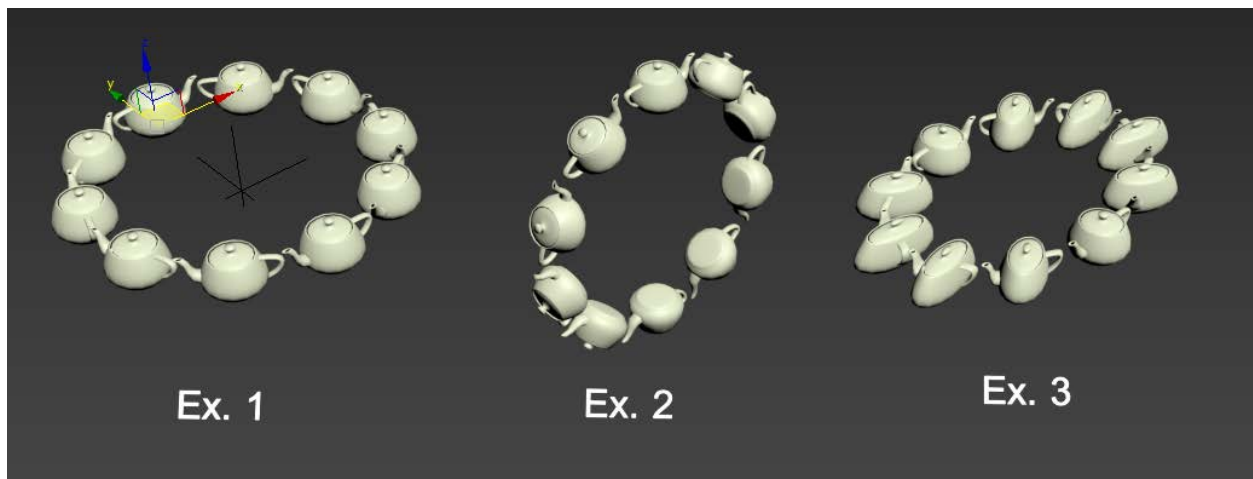
The center parameters control the center of rotation operations primarily, but can also affect the center of scale operations depending on the [Transform Order](#) set in the [Advanced](#) rollout. The center offset is particularly useful when you need your objects pivot to remain stationary but need to produce a radial array around an external center point (Ex. 1).

The center point also affects the main gizmos center of transform, which is useful when you need to snap your array spacing to other geometry.



You can rotate the center to change the axis of rotation, however, it's typically more straight-forward to just rotate on another axis. Once you rotate the center then all rotational axes are changed. For example (Ex. 2), if you rotate the center 90 degrees about the X axis, then rotations made in the Z axis will now result as if they were rotated about the Y axis.

Scaling the center is also something that can cause strange or unintended results. Scaling uniformly should have no effect on the array. However, scaling non-uniformly will change the array dramatically in some cases. Though, it depends on what other parameters you've changed. For example (Ex. 3), if you have a rotational array that rotates your items around the Z axis, with the center transformed in the Y direction. If you change the X scale it will squash or stretch your array along the X direction.



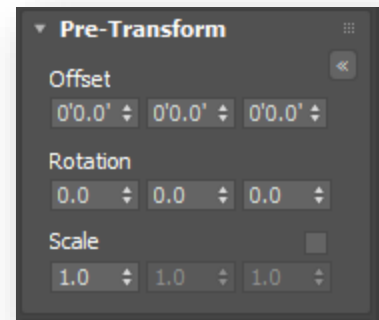
## Pre-Transform | Hotkey 3

The pre-transform allows you to transform your geometry parametrically before the remaining transformations happen. Works great when doing radial arrays where you want the center of your rotation to occur at the pivot for your node. Pre-transform can eliminate the need to use an xform modifier in the stack to get similar effects.

The pre-transform's offset can be used to change the radius of rotation when creating radial arrays using the base parameters rotation. Can also be used to adjust geometry without going down the stack.

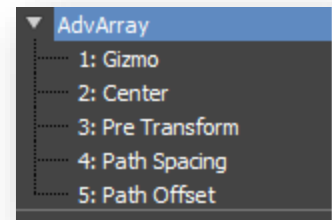
Depending on the Transform Order in the **Advanced** rollout, the pre-transform rotation can be used to add an overall rotation to the array. When doing radial arrays this can easily be used for adding uniform rotational animation. The rotation of the Pre-Transform is affected by the Center transform.

Use the scale to adjust the size of the geometry being arrayed to fit its situation. Use the checkbox to enable non-uniform scaling of the object.



## Gizmos

The gizmos can be used to graphically adjust the parameters in the modifier. They can also be used with other tools such as the **Align Tool** to align the parameters to various positions in your scene. I've included the number on your keyboard that enables each of the gizmos for quick visual reference.



Special note, depending on what **Transform Units** you specify on the **Base Parameters** panel and on the **Transform Order** you specify on the **Advanced** rollout, the gizmos may not always function exactly as you move the transforms graphically. This has a lot to do with the transform order that 3ds Max uses internally compared to what's chosen for the array. A future update may be able to correct the variations caused by the **Final Transform** though.

### 1: Gizmo

The gizmo drives the parameters on the base **Parameters** rollout. Often times it's much easier to use this gizmo than to use the parameters spinners as the distances you have to adjust are based on the viewport you're working in rather than on real world distances that can often times take a couple of rolls on the spinners.

### 2: Center

Use this to graphically align the center of your array. You can use this mode along with the **Align Tool** to align your center as you deem necessary. However, be sure to be in the local transform mode and also reset your center just before you align it.

### 3: Pre Transform

Use this to graphically align the transformation of the input object. Make use of the **Align Tool** here as well, again, make sure you reset the pre-transform parameters and switch to local transform mode first.

### 4: Path Spacing

*Yet to be implemented*

Can be used to graphically adjust the spacing along the path. Can only be used to move along the x axis. All other transformations such as rotation and scale will be ignored.

### 5: Path Offset

*Yet to be implemented*

Can be used to graphically adjust the offset along the path. Can be used to move along the x axis. But can also be rotated about the x axis to adjust the banking.



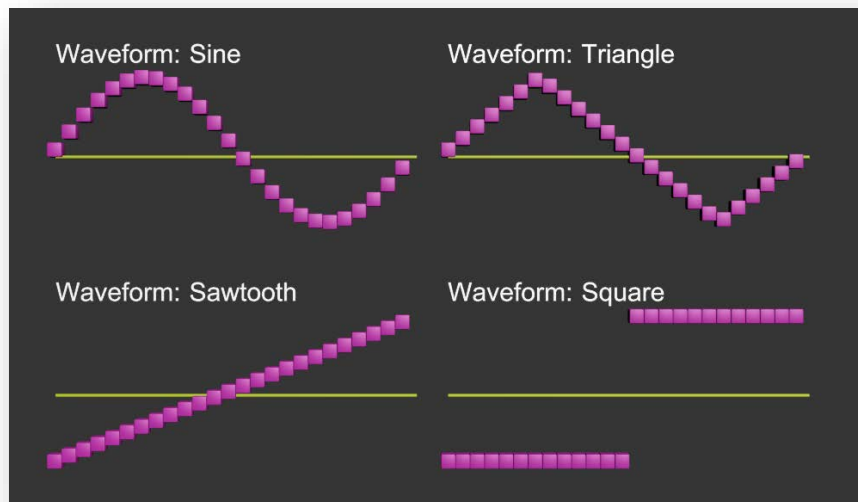
## Oscillate

The oscillate panel may seem like there's a lot going on, but it's very similar to the other rollouts. There's just a few more adjustments per dimension that need to be messed with to get the proper results.

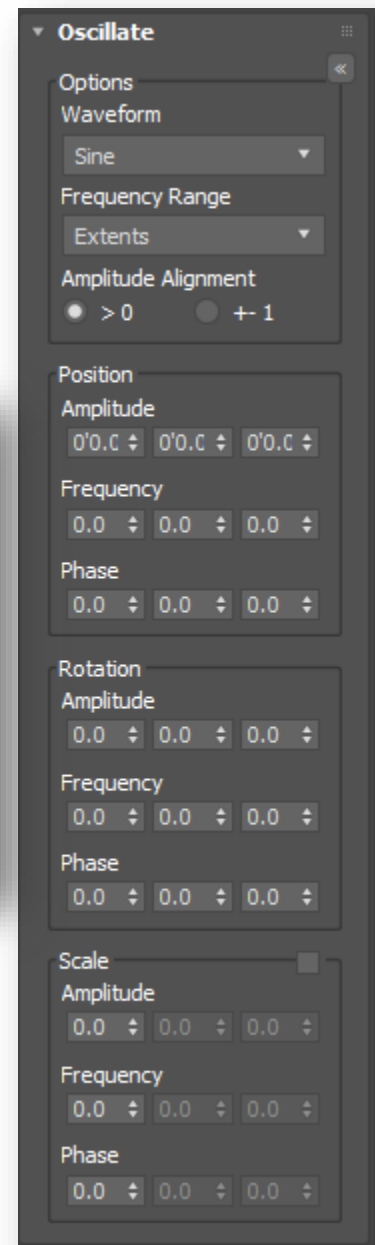
The oscillation functions as a sound wave might function. You have various waveforms that can be used to achieve different results.

### Waveform

The waveform defines the shape of the oscillation.



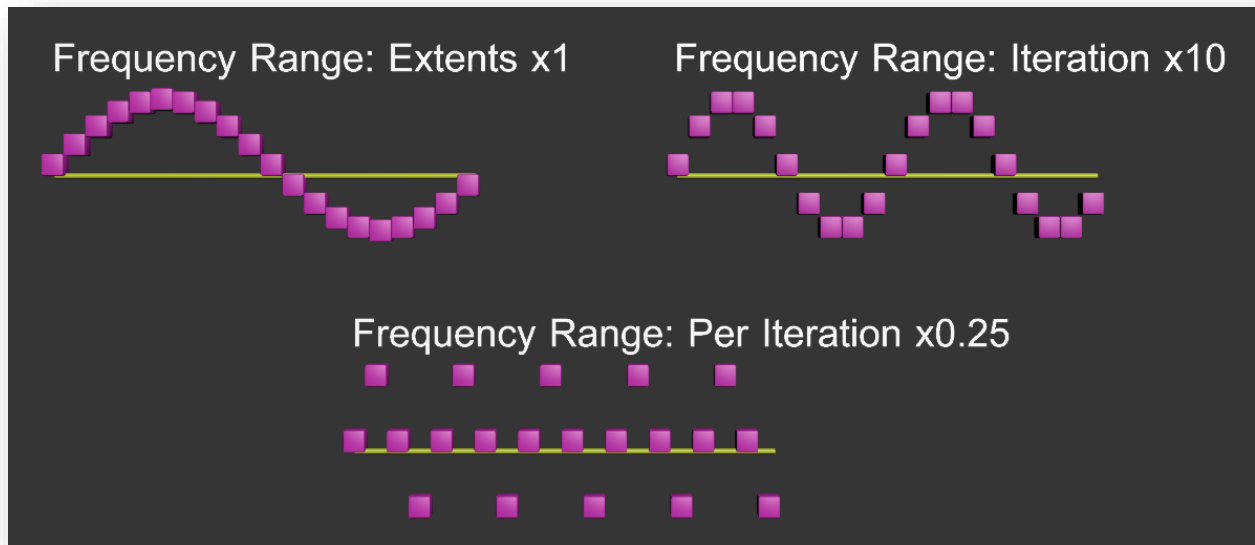
- **Sine:** The sine wave is a smooth waveform calculated using the same basic function used in trigonometry. Offset the phase by 0.25 to achieve a cosine type function.
- **Triangle:** Similar to the sine wave, the triangle wave oscillates back and forth, except the triangle wave is linear rather than smooth.
- **Sawtooth:** Similar to the triangle wave in being linear, the sawtooth waveform does not oscillate back and forth, but rather skips directly back to the start of the oscillation when it reaches the end.
- **Square:** The square waveform is binary in nature, either it's all on, or all off.



### Frequency Range

The frequency range controls how the cycle period is calculated. Similar to the Transform Mode on the Base Parameters rollout, it can be used to automate cycle distribution evenly across the array, or be set

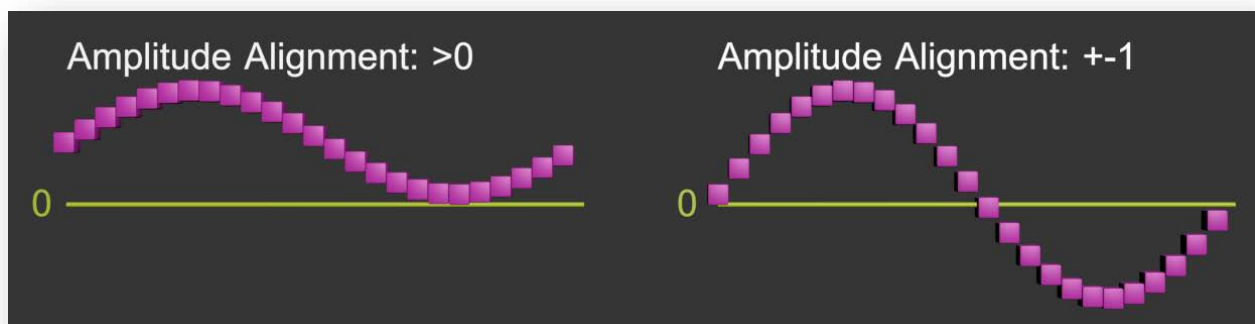
in a way to enable absolute control to you as the artist



- **Extents:** This mode will evenly distribute the oscillation cycles evenly across the entire array based on the frequency defined.
- **Iterations:** The oscillation cycle will occur over the period of iterations defined in the frequency. Just because the iteration count is a whole number though doesn't limit the frequency to whole numbers.
- **Per Iteration:** Kind of an inverted multiplier in comparison to the Iterations Frequency Range. This will calculate the cycles based on a number of cycles per iteration.

### Amplitude Alignment

The amplitude alignment defines the range that the amplitude affects. There's two options,  $>0$  and  $+/-1$ .  $>0$  will cause the amplitude range to extend away from zero, depending on whether you use a positive amplitude or a negative amplitude.  $+/-1$  will cause the amplitude range to straddle zero, with the range on half of the cycle to extend the direction defined by the sign of the amplitude, while the other half will extend in the opposite direction.



### Amplitude, Frequency, and Phase

If you're familiar with sound processing, these terms should be very familiar. Amplitude is how much of the effect to apply to a particular dimension. Frequency defines how often the cycle happens. And phase is used for offsetting the cycle. When animating, phase will definitely be your best friend in here.

### Position, Rotation, and Scale

As with the previous rollouts, there's control for each aspect of the iterations transform, and for each, each dimension is controllable.

*Ignore my notation inconsistency here, position here is the same as offset is elsewhere in the plugin. Either term is used interchangeably with the translation aspect of the transform.*

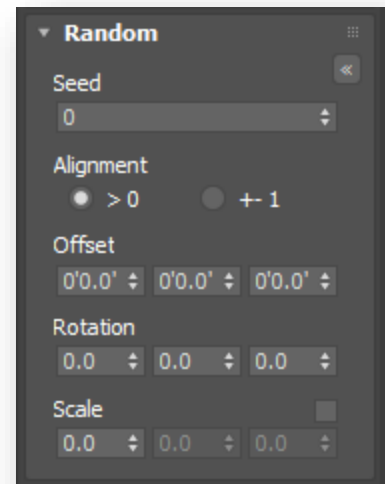
## Random

Nothing should ever look perfect, adding just a bit of random to each iteration of the array can really add a sense of life to your array.

### Seed

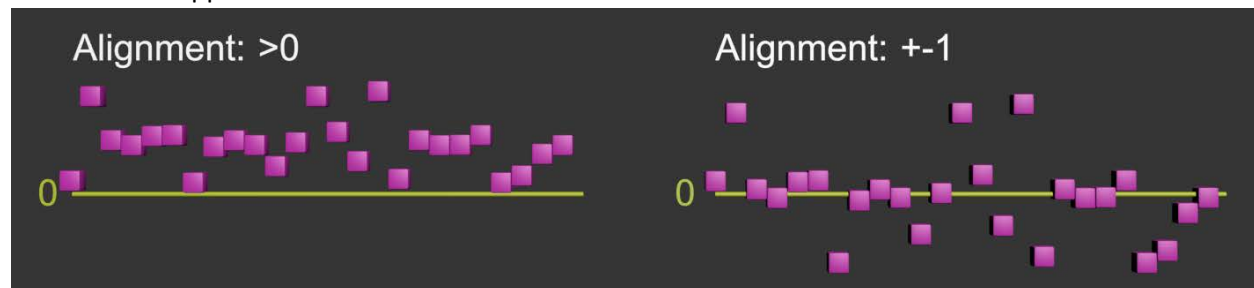
The seed gives you control of the randomization. If you're getting too much overlap between arrays, just change the seed of one of them. It's designed in such a way that using the same seed will give the same results every time.

Special note, this seed also controls the random values used in other areas of the modifier, such as the randomization of the material IDs.



### Alignment

As is the case with the oscillators' amplitude alignment, the random alignment defines the value range that a particular random dimension affects. There's two options, >0 and +-1. >0 will cause the random range to extend away from zero, depending on whether you use a positive value or a negative value. +-1 will cause the random range to straddle zero, with the range on half of the value range to extend the direction defined by the sign of the value, while the other half will extend in the opposite direction.



### Offset, Rotation, and Scale

The values for the dimensions here are the maximum that the random will extend. Being that it's random, you will find that most of the time, the transform does not extend all the way to this value.

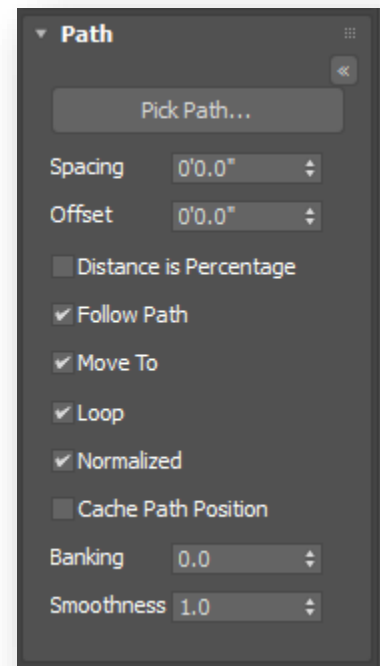


## Path

Using a path to define the placement of items of an array is the easiest and definitely the best way to get fully controllable non-linear arrays. A path can be used in conjunction with other transformation options in the array. Particularly well with the oscillator and the random transforms. But you will definitely see that it can be difficult to control placement when using the offset from the base parameters rollout. Though, there's some interesting effects that can be achieved for sure. So make sure to play around a bit to see if you can find your own interesting results. Rotation and scale from the base parameters should have little or no differing results than expected when combined with the path. Especially rotation can produce helix style wrapping along the path.

### Spacing and Offset | Hotkeys 4 & 5

The spacing defines the spacing between items along the path. Use the offset to define where along the path that the first item is placed along the path.



### Distance is Percentage

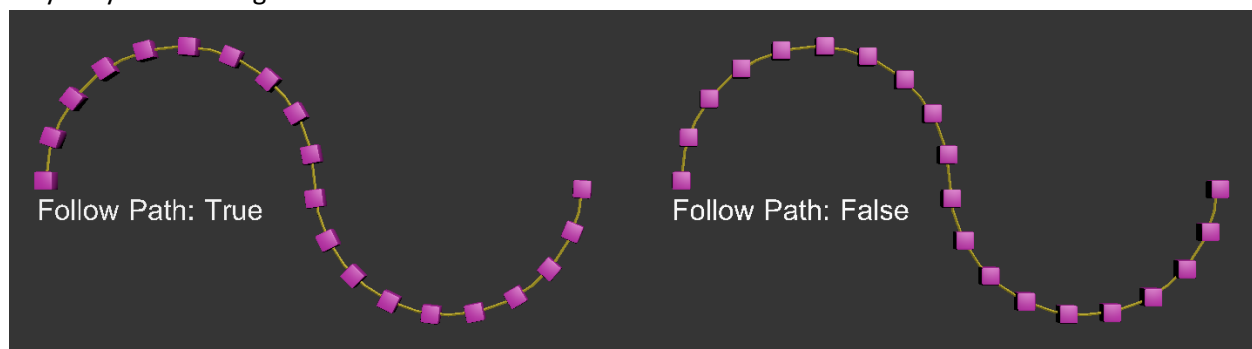
With this option enabled, the spacing and the offset are treated as a percentage of the entire length of all splines and segments within the shape object. When disabled, the values are treated as the unit length you have defined in your 3ds Max Units Setup.

When one desires to have the array fill the length of a path, try setting the Transform Mode on the base parameters to Final Transform, then set Distance is Percentage to true, set Spacing to 1.0, and disable Loop. This will enable having the array space itself evenly along the spline no matter how it changes.

*Disabling Loop due to rounding that often occurs that pushes the last item beyond 1.0 thereby causing it to loop back to the beginning of the path.*

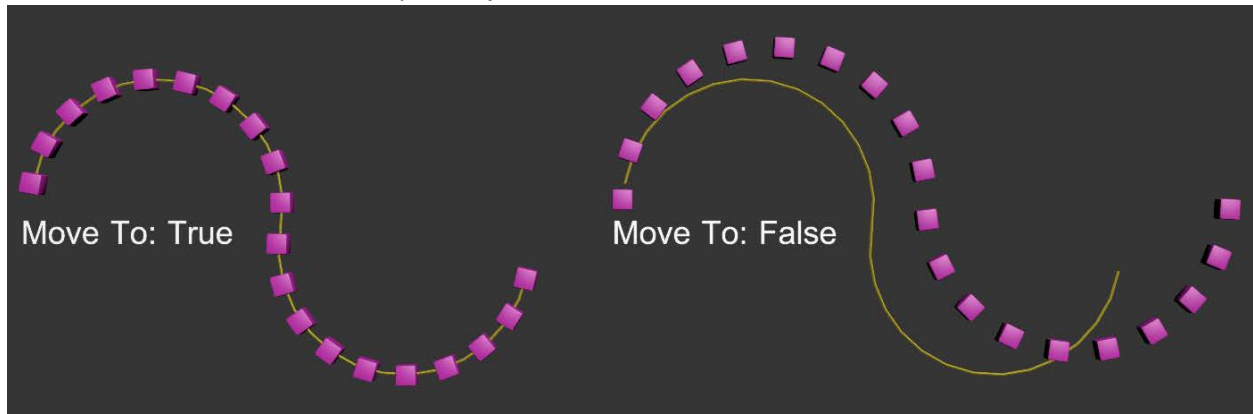
### Follow Path

The same as is the case with the built in path constraint, enabling the follow option will cause each of the iterations to rotate so that they follow the path vector. Disabling it will keep the objects oriented the way they were to begin with.



### Move To

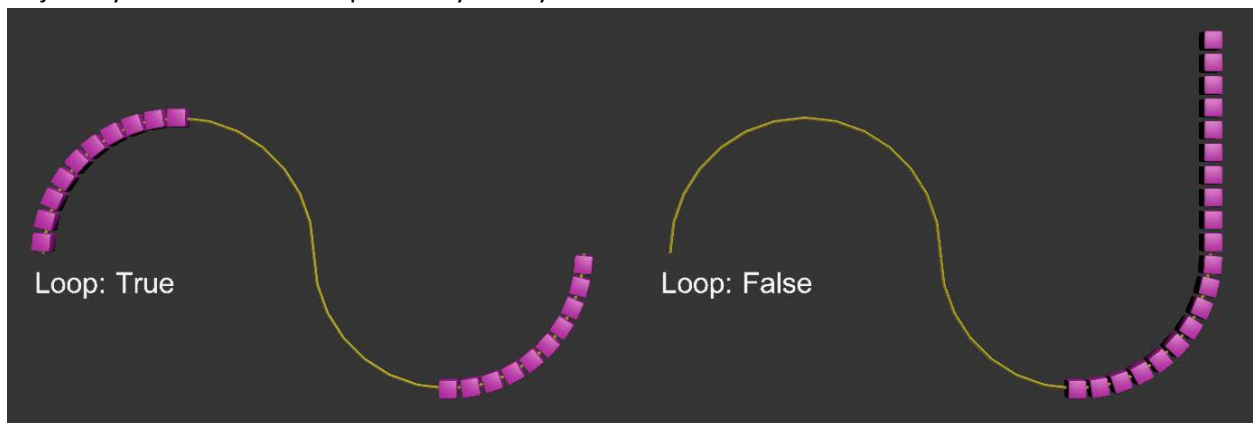
Causes the items to move to the path when enabled. When disabled, will array the items as if the path node was transformed to match your objects transform.



*When using the standard object space (OSM) variation of the modifier, and the modifier is being instanced to multiple objects, the Move To transform is only valid for one of the objects that instances the modifier. This has to do with the way that 3ds Max works. You can get around this by having all the objects that use the instanced modifier share the exact same transform in world space. Another option is to use the world space (WSM) variation of the modifier. This will consistently transform the items in the array to the path no matter where the nodes that instance the modifier exist in space. Careful though, the WSM version will essentially nullify transformation animations.*

### Loop

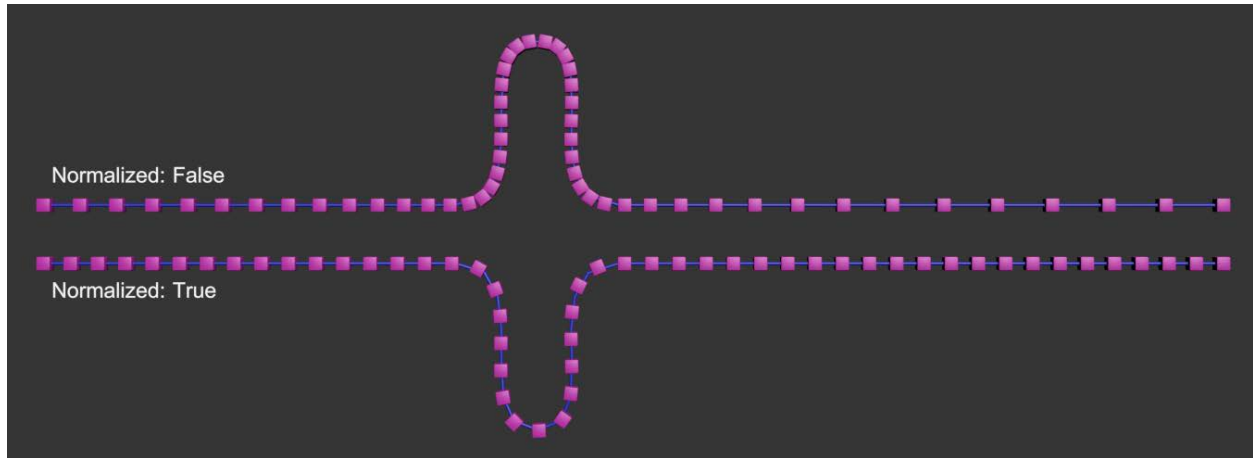
Enabling the loop option here will cause any items on the array that have spacing or an offset that would put them beyond the limits of the path back around to the opposite end of the path. So an offset of 101% would end up at 1% along the spline length, or -2% would end up at 98% along the spline. Disabling this option, and iterations that offset beyond the limits of the array would continue in the trajectory of the end of the spline they're beyond.





### Normalized

When enabled, the spacing along the path should be uniform no matter how the segments within the shape may vary. When disabled, the length of the segments within the shape affect the spacing, causing some iterations to be closer or farther apart depending on the position along the spline.



### Cache Path Position

When enabled, the current position of the path's node will be stored and used for the move to transformation calculation. Changes within the spline will still be reflected, however, if the paths node is transformed in space, those changes will not be reflected in the array. Can be used when aligning the spline in space where you want the array to occur, lock the position, then move the spline to the next place for additional arrays.

### Banking and Smoothness

*Planned for a future update.*

Banking will allow you to control how much of the curvature of the spline affects the roll of the placement of iterations of the array. Smoothness will define how much of the spline contributes to the banking of the array.

## Material IDs

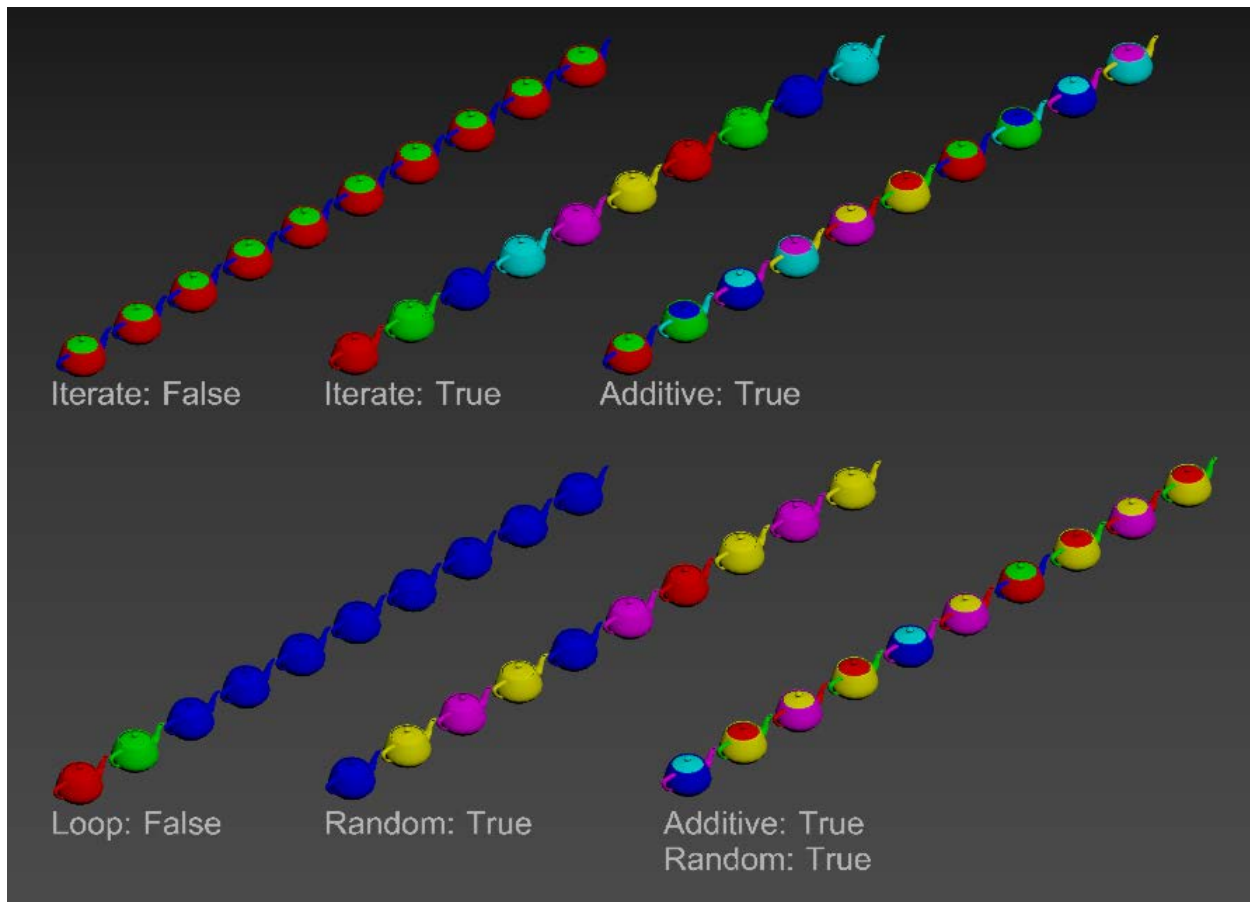
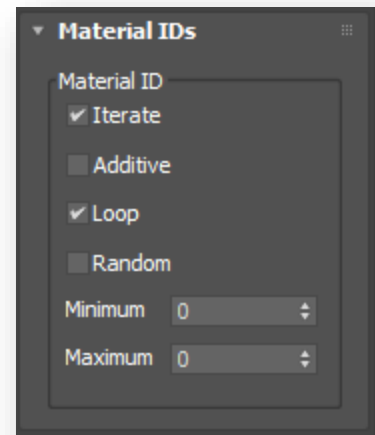
Why have uniform material application when you can apply a multi/sub-object material and drive variation with each iteration of the array!

### Iterate

Enable Iterate to enable iterating the material IDs. Having it enabled will set the material ID of all faces for each iteration based on the minimum and maximum values.

### Additive

Enabling Additive will, instead of setting the material ID to the same material ID for every face in each iteration, instead will add to the material ID for each face that was already defined previously on the mesh.



### Loop

Disable Loop to cause the iteration to stop at the maximum defined. Can be used for animating the maximum.

### Random

Randomly assigns the material ID per iteration. Refer to the [Random](#) rollout to adjust the seed for the random calculation used here.

### Minimum and Maximum

These spinners define the minimum and maximum material IDs to use. If they're both set to zero, then it will iterate between 0 and 65535, thusly enabling the Multi/Sub-Object material to handle the looping. If the minimum is set above the maximum, it will swap the values internally.

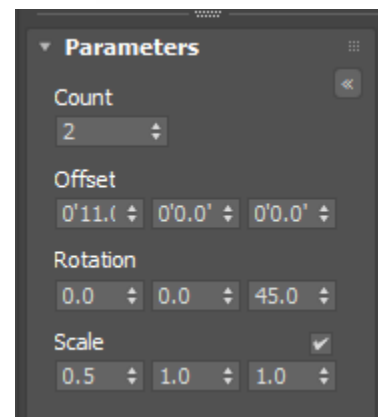
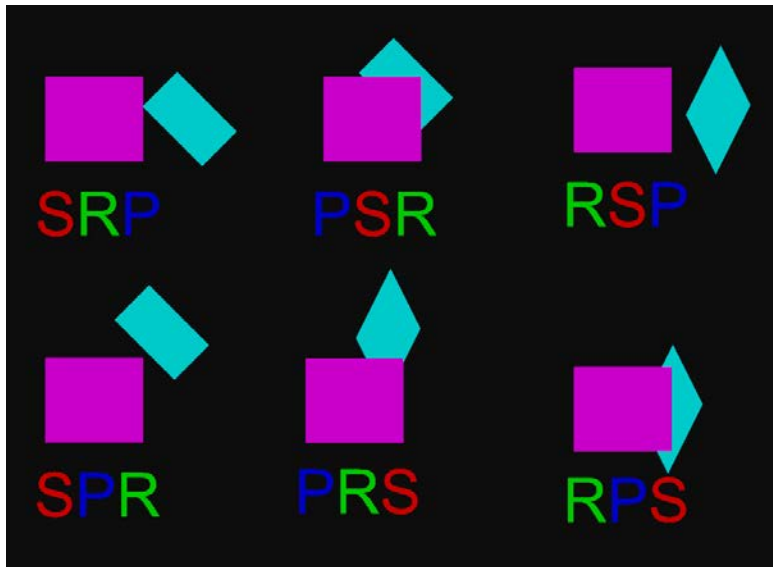
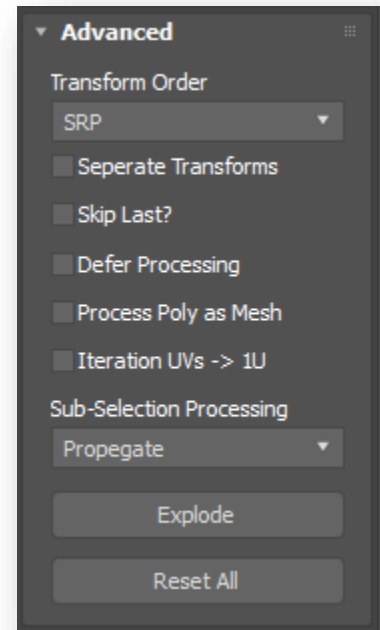
## Advanced

The advanced panel has many advanced options as well as a few that just didn't really fit elsewhere. Many options here have a kind of global effect, or an effect that isn't necessarily array transform related.

### Transform Order

- S = Scale
- R = Rotation
- P = Position

The transform order drives what order that each of the transforms is applied globally. To be consistent with 3ds Max, and for your gizmo transforms to work as expected, you'll want to use the default Scale, Rotation, Position (SRP) order. But feel free to mess with different orders to get different effects. There are definitely times when it's appropriate to utilize a different transform order to get things working just the way you need. In the list you will find the 9 unique combinations possible.



In this example, each of these arrays shares the same base parameters, they just have their transform orders

modified so you can see the affect. You can see that each of the transform operations that happen last are affected by the previous operations. Take the SRP order, first the box is scaled by 0.5 in the X direction, then rotated 45 degrees about the Z axis, then transformed 11" along the X direction. This keeps the box in the same basic box shape. Conversely, take a look at the PRS example, which seems like the logical order at first glance. You can see that first it's offset the box by 11" in the x direction, then rotated 45 degrees about the Z axis, which rotates about the base point as the center. All well and good in the right conditions. But lastly the scale is applied in the X direction. This skews the box due to the fact that the points are no longer aligned to that axis. It also pulls the duplicated box back into the initial object, because its original position offset has now also been scaled back towards the base point.

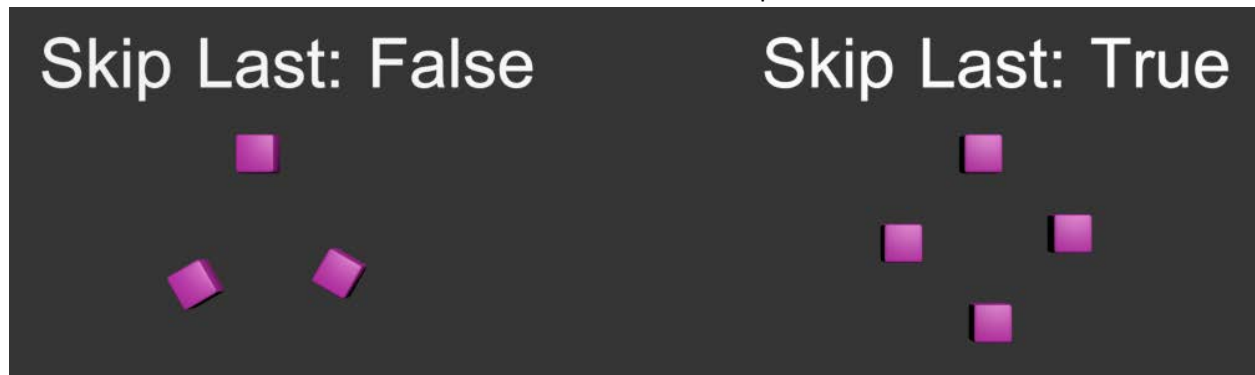
### Separate Transforms

Sometimes things just don't work when applying the adjustments in a single transform. Enabling the Separate Transforms option will process the Scale, Rotation, and Position in separate transforms and then combine the results at the end of the operation, producing often times very different results. Good for things like rotating your objects as they're offset without rotating the offset as well.



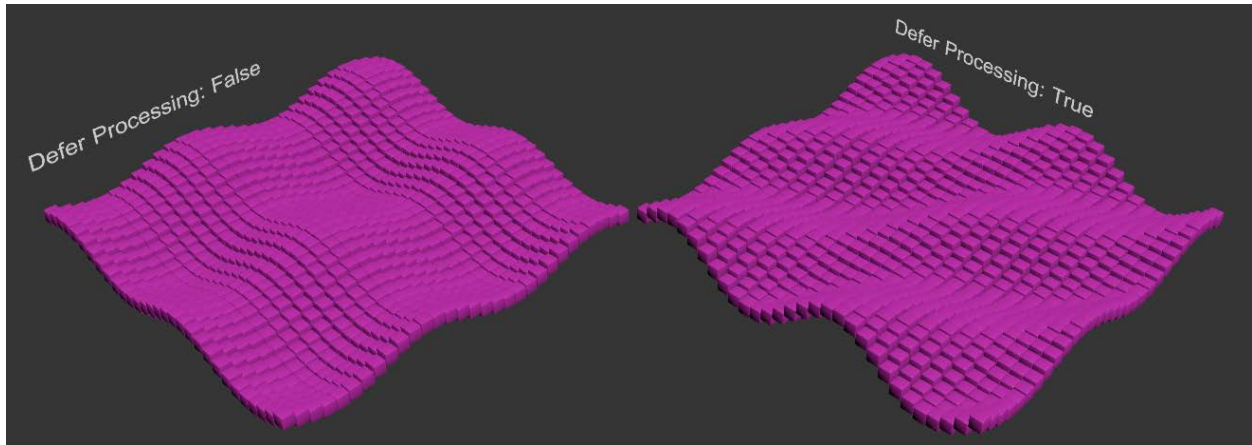
### Skip Last

This option will skip the last item in the array. Why would you want something like this? In particular, it's extremely useful when using extents transform range with a 360 degree radial array. Typically this would create an overlapping last item and the count would appear to be 1 less than what you have in the count. You could add one, which still leaves the overlap, or you could adjust the rotation to be something like:  $360 - (360 / \text{Count})$ . But that kind of defeats the parametric nature of the plugin. So instead just enable this, set the count to what you want the end result to be, and set your rotation to 360. This will internally add one to the count when your Transform Mode is Final Transform to accommodate this situation. Other transform modes will end up one short.



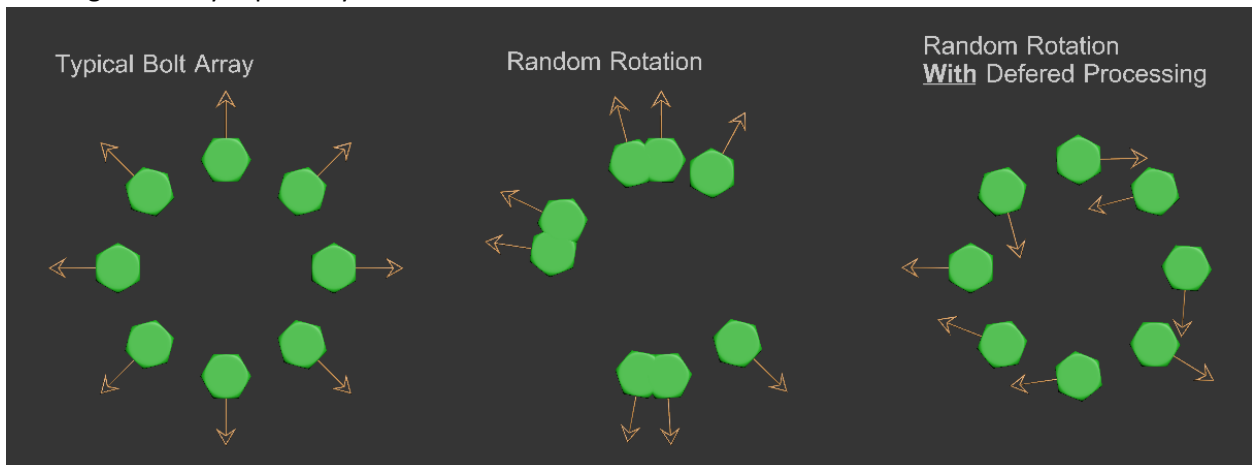
## Defer Processing

Defer processing will pass the geometry up the stack. If the next modifier up is an AdvArray, the modifier will get the settings from the deferring modifier and process the array. This enables certain effects not normally possible.



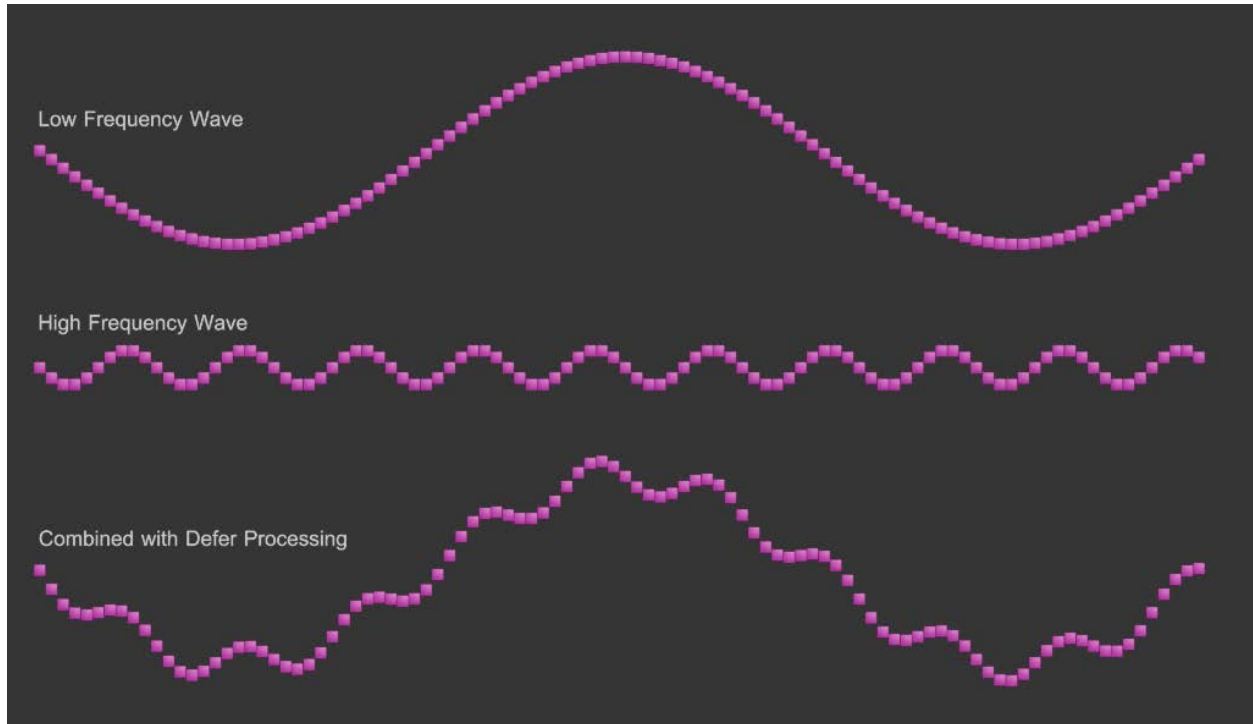
*But be aware, this can cause a significant performance drop. This is because when not deferring, it will make the duplicates once and pass that up the stack, then the next array will process that arrayed geometry. When defer processing is enabled however, for each iteration of the processing modifier, it will process the deferring modifiers settings. It becomes a multiplication of the processing which can be very heavy. You can do some more advanced things with this option that just isn't possible otherwise. Certain aspects of the array can affect the processing array even if the deferring array has a count of one.*

Here's an example. If you had a bolt or something that you wanted to array around a flange. Typically you would apply and array, set the Count you want, Rotation to 360, Transform Mode to Final Transform, then enable Skip Last. Use the Pre-Transform (or Center) to offset the object. Now you have a radial array. But everything is very uniformly rotated along that array, that wouldn't look very realistic. So what you would want to do is to add some random rotation along the rotation axis, but now all the bolts would be rotated incorrectly. This is where the defer processing comes in. Get rid of the random rotation, then add another AdvArray below the current one, leave the Count at 1, but add some random rotation. Initially it just rotates all of them, but enable the defer processing, and voila, all bolts are now rotating randomly separately!





Another example would be if you were doing an oscillating wave and wanted to have higher frequency waves moving along a lower frequency wave. What you would do is add one AdvArray to the stack, set the **amplitude**, then the **frequency** to a low frequency setting. Then add another AdvArray below this one in the stack. Leave the **Count** at 1. In the **Oscillator**, make sure to change the **Frequency Range** to **Iterations**, as this is required for this sort of combining. Set the amplitude, then the frequency to a high frequency setting, then enable **Defer Processing**, and now you have a multi frequency wave!

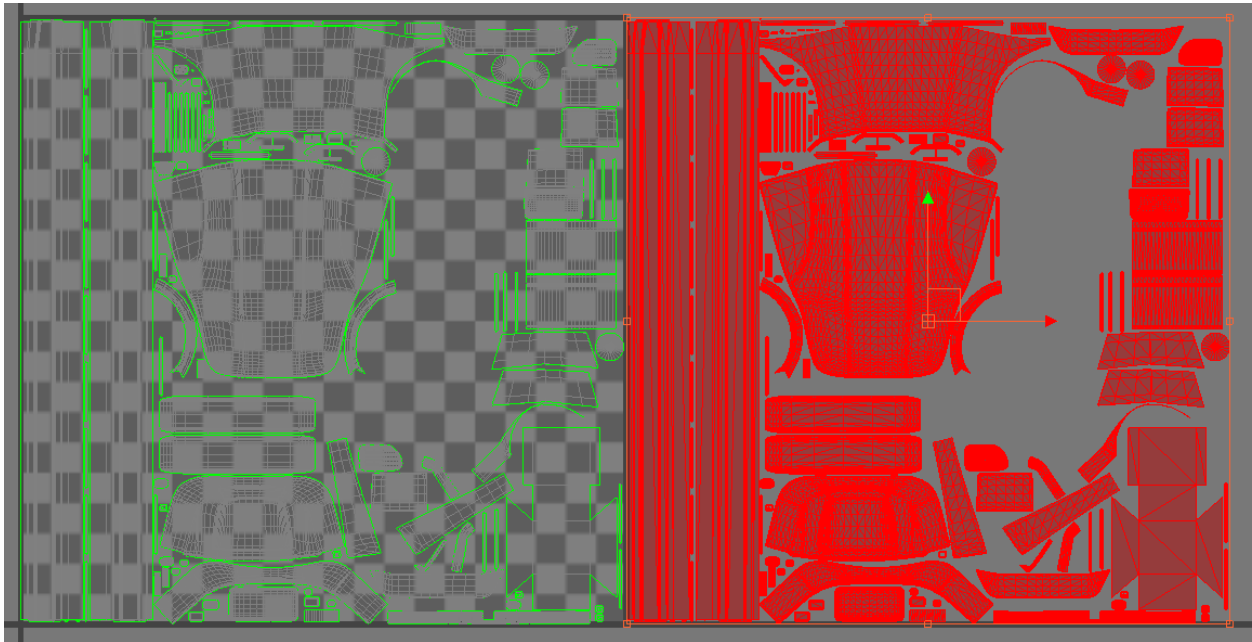


### Process Poly as Mesh

Enabling **Process Poly as Mesh** can really improve performance if the input geometry is a poly type object. The poly processing carries much more information than the mesh processing. Like additional uv maps, edge and vertex crease data, and other vertex data. Future versions will process this additional data with mesh processing, but currently poly objects are the only one that can. But this additional copying can really impact performance. So use this feature to speed up processing by a lot. And it's easy enough to enable while tweaking and then just disable it when you have things all set.

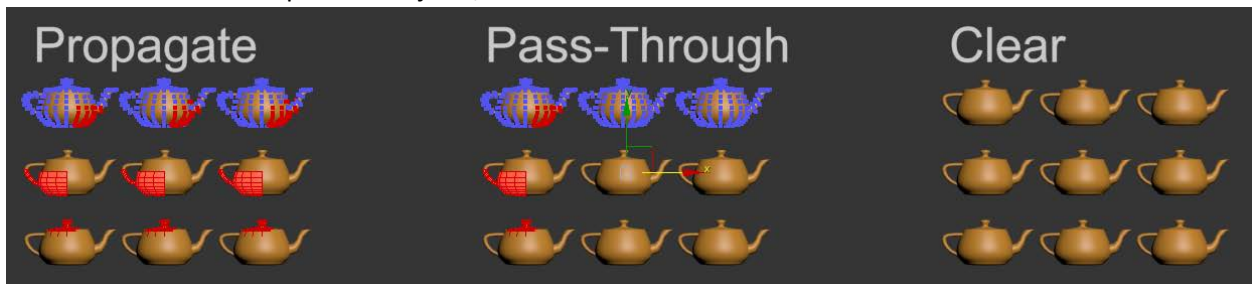
### Iteration UVs -> 1U

As I've seen some people like to re-use UV space for cloned objects, I've included this feature. It moves the UVs for iterations 2 and beyond from the 0.0 – 1.0 U space into the 1.0 – 2.0 U space.



### Sub-Selection Processing

This option enables you to define how the sub-selection is handled in the array. Use these options to set the sub-selection on duplicated objects, or to remove it.



- *Propagate*: The default option will take the selection that is active on the geometry entering the modifier and will copy it to the rest of the duplicated geometry in the array.
- *Pass-Through*: This option will pass only the selection that is applied to the geometry entering the modifier, and will leave duplicated geometry unselected.
- *Clear*: Use this option to just wipe out the selection all together.

### Explode

*Planned for a future update.*

Explode the array into individual nodes transformed by the array. Options on how duplicates are handle, such as instancing, etc...

### Reset All

Reset all parameters to default. Similar to how the Reset Rollout Buttons work on individual rollouts.

## Additional Information

Some additional information you may find useful when using the AdvArray modifier.

### Reset Rollout Buttons

You may have noticed that most rollouts have a little button in the upper right corner. This is to reset the values of that particular rollout to its default values. This can be useful when things get all out of whack and you need to get back to a rational setup for just one rollout. It's also nice to be able to reset certain rollouts when you duplicate an object and want all the settings the same except on that one particular rollout.

### Align Tool

Using the align tool is a great way to control your gizmos, however, the align tool has a known bug that needs to be properly handled. Be sure to be in local coordinates when using the align tool; and always reset the matching rollout before you use the align tool. Use the handy Reset button in the upper right of the rollout to help with that.

## Conclusion

If you have any issues, questions, or comments, please feel free to contact me at [tim\\_catalano@hotmail.com](mailto:tim_catalano@hotmail.com).

Also, thanks to my friend Henning Lande who has helped me to really get this modifier up to production ready. Please visit his Art Station page to see some of his incredible work!  
<https://landetls.artstation.com/>