




Hypredrive: High-level interface for solving linear systems with *hypre*

Victor A. P. Magri ¹

¹ Lawrence Livermore National Laboratory, CA, USA

DOI: [10.21105/joss.06654](https://doi.org/10.21105/joss.06654)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Lucy Whalley](#)  

Reviewers:

- [@DamynChipman](#)
- [@mayrmt](#)

Submitted: 30 March 2024

Published: 24 June 2024

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

Summary

Solving sparse linear systems of equations is an essential task for many application codes in computational science and engineering. Commonly used mathematical libraries for addressing these problems include, among others, [trilinos](#) ([Heroux et al., 2005](#)), [PETSc](#) ([Balay et al., 2024](#)) and [hypre](#) ([Falgout & Yang, 2002](#)). The first two allow for quick testing of different solution strategies through input files, a feature that *hypre* currently lacks. *Hypredrive* aims to fill this gap by providing a simple and user-friendly interface to *hypre*. Inspired by the solver composability support in PETSc via [.petscrc files](#) and the flexible configuration framework used in [spack](#) ([Gamblin et al., 2015](#)), *hypredrive* allows users to easily configure and switch solver options in *hypre* through input files in [YAML](#) format, making experimentation with different solver techniques more accessible to researchers and software developers who work with numerical simulation codes.

Statement of need

Hypre is a widely used and efficient linear solver package; however, the complexity associated with its direct use might limit *hypre* users and application developers to explore with different solver options. *Hypredrive* bridges this gap by providing a high-level and lightweight interface to *hypre*, encapsulating its complexity while retaining its capabilities with minimal computational overhead.

Software capabilities

Hypredrive is a software package written in C that includes a library with APIs designed to simplify the interaction with *hypre* and an executable driver for performing the solution of linear systems defined via [YAML](#) input files. The types of linear systems solvable with *hypredrive* are determined by *hypre*. Key features of the software are:

- **Encapsulation:** `libHYPREDRV` wraps the function calls for building solvers and preconditioners in *hypre* through an intuitive [YAML](#) interface driven by configuration files. This design ensures a straightforward way of setting up solvers in *hypre* and sharing options without recompiling the user's application code.
- **Prototyping:** *hypredrive* allows users to prototype rapidly, comparing the performance of various solver options and tweaking parameters directly through the [YAML](#) configuration file. This flexibility encourages experimentation, helping users identify the most effective solver strategies for their specific problems.
- **Testing:** *hypredrive* enables the creation of an integrated testing framework to evaluate solvers against a set of predefined linear systems. This feature lets users understand

whether updates to *hypre* lead to different solver convergence or performance for their problems of interest.

Example usage

As an example usage of *hypredrive*, consider the solution of a linear system arising from a seven points finite difference discretization of the Laplace equation on a 10x10x10 cartesian grid. Both sparse matrix and right hand side vector are stored at `data/ps3d10pt7/np1/`. For solving this linear system with algebraic multigrid (BoomerAMG) as a preconditioner to the conjugate gradient iterative solver, the YAML input file `input.yml` would look like:

```
linear_system:
  rhs_filename: data/ps3d10pt7/np1/IJ.out.b
  matrix_filename: data/ps3d10pt7/np1/IJ.out.A
solver: pcg
preconditioner: amg
```

while *hypredrive* can be executed via

```
$ mpirun -np 1 ./hypredrive input.yml
```

yielding an output that looks like:

```
Date and time: YYYY-MM-DD HH:MM:SS
```

```
Using HYPRE_DEVELOP_STRING: HYPRE_VERSION_GOES_HERE
```

```
Running on 1 MPI rank
```

```
-----
linear_system:
  rhs_filename: data/ps3d10pt7/np1/IJ.out.b
  matrix_filename: data/ps3d10pt7/np1/IJ.out.A
solver: pcg
preconditioner: amg
-----
```

```
=====
Solving linear system #0 with 1000 rows and 6400 nonzeros...
=====
```

```
STATISTICS SUMMARY:
```

```
+-----+-----+-----+-----+-----+-----+
|      | LS build |      setup |      solve | relative |      |
| Entry |      times |      times |      times | res. norm | iters |
+-----+-----+-----+-----+-----+-----+
|      0 |      0.004 |      0.002 |      0.001 | 4.98e-08 |      6 |
+-----+-----+-----+-----+-----+-----+
```

```
Date and time: YYYY-MM-DD HH:MM:SS
```

```
${HYPREDRIVE_PATH}/hypredrive done!
```

This example shows the minimal set of options in *hypredrive*'s input file needed for running the executable. Specific parameter/value pairs for controlling the setup of preconditioners and solvers can be added to the input file, leading to different convergence behaviors. For a complete list of available parameters and more detailed input file examples, see [hypredrive's manual](#).

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-JRNL-865058).

References

- Balay, S., Abhyankar, S., Adams, M. F., Benson, S., Brown, J., Brune, P., Buschelman, K., Constantinescu, E., Dalcin, L., Dener, A., Eijkhout, V., Faibussowitsch, J., Gropp, W. D., Hapla, V., Isaac, T., Jolivet, P., Karpeev, D., Kaushik, D., Knepley, M. G., ... Zhang, J. (2024). *PETSc/TAO users manual* (ANL-21/39 - Revision 3.21). Argonne National Laboratory. <https://doi.org/10.2172/2205494>
- Falgout, R. D., & Yang, U. M. (2002). Hypre: A library of high performance preconditioners. In P. M. A. Sloot, A. G. Hoekstra, C. J. K. Tan, & J. J. Dongarra (Eds.), *Computational science — ICCS 2002* (pp. 632–641). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-47789-6_66
- Gamblin, T., LeGendre, M., Collette, M. R., Lee, G. L., Moody, A., Supinski, B. R. de, & Futral, S. (2015). *The Spack package manager: Bringing order to HPC software chaos*. <https://doi.org/10.1145/2807591.2807623>
- Heroux, M. A., Bartlett, R. A., Howle, V. E., Hoekstra, R. J., Hu, J. J., Kolda, T. G., Lehoucq, R. B., Long, K. R., Pawlowski, R. P., Phipps, E. T., Salinger, A. G., Thornquist, H. K., Tuminaro, R. S., Willenbring, J. M., Williams, A., & Stanley, K. S. (2005). An overview of the Trilinos project. *ACM Trans. Math. Softw.*, *31*(3), 397–423. <https://doi.org/10.1145/1089014.1089021>