



# Custom Analysis using Raster Cell Iterator

Nawajish Noman

2021 ESRI  
DEVELOPER SUMMIT

# Outline

- What is the Raster Cell Iterator (RCI)?
- Neighborhood Notation
  - Demo: Iterating Cells in a Raster
- RasterCellIterator Class
- Creating an Empty Raster
  - Demo: RasterCellIterator in Action
- Custom Analysis: Computing Contour Curvature
  - Demo: Custom Analysis
- Additional Resources

# What is Raster Cell Iterator (RCI)?

- RCI allows you to visit each cell in a raster
- You can refer to a cell using its location by index in a **Raster** object
- The index starts from 0
- You can read a cell value
- You can assign a cell value
  - Set **Raster.readOnly** property to False
  - Save the raster using **Raster.save()** method

|             |             |     |     |     |     |
|-------------|-------------|-----|-----|-----|-----|
|             | Column(c) → |     |     |     |     |
|             | 0,0         | 0,1 | 0,2 | 0,3 | 0,4 |
| Row(r)<br>↓ | 1,0         | 1,1 | 1,2 | 1,3 | 1,4 |
|             | 2,0         | 2,1 | 2,2 | 2,3 | 2,4 |
|             | 3,0         | 3,1 | 3,2 | 3,3 | 3,4 |
|             | 4,0         | 4,1 | 4,2 | 4,3 | 4,4 |
|             |             |     |     |     |     |

```
## Getting cell values
in_ras = Raster("myRaster")
for r,c in in_ras:
    cell_value = in_ras[r,c]
```

# Neighborhood Notation

- While visiting a cell, you can use relative indices to access the surrounding cells
- This capability is useful for custom focal analysis

|          |     |             |     |     |     |
|----------|-----|-------------|-----|-----|-----|
|          |     | Column(c) → |     |     |     |
| Row(r) ↓ | 0,0 | 0,1         | 0,2 | 0,3 | 0,4 |
|          | 1,0 | 1,1         | 1,2 | 1,3 | 1,4 |
|          | 2,0 | 2,1         | 2,2 | 2,3 | 2,4 |
|          | 3,0 | 3,1         | 3,2 | 3,3 | 3,4 |
|          | 4,0 | 4,1         | 4,2 | 4,3 | 4,4 |
|          |     |             |     |     |     |

|         |         |       |         |         |
|---------|---------|-------|---------|---------|
| r-2,c-2 | r-2,c-1 | r-2,c | r-2,c+1 | r-2,c+2 |
| r-1,c-2 | r-1,c-1 | r-1,c | r-1,c+1 | r-1,c+2 |
| r,c-2   | r,c-1   | r,c   | r,c+1   | r,c+2   |
| r+1,c-2 | r+1,c-1 | r+1,c | r+1,c+1 | r+1,c+2 |
| r+2,c-2 | r+2,c-1 | r+2,c | r+2,c+1 | r+2,c+2 |

```
for r,c in in_ras:  
    UL_cell_value = in_ras[r-1,c-1]  
    UC_cell_value = in_ras[r-1, c]  
    UR_cell_value = in_ras[r-1,c+1]  
  
    CL_cell_value = in_ras[r,c-1]  
    CC_cell_value = in_ras[r, c]  
    CR_cell_value = in_ras[r,c+1]  
  
    LL_cell_value = in_ras[r+1,c-1]  
    LC_cell_value = in_ras[r+1, c]  
    LR_cell_value = in_ras[r+1,c+1]
```

|      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|
| 1010 | 1020 | 1030 | 1040 | 1050 | 1060 | 1070 | 1080 | 1090 |
| 2010 | 2020 | 2030 | 2040 | 2050 | 2060 | 2070 | 2080 | 2090 |
| 3010 | 3020 | 3030 | 3040 | 3050 | 3060 | 3070 | 3080 | 3090 |
| 4010 | 4020 | 4030 | 4040 | 4050 | 4060 | 4070 | 4080 | 4090 |
| 5010 | 5020 | 5030 | 5040 | 5050 | 5060 | 5070 | 5080 | 5090 |
| 6010 | 6020 | 6030 | 6040 | 6050 | 6060 | 6070 | 6080 | 6090 |
| 7010 | 7020 | 7030 | 7040 | 7050 | 7060 | 7070 | 7080 | 7090 |
| 8010 | 8020 | 8030 | 8040 | 8050 | 8060 | 8070 | 8080 | 8090 |
| 9010 | 9020 | 9030 | 9040 | 9050 | 9060 | 9070 | 9080 | 9090 |

# Iterating Cells in a Raster

Nawajish Noman



# RasterCellIterator Class

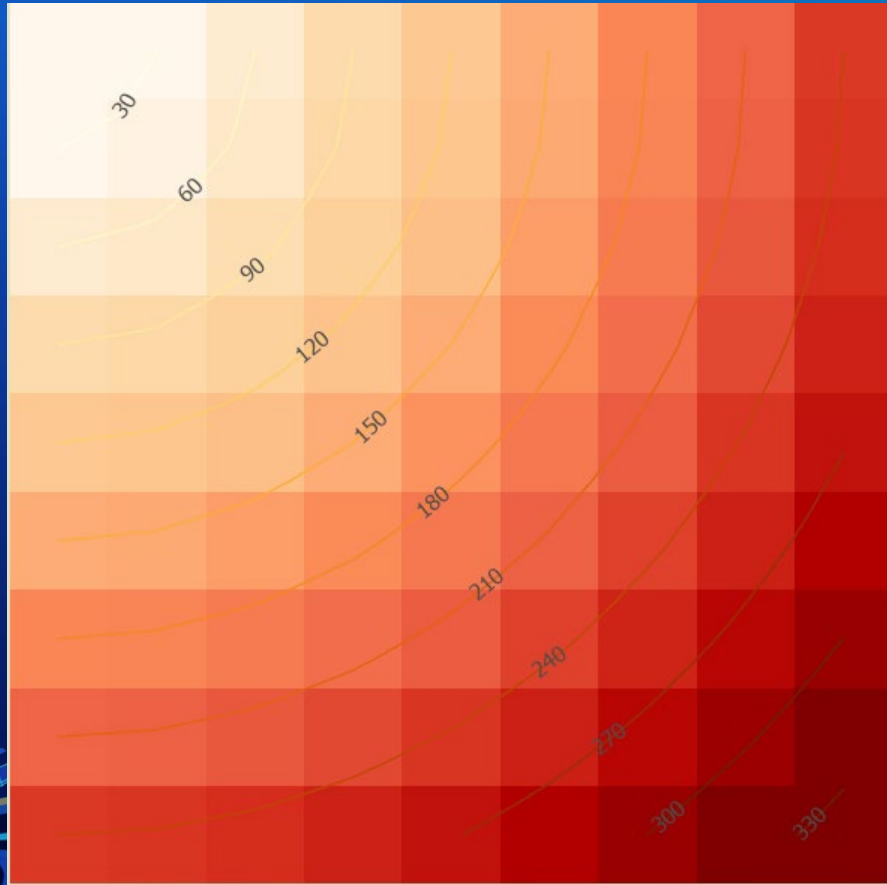
- Defines an iterator object to iterate through the cells of one or more rasters.
- It is an explicit iterator, useful for iterating through multiple rasters while performing analysis using one or more inputs and creating output(s).
- The `rasters` key indicates the list of input and output rasters.
- The `padding` key lets you define a padding factor to improve the performance for accessing neighboring cells
- The `skipNoData` key lets the iterator skip the NoData cells automatically. Useful for processing raster representing discrete features like roads, rivers, etc.

```
RasterCellIterator({'rasters': [in_rasterobj1, in_rasterobj2, ...],  
                   'padding': padding_factor,  
                   'skipNoData': [in_rasterobj1, in_rasterobj2, ...]})
```

# Creating a New Raster Filled with NoData

- Create **Raster** object using **RasterInfo** object as an argument.
- You can create an empty **RasterInfo** object and set necessary properties.
- You can also get a **RasterInfo** object from an existing rasters using **Raster.getRasterInfo()** method to use it as a template.
  - Update the new raster specific properties such as pixel type.
- Save the new raster using **Raster.save()** method

```
in_ras = Raster("myRaster")
ras_info = in_ras.getRasterInfo()
ras_info.setPixelType("F32")
out_new_ras = Raster(ras_info)
```



# RasterCellIterator in Action

Nawajish Noman



# Contour Curvature: Custom Analysis using Finite Difference Approximation

$$K_C = -\frac{f_{xx}f_y^2 - 2f_{xy}f_xf_y + f_{yy}f_x^2}{(f_x^2 + f_y^2)^{3/2}}$$

$$f_x = (z6 - z4) / 2 * cs$$

$$f_y = (z8 - z2) / 2 * cs$$

$$f_{xx} = (z6 + z4 - 2 * z5) / cs^2$$

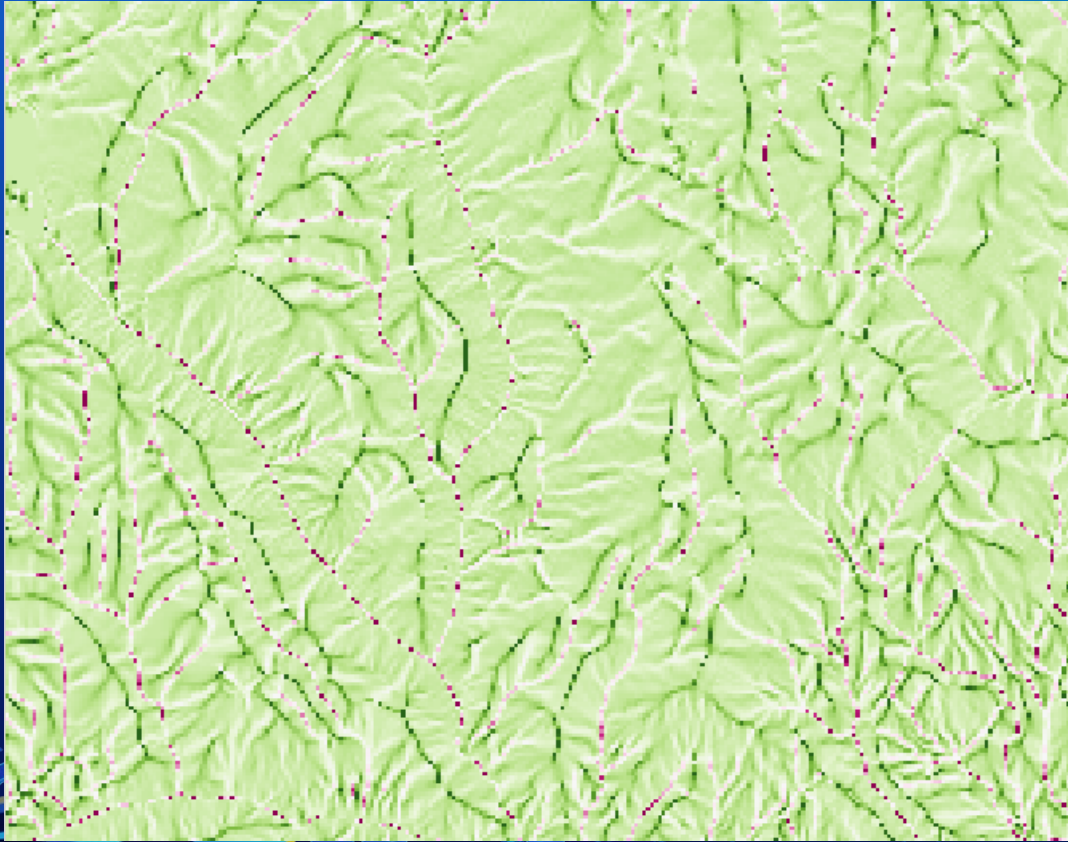
$$f_{yy} = (z8 + z2 - 2 * z5) / cs^2$$

$$f_{xy} = (z9 - z7 - z3 + z1) / (4 * cs^2)$$

|                   |                 |                   |
|-------------------|-----------------|-------------------|
| $z1$<br>$r-1,c-1$ | $z2$<br>$r-1,c$ | $z3$<br>$r-1,c+1$ |
| $z4$<br>$r,c-1$   | $z5$<br>$r,c$   | $z6$<br>$r,c+1$   |
| $z7$<br>$r+1,c-1$ | $z8$<br>$r+1,c$ | $z9$<br>$r+1,c+1$ |

$z1, z2, \dots$  = Cell values

$cs$  = Cell size



# Custom Analysis

Nawajish Noman

# Additional Resources

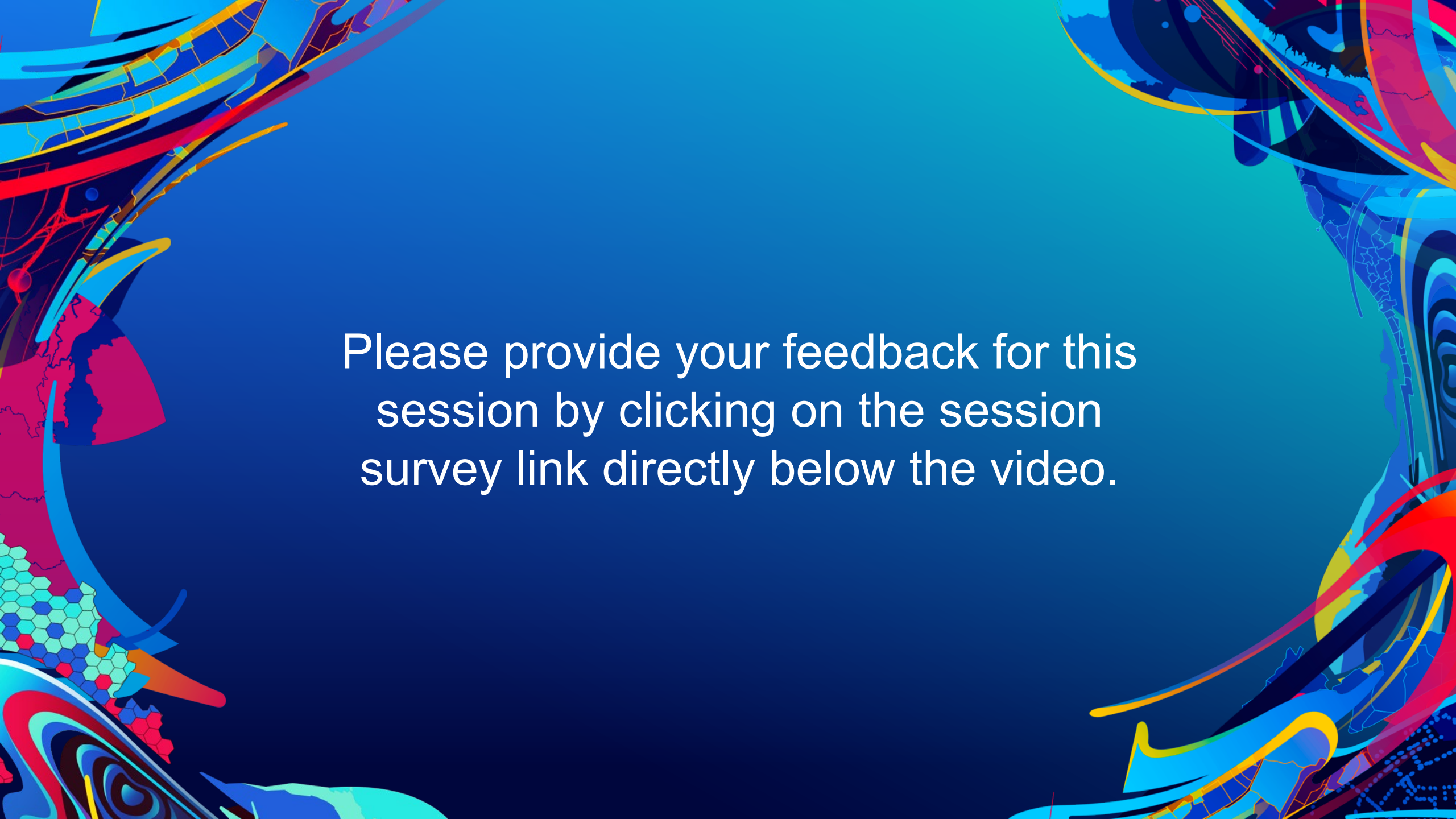
This screenshot shows the ArcGIS Pro help page for the Raster Cell Iterator. The page has a dark header with the Esri logo and navigation links. Below the header, there's a search bar and a breadcrumb trail: Home > Get Started > Help > Tool Reference > Python > SDK. The main content area is titled "A quick tour of using Raster Cell Iterator" and includes a sub-header "ArcGIS Pro 2.7 | Other versions". The text explains that the Raster Cell Iterator (RCI) is used for custom raster analysis. A sidebar on the left lists various topics, and a "In this topic" section on the right provides links to "Basic operations using Raster Cell Iterator" and "Advanced operations using Raster Cell Iterator".

This screenshot shows the ArcGIS Pro help page for the RasterCellIterator class. The page has a dark header with the Esri logo and navigation links. Below the header, there's a search bar and a breadcrumb trail: Python > Spatial Analyst module > Classes. The main content area is titled "RasterCellIterator" and includes a sub-header "ArcGIS Pro 2.7 | Other versions". The text defines the RasterCellIterator as an object to iterate through the cells of one or more rasters. A sidebar on the left lists various topics, and a "In this topic" section on the right provides links to "Summary", "Discussion", "Syntax", and "Code sample".

This screenshot shows the ArcGIS Blog post titled "Introducing the Raster Cell Iterator". The post is dated February 06, 2020, and is categorized under "Analytics". The author is Neeraj Rajasekar. The text explains that the Raster Cell Iterator (RCI) is a new ArcPy class added in ArcGIS Pro 2.5, which allows users to interact with raster datasets at an individual cell level. The post includes a sidebar with social media links and a "Topics" dropdown menu.

This screenshot shows the ArcGIS Blog post titled "Unleash the power of RasterCellIterator to perform custom raster analysis". The post is dated February 06, 2020, and is categorized under "Analytics". The authors are Hao Hu and Yu Wang. The text explains that the Raster Cell Iterator (RCI) is a new ArcPy class added in ArcGIS Pro 2.5, which allows users to interact with raster datasets at an individual cell level. The post includes a sidebar with social media links and a "Topics" dropdown menu.



The background is a vibrant, abstract composition. It features a central area of solid blue and teal. This is framed by dynamic, swirling patterns in shades of red, yellow, and blue. On the left side, there are intricate geometric patterns, including a hexagonal grid in light blue and green, and a series of red and blue dots. The overall effect is one of high energy and modern design.

Please provide your feedback for this session by clicking on the session survey link directly below the video.



esri®

THE  
SCIENCE  
OF  
WHERE®