Radial Basis Function Interpolation

Rahul Narkhede

02.09.2024

Task

Aims

- Write program to perform RBF interpolation of surface using x, y, z data.
- Apply this program to identify trends in quality of interpolation by varying input parameters.

Program Input

- x, y, z data of surface ($N = 40000$ points).
- Number of basis functions (n_x) in one direction. n_y to be determined from aspect ratio. Center points of basis functions are homogeneously distributed over area.
- Width parameter σ of the Gaussian kernel, considered constant for all kernels.

$$G(x, y, z) = \frac{1}{2\pi\sigma^2} exp\Big(- \frac{x^2 + y^2}{2\sigma^2} \Big) \tag{1}$$

Program Output

- Interpolated surface.
- Residual

RBF Interpolation

RBF Interpolant

Given the center points c_i the RBF interpolant $s(x, y)$ for the surface can be expressed as:

$$s(x, y) = \sum_{j=1}^{M} \lambda_j G(\|(x, y) - c_j\|, \sigma) \tag{2}$$

where λ_j are the coefficients to be determined, $M = n_x n_y$ is the number of centers, and $G(\cdot)$ is the Gaussian kernel.

System of Linear Equations

To find the coefficients λ_j, solve $s(x_i, y_i) = z_i$ for each data point $i \in 1, 2, ..., N$, leading to system of linear equations $\mathrm{A}\lambda = \mathrm{z}$. Here,

- $\mathrm{A} \in \mathbb{R}^{N \times M}$ is the interpolation matrix with entries $A_{ij} = G(\|(x_i, y_i) - c_j\|, \sigma)$ and $j \in \{1, 2, ..., M\}$,
- $\lambda \in \mathbb{R}^M$ is the vector of coefficients,
- $\mathrm{z} \in \mathbb{R}^N$ is the vector of the (z_j) values.

Solve System

Solve this linear system for λ using exact solution of Least Squares Interpolation problem: $\lambda = (\mathrm{A}^T\mathrm{A})^{-1}\mathrm{A}^T\mathrm{z}$. Using coefficients λ_j, the interpolated surface can be evaluated at any point (x, y) using equation 2

Centers of Basis Functions

Figure 1 shows centers selected over the area using the following three methods to obtain centers of basis functions.

- **Grid Sub-sampling**: Form equal-sized grids on the area and select points closest to the center of each grid as basis function centers.
- **KMeans Clustering**: Use KMeans Clustering to determine the center points from the given data.
- **Uniform Grid**: Form a gird of equidistant points on the area and use them as basis function centers. These grid points do not necessarily coincide with the points from the given data.

Grid sub-sampling has been used in this work since the points in the given data are originally uniformly distributed over the area.

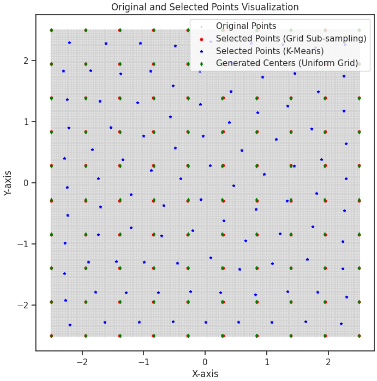

Figure 1: Selected centers ($M = 100$) using the three methods.

Trends in Quality of Interpolation

Design of Experiment

Different interpolations are performed by varying the number of basis functions in the x-axis (n_x) and the width parameter of the Gaussian kernels σ. The levels of these two parameters are as follows:

- $n_x \in \{10, 20, 30, 40, 50, 60\}$, corresponding to $M \in \{100, 400, 900, 1600, 2500, 3600\}$.
- $\sigma \in \{0.01, 0.025, 0.05, 0.1, 0.25, 0.5\}$

With a full-factorial design, this leads to 36 combinations.

Use of GPU for Computations

Using Python's `TensorFlow` library, the computations are also performed on GPU and the results are compared with CPU-based computations which use the `NumPy` library. LU decomposition has been applied for computing the inverse of $\mathrm{A}^T\mathrm{A}$ in both CPU- and GPU-based computations.

Metrics for Comparison

Three indicators are considered:

- Computation time in seconds for comparing GPU- and CPU-based computations.
- MSE between the original z values and the interpolated $s(x_i, y_i)$ values for all points of the given data.
- Condition number of the interpolation matrix A

CPU vs GPU Computations

As shown in figure 2, the computation using Python's `TensorFlow` library offers faster computations. GPU-based computations also offer a lower MSE in comparison to CPU-based computations as shown in figure 3. Thus, GPU-based computations are used for the interpolation of surfaces.

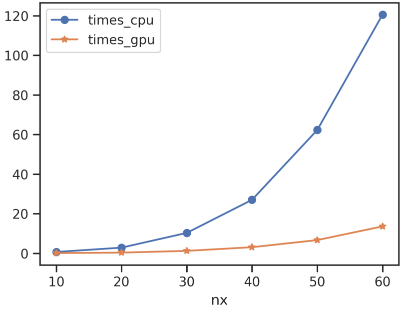

Figure 2: Comparison of CPU and GPU times (in seconds). The plot shows mean values of times over all σ values.

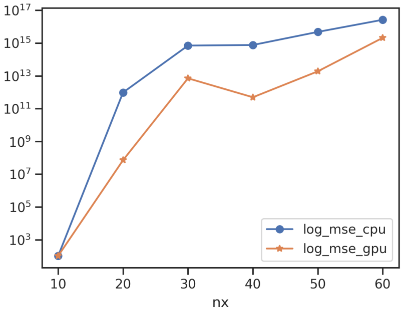

Figure 3: Comparison of CPU and GPU MSE. The plot shows mean values of times over all σ values.

MSE and Condition Number of the 36 Experiments

The MSE and condition numbers for all experiments are visualized using heatmaps as shown in figures 4 and 5. The rows represent n_x and columns represent σ values.

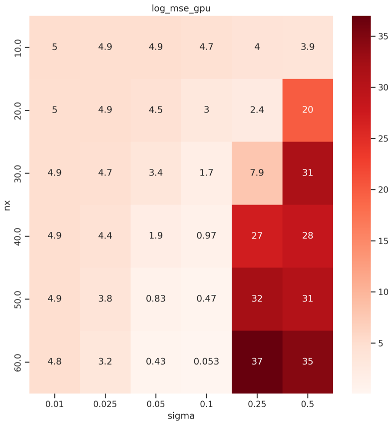

Figure 4: Heatmap of the logarithm of MSE for all experiments.

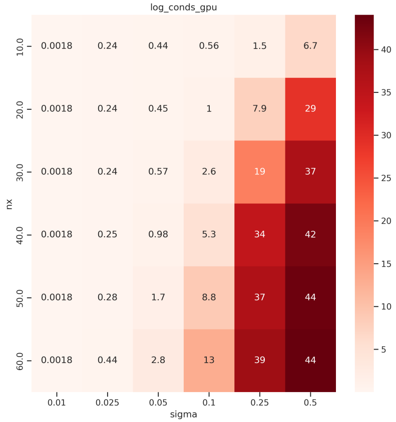

Figure 5: Heatmap of the logarithm of condition number of the matrix A for all experiments.

Trends in Quality of Interpolation

Observations

From the heatmaps in figures 4 and 5 and plots in figure 6, the following observations can be made:

- The MSE decreases as σ and n_x increase. However, the high condition number of matrix A for $\sigma \in \{0.25, 0.5\}$ leads to large deviations in the interpolated values and a high MSE.
- With the increase of either σ or n_x, the condition number increases super-exponentially.
- Interestingly, for $(n_x = 60, \sigma = 0.1)$, which results in the lowest MSE, the condition number of matrix A is high, in the order of 10^{13}.
- Thus, an increase in the condition number does not necessarily lead to a bad quality of interpolation. Beyond a certain order of the condition number, the MSE increases abruptly, indicating numerical instabilities.

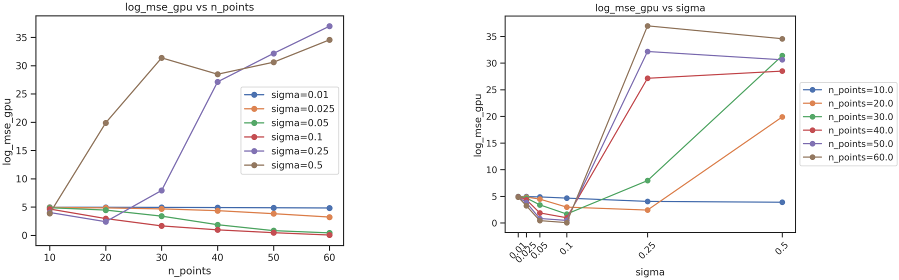

Figure 6: Effect of n_x and σ on logarithm of MSE.

Correlation between n_x, σ, condition number and MSE

Figure 7 shows the correlation heatmap between n_x, σ and logarithms of MSE and condition number.

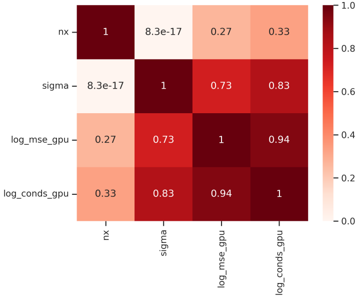

Figure 7: Heatmap showing correlation of parameters, MSE and condition number.

- The correlation heatmap shows that σ has a stronger influence on the MSE and condition number of the matrix A, as compared to n_x.
- The condition number of the matrix A is strongly correlated to the MSE.

Final Output

Using $n_x = 60$ and $\sigma = 0.1$, RBF interpolation is performed for surfaces 1 and 2 using the given data, leading to a MSE of 1.05 and 35.58 respectively. Higher MSE for surface 2 is expected due to its higher noise, as can be noticed from figures 8 and 9. The surfaces are plotted by interpolating the obtained function $s(x, y)$ over a meshgrid of $(200, 200)$ points.

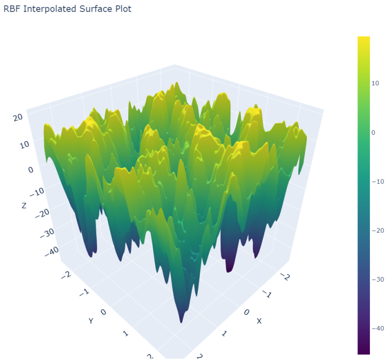

Figure 8: Interpolated surface 1

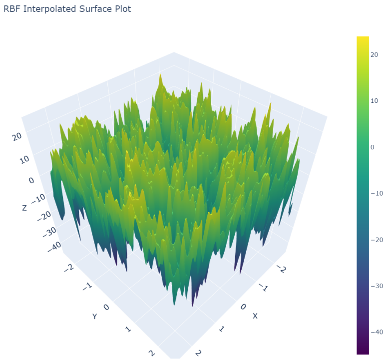

Figure 9: Interpolated surface 2

Conclusion

- In general, the quality of interpolation increases as the width parameter σ and the number of radial basis functions n_x increases.
- The condition number of the interpolation matrix A poses a limitation in obtaining the best quality of interpolation. The condition number grows super-exponentially with an increase in n_x and σ.
- A high condition number up to a certain threshold can still lead to a good quality of interpolation, as observed in the case of $(n_x = 60, \sigma = 0.1)$.

Alternative approaches as future scope

- There are some proposed algorithms to overcome the ill-conditioning of the linear system, e.g. Multilevel and Multipole methods.
- Computational limitations for interpolating a large number of points are addressed by algorithms using parallel computing, e.g. PetRBF by Yokota et al.
- RBF Networks provide a promising approach to interpolate the surface with a large number of points. Allowing the network to learn the σ of the basis functions can lead to different Gaussian kernels with individual width parameters, enabling a better quality of interpolation.

  I attempted using an RBF Network to interpolate surface 1 and could obtain an MSE close to 6 with the Adam optimizer and a learning rate scheduler. The training is not yet perfect and the learning rate and possibly the optimizer need to be changed for better results.