The Journal of Open Source Software

SparseGridsKit.jl: A Julia sparse grids approximation implementation

Benjamin M. Kent [1*¶]

1 CNR-IMATI, Pavia, Italy ¶ Corresponding author * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Approximation of functions with high dimensional domains is an important tool for modern scientific and engineering modelling. Surrogate models are often constructed to give computationally cheap yet accurate approximations that can be used in applications such as uncertainty quantification, optimisation, parameter estimation. Surrogates may use global polynomial approximation and a common approach is the use of *sparse grid* approximation techniques. In particular, sparse grid polynomial interpolation techniques allow a practitioner to approximate solutions to parametric formulations of such problems in a non-intrusive way using existing numerical solvers.

`SparseGridsKit.jl` provides tools to manually and adaptively construct sparse grid polynomial approximations. Interpolation and quadrature routines allow evaluation and integration of the surrogate models. Multi-fidelity approximation via the multi-index stochastic collocation algorithm is also possible. Approximations can be represented either in a basis of Lagrange interpolation polynomials or in a basis of spectral-type polynomials.

Statement of need

Surrogate modelling is an important theme in many science and engineering applications. In particular, sparse grid approximation is a well developed methodology and is featured in many survey articles and textbook chapters, e.g. Cohen & DeVore (2015). The need for sparse grid surrogate modelling is demonstrated by its use in many applications, from simpler elliptic and parabolic PDEs to complex practical engineering problems e.g. Piazzola et al. (2022). The `SparseGridsKit.jl` implementation offers a rich set of features to enable this.

Specifically, a Julia implementation of sparse grid approximation methods is offered by `SparseGridsKit.jl`. This uniquely offers - native algorithm implementation in Julia to give high performance and simple programming, - dynamical typing, allowing surrogate models mapping parameters $\vec{y}$ to any type offering vector space operations. Existing sparse grid approximation packages in Julia include `Tasmanian.jl`, wrapping the Tasmanian library, `AdaptiveSparseGrids.jl` and `DistributedSparseGrids.jl`. `SparseGridsKit.jl` offers a more complete set of functionality, with close resemblance to the popular `Sparse Grids MATLAB Kit` (Piazzola & Tamellini, 2024).

Other software packages implementing sparse grid approximation include: - `Sparse Grids MATLAB Kit` A MATLAB package on which the `SparseGridsKit.jl` is loosely based (Piazzola & Tamellini, 2024), - `spinterp` A MATLAB toolbox for sparse grid interpolation (Klimke & Wohlmuth, 2005), - `Dakota` A C++ library for optimisation and surrogate modelling (B. M. Adams & Winokur, 2024). - `PyApprox` A Python package for high-dimensional approximation (Jakeman, 2023), `SparseGridsKit.jl` offers specific toolkit with minimal complexity for fast algorithm development in Julia.

SparseGridsKit.jl

Knots, Quadrature and Interpolation

Interpolation and quadrature methods are built out of one-dimensional knot functions which return knots x and corresponding quadrature weights w. Sparse grid constructions require sequences of approximation operators. In this interpolation setting, we consider a sequence of interpolation operators I^α indexed by $\alpha \in \mathbb{N}$ which use sequentially growing sets of interpolation points. This is achieved using the abstract `Points` and `Level` types in `SparseGridsKit.jl`.

Sparse grid methods extend one dimensional rules to many dimensions. This originates in the work of Smolyak (Smolyak, 1963) and developed for interpolation, quadrature and parametric PDEs (Thomas Gerstner & Griebel, 1998),(Novak & Ritter, 1999),(Barthelmann et al., 2000),(Babuska et al., 2004),(Xiu & Hesthaven, 2005),(Nobile et al., 2008). In particular, we use the combination technique formulation in which the approximation is a cleverly chosen linear combination of tensor product interpolation operators. The sparse grid interpolation operator is

$$I = \sum_{\underline{\alpha} \in A} c_{\alpha} \bigotimes_{i=1}^{n} I^{\alpha_i}$$

where $c_{\underline{\alpha}}$ is the combination technique coefficient and A is a set of multi-indices defining the approximation.. For further details we refer to the vast literature, e.g. (Piazzola et al., 2021). Construction of an approximation using the sparse grid interpolation operator requires evaluation of the target function at a set of collocation points Z which is implicitly defined. Nesting of the underlying one-dimensional interpolation rules means that many grid points are coincident hence the *sparse* nature of the grid.

Surrogate Models

For a sparse grid approximation with corresponding function evaluations $\{f(z)\}_{z\in Z}$, we can apply interpolation to approximate the function values at non-interpolation points. This is implemented by the `interpolate_on_sparsegrid` function. A `SparseGridApproximation` structure wraps the sparse grid approximation operator and a set of function evaluations which can be treated as a function to evaluate the approximation. Similarly, the surrogate model can be integrated using the `integrate_on_sparsegrid` function.

Adaptive Sparse Grids

A sparse grid can be constructed by using a user specified multi-index set. Generally, the underlying function is unknown and we wish to adaptively construct the approximation space to capture the target function behaviour. This is often achieved in a greedy iterative manner. Adaptive sparse grid approximation is implemented as `adaptive_sparsegrid`. This is based on the ubiquitous Gerstner-Griebel dimensional adaptive algorithm (T. Gerstner & Griebel, 2003).

Other functionality

`SparseGridsKit.jl` includes functionality for multi-fidelity approximation via the multi-index stochastic collocation algorithm and limited support for differentiation via automatic differentiation.

The functionality described above is all tested and documented with examples.

Acknowledgements

References

B. M. Adams, K. R. D., W. J. Bohnhoff, & Winokur, J. G. (2024). *Dakota 6.21.0 documentation. Technical report SAND2024-15492O*. Sandia National Laboratories, Albuquerque, NM. http://snl-dakota.github.io

Babuska, I., Tempone, R., & Zouraris, G. E. (2004). Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM Journal on Numerical Analysis*, *42*(2), 800–825. https://doi.org/10.1137/s0036142902418680

Barthelmann, V., Novak, E., & Ritter, K. (2000). High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics*, *12*(4), 273–288. https://doi.org/10.1023/a:1018977404843

Bungartz, H.-J., & Griebel, M. (2004). Sparse grids. *Acta Numerica*, *13*, 147–269. https://doi.org/10.1017/s0962492904000182

Cohen, A., & DeVore, R. (2015). Approximation of high-dimensional parametric PDEs. *Acta Numerica*, *24*, 1–159. https://doi.org/10.1017/s0962492915000033

Gerstner, Thomas, & Griebel, M. (1998). Numerical integration using sparse grids. *Numerical Algorithms*, *18*(3), 209–232. https://doi.org/10.1023/A:1019129717644

Gerstner, T., & Griebel, M. (2003). Dimension–adaptive tensor–product quadrature. *Computing*, *71*(1), 65–87. https://doi.org/10.1007/s00607-003-0015-5

Jakeman, J. D. (2023). PyApprox: A software package for sensitivity analysis, bayesian inference, optimal experimental design, and multi-fidelity uncertainty quantification and surrogate modeling. *Environmental Modelling & Software*, *170*, 105825. https://doi.org/10.1016/j.envsoft.2023.105825

Klimke, A., & Wohlmuth, B. (2005). Algorithm 847: Spinterp: Piecewise multilinear hierarchical sparse grid interpolation in MATLAB. *ACM Transactions on Mathematical Software*, *31*(4), 561–579. https://doi.org/10.1145/1114268.1114275

Li, Y., Zoccarato, C., Piazzola, C., Bru, G., Tamellini, L., Guardiola-Albert, C., & Teatini, P. (2024). *Characterizing aquifer properties through a sparse grid-based bayesian framework and InSAR measurements: A basin-scale application to alto guadalentín, spain*. https://doi.org/10.22541/essoar.172373105.53381390/v1

Nobile, F., Tempone, R., & Webster, C. G. (2008). A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, *46*(5), 2309–2345. https://doi.org/10.1137/060663660

Novak, E., & Ritter, K. (1999). Simple cubature formulas with high polynomial exactness. *Constructive Approximation*, *15*(4), 499–522. https://doi.org/10.1007/s003659900119

Piazzola, C., & Tamellini, L. (2024). Algorithm 1040: The sparse grids matlab kit - a matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification. *ACM Transactions on Mathematical Software*, *50*(1), 1–22. https://doi.org/10.1145/3630023

Piazzola, C., Tamellini, L., Pellegrini, R., Broglia, R., Serani, A., & Diez, M. (2022). Comparing multi-index stochastic collocation and multi-fidelity stochastic radial basis functions for forward uncertainty quantification of ship resistance. *Engineering with Computers*, *39*(3), 2209–2237. https://doi.org/10.1007/s00366-021-01588-0

Piazzola, C., Tamellini, L., & Tempone, R. (2021). A note on tools for prediction under uncertainty and identifiability of SIR-like dynamical systems for epidemiology. *Mathematical

Biosciences, *332*, 108514. https://doi.org/10.1016/j.mbs.2020.108514

Schwab, C., & Gittelson, C. J. (2011). Sparse tensor discretizations of high-dimensional parametric and stochastic PDEs. *Acta Numerica*, *20*, 291–467. https://doi.org/10.1017/s0962492911000055

Smolyak, S. A. (1963). Quadrature and interpolation formulae on tensor products of certain function classes. *Soviet Math. Dokl.*, *4*(5), 240–243.

Xiu, D., & Hesthaven, J. S. (2005). High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, *27*(3), 1118–1139. https://doi.org/10.1137/040615201