Parafields is a Python package for the generation of stationary Gaussian random fields with well-defined, known statistical properties. The use of such fields is a key ingredient of simulation workflows that involve uncertain, spatially heterogeneous parameters. As such, Gaussian random fields play a dominant role in geostatistics, e.g., in the modeling of particulate matter concentration, temperature distributions and subsurface flow (Cameletti et al., 2013) (Sain et al., 2011) (Dodwell et al., 2015). Outside these traditional applications, Gaussian random fields are also used in biomedical imaging (Penny et al., 2005), material sciences (Torquato & Haslach Jr, 2002) or within Markov-Chain Monte-Carlo methods in Bayesian estimation (Scheichl et al., 2017).

Parafields is also able to run in parallel using the Message Passing Interface (MPI) standard through mpi4py (Dalcin & Fang, 2021). In this case, the computational domain is split and only the part of the random field relevant to a certain process is generated on that process. The generation process is implemented in a performance-oriented C++ backend library and exposed in Python though an intuitive Python interface.

Other packages for the generation of stationary Gaussian processes exist, e.g., the R package lgcp (Davies & Bryant, 2013), the Julia package GaussianRandomFields.jl (Robbe, 2023), and the Python package GSTools (Müller et al., 2022). In comparison with these alternative packages, parafields is specifically designed and adapted to the sampling of very large Gaussian

---

random fields within a HPC workflow. This was a major concern in the development of the backend and is among other things, reflected in the ability to create Gaussian processes in an MPI-distributed fashion.

**Implementation**

Parafields has over ten years of development history: it was first implemented as an extension to the Dune framework (Bastian et al., 2021) for the numerical solution of partial differential equations. This restricted the potential userbase to users of that software framework, although there was quite some interest in the software from outside this community. In 2022, we started a huge refactoring: the previous C++ code base (Klein, 2017) was rewritten to have a weaker dependency on Dune, which e.g. included a rewrite of the CMake build system (Klein & Kempf, 2022). In order to open up to a wider userbase, a Python interface written in pybind11 (Jakob et al., 2017) was added.

When engineering the Python package, we put special emphasis on the following usability aspects: installability, customizability and embedding into existing user workflows.

The recommended installation procedure for parafields is perfectly aligned with the state-of-the-art of the Python language: it is installable through pip and automatically compiles using the CMake build system of the project through scikit-build (Fillion-Robin et al., 2018). Required dependencies of the C++ library are automatically fetched and built in the required configuration. For sequential usage we also provide pre-compiled Python wheels. They are built against the sequential MPI stub library FakeMPI (Kempf & PetSc Developers, 2022), which allows us to build the sequential and the parallel version from the same code base. Users who want to leverage MPI through mpi4py will instead build the package from source against their system MPI library.

It was a goal of the design of the Python API to expose as much of the flexibility of the underlying C++ framework as possible. In order to do so, we use pybind11’s capabilities to pass Python callables to the C++ backend. This allows users to, e.g., implement custom covariance functions or use different random number generators. Furthermore, we acknowledge the fact that many Python users write scientific applications within Jupyter: our fields render nicely as images in Jupyter and field generation can optionally be configured through an interactive widget frontend within Jupyter.

**Acknowledgments**

The authors thank all contributors of the dune-randomfield project for their valuable contributions that are now part of the parafields-core library. Dominic Kempf is employed by the Scientific Software Center of Heidelberg University which is funded as part of the Excellence Strategy of the German Federal and State Governments. Ole Klein’s work is supported by the federal ministry of education and research of Germany (Bundesministerium für Bildung und Forschung) and the ministry of science, research and arts of the federal state of Baden-Württemberg (Ministerium für Wissenschaft, Forschung und Kunst Baden-Württemberg).


