LaplaceInterpolation.jl: A Julia package for fast interpolation on a grid

Vishwas Rao[1], **Charlotte L. Haley**[1], **and Mihai Anitescu**[1,2]

1 Argonne National Laboratory **2** University of Chicago

DOI: 10.21105/joss.03766

Software

- Review
- Repository
- Archive

Editor: Viviane Pons

Reviewers:

- @wkearn
- @eviatarbach

Submitted: 09 September 2021
Published: 27 September 2021

Summary

We implement a linear-time algorithm for interpolation on a regular multidimensional grid in the Julia language. The algorithm is an approximate Laplace interpolation (Press, 1992) when no parameters are given; and when parameters $m \in \mathbb{Z}$ and $\epsilon > 0$ are set, the interpolant approximates a Matérn kernel, of which radial basis functions and polyharmonic splines are a special case. We implement, in addition, Neumann, Dirichlet (trivial), and average boundary conditions with potentially different aspect ratios in the different dimensions. The interpolant functions in arbitrary dimensions.

Mathematical Background

Radial basis functions and splines can be unified conceptually through the notion of Green's functions and eigenfunction expansions (Fasshauer, 2012). The general multivariate Matérn kernels are of the form

$$K(\mathbf{x};\mathbf{z}) = K_{m-d/2}(\epsilon||\mathbf{x}-\mathbf{z}||)(\epsilon||\mathbf{x}-\mathbf{z}||)^{m-d/2}$$

for $m > d/2$, where K is the modified Bessel function of the second kind with parameter ν and can be obtained as Green's kernels of

$$L = (\epsilon^2 I - \Delta)^m,$$

where Δ denotes the Laplacian operator in d dimensions. Polyharmonic splines, including thin plate splines, are a special case of the above, and this class includes the thin plate splines.

The discrete gridded interpolation seeks to find an interpolation $u(\mathbf{x})$ that satisfies the differential operator in d dimensions on the nodes $\mathbf{x}_i$ where there is no data and equals y_i everywhere else. Discretely, one solves the matrix problem

$$\mathbf{C}(\mathbf{u}-\mathbf{y}) - (1-\mathbf{C})L\mathbf{u} = 0,$$

where $\mathbf{y}$ contains the y_i's and placeholders where there is no data, L denotes the discrete matrix operator, and C is a diagonal matrix that indicates whether node $\mathbf{x}_i$ is observed.

In $d-$ dimensions the matrix $A^{(d)}$ of size $M \times M$ expands the first-order finite difference curvature, and its (i,j)th entry is -1 when node j is in the set of neighbors of the node $\mathbf{x}_i$ and has the number of such neighbors on the diagonal. Note that if node i is a boundary node, the ith row of $A^{(d)}$ has -1s in the neighboring node spots and the number of such nodes on the diagonal. In general, the rows of $A^{(d)}$ sum to zero.

Denote by $L = A^{(d)}$ the discrete analog of the Laplacian operator. To use the Matern operator, one substitutes

$$L = B^{(d)}(m, \epsilon) = ((A^{(d)})^m - \epsilon^2 I).$$

Importantly, A is sparse, containing at most 5 nonzero entries per row when $d = 2$ and 7 nonzero entries per row when $d = 3$ and so on. The Matérn matrix $B^{(d)}(m, \epsilon)$ is also sparse, having $2(m + d) - 1$ nonzero entries per row. The sparsity of the matrix allows for the interpolation to solve in linear time.

Statement of Need

While numerous implementations of interpolation routines exist that fill missing data points on arbitrary grids, these are largely restricted to one and two dimensions and are slow to run. The implementation we propose is dimension-agnostic, based on a linear-time algorithm, and implements an approximate Matérn kernel interpolation (of which thin plate splines, polyharmonic splines, and radial basis functions are a special case).

Why Is It So Fast?

The implementation is fast because the problem largely boils down to the solution of $Ax = b$ (Mainberger et al., 2011), where the square matrix A's size is the product of the number of points in each of the dimensions and is dense. For the special case where the data points are on a regular grid and the Matérn kernel interpolant is used, a remarkable simplification occurs in which a discrete approximation to the Green's function for the operator results in an interpolant having sparse matrix representation.

Other Software for Interpolation

As of the time of this writing, related software includes the following:

Julia

- Interpolations.jl, which does B-splines and Lanczos interpolation and has support for irregular grids
- Dieerckx.jl, a Julia-wrapped Fortran package for 1-D and 2-D splines
- GridInterpolations.jl
- Laplacians.jl, whose function `harmonic_interp` is similar to our vanilla implementation

Python

- astropy.convolve, which will interpolate gridded data by rescaling a convoution kernel when it encounters missing values
- scipy.interpolate.RBF

Python Wrapper

- gridinterppy, which serves as a Python wrapperr for LaplaceInterpolation.jl

Acknowledgments

The authors would like to thank Ray Osborn, Matt Krogstad, and Stefan Rosenkranz for their input on this work.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, under contract DE-AC02-06CH11357.

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. http://energy.gov/downloads/doe-public-access-plan

References

Fasshauer, G. E. (2012). Green's functions: Taking another look at kernel approximation, radial basis functions, and splines. In *Approximation theory XIII: San Antonio 2010* (pp. 37–63). Springer.

Mainberger, M., Hoffmann, S., Weickert, J., Tang, C. H., Johannsen, D., Neumann, F., & Doerr, B. (2011). Optimising spatial and tonal data for homogeneous diffusion inpainting. *International Conference on Scale Space and Variational Methods in Computer Vision*, 26–37.

Press, W. H. (1992). *Numerical recipes in Fortran 77: The art of scientific computing* (3rd ed.). Cambridge University Press.