ARBTools is a Python library containing a Lekien-Marsden type tricubic spline method for interpolating three-dimensional scalar or vector fields presented as a set of discrete data points on a regular cuboid grid. ARBTools was developed for simulations of magnetic molecular traps, in which the magnitude, gradient and vector components of a magnetic field are required. Numerical integrators for solving particle trajectories are included, but the core interpolator can be used for any scalar or vector field. The only additional system requirements are NumPy.

Keywords: Python; three-dimensional interpolation; spline; vector field; scalar field; smoothing

Funding statement: This software was developed as part of research funded by EPSRC grant number EP/N509462/1.
Although produced for the specific application of modelling low-field seeking neutral particles, this software has been developed to be more general. It can work directly with either a scalar or vector field input, and is suitable for a variety of applications with any field supplied across a regular, cuboid grid.

**Implementation and architecture**

ARBTools is written in Python [8], with extensive use of NumPy [9]. 'ARBInterp' contains the tricubic interpolator and query methods. Any source data presented across a regular grid as either a scalar or vector field can be input into the interpolator, allowing the values of the data to be calculated for arbitrary points within the set. For scalar data the derivatives are directly accessible. Although designed for magnetic fields this software could be used with a wide variety of systems, such as modelling heat flow, or in data processing to smooth contour plots or heatmaps. Example magnetic fields, as both magnitudes and vectors, are available to download along with scripts illustrating the use of the interpolator. These example files also illustrate the expected input format of the data.

The 'ARBTraj' module contains functions to create a random sample of argon atoms and solve its motion through a quadrupole field. By simply changing mass and magnetic moment values this can be adapted for different atomic species, or a differently shaped magnetic potential could be specified. The included Runge-Kutta integrator can be easily modified to solve particle motion in alternative systems – for example, we have recently discussed simulating the operation of an atomic tweezer apparatus with a colleague.

**Installation**

To install on Linux run `sudo python setup.py install`. The interpolator is contained in a file called 'ARBInterp.py' and the command 'from ARBTools.ARBInterp import tricubic' will import the interpolation class.

**Usage**

To instantiate the class, pass it a source field – e.g. `interp = tricubic (sourcefield)` will create an instance called `interp`. Input can be either a scalar field `U(x, y, z)` as an `N x 4` (x, y, z, U) array or a vector field `B(x, y, z)` as an `N x 6` (x, y, z, Bx, By, Bz) array. If an `N x 4` field is passed, the interpolator will automatically default to return the magnitude and gradient of the field. If an `N x 6` field is passed it will accept an optional 'mode' keyword argument to select one of three modes, (e.g. `interp = tricubic (sourcefield, mode = ‘kw’)):

- Norm: takes the norm of the vector field and return the magnitude and gradient (as three partial derivatives)
- Vector: returns the interpolated vector components
- Both: takes vector norm, and returns the magnitude and norm of the vector plus the vector components at the interpolation point

If no keyword is passed, the interpolator defaults to vector mode. Two query modes are implemented: 'sQuery' interpolates a single point within the volume, accepting an input in the form `[x, y, z]`. 'rQuery' accepts a range of coordinates for simultaneous interpolation, as an array `[x1, y1, z1]…[xn, yn, zn]`. For multiple queries rQuery is much more efficient than running sQuery in a loop.

**Quality control**

ARBTools was written with Python 2.7.12 and NumPy 1.13.3 on Linux Mint 18.3, and has been tested with Python 3.5.2 on the same platform. It has also been tested on Enthought Canopy v2.1.9 on Microsoft Windows 7 and 10.

Example input fields and query scripts are available to download from the source repository. Performance benchmarking on 64-bit Linux with an Intel Core i7 CPU shows 100 unique interpolations for a given data set take between 20 and 50 ms, depending on which components are being returned. As expected, there is a linear relationship between number of queries and run time.

---

Figure 1: A quadrupole electric field, left, 400 x 400 pixel analytic solution, centre, 40 x 40 pixel exported subset, right, 400 x 400 pixel interpolation of the subset.
The main constraint when using ARBTools is the amount of memory required to load the source file; this is determined by the size of the input grid. For example; a cubic volume 20 mm on a side with a grid spacing of 0.5 mm contains \(41^3 = 68921\) grid points, which will load in less than a second with negligible memory usage. The same data sampled at 0.25 mm intervals contains 531441 points, this may take several seconds to load and consume \(\approx 500\) MB memory. At 0.125 mm intervals we have 4173281 points, this may take up to a minute to load and consume over 5 GB of memory. Once loaded, however, querying these different datasets takes almost exactly the same amount of time.

The tricubic interpolation method values smoothness of the interpolated function and its first derivatives over absolute accuracy [4]. In order to quantify the errors in this method two types of model were considered; the quadrupole electric field produced by a series of point charges, which can be solved analytically (Figure 1), and a magnetic field produced by a permanent magnet, calculated using finite-element analysis with the ‘FEMM’ [11] software package (see Figure 2). (Of course, if an analytic solution is available there is no need to interpolate – this is simply a useful calibration tool!).

For both cases a high-resolution source dataset was created, and then a sparse subset of this data was interpolated and compared with it. Figure 3 shows the root-mean-squared errors between the interpolated and ‘true’ values of the calculated fields for a variety of grid intervals. It can be seen that for a given level of accuracy the analytic solution can tolerate a larger grid spacing – this is due to the high gradients at the interface between two materials in finite-element (or boundary volume integral [12]) analysis. In general, consideration

Figure 2: The magnitude of the magnetic field around a ring magnet, left, 400 × 400 pixel finite-element analysis model, centre, 40 × 40 pixel exported subset, right, 400 × 400 pixel interpolation of the subset.

Figure 3: Root-mean-square error in interpolated data as compared to ‘true’ values from either a finite-element analysis model or an analytic solution.
of the nature of the data set being interpolated and its structure will inform the grid spacing chosen, which is a compromise between inaccuracy and unwieldiness. These tests were repeated with the interpolator in the ‘EQ Tools’ library [13]; although it does not provide the field derivatives, the magnitudes were found to be the same to within $1 \times 10^{-6}$.

(2) Availability

Operating system
ARBTools was developed on Linux Mint 18.3, and has been tested on Windows 7 and 10.

Programming language
ARBTools was developed in Python 2.7.12 and has been tested on 3.5.2. Any version of Python from 2.7 upwards should be suitable.

Additional system requirements
Several GB of RAM should be suitable for most applications. ARBTools has been used with large datasets on the Durham university supercomputer.

Dependencies
Written using NumPy 1.13.3. Earlier versions may work.

Software location
Name: ARBTools
Persistent identifier: https://doi.org/10.5281/zenodo.2548609
Publisher: Paul A. Walker
Version published: v1.3
Repository: GitHub
Persistent identifier: https://github.com/DurhamDecLab/ARBInterp
Licence: GPL-3.0
Date published: 15/02/2019

Language
English

(3) Reuse potential

The core of ARBTools is the tricubic interpolator, which can be used with any suitably-formatted input field, for many possible tasks – for example, visualising the shape of a three-dimensional potential or extracting coherent Lagrangian structures from a time-dependent two-dimensional flow. The interpolator has been designed to be imported as a library in Python, and the output from the evaluation methods can easily be passed into third-party code, or output to file for use in non-Python systems.

As is, ARBTools can be used to model the trajectories of low-field-seeking argon atoms in a magnetic field. Simply altering the mass and magnetic moment parameters would allow other species to be modelled. If the functions defining the acceleration due to a potential are replaced, trajectories in alternative systems could easily be modelled, for example, the motion of charges in electric fields, or masses moving under gravity.

Support may be requested through the project GitHub page: (https://github.com/DurhamDecLab/ARBInterp). The source code is available and may be reused or modified at will subject to the details of the GPL-3.0 licence.

Acknowledgements

Many thanks to Dr. Lewis McArd for his invaluable advice on this and other projects.

Competing Interests

The authors have no competing interests to declare.

References
