Scalar Field Visualization I
What is a Scalar Field?

- The approximation of certain scalar function in space $f(x,y,z)$.
What is a Scalar Field?

• The approximation of certain scalar function in space \( f(x,y,z) \).

• Most of time, they come in as some scalar values defined on some sample points.
What is a Scalar Field?

• The approximation of certain scalar function in space $f(x,y,z)$.

• Most of time, they come in as some scalar values defined on some sample points.

• Visualization primitives:
  – Geometry:
    • iso-contours (2D), iso-surfaces (3D),
  – Attributes:
    • colors, transparency (3D), 3D textures.
Generate 2D color plots
Do we need to construct/extract additional geometry?

How to generate color plots?
Steps of generating 2D color plots

1. Design color transfer function
2. Color interpolation
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

Scalar values \([s_{\text{min}}, s_{\text{max}}]\) \(\rightarrow\) \([0, 1]\) \(\rightarrow\) Colors
To create a color plot, we need to define a proper Transfer Function to set Color as a function of Scalar Value. The following shows a simple transfer function.

Scalar values -> [0,1]

\[ t = \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}} \] normalization
To create a color plot, we need to define a proper *Transfer Function* to set *Color* as a function of Scalar Value. The following shows a simple transfer function.

![Graph showing color mapping]

What colors should they map to?
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

![Color plot diagram](image)

[0,1] -> Colors

- **smallest**
- **largest**

What colors should they map to?

- 0 -> blue (hue=240)
- 1 -> red (hue=0)

Ex. Rainbow color
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

Ex. Rainbow color

<table>
<thead>
<tr>
<th>smallest</th>
<th>largest</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

What colors should they map to?

\[
\begin{align*}
0 & \rightarrow \text{blue (hue=240)} \\
1 & \rightarrow \text{red (hue=0)} \\
\end{align*}
\]

*Ex. Rainbow color*
To create a color plot, we need to define a proper **Transfer Function** to set **Color** as a function of Scalar Value. The following shows a simple transfer function.

Scalar values ->[0,1] -> Colors

\[ \text{Hue} = 240. - 240. \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}} \]

Verify low and high!
How to achieve the following color scale?

Scalar values -> [0,1]

\[ t = \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}} \]

normalization
How to achieve the following color scale?

Scalar values -> [0,1]

0 -> white
1 -> full saturated green (hue=120)
How to achieve the following color scale?

Scalar values -> [0, 1]

0 -> white
1 -> full saturated green (hue=120)

Saturation=0

What is between 0 and 1?

Saturation=1
How to achieve the following color scale?

Scalar values -> [0, 1]

0 -> white

Saturation = 0

1 -> full saturated green (hue=120)

\[ Saturation = t, \text{ where } t \text{ is in } [0, 1] \]

Saturation = 1
Use the Right **Transfer Function** Color Scale to Represent a Range of Scalar Values
Hue = 240. − 240. \frac{S - S_{\text{min}}}{S_{\text{max}} - S_{\text{min}}}

Rainbow Scale
Gray Scale

How to generate gray scale color scheme?
Gray Scale

How to generate gray scale color scheme?

Set r=g=b= the same value
Intensity and Saturation Color Scales
Two-Color Interpolation

How to achieve the above color scheme?
Two-Color Interpolation

How to achieve the above color scheme?

You may try a simple linear interpolation of the three color-channels.
Heated Object Color Scale

Implementation: add one color component at a time

R+G+B
R+G+B

[0, 1/3, 2/3, 1]
For $s$ in $[0, 1/3]$, fixed $G=B=0$, map $s$ to $R$ from $[0, 1/3]$ to $[0, 1]$, that is, $R=3.0*s$
For $s$ in $[0, 1/3]$, fixed $G=B=0$, map $s$ to $R$ from $[0, 1/3]$ to $[0, 1]$, that is, $R=3.0s$

For $s$ in $(1/3, 2/3]$, fixed $R=1$, $B=0$, map $s$ to $G$ from $(1/3, 2/3]$ to $[0, 1]$, that is, $G=$?

For $s$ in $(2/3, 1]$, fixed $R=G=1$, map $s$ to $B$ from $(2/3, 1]$ to $[0, 1]$, that is, $B=$?
Add-One-Component at a time -- an extension from the heated object color scale
Blue-White-Red Color Scale

Diverging color scheme!
Blue-White-Red Color Scale

How to achieve it?
Blue-White-Red Color Scale

How to achieve it?

Scalar value -> [0, ½, 1]
Blue-White-Red Color Scale

How to achieve it?

Scalar value -> [0, ½, 1]

For [0, ½], saturation reduces from 1 to 0, hue=240
For (½, 1], saturation increases from 0 to 1, hue=0
Use VTK color look-up table

```python
lut = vtk.vtkLookupTable()  # Initialize the vtk lookup table
nc = 256  # the size of the table - increase this number to obtain more precise color map
lut.SetNumberOfTableValues(nc)
lut.Build()
```

Generate the individual entries of the look up table (rainbow color is shown)

```python
sMin = 0.
sMax = 1.

hsv = [0.0, 1.0, 1.0]

for i in range(0, nc):
    s = float(i) / nc  # uniformly interpolate the scalar values
    hsv[0] = 240. - 240. * (s-sMin)/(sMax-sMin)  # rainbow color calculation
    rgba = hsvRgb(hsv)  # convert hsv to rgb
    rgba.append(1.0)  # set alpha (or opacity) channel
    lut.SetTableValue(i, *rgba)
```

<table>
<thead>
<tr>
<th>sMin</th>
<th>entry 0</th>
<th></th>
<th></th>
<th>color 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color nc-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color nc-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color nc-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>color nc-1</td>
</tr>
</tbody>
</table>

sMin ➔ entry 0
sMax ➔ entry nc-1
Generate 2D color plots

1. Color transfer function
2. Color interpolation
2D Interpolated Color Plots

- How can we turn the discrete samples into a continuous color plot?
- Here’s the input: we have a 2D grid of data points. At each node, we have an X, Y, Z, and a scalar value S. We know $S_{\text{min}}$, $S_{\text{max}}$, and a Transfer Function.

Even though this is a 2D technique, we keep around the X, Y, and Z coordinates so that the grid doesn’t have to lie in any particular plane.
2D Interpolated Color Plots

- Let us look at one **square (or quad)** of the mesh at a time

For each scalar value at a vertex

```c
float arrays hsv[3], rgb[3]
hsv[0] = 240. -240. \frac{S-S_{\text{min}}}{S_{\text{max}}-S_{\text{min}}};
HsvRgb (hsv, rgb);
```

Convert hsv color to rgb color as monitor uses additive color model!
Use VTK color look-up table with a mapper

Get scalar value range

```python
min_scalar, max_scalar = vtk_geometry.GetOutput().GetPointData().GetArray("s").GetRange()
```

Create a vtkPolyDataMapper object and connect it to the geometry of the data domain (e.g., the grid). Assume a look up table, `lut`, has been built.

```python
vtk_poly_mapper = vtk.vtkPolyDataMapper()
vtk_poly_mapper.SetInputData(vtk_geometry.GetOutput())

vtk_poly_mapper.SetScalarModeToUsePointData()
vtk_poly_mapper.SetLookupTable(lut)
vtk_poly_mapper.SetScalarRange(min_scalar, max_scalar)  # Specify range in terms of scalar minimum and maximum (smin, smax). These values are used to map scalars into lookup table
vtk_poly_mapper.SelectColorArray(scalar_field)  # set the name of scalar field
```

```
\begin{array}{c|c}
  \text{sMin} & \text{entry 0} \\
  \vdots & \vdots \\
  \text{sMax} & \text{entry nc-1} \\
\end{array}
```

```
\begin{array}{c|c}
  \text{color 0} & \\
  \vdots & \\
  \text{color nc-1} & \\
\end{array}
```
2D Interpolated Color Plots

- What happen underneath is the calling of OpenGL drawing functions

Loop through the individual quads to perform the following for each quad

```c
// compute color at V0
glColor3fv (rgb0);
glVertex3f (x0, y0, z0);

// compute color at V1
glColor3fv (rgb1);
glVertex3f (x1, y1, z1);

// compute color at V3
glColor3fv (rgb3);
glVertex3f (x3, y3, z3);

// compute color at V2
glColor3fv (rgb2);
glVertex3f (x2, y2, z2);
```
If the grid is unstructured like a triangle mesh, then...

Loop through the individual triangles to perform the following for each triangle

```c
// compute color at V0
glColor3fv(rgb0);
glVertex3f(x0, y0, z0);

// compute color at V1
glColor3fv(rgb1);
glVertex3f(x1, y1, z1);

// compute color at V3
glColor3fv(rgb3);
glVertex3f(x3, y3, z3);
```

OpenGL mechanism