King Abdullah University of Science and Technology KAUST

CS 247 – Scientific Visualization
Lecture 9: Scalar Field Visualization, Pt. 3

Markus Hadwiger, KAUST

Reading Assignment #5 (until Mar 5)

Read (required):

- Gradients of scalar-valued functions

  `https://en.wikipedia.org/wiki/Gradient`

- Critical points

  `https://en.wikipedia.org/wiki/Critical_point_(mathematics)`

- Multivariable derivatives and differentials

  `https://en.wikipedia.org/wiki/Total_derivative`

  `https://en.wikipedia.org/wiki/Differential_of_a_function#`
  `Differentials_in_several_variables`

  `https://en.wikipedia.org/wiki/Hessian_matrix`

- Dot product, inner product (more general)

  `https://en.wikipedia.org/wiki/Dot_product`

  `https://en.wikipedia.org/wiki/Inner_product_space`

Quiz #1: Mar 1

Organization

- First 30 min of lecture
- No material (book, notes, ...) allowed

Content of questions

- Lectures (both actual lectures and slides)
- Reading assignments (except optional ones)
- Programming assignments (algorithms, methods)
- Solve short practical examples

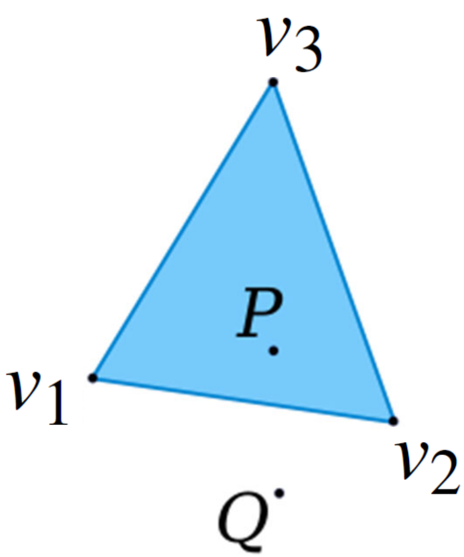
Linear Interpolation / Convex Combinations

Linear combination (n-dim. space):

$$\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = \sum_{i=1}^{n} \alpha_i v_i$$

Affine combination: Restrict to $(n-1)$-dim. subspace:

$$\alpha_1 + \alpha_2 + \ldots + \alpha_n = \sum_{i=1}^{n} \alpha_i = 1$$



Convex combination: $\alpha_i \geq 0$

(restrict to simplex in subspace)

Linear Interpolation / Convex Combinations

The weights α_i are the n normalized ***barycentric*** coordinates

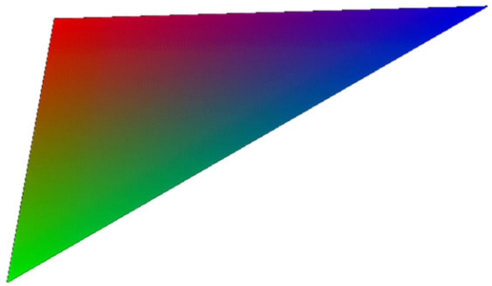
→ linear attribute interpolation in simplex

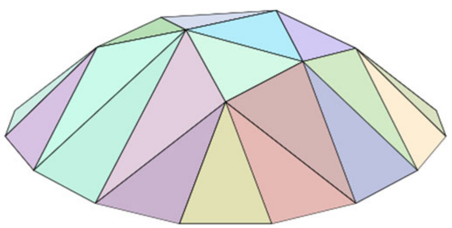
$$\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = \sum_{i=1}^{n} \alpha_i v_i$$

$$\alpha_1 + \alpha_2 + \ldots + \alpha_n = \sum_{i=1}^{n} \alpha_i = 1$$

$$\alpha_i \geq 0$$

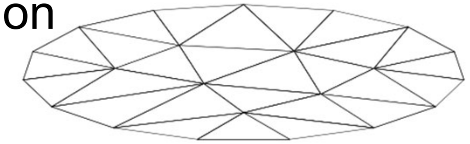
attribute interpolation





spatial position interpolation

wikipedia



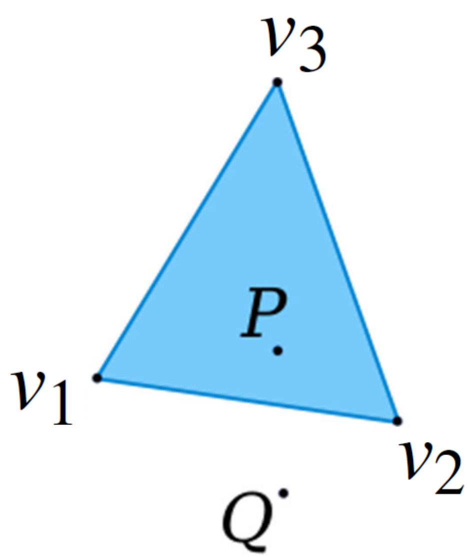
Linear Interpolation / Convex Combinations

$$\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = \sum_{i=1}^{n} \alpha_i v_i$$

$$\alpha_1 + \alpha_2 + \ldots + \alpha_n = \sum_{i=1}^{n} \alpha_i = 1$$

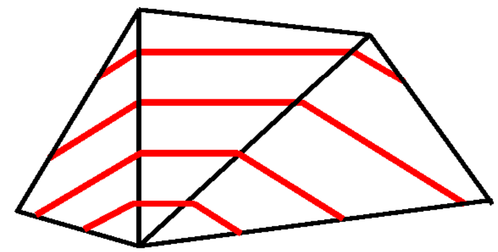
Can re-parameterize to get $(n-1)$ ***affine*** coordinates:

$$\begin{aligned} \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 &= \\ \tilde{\alpha}_1 (v_2 - v_1) + \tilde{\alpha}_2 (v_3 - v_1) + v_1 \\ \tilde{\alpha}_1 &= \alpha_2 \\ \tilde{\alpha}_2 &= \alpha_3 \end{aligned}$$



Contours in triangle/tetrahedral cells

Linear interpolation of cells implies piece-wise linear contours.

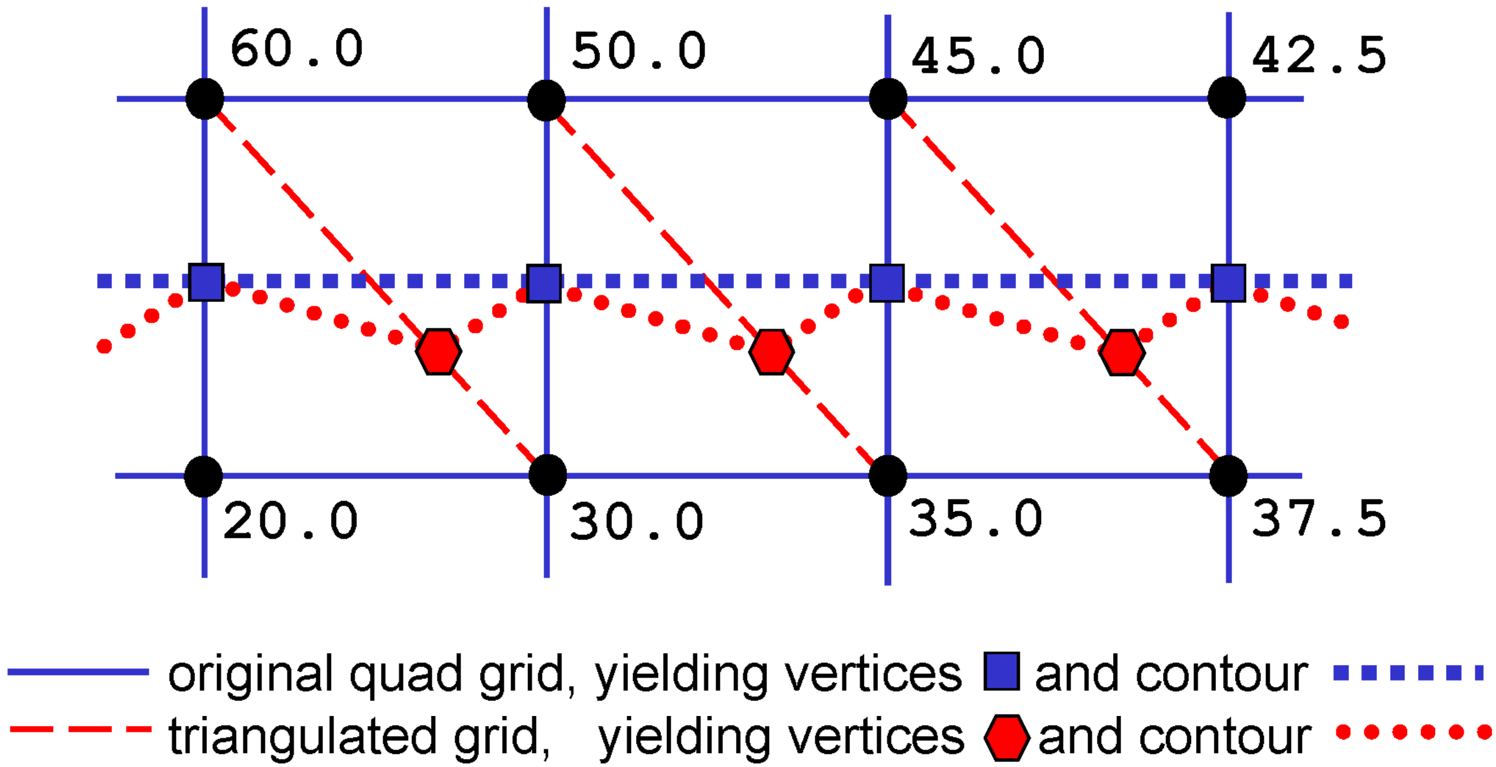


Contours are unambiguous, making "marching triangles" even simpler than "marching squares".

Question: Why not split quadrangles into two triangles (and hexahedra into five or six tetrahedra) and use marching triangles (tetrahedra)?

Answer: This can introduce periodic artifacts!

Illustrative example: Find contour at level *c*=40.0 !

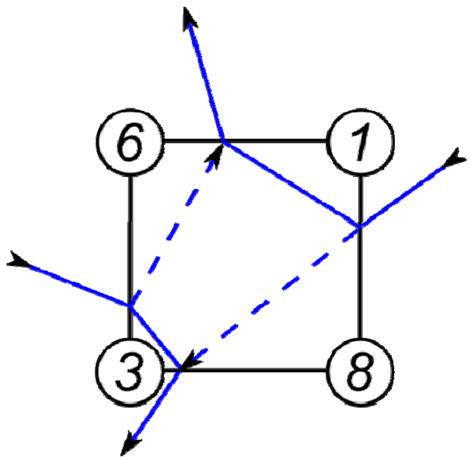


Ambiguities of contours

What is the correct contour of *c*=4?

Two possibilities, both are orientable:

- connect high values ———————
- connect low values - - - - - - - - - -



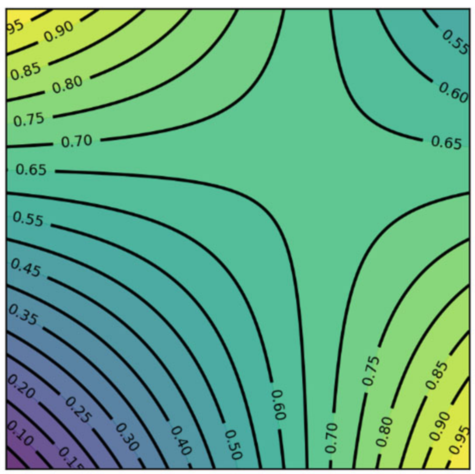
Answer: correctness depends on interior values of $f(x)$.

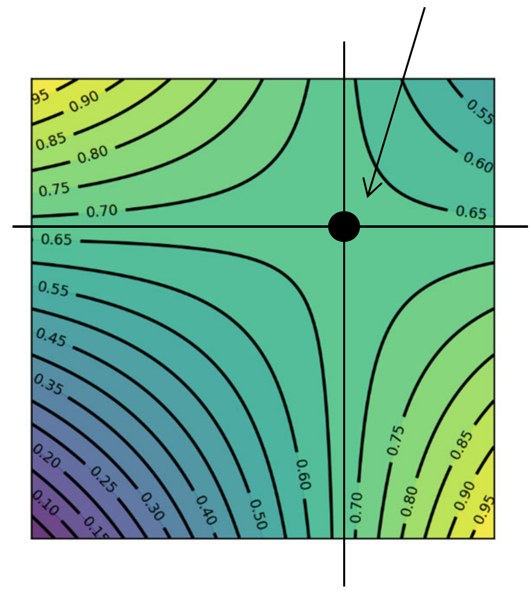
But: different interpolation schemes are possible.

Better question: What is the correct contour with respect to bilinear interpolation?

Bi-Linear Interpolation: Critical Points

Critical points are where the gradient vanishes (i.e., is the zero vector)

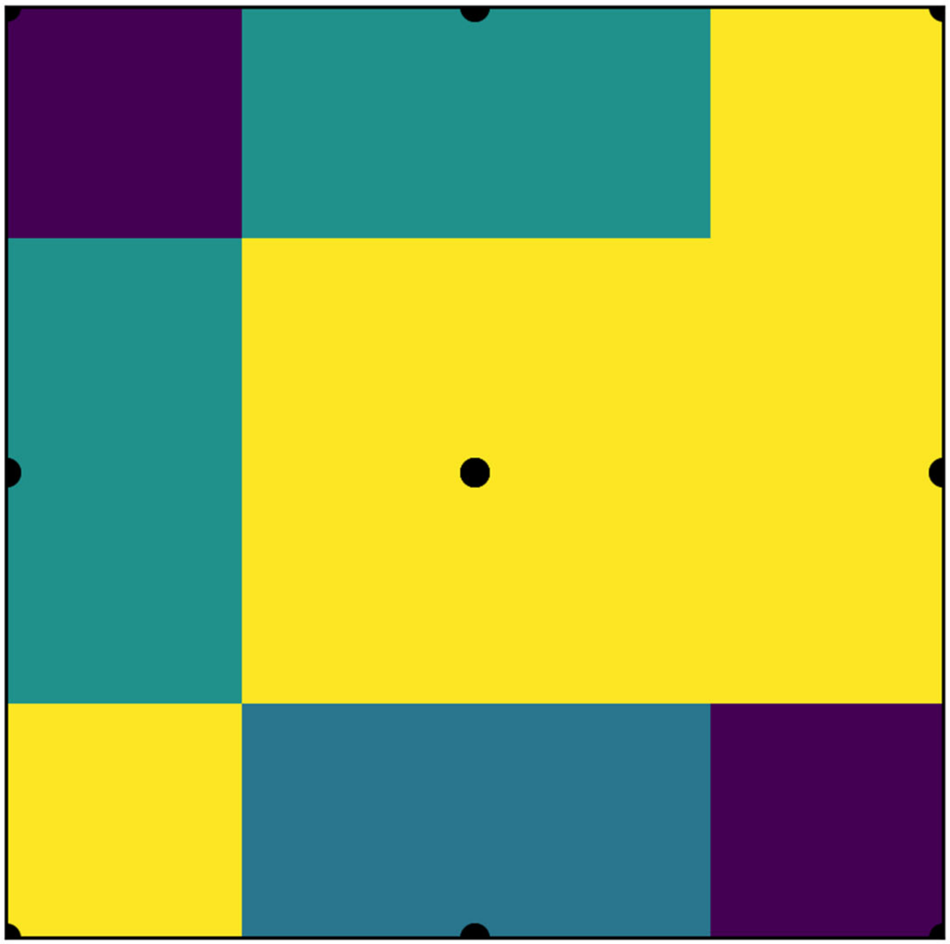




“Asymptotic decider”: resolve ambiguous configurations (6 and 9) by
comparing specific iso-value with critical value (scalar value at critical point)

Bi-Linear Interpolation: Comparisons

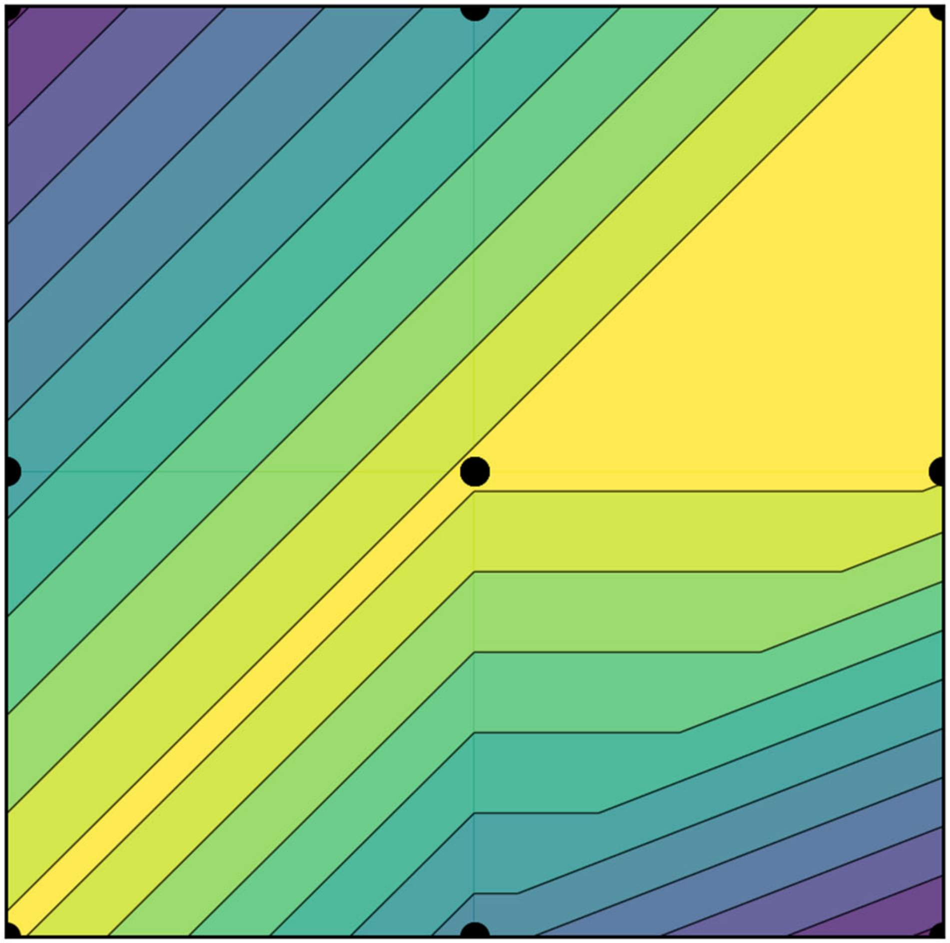
nearest-neighbor



Bi-Linear Interpolation: Comparisons

piecewise linear

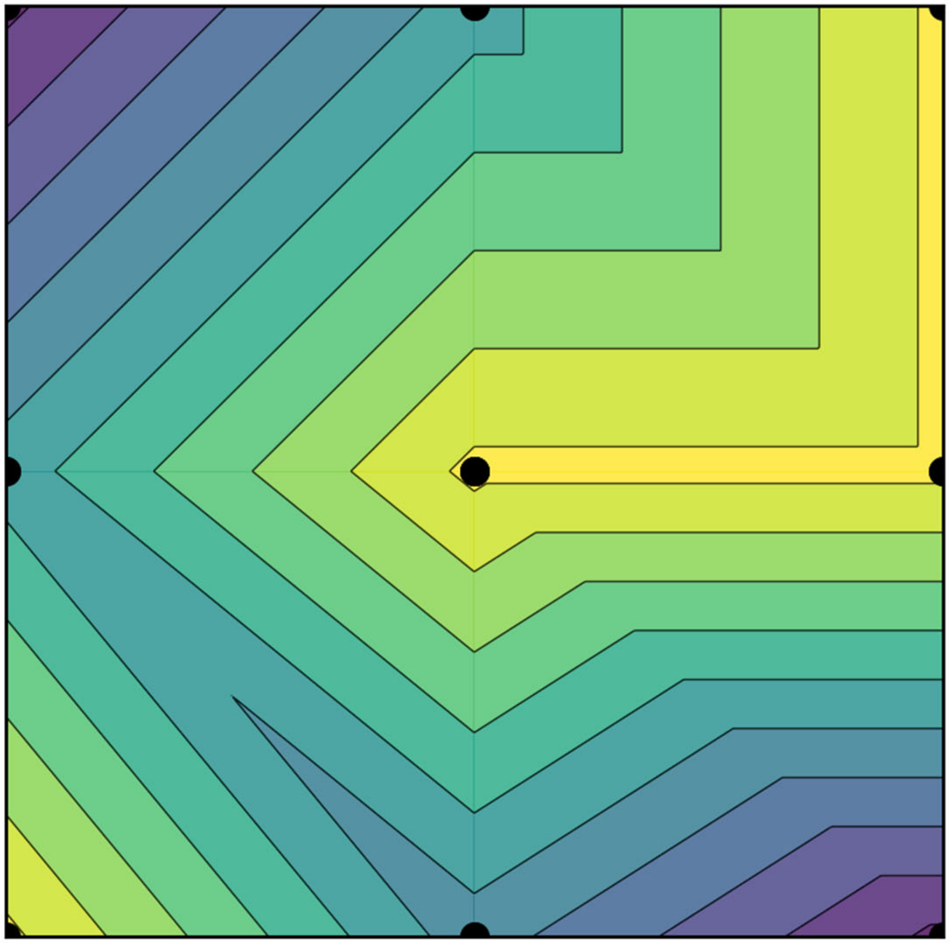
(2 triangles per quad;
diagonal:
bottom-left,
top-right)



Bi-Linear Interpolation: Comparisons

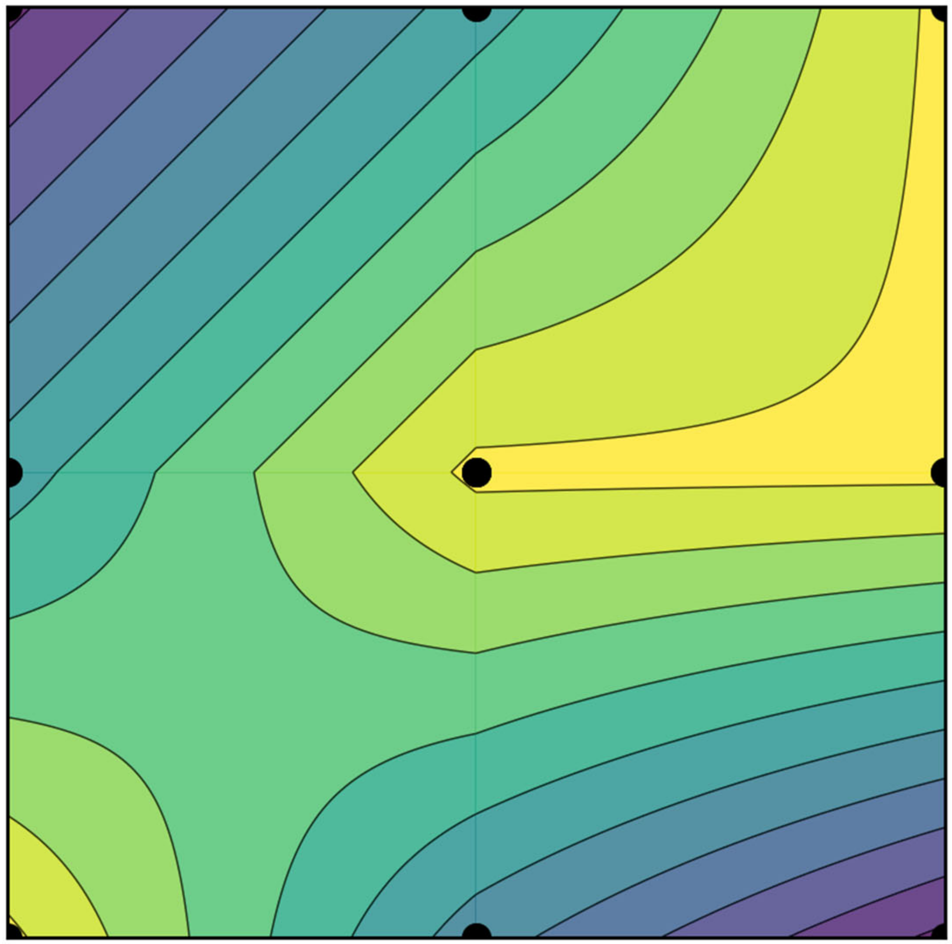
piecewise linear

(2 triangles per quad;
diagonal:
top-left,
bottom-right)



Bi-Linear Interpolation: Comparisons

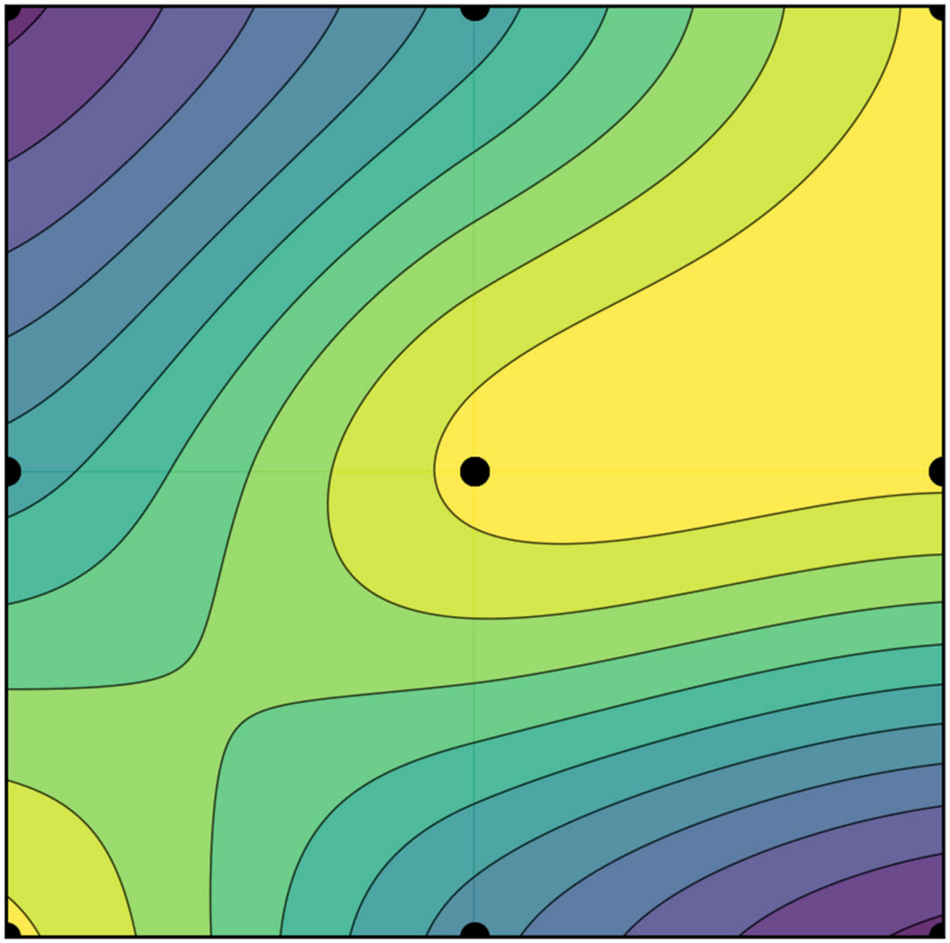
bi-linear



Bi-Linear Interpolation: Comparisons

bi-cubic

(Catmull-Rom spline)



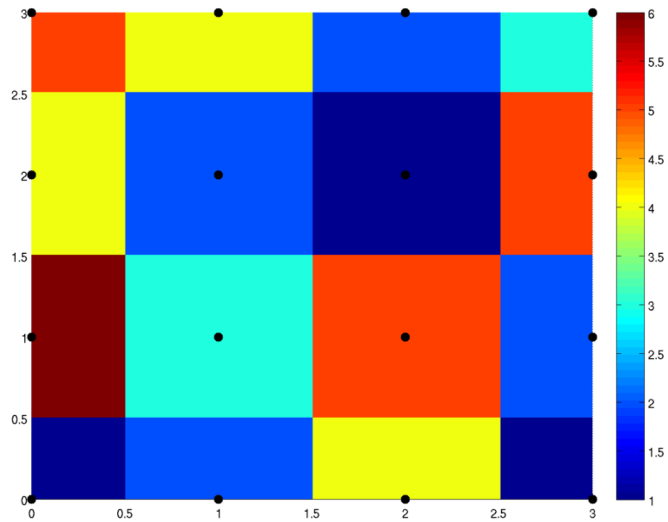
Texture Filtering Example (Magnification)

Original texture image

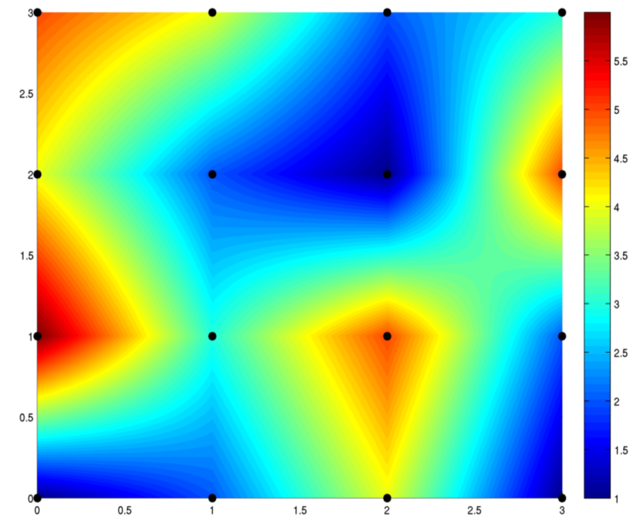
Nearest neighbor

Bi-linear filtering

Bi-Linear Interpolation vs. Nearest Neighbor



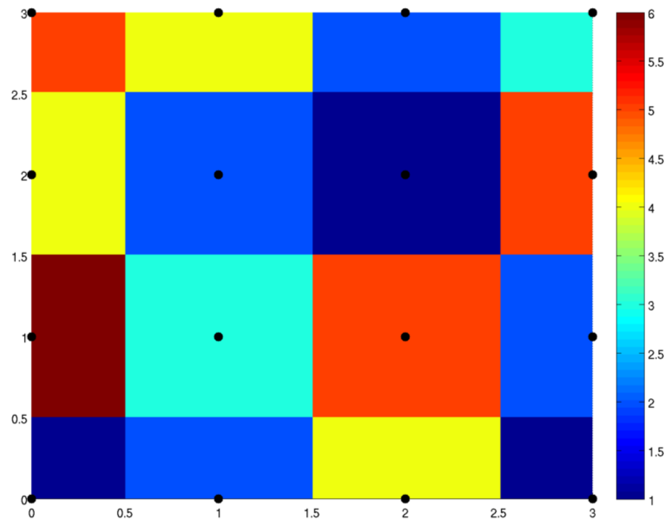
nearest-neighbor



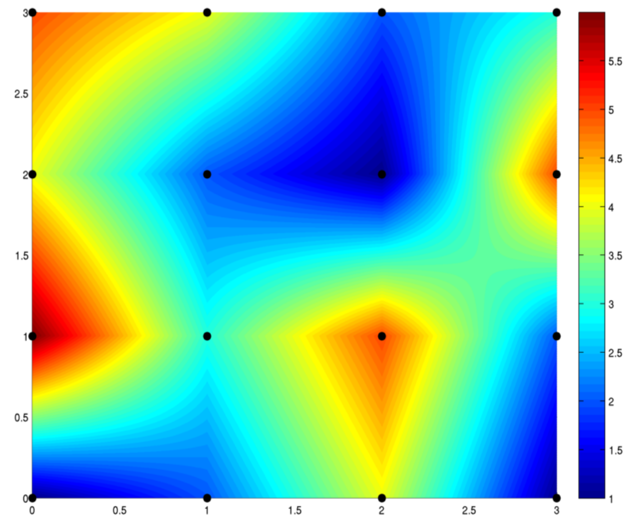
bi-linear

wikipedia

Bi-Linear Interpolation vs. Nearest Neighbor



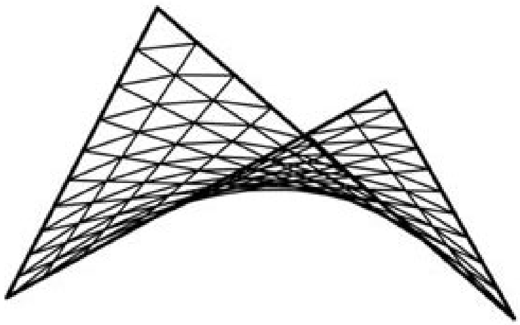
nearest-neighbor

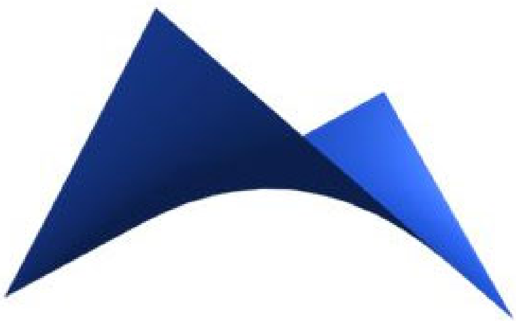


bi-linear

wikipedia

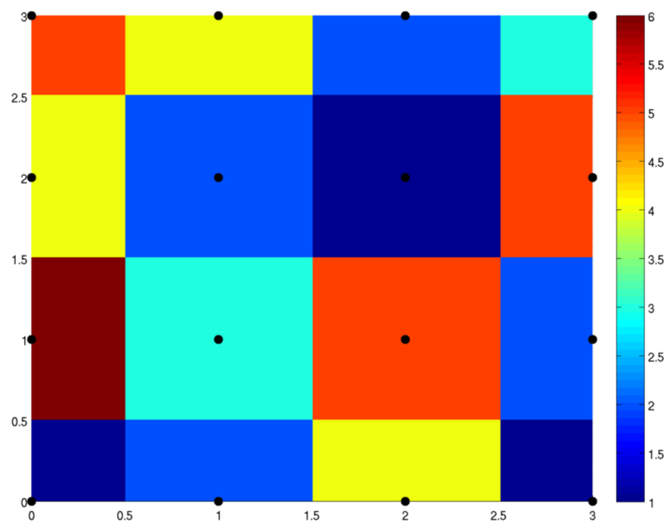
for surfaces,
height interpolation:

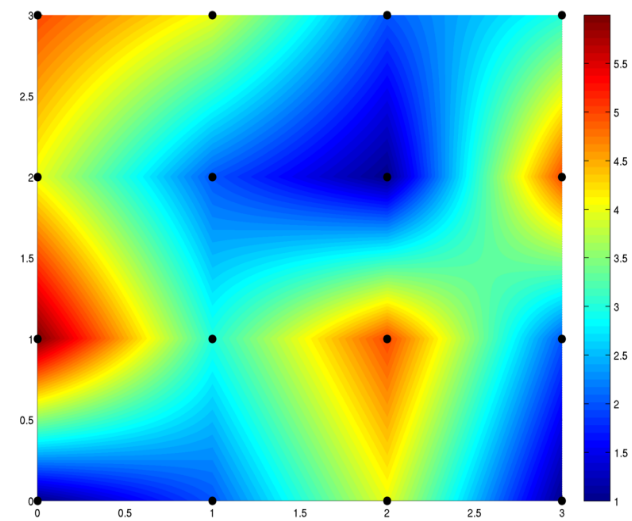




Bilinear patch (courtesy J. Han)

Bi-Linear Interpolation vs. Nearest Neighbor



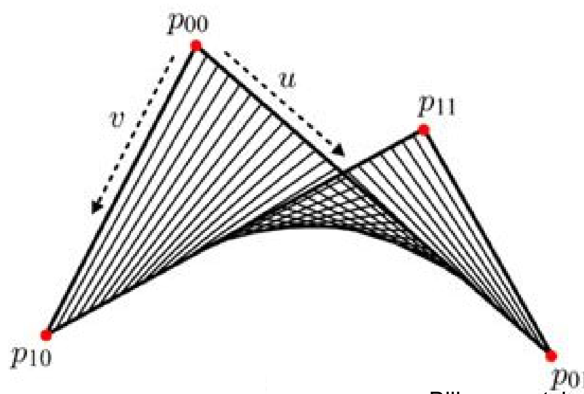


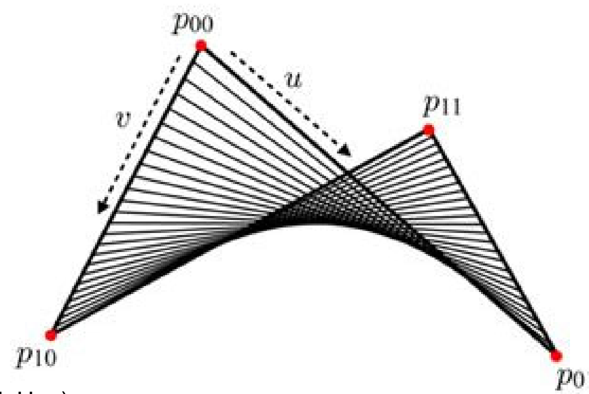
wikipedia

nearest-neighbor

bi-linear

for surfaces,
height interpolation:



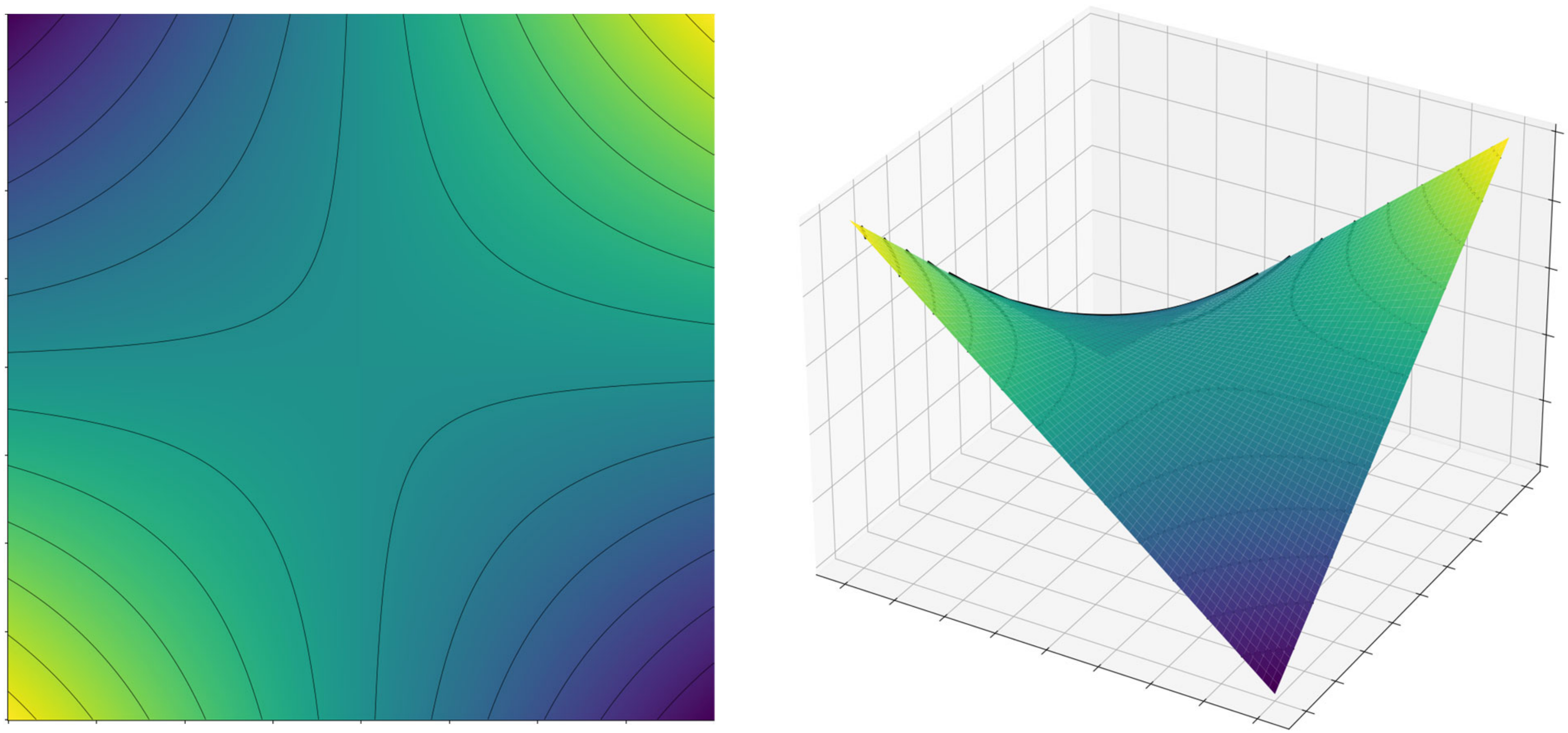


Bilinear patch (courtesy J. Han)

Bi-Linear Interpolation

Consider area between 2x2 adjacent samples (e.g., pixel centers)

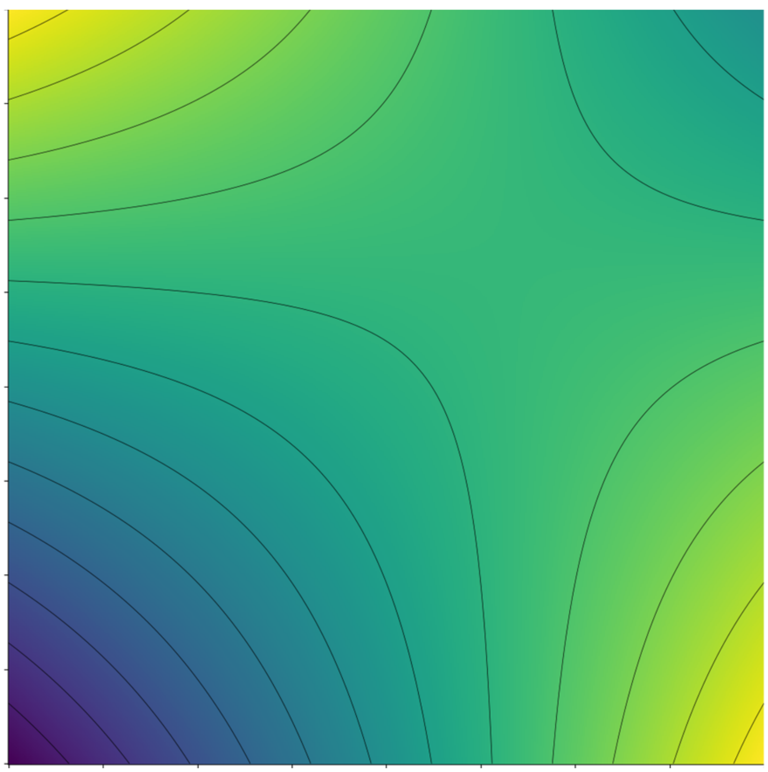
Example #1: 1 at bottom-left and top-right, 0 at top-left and bottom-right

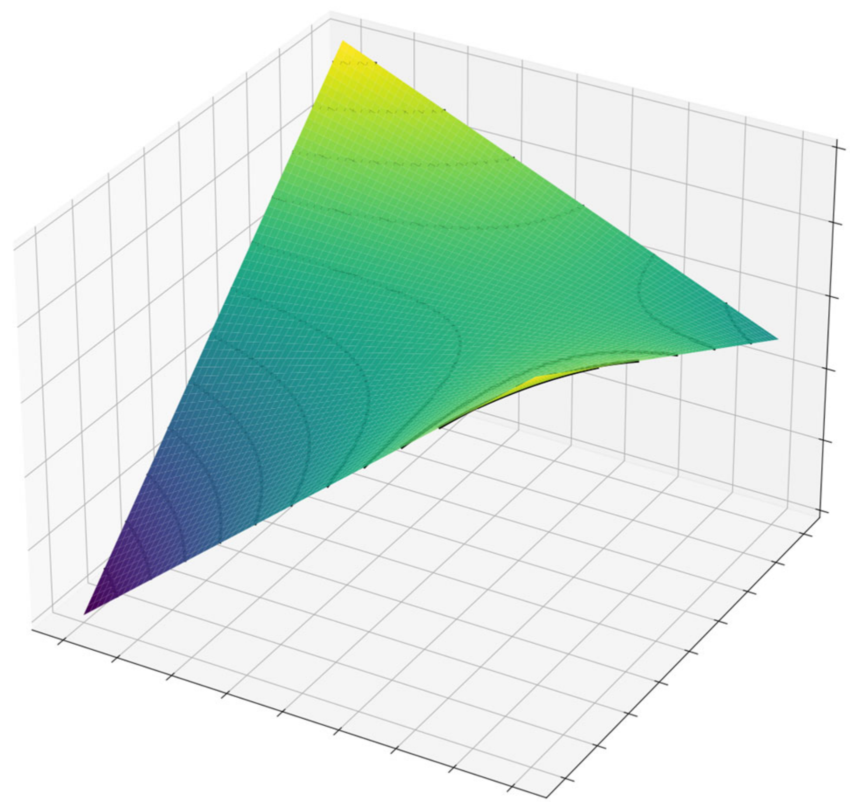


Bi-Linear Interpolation

Consider area between 2x2 adjacent samples (e.g., pixel centers)

Example #2: 1 at top-left and bottom-right, 0 at bottom-left, 0.5 at top-right





Bi-Linear Interpolation

Consider area between 2x2 adjacent samples (e.g., pixel centers):

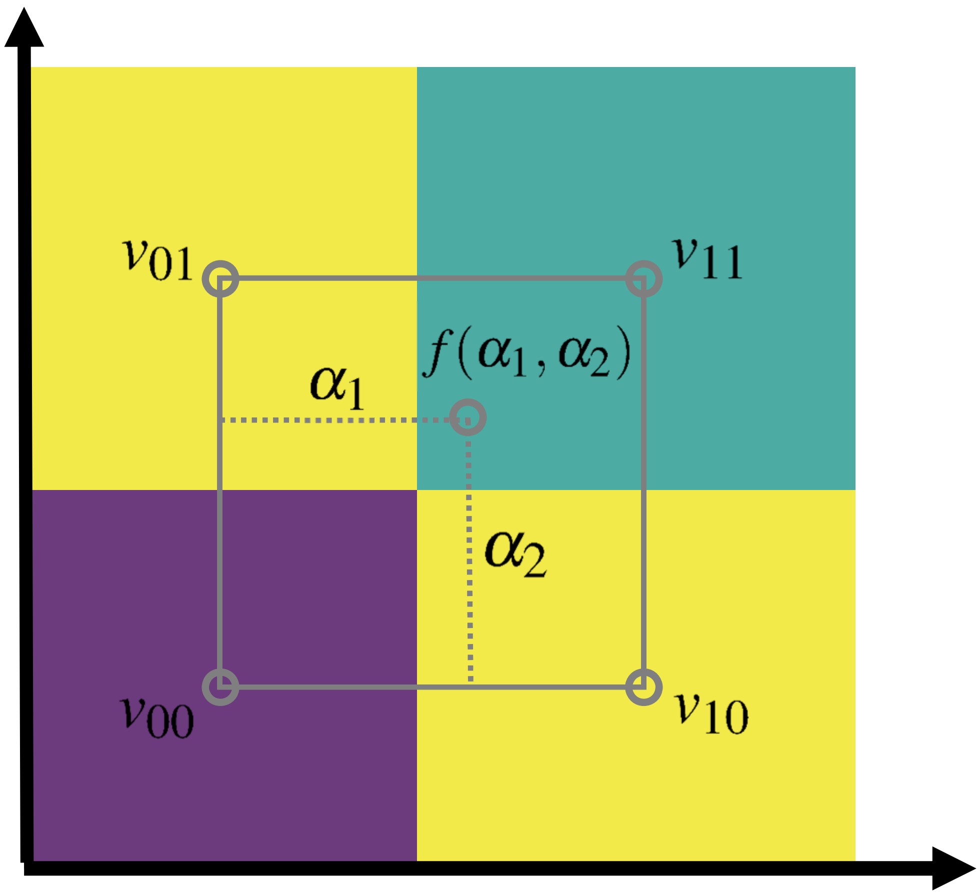
Given any (fractional) position

$$\alpha_1 := x_1 - \lfloor x_1 \rfloor \qquad \alpha_1 \in [0.0, 1.0)$$
$$\alpha_2 := x_2 - \lfloor x_2 \rfloor \qquad \alpha_2 \in [0.0, 1.0)$$

and 2x2 sample values

$$\begin{bmatrix} v_{01} & v_{11} \\ v_{00} & v_{10} \end{bmatrix}$$

Compute: $f(\alpha_1, \alpha_2)$



Bi-Linear Interpolation

Consider area between 2x2 adjacent samples (e.g., pixel centers):

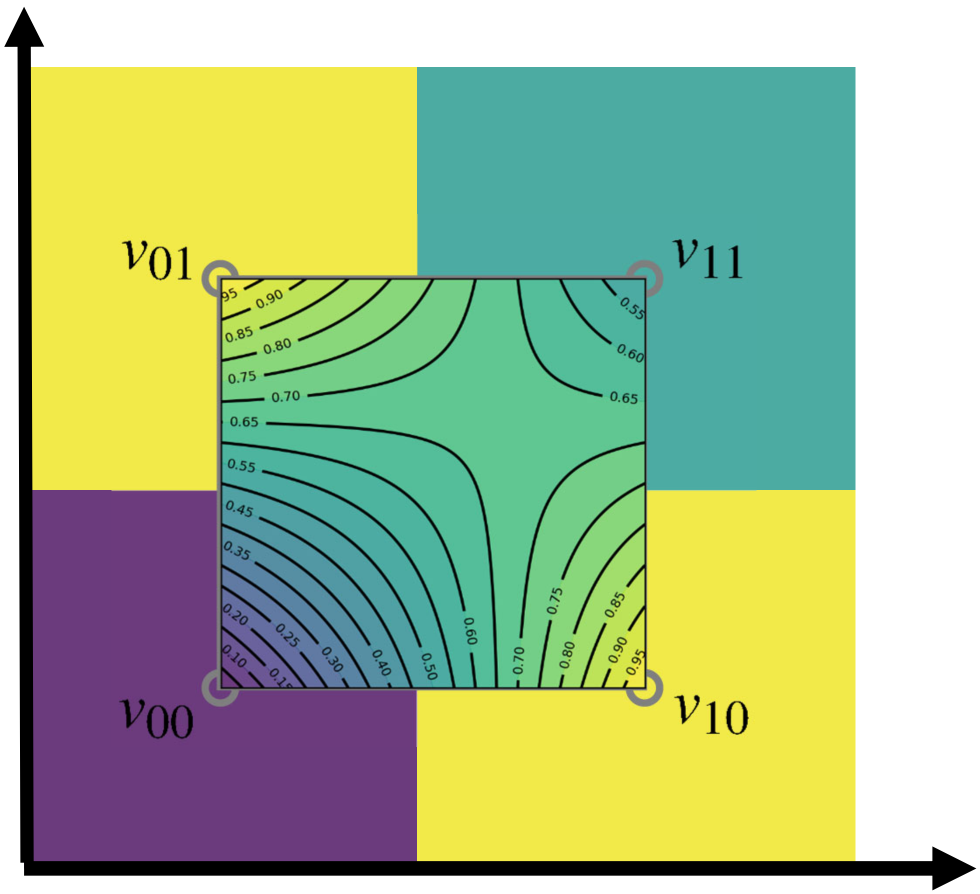
Given any (fractional) position

$$\alpha_1 := x_1 - \lfloor x_1 \rfloor \qquad \alpha_1 \in [0.0, 1.0)$$
$$\alpha_2 := x_2 - \lfloor x_2 \rfloor \qquad \alpha_2 \in [0.0, 1.0)$$

and 2x2 sample values

$$\begin{bmatrix} v_{01} & v_{11} \\ v_{00} & v_{10} \end{bmatrix}$$

Compute: $f(\alpha_1, \alpha_2)$



Bi-Linear Interpolation

Interpolate function at (fractional) position (α_1, α_2) :

$$f(\alpha_1, \alpha_2) = \begin{bmatrix} \alpha_2 & (1-\alpha_2) \end{bmatrix} \begin{bmatrix} v_{01} & v_{11} \\ v_{00} & v_{10} \end{bmatrix} \begin{bmatrix} (1-\alpha_1) \\ \alpha_1 \end{bmatrix}$$

$$= \begin{bmatrix} \alpha_2 & (1-\alpha_2) \end{bmatrix} \begin{bmatrix} (1-\alpha_1)v_{01} + \alpha_1 v_{11} \\ (1-\alpha_1)v_{00} + \alpha_1 v_{10} \end{bmatrix}$$

$$= \begin{bmatrix} \alpha_2 v_{01} + (1-\alpha_2)v_{00} & \alpha_2 v_{11} + (1-\alpha_2)v_{10} \end{bmatrix} \begin{bmatrix} (1-\alpha_1) \\ \alpha_1 \end{bmatrix}$$

Bi-Linear Interpolation

Interpolate function at (fractional) position (α_1, α_2) :

$$f(\alpha_1, \alpha_2) = \begin{bmatrix} \alpha_2 & (1-\alpha_2) \end{bmatrix} \begin{bmatrix} v_{01} & v_{11} \\ v_{00} & v_{10} \end{bmatrix} \begin{bmatrix} (1-\alpha_1) \\ \alpha_1 \end{bmatrix}$$

$$= (1-\alpha_1)(1-\alpha_2)v_{00} + \alpha_1(1-\alpha_2)v_{10} + (1-\alpha_1)\alpha_2 v_{01} + \alpha_1 \alpha_2 v_{11}$$

$$= v_{00} + \alpha_1(v_{10} - v_{00}) + \alpha_2(v_{01} - v_{00}) + \alpha_1 \alpha_2(v_{00} + v_{11} - v_{10} - v_{01})$$

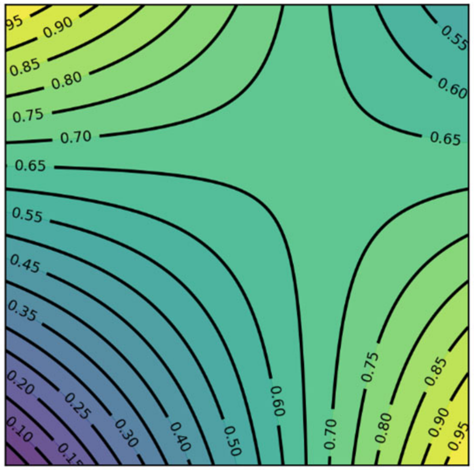
Bi-Linear Interpolation: Contours

Find one specific iso-contour (can of course do this for any/all isovalues):

$$f(\alpha_1, \alpha_2) = c$$

Find all (α_1, α_2) where:

$$v_{00} + \alpha_1 (v_{10} - v_{00}) + \alpha_2 (v_{01} - v_{00}) + \alpha_1 \alpha_2 (v_{00} + v_{11} - v_{10} - v_{01}) = c$$



Bi-Linear Interpolation: Critical Points

Compute gradient (critical points are where gradient is zero vector):

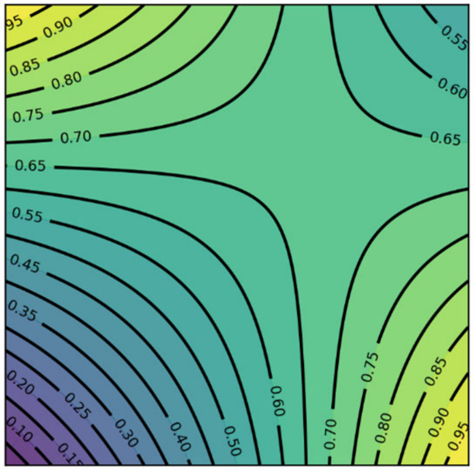
$$\frac{\partial f(\alpha_1, \alpha_2)}{\partial \alpha_1} = (v_{10} - v_{00}) + \alpha_2 (v_{00} + v_{11} - v_{10} - v_{01})$$

$$\frac{\partial f(\alpha_1, \alpha_2)}{\partial \alpha_2} = (v_{01} - v_{00}) + \alpha_1 (v_{00} + v_{11} - v_{10} - v_{01})$$

Where are lines of constant value / critical points?

$$\frac{\partial f(\alpha_1, \alpha_2)}{\partial \alpha_1} = 0 : \qquad \alpha_2 = \frac{v_{00} - v_{10}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

$$\frac{\partial f(\alpha_1, \alpha_2)}{\partial \alpha_2} = 0 : \qquad \alpha_1 = \frac{v_{00} - v_{01}}{v_{00} + v_{11} - v_{10} - v_{01}}$$



Bi-Linear Interpolation: Critical Points

Compute gradient (critical points are where gradient is zero vector):

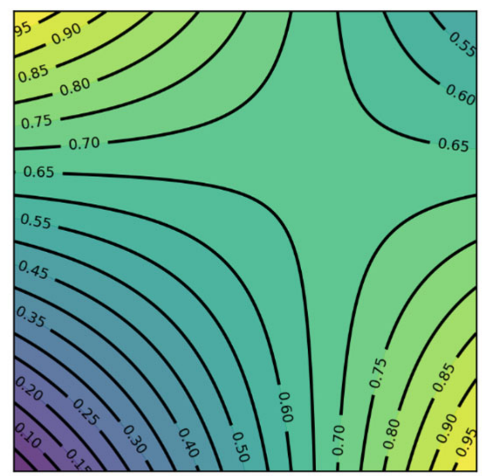
$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = (v_{10} - v_{00}) + \alpha_2 (v_{00} + v_{11} - v_{10} - v_{01})$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = (v_{01} - v_{00}) + \alpha_1 (v_{00} + v_{11} - v_{10} - v_{01})$$

Where are lines of constant value / critical points?

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = 0: \qquad \alpha_2 = \frac{v_{00} - v_{10}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = 0: \qquad \alpha_1 = \frac{v_{00} - v_{01}}{v_{00} + v_{11} - v_{10} - v_{01}}$$



if denominator is zero, bi-linear interpolation has degenerated to linear interpolation (or const)! (also means: no isolated critical points!)

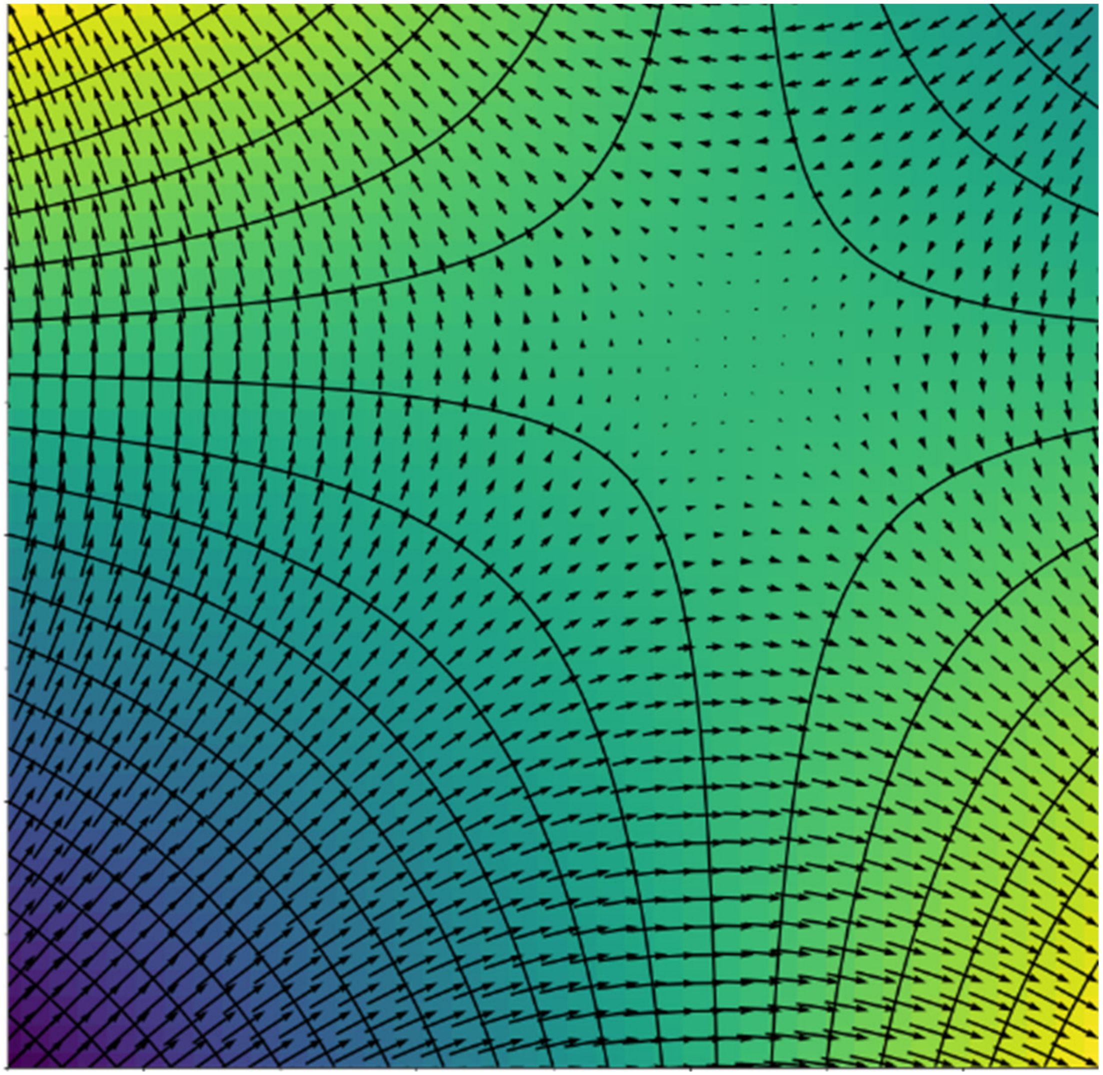
Bi-Linear Interpolation: Critical Points

Compute gradient

Note that isolines are farther apart where gradient is smaller

Note the horizontal and vertical lines where gradient becomes vertical/horizontal

Note the critical point



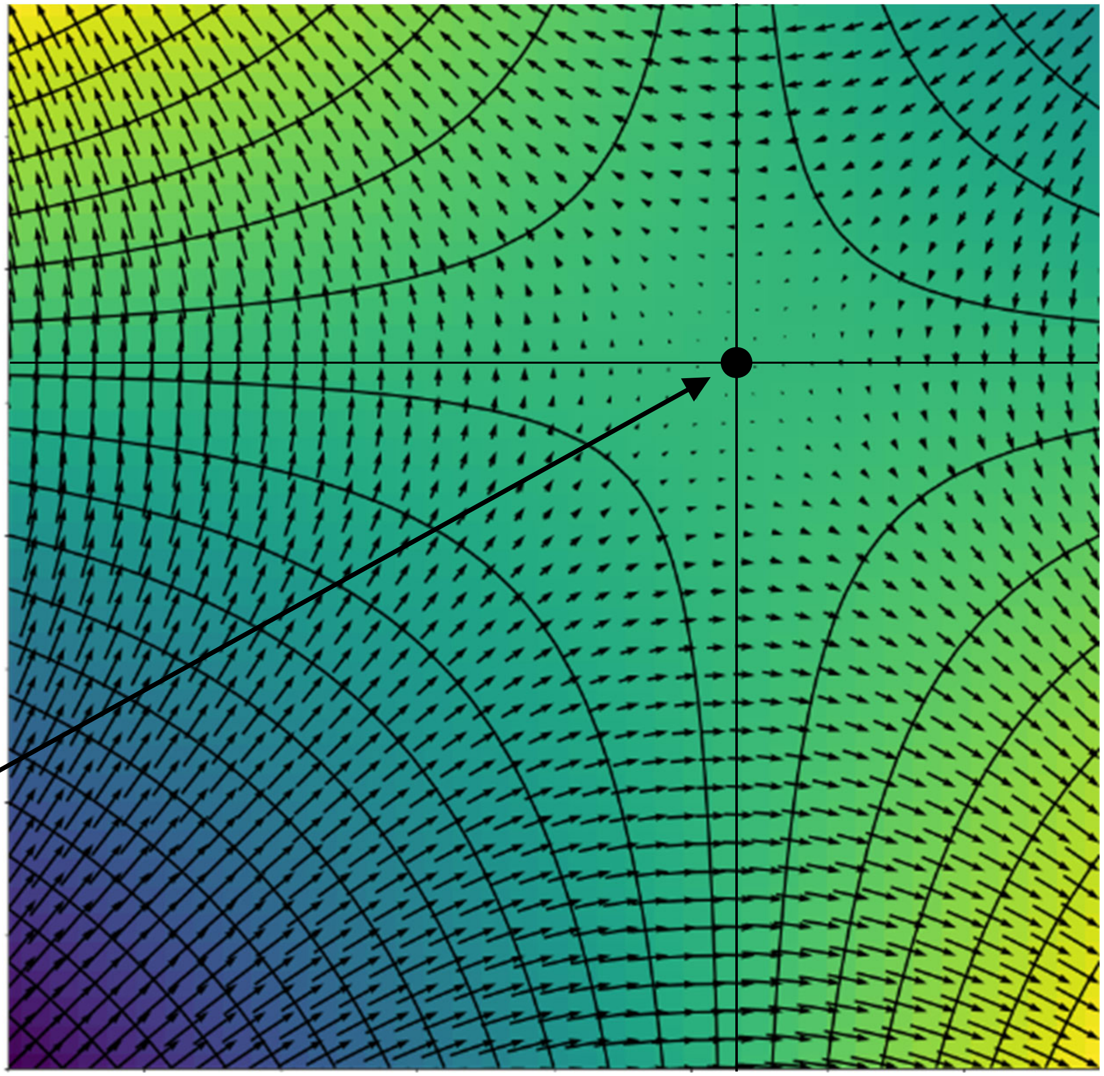
Bi-Linear Interpolation: Critical Points

Compute gradient

Note that isolines are
farther apart where
gradient is smaller

Note the horizontal and
vertical lines where
gradient becomes
vertical/horizontal

Note the critical point



Bi-Linear Interpolation: Critical Points

Compute gradient (critical points are where gradient is zero vector):

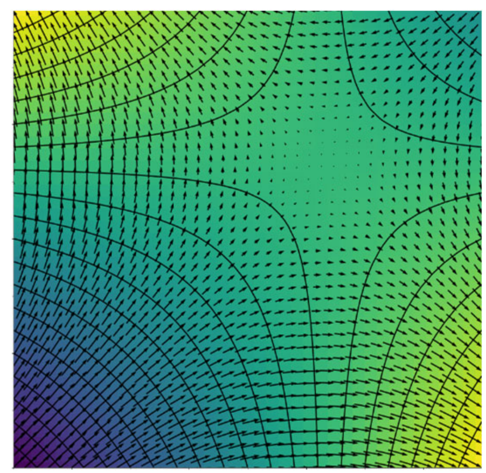
$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = (v_{10} - v_{00}) + \alpha_2(v_{00} + v_{11} - v_{10} - v_{01})$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = (v_{01} - v_{00}) + \alpha_1(v_{00} + v_{11} - v_{10} - v_{01})$$

Where are lines of constant value / critical points?

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = 0: \qquad \alpha_2 = \frac{v_{00} - v_{10}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = 0: \qquad \alpha_1 = \frac{v_{00} - v_{01}}{v_{00} + v_{11} - v_{10} - v_{01}}$$



Bi-Linear Interpolation: Critical Points

Compute gradient (critical points are where gradient is zero vector):

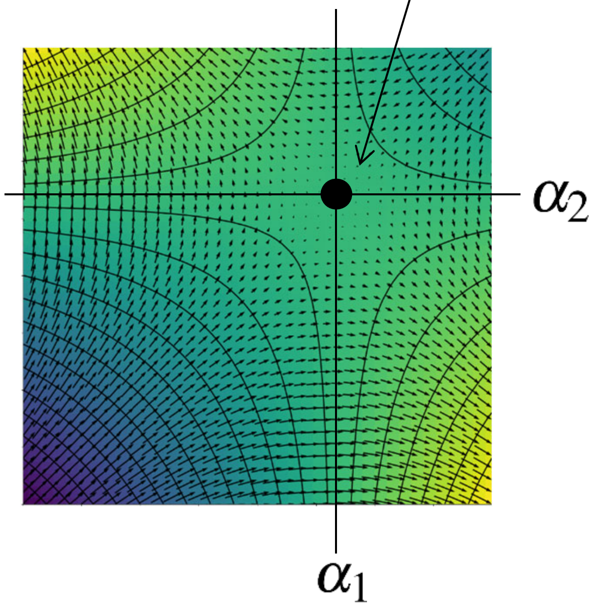
$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = (v_{10} - v_{00}) + \alpha_2(v_{00} + v_{11} - v_{10} - v_{01})$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = (v_{01} - v_{00}) + \alpha_1(v_{00} + v_{11} - v_{10} - v_{01})$$

Where are lines of constant value / critical points?

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = 0: \quad \alpha_2 = \frac{v_{00} - v_{10}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = 0: \quad \alpha_1 = \frac{v_{00} - v_{01}}{v_{00} + v_{11} - v_{10} - v_{01}}$$



Bi-Linear Interpolation: Critical Points

Compute gradient (critical points are where gradient is zero vector):

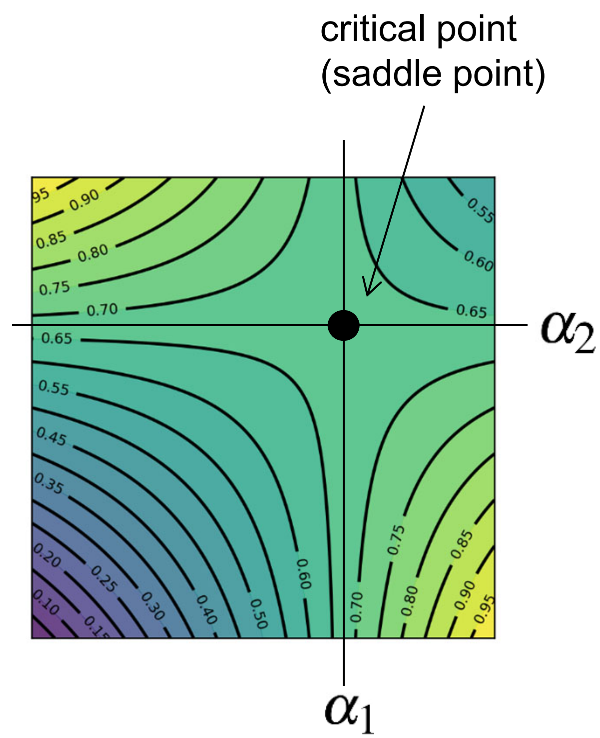
$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = (v_{10} - v_{00}) + \alpha_2 (v_{00} + v_{11} - v_{10} - v_{01})$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = (v_{01} - v_{00}) + \alpha_1 (v_{00} + v_{11} - v_{10} - v_{01})$$

Where are lines of constant value / critical points?

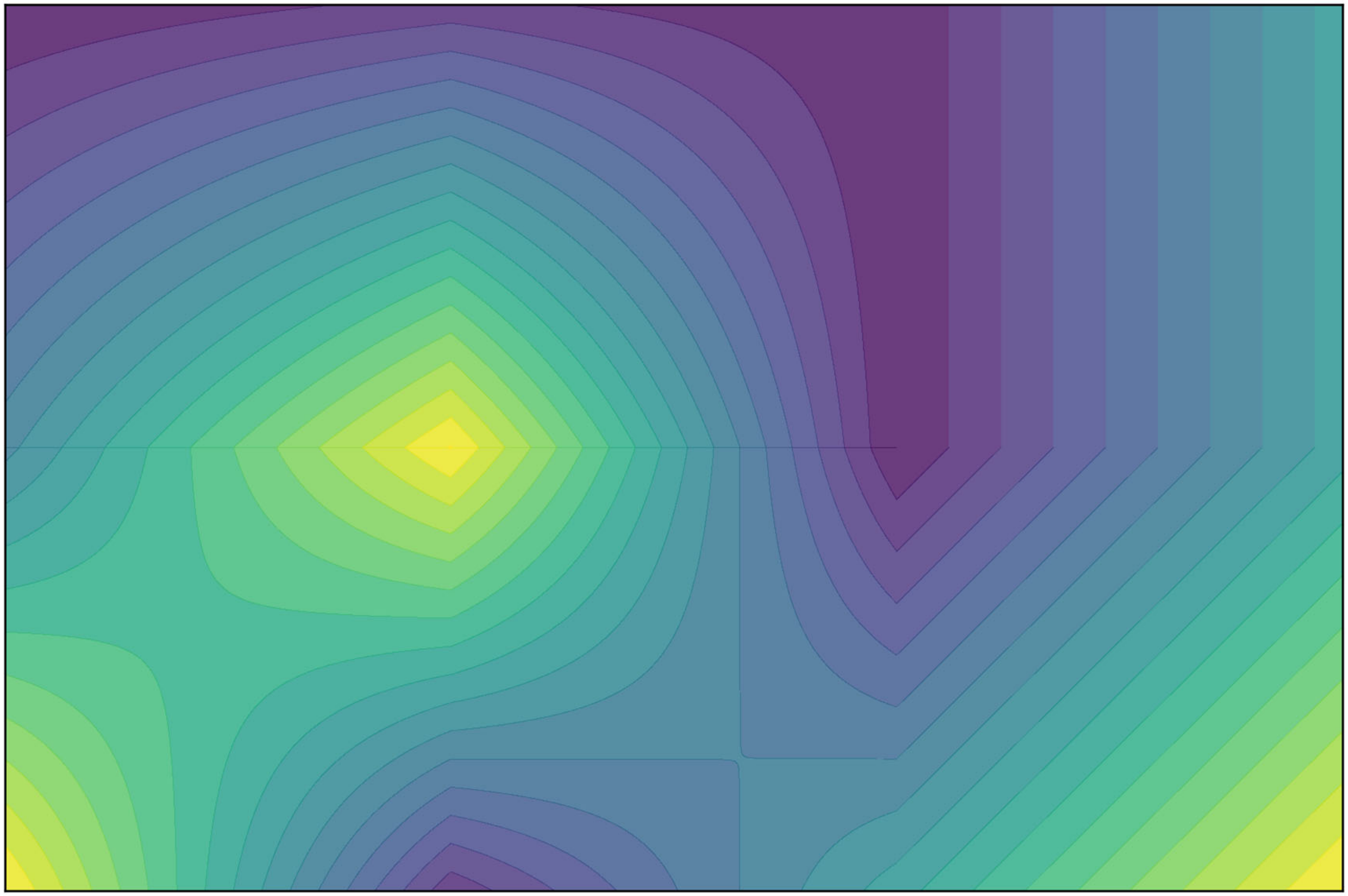
$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_1} = 0: \qquad \alpha_2 = \frac{v_{00} - v_{10}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

$$\frac{\partial f(\alpha_1,\alpha_2)}{\partial \alpha_2} = 0: \qquad \alpha_1 = \frac{v_{00} - v_{01}}{v_{00} + v_{11} - v_{10} - v_{01}}$$

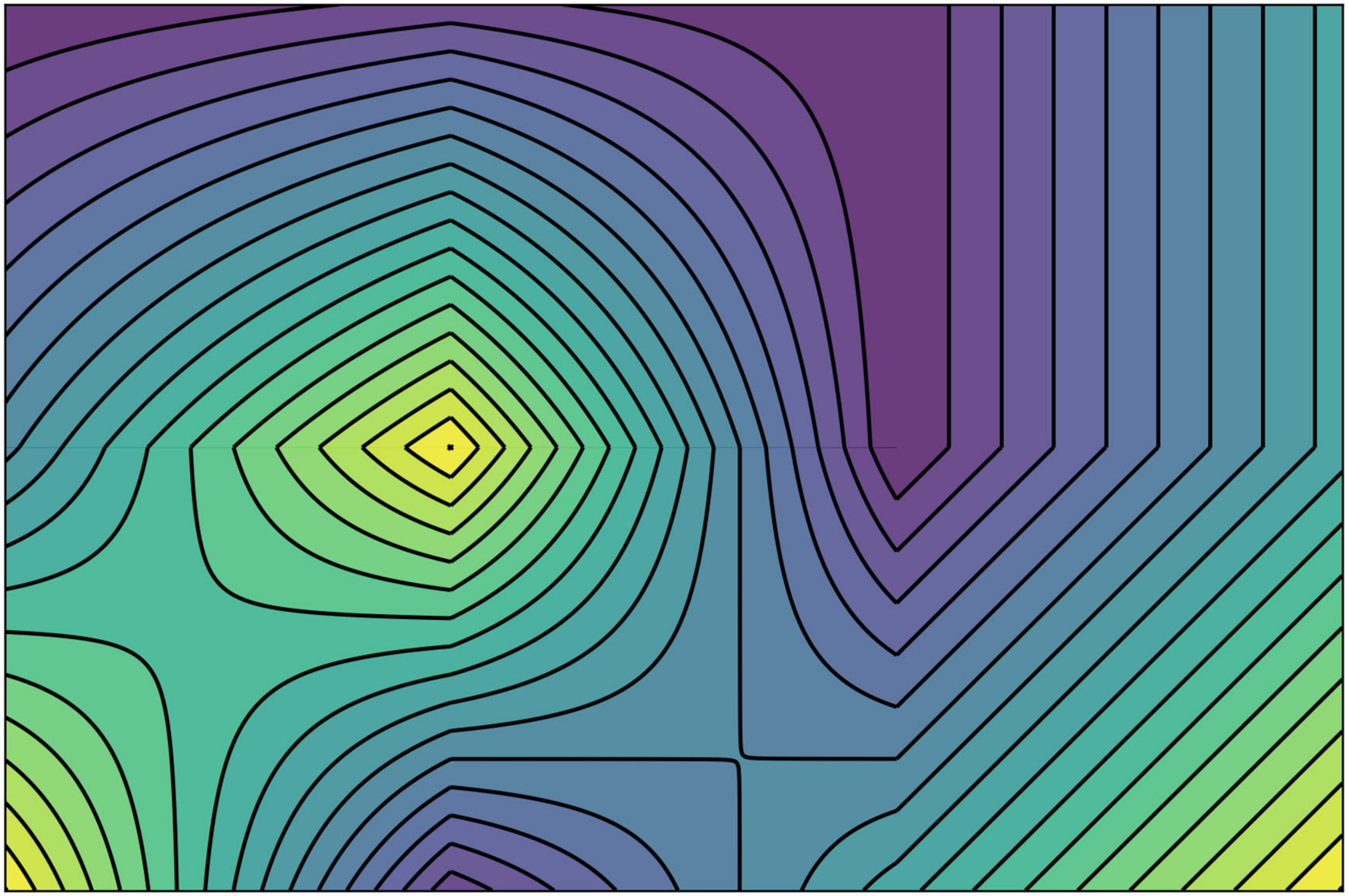


More examples

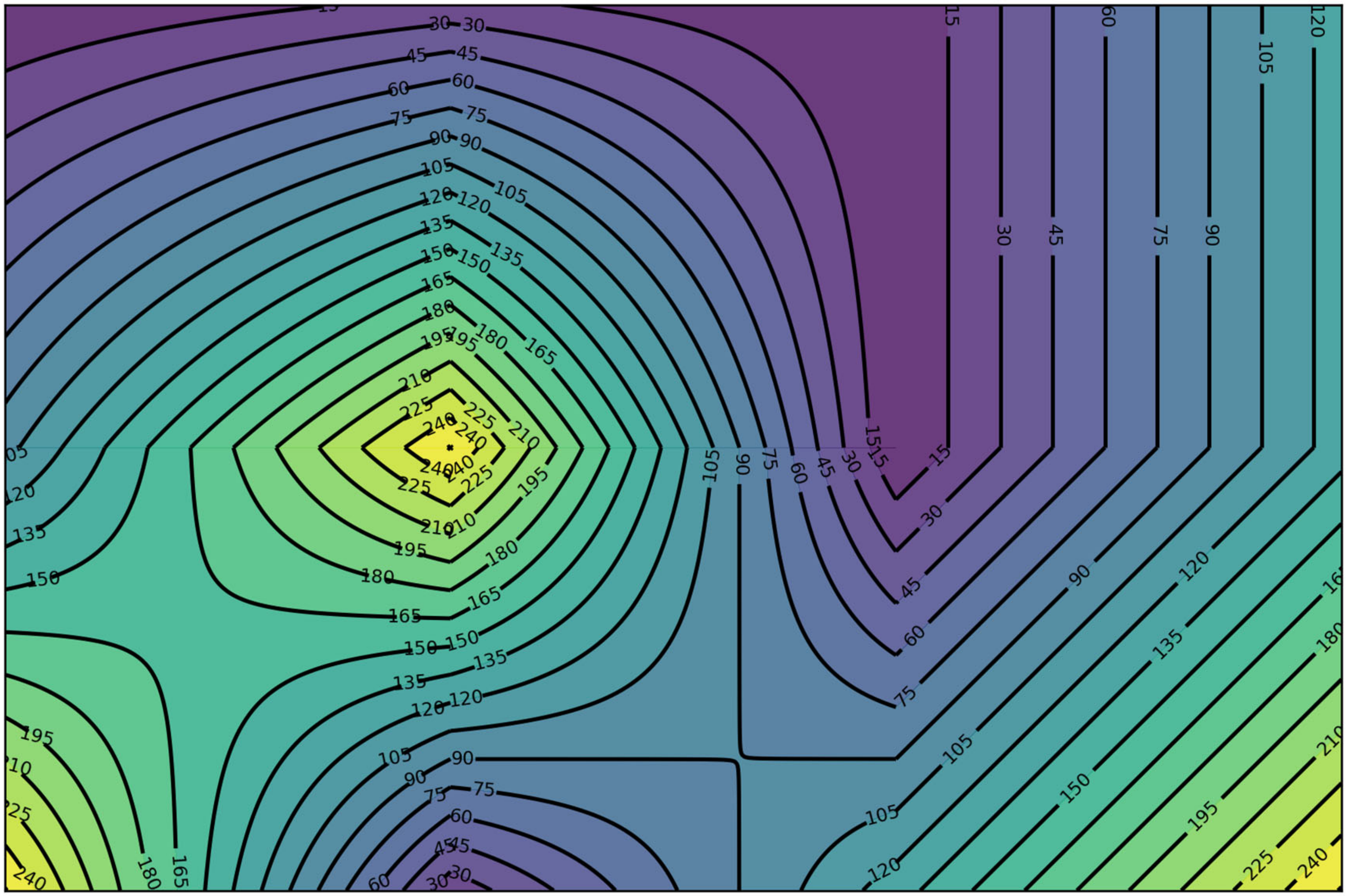
Piecewise Bi-Linear (Example: 3x2 Cells)



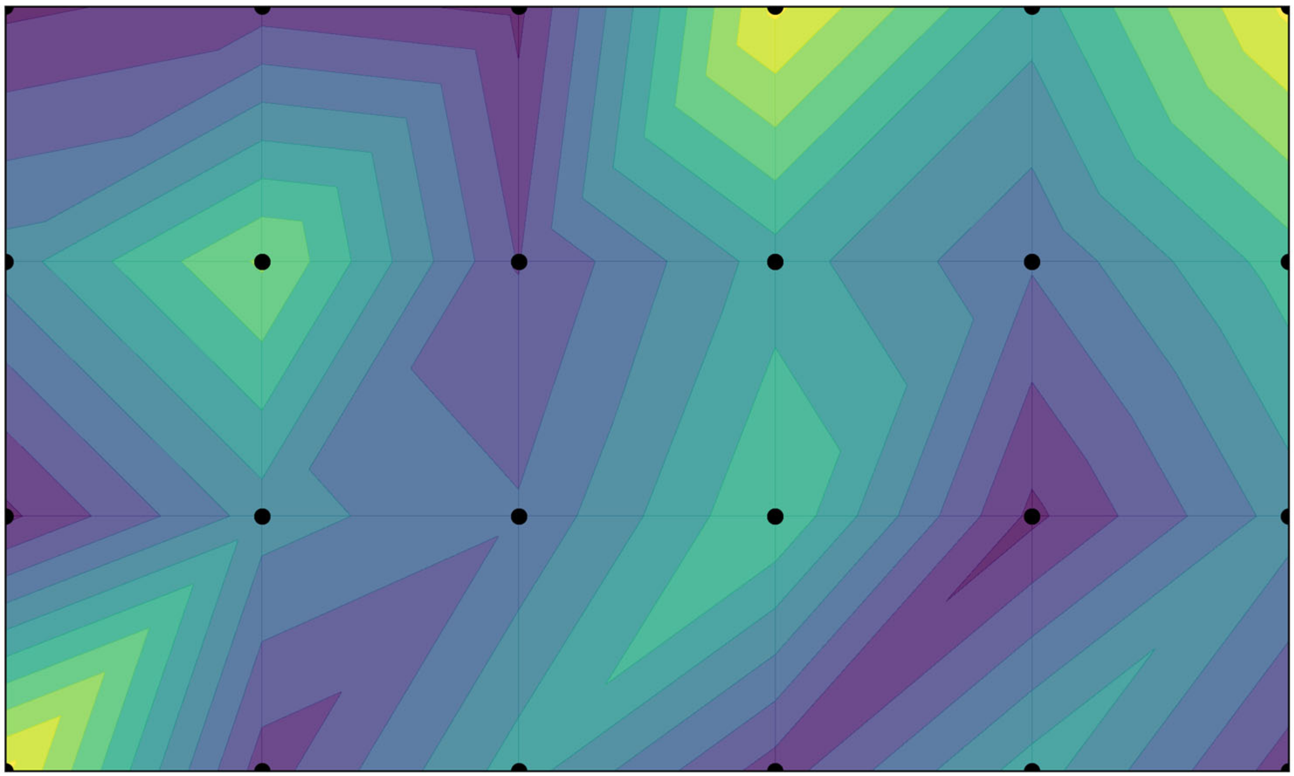
Piecewise Bi-Linear (Example: 3x2 Cells)



Piecewise Bi-Linear (Example: 3x2 Cells)

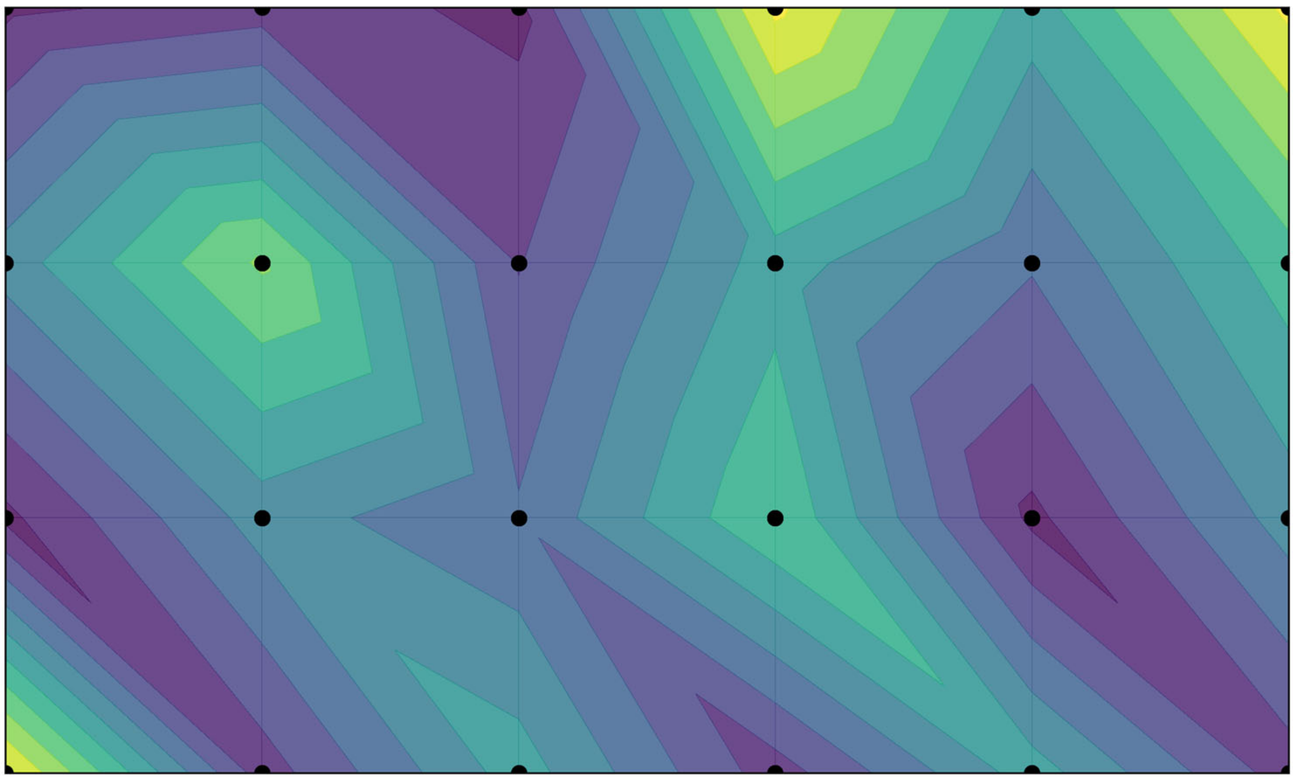


Bi-Linear Interpolation: Comparisons



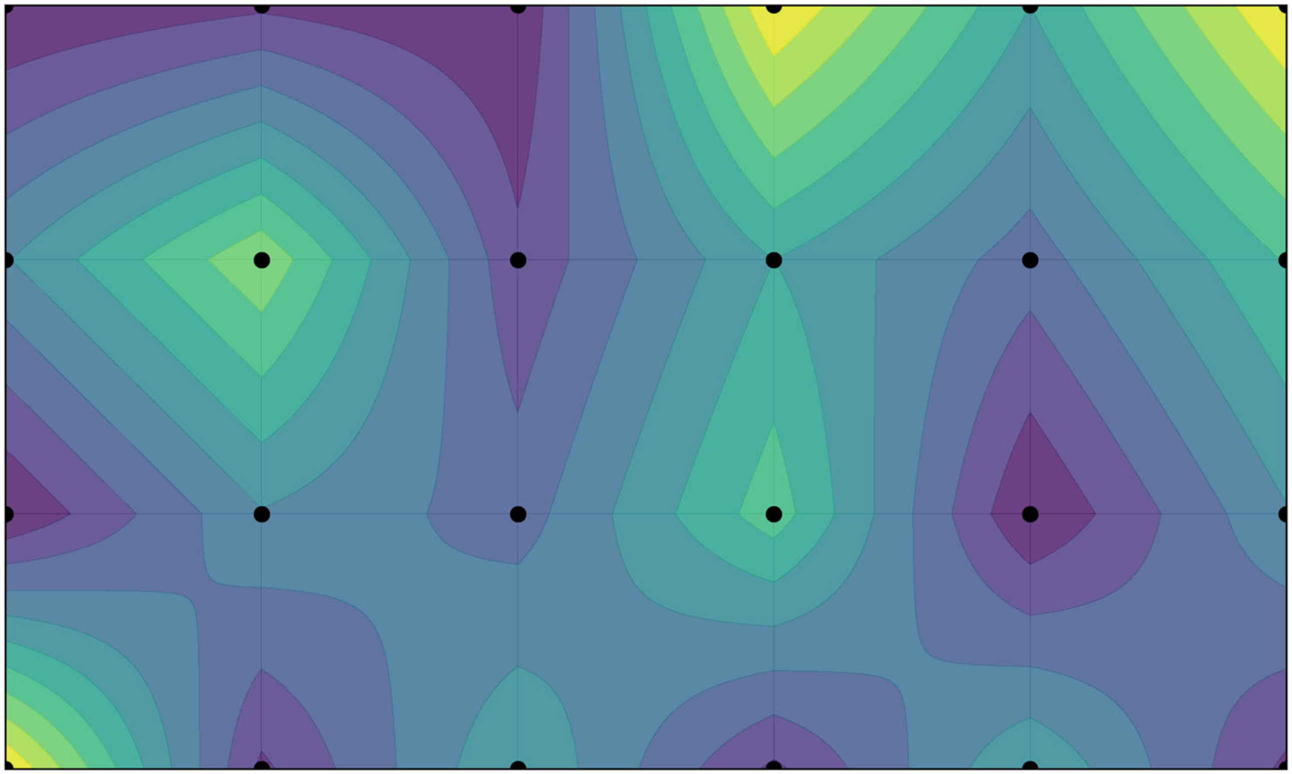
linear (diagonal 1)

Bi-Linear Interpolation: Comparisons



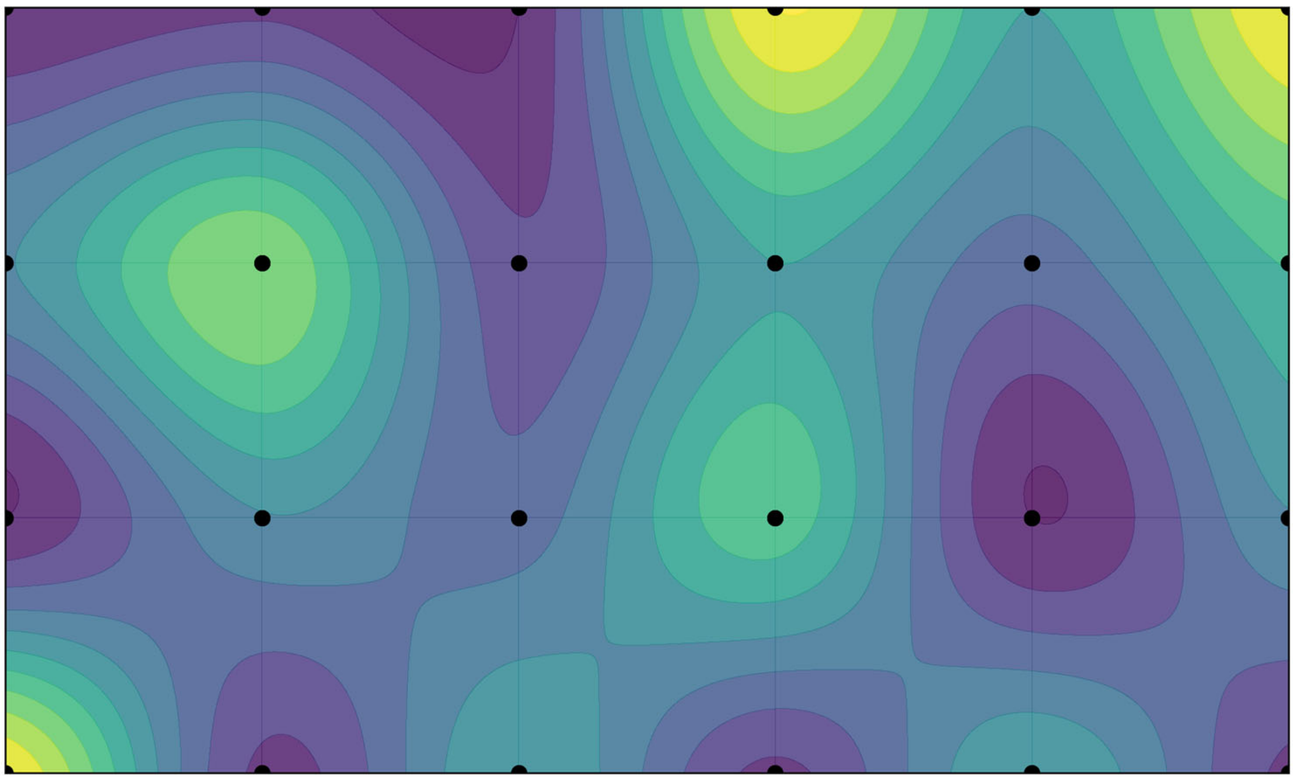
linear (diagonal 2)

Bi-Linear Interpolation: Comparisons



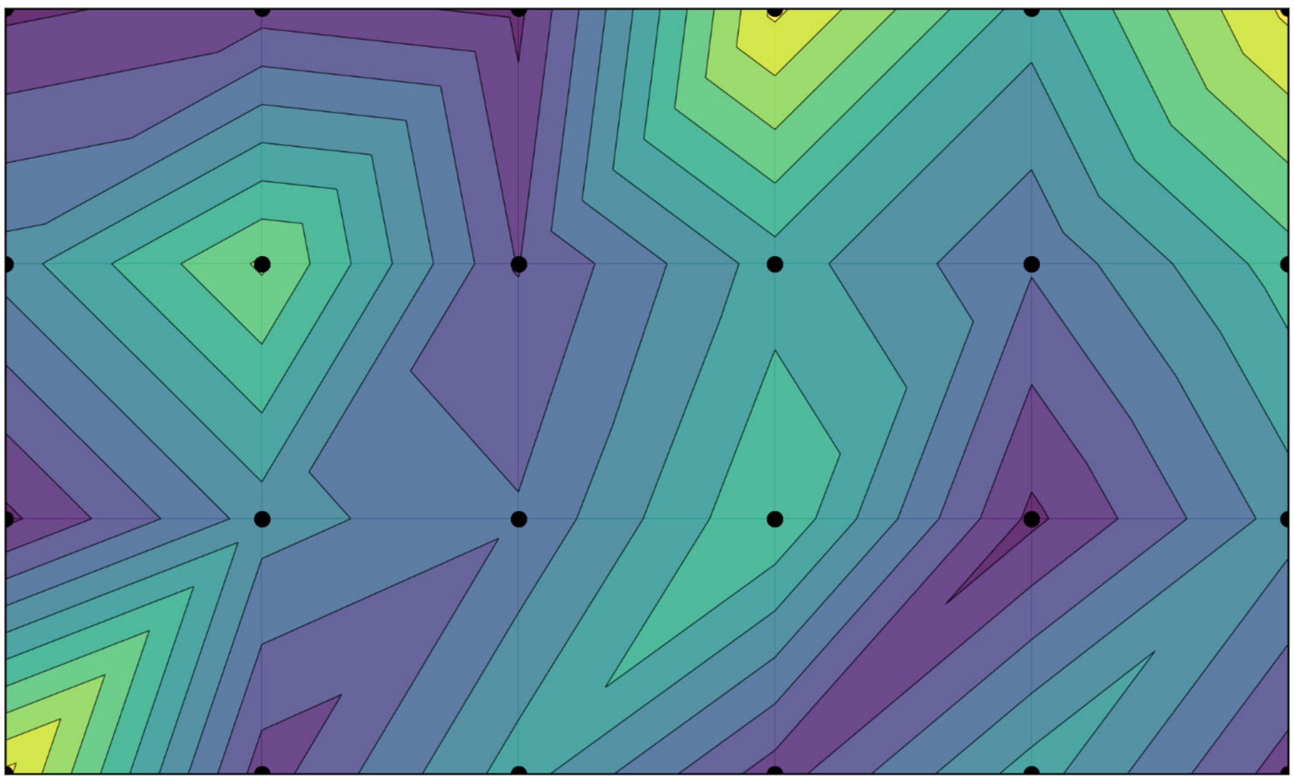
bi-linear (in 3D: tri-linear)

Bi-Linear Interpolation: Comparisons



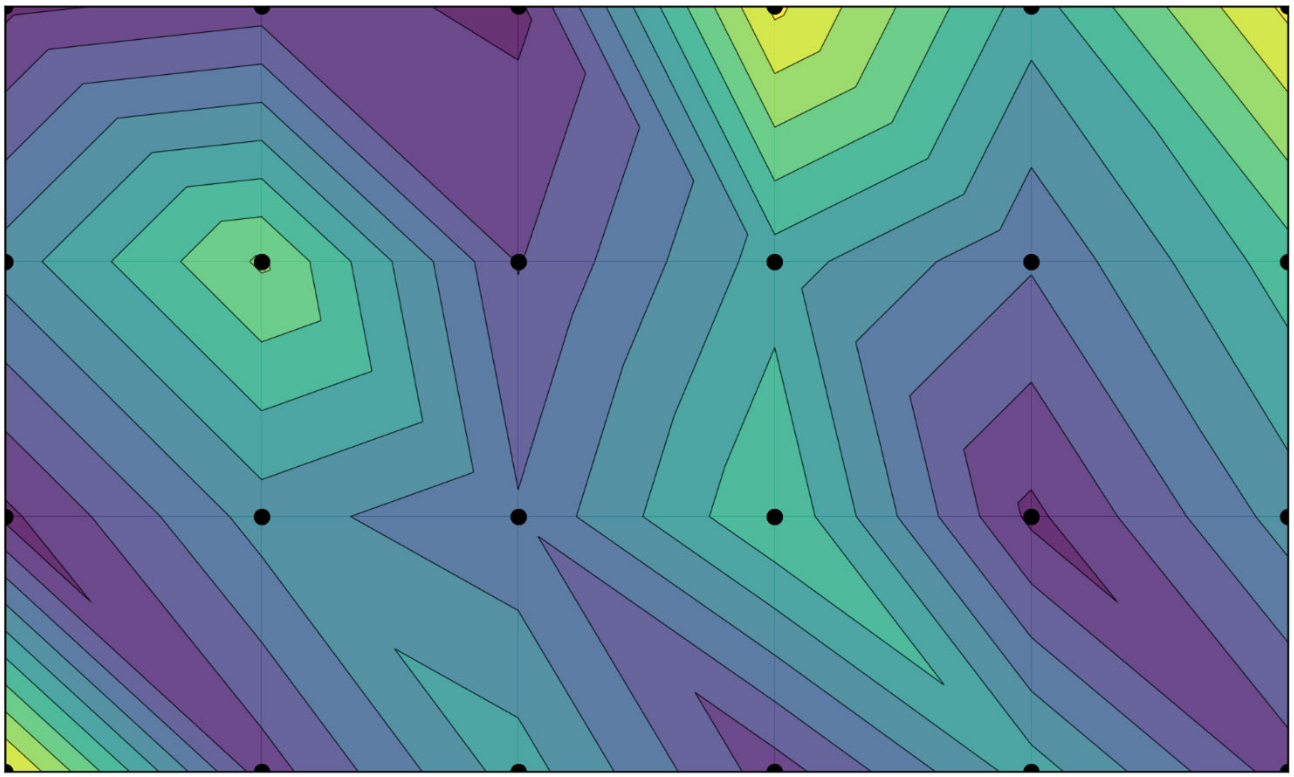
bi-cubic (in 3D: tri-cubic)

Bi-Linear Interpolation: Comparisons



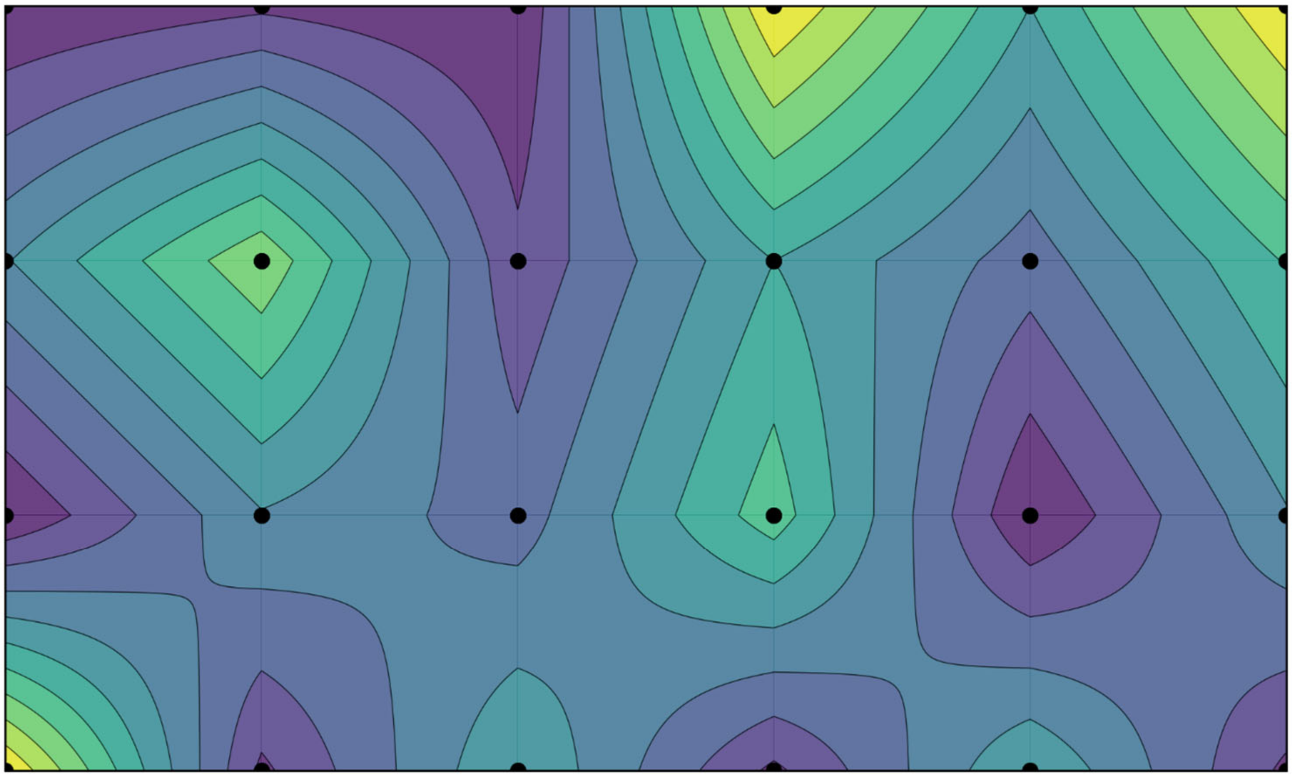
linear (diagonal 1)

Bi-Linear Interpolation: Comparisons



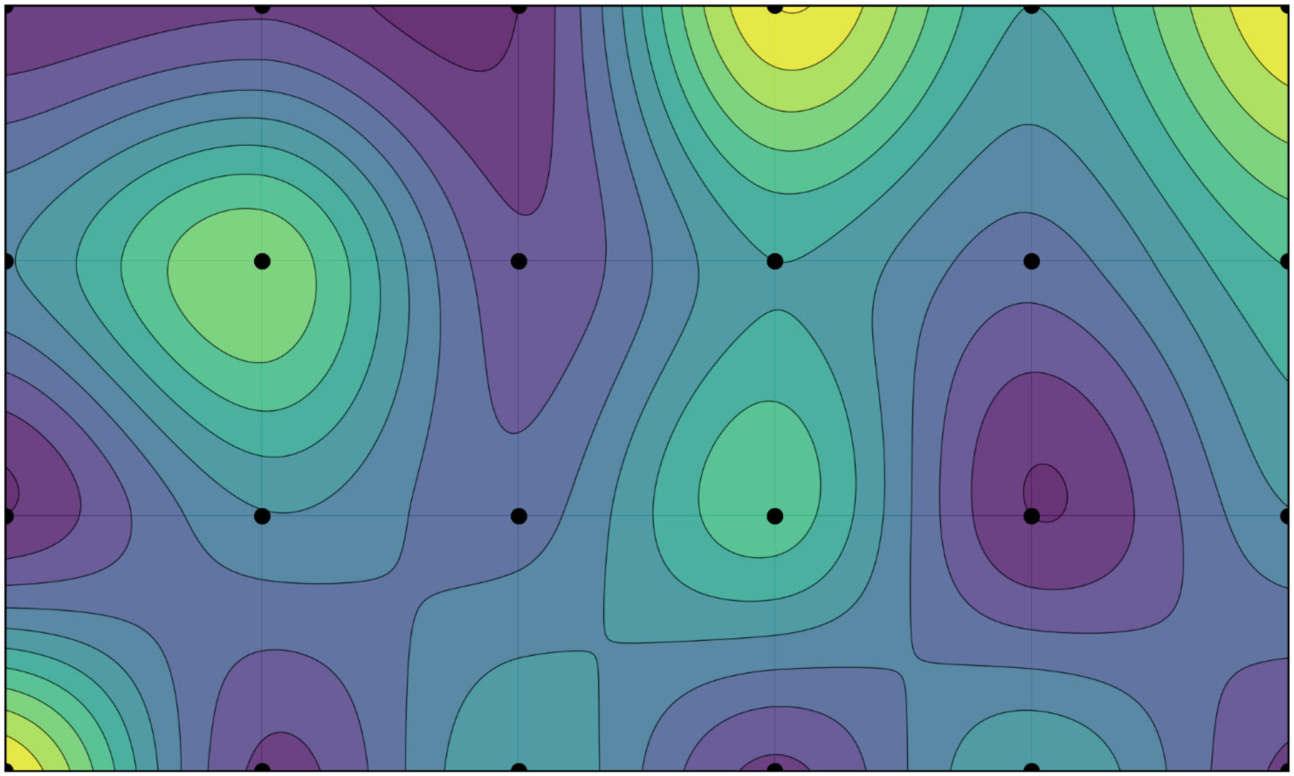
linear (diagonal 2)

Bi-Linear Interpolation: Comparisons



bi-linear (in 3D: tri-linear)

Bi-Linear Interpolation: Comparisons



bi-cubic (in 3D: tri-cubic)

Thank you.

Thanks for material

- Helwig Hauser
- Eduard Gröller
- Daniel Weiskopf
- Torsten Möller
- Ronny Peikert
- Philipp Muigg
- Christof Rezk-Salama