hypredrive

Release 0.1

Victor A. P. Magri

Feb 05, 2024

CONTENTS:

CHAPTER

ONE

INTRODUCTION

1.1 What is hypredrive?

hypredrive is a high-level interface driver for hypre, a package for solving sparse linear systems of equations. It's designed to make the process of building and solving linear systems easier by all steps can be configured via an input file in YAML format. Whether you are a researcher or an application library developer, hypredrive offers an easy way with low overhead to test or access the linear solvers provided by hypre

1.2 Key Features

- **Encapsulation** hides the complexity of the Hypre library by offering an intuitive interface driven by YAML configurations. This allows for straightforward and error-resistant setup, enabling easy adjustments and sharing of solver settings.
- **Prototyping** with a variety of solver options. Users can effortlessly compare solver performance and adjust parameters through the YAML configuration, fostering experimentation and optimal solver strategy identification.
- **Testing** through an integrated framework to evaluate solvers against a diverse set of predefined linear system problems. Thus ensuring that future hypre developments do not negatively impact solver convergence and performance for these problems.

1.3 Getting Started

To get started with hypredrive, you should first ensure that the software is properly installed on your system. For detailed installation instructions, please refer to the *Installation* section.

Once the installation is complete, familiarize yourself with the the input file structure for hypredrive by reading through the *Input File Structure* section. This will provide you with a good understanding of how to configure and run hypredrive for your specific needs.

Here's an example command to run hypredrive on a single process with a basic configuration file:

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

In this command, `input.yml` should be replaced with the path to your actual configuration file. You can find input file examples and detailed explanations in the *Input File Examples* section.

1.4 Contributing

We welcome contributions from the community and are pleased that you're interested in helping improve hypredrive! This document provides guidelines and information on how you can contribute.

1.4.1 Ways to Contribute

There are many ways to contribute to hypredrive:

- **Reporting Bugs:** If you encounter issues or bugs, please report them by opening an issue on our GitHub issues page. Please provide as much detail as possible to help us understand and address the issue.
- **Feature Requests:** Are you a developer with ideas for new features or improvements? Feel free to submit them as issues, labeling them as feature requests.
- **Submitting Patches:** If you've fixed a bug or implemented a feature, you can submit a pull request. Make sure your code adheres to the project's coding standards and include tests if possible.

If you plan to submit a pull request, before you start, it's a good idea to get familiar with the following:

- **Project Structure:** Understand how the project is organized.
- **Coding Standards:** Follow the coding style and guidelines of the project to ensure consistency.
- **Testing:** Write and run tests to make sure your changes don't introduce new issues.

1.4.2 Submitting a Pull Request

1. **Fork the Repository:** Start by forking the repository on GitHub.
2. **Clone Your Fork:** Clone your fork locally and create a new branch for your contribution.
3. **Make Your Changes:** Implement your fix or feature.
4. **Test Your Changes:** Ensure your changes pass all tests and don't introduce new issues.
5. **Commit Your Changes:** Commit your changes with a clear, descriptive commit message.
6. **Push Your Changes:** Push your changes to your fork on GitHub.
7. **Submit a Pull Request:** Open a pull request from your fork to the main *hypredrive* repository.

1.4.3 Code Review Process

After you submit a pull request, the project maintainers will review your changes. During the review process:

- Be open to feedback and willing to make revisions.
- Discuss any suggestions or issues that reviewers bring up.
- Once your pull request is approved, a maintainer will merge it into the project.

CHAPTER TWO

INSTALLATION

Installing *hypredrive* is straightforward. Follow these steps to get it up and running on your system.

2.1 Prerequisites

Before installing *hypredrive*, ensure you have the following prerequisites installed:

- m4: GNU package for expanding and processing macros.
- Autoconf: GNU package for generating portable configure scripts.
- Automake: GNU package for generating portable Makefiles.
- hypre: high-performance preconditioners library.

Note: The GNU packages (`m4`, `autoconf`, and `automake`) are generally pre-installed in Unix distributions. If they are not present, they can be easily installed via package managers (`apt`, `yum`, `pacman`, `homebrew`).

2.2 Installing *hypredrive*

Users can install *hypredrive* by compiling from source, according to the steps bellow:

1. Download *hypredrive's* source code. This can be accomplished via `git`:

   ```
   git clone https://github.com/victorapm/hypredrive.git
   ```

   Another option, which does not download the full repository history, is to use `wget`:

   ```
   wget https://github.com/victorapm/hypredrive/archive/refs/heads/master.zip
   unzip master.zip
   rm master.zip
   mv hypredrive-master hypredrive
   ```

2. Navigate to the cloned directory and run the `autogen` script:

   ```
   cd hypredrive
   ./autogen.sh
   ```

3. Run the configure script while informing where the *hypre* library and include files can be found:

```
./configure --prefix=${HYPREDRIVE_INSTALL_DIR} \
            --with-hypre-include=${HYPRE_INSTALL_DIR}/include \
            --with-hypre-lib=${HYPRE_INSTALL_DIR}/lib
```

Replace `${HYPREDRIVE_INSTALL_DIR}` with your desired installation path for *hypredrive*, and `${HYPRE_INSTALL_DIR}` with the path to your installation of *hypre*.

4. Run `make`:

```
make -j
make install
```

2.3 Verifying the Installation

After installation, you can verify that *hypredrive* is installed correctly by running:

```
make check
```

You should see the output below:

```
"Running with 1 MPI process... passed!"
```

2.4 Troubleshooting

If you encounter any issues during the installation of *hypredrive*, please open a GitHub issue and include a copy of the `config.log` file, which is generated after running the `configure` script.

CHAPTER THREE

INPUT FILE STRUCTURE

hypredrive uses a configuration file in YAML format to specify the parameters and settings for the program's execution. Below is a detailed explanation of each section in the configuration file. In general, the various keywords are optional and, if not explicitly defined by the user, default values are used for them. On the other hand, some keywords such as `linear_system` are mandatory and, thus, are marked with *required* or *possibly required*, depending on the value of other keywords.

Note: The YAML file parser in *hypredrive* is case-insensitive, meaning that it works regardless of the presence of lower-case, upper-case, or a mixture of both when defining keys and values in the input file.

3.1 General Settings

The `general` section contains global settings that apply to the entire execution of *hypredrive*. This section is optional.

- `warmup` - If set to *yes*, *hypredrive* will perform a warmup execution to ensure more accurate timing measurements. If *no*, no warmup is performed. The default value for this parameter is *yes*.
- `statistics` - If set to *yes*, *hypredrive* will display a statistics summary at the end of the run reporting execution times. If *no*, no statistics reporting is performed. The default value for this parameter is *yes*.
- `num_repetitions` - Specifies the number of times the operation should be repeated. Useful for benchmarking and profiling. The default value for this parameter is *1*.

An example code block for the `general` section is given below:

```yaml
general:
  warmup: yes
  statistics: yes
  num_repetitions: 1
```

3.2 Linear System

The `linear_system` section describes the linear system that *hypredrive* will solve. This section is required.

- `exec_policy` - Determines whether the linear system is to be solved on the `host` (CPU) or `device` (GPU). The default value for this parameter is `host`.
- `type` - The format of the linear system matrix. Available options are `ij` and `mtx`. The default value for this parameter is `ij`.

- `matrix_filename` - (Required) The filename of the linear system matrix. This parameter does not have a default value.
- `precmat_filename` - The filename of the linear system matrix used for computing the preconditioner, which, by default, is set to the original linear system matrix.
- `rhs_filename` - (Possibly required) The filename of the linear system right hand side vector. This parameter does not have a default value and it is required when the `rhs_mode` is set to `file`.
- `x0_filename` - (Possibly required) The filename of the initial guess for the linear system left hand side vector. This parameter does not have a default value and it is required when the `init_guess_mode` is set to `file`.
- `dofmap_filename` - (Possibly required) The filename of the degrees of freedom maping array (*dofmap*) for the linear system. This parameter does not have a default value and it is required when the `mgr` preconditioner is used.
- `init_guess_mode` - Choice of initial guess vector. Available options are:
  - `zeros`: generates a vector of zeros.
  - `ones`: generates a vector of ones.
  - `random`: generates a vector of random numbers between *0* and *1*.
  - `file`: vector is read from file.

  The default value for this parameter is `file`.
- `rhs_mode` - Choice of initial guess vector. Available options are the same as for `init_guess_mode`.

An example code block for the `linear_system` section is given below:

```
linear_system:
  type: ij
  x0_filename: IJ.out.x0
  rhs_filename: IJ.out.b
  matrix_filename: IJ.out.A
  precmat_filename: IJ.out.A
  dofmap_filename: dofmap
  rhs_mode: file
  init_guess_mode: file
  exec_policy: device
```

3.3 Solver

The `solver` section is mandatory and it specifies the Krylov solver configuration. The available options for the Krylov solver type are:

- `pcg` - preconditioned conjugate gradient.
- `bicgstab` - bi-conjugate gradient stabilized.
- `gmres` - generalized minimal residual.
- `fgmres` - flexible generalized minimal residual.

The solver type must be entered as a key in a new indentation level under `solver`.

3.3.1 PCG

The available keywords to further configure the preconditioned conjugate gradient solver (pcg) are all optional and given below:

- max_iter - Maximum number of iterations. Available values are any positive integer.
- two_norm - Turn on/off L2 norm for the residual. Available values are yes or no. Default value is yes.
- rel_change - Turn on/off an additional convergence criteria that checks for a relative change in the solution vector. Available values are yes or no. Default value is no.
- print_level - Verbosity level for the iterative solver. *1* turns on convergence history reporting. Default value is *0*.
- relative_tol - Relative tolerance based on the norm of the residual vector and used for determining convergence of the iterative solver. Available values are any positive floating point number. Default value is 1.0e-6.
- absolute_tol - Absolute tolerance used for determining convergence of the iterative solver. Available values are any positive floating point number. Default value is 0.0, meaning that the absolute tolerance-based convergence criteria is inactive.
- residual_tol - Tolerance used for determining convergence of the iterative solver and based on the norm of the difference between subsequent residual vectors. Available values are any positive floating point number. Default value is 0.0, meaning that the residual tolerance-based convergence criteria is inactive.
- conv_fac_tol - Tolerance used for determining convergence of the iterative solver and based on the convergence factor ratio of subsequent iterations. Available values are any positive floating point number. Default value is 0.0, meaning that the convergence factor tolerance-based convergence criteria is inactive.

The code block representing the default parameter values for the solver:pcg section is given below:

```
solver:
  pcg:
    max_iter: 100
    two_norm: yes
    rel_change: no
    print_level: 1
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    residual_tol: 0.0
    conv_fac_tol: 0.0
```

3.3.2 BiCGSTAB

The available keywords to further configure the bi-conjugate gradient stabilized solver (bicgstab) are all optional and given below:

- min_iter - Minimum number of iterations. Available values are any positive integer.
- max_iter, print_level, relative_tol, absolute_tol, residual_tol, and conv_fac_tol - See *PCG* for a description of these variables.

The code block representing the default parameter values for the solver:bicgstab section is given below:

```
solver:
  bicgstab:
    min_iter: 0
```

(continues on next page)

(continued from previous page)

```
    max_iter: 100
    print_level: 1
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    residual_tol: 0.0
    conv_fac_tol: 0.0
```

3.3.3 GMRES

The available keywords to further configure the generalized minimal residual solver (`gmres`) are all optional and given below:

- `skip_real_res_check` - Skip calculation of the real residual when evaluating convergence. Available values are *yes* and *no*. Default value is *no*.
- `krylov_dim` - Dimension of the krylov space. Available values are any positive integer. Default value is *30*.
- `min_iter`, `max_iter`, `print_level`, `rel_change`, `relative_tol`, `absolute_tol`, and `conv_fac_tol` - See *PCG* for a description of these variables.

The code block representing the default parameter values for the `solver:gmres` section is given below:

```
solver:
  gmres:
    min_iter: 0
    max_iter: 300
    skip_real_res_check: no
    krylov_dim: 30
    rel_change: no
    print_level: 1
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    conv_fac_tol: 0.0
```

3.3.4 FGMRES

The available keywords to further configure the flexible generalized minimal residual solver (`fgmres`) are all optional and given below:

- `min_iter`, `max_iter`, `krylov_dim`, `print_level`, `relative_tol`, `absolute_tol` - See *GMRES* for a description of these variables.

The code block representing the default parameter values for the `solver:fgmres` section is given below:

```
solver:
  fgmres:
    min_iter: 0
    max_iter: 300
    krylov_dim: 30
    print_level: 1
    relative_tol: 1.0e-6
    absolute_tol: 0.0
```

3.4 Preconditioner

The `preconditioner` section is mandatory and it specifies the preconditioner configuration. Available options for the preconditioner type are:

- `amg` - algebraic multigrid (BoomerAMG).
- `ilu`: incomplete LU factorization.
- `fsai`: factorized sparse approximate inverse.
- `mgr`: multigrid reduction.

The preconditioner type must be entered as a key in a new indentation level under `preconditioner`.

3.4.1 AMG

The algebraic multigrid (BoomerAMG) preconditioner can be further configured by the following optional keywords:

- `max_iter` - number of times the preconditioner is applied when it is called. Available values are any positive integer. Default value is *1*.
- `tolerance` - convergence tolerance of AMG when applied multiple times. Available values are any positive floating point number. Default value is *0.0*.
- `print_level` - Verbosity level for the preconditioner. Default value is *0*
  - `0` - no printout.
  - `1` - print setup statistics.
  - `2` - print solve statistics.
- `interpolation` - subsection detailing interpolation options:
  - `prolongation_type` - choose the prolongation operator. For available options, see HYPRE_BoomerAMGSetInterpType. Default value is *6*.
  - `restriction_type` - choose the restriction operator. For available options, see HYPRE_BoomerAMGSetRestriction. Default value is *0*.
  - `trunc_factor` - truncation factor for computing interpolation. Available values are any non-negative floating point number. Default value is *0.0*.
  - `max_nnz_row` - maximum number of elements per row for interpolation. Available values are any non-negative integer. Default value is *4*.
- `coarsening` - subsection detailing coarsening options:
  - `type` - choose the coarsening method. For available options, see HYPRE_BoomerAMGSetCoarsenType. Default value is *10* (HMIS).
  - `strong_th` - strength threshold used for computing the strength of connection matrix. Available values are any non-negative floating point number. Default value is *0.25*.
  - `seq_amg_th` - maximum size for agglomeration or redundant coarse grid solve. Smaller system are then solved with a sequential AMG. Available values are any non-negative integer. Default value is *0*.
  - `max_coarse_size` - maximum size of the coarsest grid. Available values are any non-negative integer. Default value is *64*.
  - `min_coarse_size` - minimum size of the coarsest grid. Available values are any non-negative integer. Default value is *0*.

   - `max_levels` - maximum number of levels in the multigrid hierarchy. Available values are any non-negative integer. Default value is *25*.
   - `num_functions` - size of the system of PDEs, when using the systems version. Available values are any positive integer. Default value is *1*.
   - `rap2` - whether or not to use two matrix products to compute coarse level matrices. Available values are any non-negative integer. Default value is *0*.
   - `mod_rap2` - whether or not to use two matrix products with modularized kernels for computing coarse level matrices. Available values are any non-negative integer. Default value is *0* for CPU runs or *1* for GPU runs.
   - `keep_transpose` - whether or not to save local interpolation transposes for more efficient matvecs during the solve phase. Available values are any non-negative integer. Default value is *0* for CPU runs or *1* for GPU runs.
   - `max_row_sum` - parameter that modifies the definition of strength for diagonal dominant portions of the matrix. Available values are any non-negative floating point number. Default value is *0.9*.

- `aggressive` - subsection detailing aggressive coarsening options:
   - `prolongation_type` - choose the prolongation type used in levels with aggressive coarsening turned on. For available options, see HYPRE_ParCSRHybridSetAggInterpType. Default value is *4* (multipass).
   - `num_levels` - number of levels with aggressive coarsening turned on. Available values are any positive integer. Default value is *0*.
   - `num_paths` - degree of aggressive coarsening. Available values are any positive integer. Default value is *1*.
   - `trunc_factor` - truncation factor for computing interpolation in aggressive coarsening levels. Available values are any non-negative floating point number. Default value is *0.0*.
   - `max_nnz_row` - maximum number of elements per row for computing interpolation in aggressive caorsening levels. Available values are any non-negative integer. Default value is *4*.
   - `P12_trunc_factor` - truncation factor for matrices P1 and P2 which are used to build 2-stage interpolation. Available values are any non-negative floating point number. Default value is *0.0*.
   - `P12_max_elements` - maximum number of elements per row for matrices P1 and P2 which are used to build 2-stage interpolation. Available values are any non-negative integer. Default value is *0*, meaning there is no maximum number of elements per row.

- `relaxation` - subsection detailing relaxation options:
   - `down_type` - relaxation method used in the pre-smoothing stage. For available options, see HYPRE_BoomerAMGSetRelaxType. Default value is *13*.
   - `up_type` - relaxation method used in the post-smoothing stage. For available options, see HYPRE_BoomerAMGSetRelaxType. Default value is *14*.
   - `coarse_type` - relaxation method used in the coarsest levels. For available options, see HYPRE_BoomerAMGSetRelaxType. Default value is *9*.
   - `down_sweeps` - number of pre-smoothing sweeps. Available values are any integer greater or equal than *-1*, which turns off the selection of sweeps at the specific cycle. Default value is *-1*.
   - `up_sweeps` - number of post-smoothing sweeps. Available values are any integer greater or equal than *-1*, which turns off the selection of sweeps at the specific cycle. Default value is *-1*.
   - `coarse_sweeps` - number of smoothing sweeps in the coarsest level. Available values are any integer greater or equal than *-1*, which turns off the selection of sweeps at the specific cycle. Default value is *-1*.

  - `num_sweeps` - number of pre and post-smoothing sweeps. Available values are any non-negative integer. Default value is *1*.
  - `order` - order in which the points are relaxed. For available options, see HYPRE_BoomerAMGSetRelaxOrder. Default value is *0*.
  - `weight` - relaxation weight for smoothed Jacobi and hybrid SOR. For available options, see HYPRE_BoomerAMGSetRelaxWt. Default value is *1.0*.
  - `outer_weight` - outer relaxation weight for hybrid SOR and SSOR. For available options, see HYPRE_BoomerAMGSetOuterWt. Default value is *1.0*.
- `relaxation` - subsection detailing complex smoother options:
  - `type` - complex smoother type. For available options, see HYPRE_BoomerAMGSetSmoothType. Default value is *5* (ILU).
  - `num_levels` - number of levels starting from the finest one where complex smoothers are used. Available values are any non-negative integer. Default value is *0*.
  - `num_sweeps` - number of pre and post-smoothing sweeps used for the complex smoother. Available values are any non-negative integer. Default value is *1*.

The default parameter values for the `preconditioner:amg` section are represented in the code block below:

```yaml
preconditioner:
  amg:
    tolerance: 0.0
    max_iter: 1
    print_level: 0
    interpolation:
      prolongation_type: 6
      restriction_type: 0
      trunc_factor: 0.0
      max_nnz_row: 4
    coarsening:
      type: 10
      strong_th: 0.25
      seq_amg_th: 0
      max_coarse_size: 64
      min_coarse_size: 0
      max_levels: 25
      num_functions: 1
      rap2: 0
      mod_rap2: 0 # 1 for GPU runs
      keep_transpose: 0 # 1 for GPU runs
      max_row_sum: 0.9
    aggressive:
      num_levels: 0
      num_paths: 1
      prolongation_type: 4
      trunc_factor: 0
      max_nnz_row: 0
      P12_trunc_factor: 0.0
      P12_max_elements: 0
    relaxation:
      down_type: 13
```

(continues on next page)

(continued from previous page)

```
      up_type: 14
      coarse_type: 9
      down_sweeps: -1
      up_sweeps: -1
      coarse_sweeps: -1
      num_sweeps: 1
      order: 0
      weight: 1.0
      outer_weight: 1.0
    smoother:
      type: 5
      num_levels: 0
      num_sweeps: 1
```

3.4.2 ILU

The incomplete LU factorization (ILU) preconditioner can be further configured by the following optional keywords:

- `max_iter`, `tolerance`, and `print_level` - See *AMG* for a description of these variables.
- `type` - ILU type. For available options, see HYPRE_ILUSetType. Default value is *0* (Block-Jacobi ILU0).
- `fill_level` - level of fill when using ILUK. Available values are any non-negative integer. Default value is *0*.
- `reordering` - reordering method. For available options, see HYPRE_ILUSetLocalReordering. Default value is *0* (no reordering).
- `tri_solve` - whether or not to turn on direct triangular solves in the preconditioner's application phase. Default value is *1*.
- `lower_jac_iters` - Number of iterations for solving the lower triangular system during the preconditioner's application phase. Available values are any positive integer. Default value is *5*. This option has effect only when `tri_solve` is set to zero.
- `lower_jac_iters` - Number of iterations for solving the upper triangular system during the preconditioner's application phase. Available values are any positive integer. Default value is *5*. This option has effect only when `tri_solve` is set to zero.
- `max_row_nnz` - Maximum number if nonzeros per row when using ILUT. Available values are any positive integer. Default value is *200*.
- `schur_max_iter` - Maximum number of the Schur system solve. Available values are any positive integer. Default value is *5*. This option has effect only when `type` is greater or equal than *10*.
- `droptol` - Dropping tolerance for computing the triangular factors when using ILUT. Available values are any non-negative floating point numbers. Default value is *1.0e-2*.
- `nsh_droptol` - Dropping tolerance for computing the triangular factors when using NSH. Available values are any non-negative floating point numbers. Default value is *1.0e-2*.

The default parameter values for the `preconditioner:ilu` section are represented in the code block below:

```
preconditioner:
  ilu:
    tolerance: 0.0
    max_iter: 1
    print_level: 0
```

(continues on next page)

(continued from previous page)

```yaml
    type: 0
    fill_level: 0
    reordering: 0
    tri_solve: 1
    lower_jac_iters: 5
    upper_jac_iters: 5
    max_row_nnz: 200
    schur_max_iter: 3
    droptol: 1.0e-2
    nsh_droptol: 1.0e-2
```

3.4.3 FSAI

The factorized sparse approximate inverse (FSAI) preconditioner can be further configured by the following optional keywords:

- `max_iter`, `tolerance`, and `print_level` - See *AMG* for a description of these variables.
- `type` - algorithm type used for building FSAI. For available options, see HYPRE_FSAISetAlgoType. Default value is *1* (Adaptive) for CPUs and *3* (Static) for GPUs.
- `ls_type` - solver type for the local linear systems in FSAI. For available options, see HYPRE_FSAISetLocalSolveType. Default value is *0* (Gauss-Jordan).
- `max_steps` - maximum number of steps for computing the sparsity pattern of G. Available values are any positive integer. Default value is *5*.
- `max_step_size` - step size for computing the sparsity pattern of G. Available values are any positive integer. Default value is *3*.
- `max_nnz_row` - maximum number of nonzeros per row for computing the sparsity pattern of G. Available values are any positive integer. Default value is *15*.
- `num_levels` - number of levels for computing the candidate pattern matrix. Available values are any positive integer. Default value is *1*.
- `eig_max_iters` - number of iterations for estimating the largest eigenvalue of G. Available values are any positive integer. Default value is *5*.
- `threshold` - Dropping tolerance for building the canditate pattern matrix. Available values are any non-negative floating point numbers. Default value is *1.0e-3*.
- `kap_tolerance` - Kaporin reduction factor. Available values are any non-negative floating point numbers. Default value is *1.0e-3*.

The default parameter values for the `preconditioner:fsai` section are represented in the code block below:

```yaml
preconditioner:
  fsai:
    tolerance: 0.0
    max_iter: 1
    print_level: 0
    algo_type: 1
    ls_type: 0
    max_steps: 5
    max_step_size: 3
```

(continues on next page)

(continued from previous page)

```
    max_nnz_row: 15
    num_levels: 1
    eig_max_iters: 5
    threshold: 1.0e-3
    kap_tolerance: 1.0e-3
```

3.4.4 MGR

The multigrid reduction (MGR) preconditioner can be further configured by the following optional keywords:

- `max_iter` and `tolerance` - See *AMG* for a description of these variables.
- `print_level` - verbosity level for the preconditioner. For available options, see HYPRE_MGRSetPrintLevel. Default value is *0* (no printout).
- `coarse_th` - threshold for dropping small entries on the coarse grid. Available values are any non-negative floating point numbers. Default value is *0.0*, which means no dropping.
- `level` - special keyword for defining specific parameters for each MGR level. Each level is identified by its numeric ID starting from *0* (finest) and placed in increasing order on the next indentation level of the YAML input.
  - `f_dofs` - (Mandatory) Array containing the identifiers of F (fine) degrees of freedom to be treated in the current level. Available values are any integer numbers from *0* to *n_dofs - 1*, where *n_dofs* represent the unique number of degrees of freedom identifiers.
  - `f_relaxation` - relaxation method targeting F points. For available options, see HYPRE_MGRSetLevelFRelaxType. Default value is *0* (Jacobi). Use `none` to deactivate F-relaxation.
  - `g_relaxation` - global relaxation method targeting F and C points. For available options, see HYPRE_MGRSetGlobalSmoothType. Default value is *2* (Jacobi). Use `none` to deactivate global relaxation.
  - `restriction_type` - algorithm for computing the restriction operator. For available options, see HYPRE_MGRSetRestrictType. Default value is *0* (Injection).
  - `prolongation_type` - algorithm for computing the prolongation operator. For available options, see HYPRE_MGRSetInterpType. Default value is *0* (Injection).
  - `coarse_level_type` - algorithm for computing the coarse level matrices. For available options, see HYPRE_MGRSetCoarseGridMethod. Default value is *0* (Galerkin).
- `coarsest_level` - special keyword for defining specific parameters for MGR's coarsest level.

The default parameter values for the `preconditioner:mgr` section are represented in the code block below:

```
preconditioner:
  mgr:
    tolerance: 0.0
    max_iter: 1
    print_level: 0
    coarse_th: 0.0
    level:
      0:
        f_dofs: [1, 2] # Example usage where DOFs 1 and 2 are treated in MGR's 1st level
        f_relaxation: single
          sweeps: 1
```

(continues on next page)

(continued from previous page)

```
        g_relaxation: none
        restriction_type: injection
        prolongation_type: jacobi
        coarse_level_type: rap

      1:
        f_dofs: [0] # Example usage where DOF 0 is treated in MGR's 2nd level
        f_relaxation: none
        g_relaxation:
          ilu: # ILU parameters can be specified with a new indentation level
        restriction_type: injection
        prolongation_type: jacobi
        coarse_level_type: rap

    coarsest_level:
      amg: # AMG parameters can be specified with a new indentation level
```

Warning: MGR cannot be fully defined by the `mgr` keyword only. Instead, it is also necessary to specify which types of degrees of freedom are treated as F points in each MGR level, i.e., the last level where a degree of freedom of a given type is present. This is done via the `f_dofs` keyword.

CHAPTER

FOUR

INPUT FILE EXAMPLES

This section provides several examples demonstrating how to set up input files and use *hypredrive* for the solution of different types of linear system problems. All example inputs can be found at the `examples` folder and a reference output for each example can be found at `examples/refOutput`.

4.1 Example 1: Minimal configuration

In this example, we solve a basic linear system using an *AMG-PCG* solver with default settings. This example showcases the minimum amount of information required in the input file to execute *hypredrive*.

We consider a linear system matrix arising from a seven points finite differences discretizaion of the Laplace equation on a *10x10x10* cartesian grid. Furthermore, the right hand side is the vector of ones. Both data are read from file and partitioned for a single MPI rank. Therefore, this example must be executed on a single process.

1. Prepare your linear system files (`matrix_filename` and `rhs_filename`).
2. Use the YAML configuration file `ex1.yml`:

```yaml
linear_system:
  rhs_filename: data/ps3d10pt7/np1/IJ.out.b
  matrix_filename: data/ps3d10pt7/np1/IJ.out.A

solver: pcg

preconditioner: amg
```

3. Run *hypredrive* with the configuration file:

```
mpirun -np 1 ./hypredrive ex1.yml
```

4. Your output should look like:

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

Using HYPRE_DEVELOP_STRING: HYPRE_VERSION_GOES_HERE

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

(continues on next page)

(continued from previous page)

```
------------------------------------------------------------------------------------
====================================================================================



STATISTICS SUMMARY:

+------------+------------+------------+------------+------------+------------+
|            |   LS build |      setup |      solve |   relative |            |
|      Entry |      times |      times |      times |  res. norm |      iters |
+------------+------------+------------+------------+------------+------------+
|          0 |   2.70e-03 |   2.86e-03 |   1.36e-03 |   4.98e-08 |          6 |
+------------+------------+------------+------------+------------+------------+

Date and time: YYYY-MM-DD HH:MM:SS
${HYPREDRIVE_PATH}/hypredrive done!
```

Warning: Make sure that *hypredrive* is executed from the top level project folder in order for the relative paths in `matrix_filename` and `rhs_filename` to be correct. Otherwise, adjust the relative paths for these entries accordingly.

4.2 Example 2: Parallel run with full AMG configuration

In this example, we solve the same problem as in the previous example, but partitioned for *4* processes. We also showcase all available input options for *PCG* and *AMG* in the configuration file.

1. Prepare your linear system files.
2. Use the YAML configuration file `ex2.yml`:

```yaml
linear_system:
  rhs_filename: data/ps3d10pt7/np4/IJ.out.b
  matrix_filename: data/ps3d10pt7/np4/IJ.out.A

solver:
  pcg:
    max_iter: 100
    two_norm: yes
    rel_change: no
    print_level: 2
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    residual_tol: 0.0
    conv_fac_tol: 0.0

preconditioner:
  amg:
    tolerance: 0.0
    max_iter: 1
    print_level: 1
```

(continues on next page)

(continued from previous page)

```yaml
    interpolation:
      prolongation_type: 6
      restriction_type: 0
      trunc_factor: 0.0
      max_nnz_row: 4
    coarsening:
      type: 10
      strong_th: 0.25
      seq_amg_th: 0
      max_coarse_size: 64
      min_coarse_size: 0
      max_levels: 25
      num_functions: 1
      rap2: 0
      mod_rap2: 0
      keep_transpose: 0
      max_row_sum: 0.9
    aggressive:
      num_levels: 0
      num_paths: 1
      prolongation_type: 4
      trunc_factor: 0
      max_nnz_row: 0
      P12_trunc_factor: 0.0
      P12_max_elements: 0
    relaxation:
      down_type: 13
      up_type: 14
      coarse_type: 9
      down_sweeps: -1
      up_sweeps: -1
      coarse_sweeps: -1
      num_sweeps: 1
      order: 0
      weight: 1.0
      outer_weight: 1.0
    smoother:
      type: 5
      num_levels: 0
      num_sweeps: 1
```

3. Run *hypredrive* with the configuration file:

```
mpirun -np 4 ./hypredrive ex2.yml
```

4. Your output should look like:

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

Using HYPRE_DEVELOP_STRING: HYPRE_VERSION_GOES_HERE

------------------------------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
linear_system:
  rhs_filename: data/ps3d10pt7/np4/IJ.out.b
  matrix_filename: data/ps3d10pt7/np4/IJ.out.A
solver:
  pcg:
    max_iter: 100
    two_norm: yes
    rel_change: no
    print_level: 2
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    residual_tol: 0.0
    conv_fac_tol: 0.0
preconditioner:
  amg:
    tolerance: 0.0
    max_iter: 1
    print_level: 1
    interpolation:
      prolongation_type: 6
      restriction_type: 0
      trunc_factor: 0.0
      max_nnz_row: 4
    coarsening:
      type: 10
      strong_th: 0.25
      seq_amg_th: 0
      max_coarse_size: 64
      min_coarse_size: 0
      max_levels: 25
      num_functions: 1
      rap2: 0
      mod_rap2: 0
      keep_transpose: 0
      max_row_sum: 0.9
    aggressive:
      num_levels: 0
      num_paths: 1
      prolongation_type: 4
      trunc_factor: 0
      max_nnz_row: 0
      P12_trunc_factor: 0.0
      P12_max_elements: 0
    relaxation:
      down_type: 13
      up_type: 14
      coarse_type: 9
      down_sweeps: -1
      up_sweeps: -1
      coarse_sweeps: -1
      num_sweeps: 1
      order: 0
```

(continues on next page)

(continued from previous page)

```
      weight: 1.0
      outer_weight: 1.0
    smoother:
      type: 5
      num_levels: 0
      num_sweeps: 1
------------------------------------------------------------------------------------


 Num MPI tasks = 4

 Num OpenMP threads = 1


BoomerAMG SETUP PARAMETERS:

 Max levels = 25
 Num levels = 4

 Strength Threshold = 0.250000
 Interpolation Truncation Factor = 0.000000
 Maximum Row Sum Threshold for Dependency Weakening = 0.900000

 Coarsening Type = HMIS
 measures are determined locally


 No global partition option chosen.

 Interpolation = extended+i interpolation

Operator Matrix Information:

             nonzero            entries/row          row sums
lev    rows  entries sparse   min  max     avg      min         max
======================================================================
  0    1000     6400  0.006     4    7     6.4   0.000e+00   3.000e+00
  1     413     7595  0.045     7   43    18.4  -1.665e-15   4.000e+00
  2      75     2523  0.449    20   60    33.6   1.198e+00   5.057e+00
  3      12      142  0.986    11   12    11.8   5.284e+00   1.196e+01


Interpolation Matrix Information:
                    entries/row        min        max            row sums
lev  rows x cols  min  max  avgW     weight      weight       min         max
================================================================================
  0  1000 x 413     1    4   4.0   6.452e-02   4.255e-01   5.000e-01   1.000e+00
  1   413 x 75      1    4   4.0   5.920e-03   4.787e-01   2.185e-01   1.000e+00
  2    75 x 12      0    4   3.7   6.512e-03   2.832e-01   0.000e+00   1.000e+00


    Complexity:   grid = 1.500000
```

(continues on next page)

(continued from previous page)

```
               operator = 2.603125
                 memory = 3.295469




BoomerAMG SOLVER PARAMETERS:

  Maximum number of cycles:         1
  Stopping Tolerance:               0.000000e+00
  Cycle type (1 = V, 2 = W, etc.):  1

  Relaxation Parameters:
   Visiting Grid:                     down   up  coarse
            Number of sweeps:            1    1     1
   Type 0=Jac, 3=hGS, 6=hSGS, 9=GE:     13   14     9
   Point types, partial sweeps (1=C, -1=F):
                  Pre-CG relaxation (down):   0
                   Post-CG relaxation (up):   0
                             Coarsest grid:   0

<b,b>: 1.000000e+03


Iters       ||r||_2     conv.rate  ||r||_2/||b||_2
-----    ------------   ---------  ------------
    1    9.722072e+00    0.307439    3.074389e-01
    2    6.390004e-01    0.065727    2.020697e-02
    3    4.085393e-02    0.063934    1.291915e-03
    4    2.841366e-03    0.069549    8.985188e-05
    5    1.973415e-04    0.069453    6.240487e-06
    6    1.260822e-05    0.063890    3.987071e-07


====================================================================================


STATISTICS SUMMARY:

+------------+------------+------------+------------+------------+-------------+
|            |   LS build |      setup |      solve |   relative |             |
|      Entry |      times |      times |      times |  res. norm |       iters |
+------------+------------+------------+------------+------------+-------------+
|          0 |   3.07e-04 |   4.42e-03 |   1.07e-03 |   3.99e-07 |           6 |
+------------+------------+------------+------------+------------+-------------+

Date and time: YYYY-MM-DD HH:MM:SS
${HYPREDRIVE_PATH}/hypredrive done!
```

4.3 Example 3: Minimal multigrid reduction strategy

In this example, we solve a linear system derived from the discretization of a compositional flow problem from GEOS. Details about how this linear system was generated can be found at `data/compflow6k/README.md`. This example uses a *MGR-GMRES* solver and showcases the minimal configuration for setting up the multigrid reduction preconditioner for this particular kind of linear system.

1. Prepare your linear system files.
2. Use the YAML configuration file `ex3.yml`:

```yaml
linear_system:
  rhs_filename: data/compflow6k/np1/IJ.out.b
  matrix_filename: data/compflow6k/np1/IJ.out.A
  dofmap_filename: data/compflow6k/np1/dofmap.out

solver: gmres

preconditioner:
  mgr:
    level:
      0:
        f_dofs: [2]

      1:
        f_dofs: [1]

    coarsest_level: amg
```

3. Run *hypredrive* with the configuration file:

```
mpirun -np 1 ./hypredrive ex3.yml
```

4. Your output should look like:

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

Using HYPRE_DEVELOP_STRING: HYPRE_VERSION_GOES_HERE

------------------------------------------------------------------------------------
linear_system:
  rhs_filename: data/compflow6k/np1/IJ.out.b
  matrix_filename: data/compflow6k/np1/IJ.out.A
  dofmap_filename: data/compflow6k/np1/dofmap.out
solver: gmres
preconditioner:
  mgr:
    level:
      0:
        f_dofs: [2]
      1:
        f_dofs: [1]
    coarsest_level: amg
------------------------------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
======================================================================================


STATISTICS SUMMARY:

+------------+------------+------------+------------+------------+------------+
|            |   LS build |      setup |      solve |   relative |            |
|      Entry |      times |      times |      times |  res. norm |      iters |
+------------+------------+------------+------------+------------+------------+
|          0 |   5.48e-03 |   8.93e-03 |   3.83e-02 |   8.86e-07 |         20 |
+------------+------------+------------+------------+------------+------------+

Date and time: YYYY-MM-DD HH:MM:SS
${HYPREDRIVE_PATH}/hypredrive done!
```

4.4 Example 4: Advanced multigrid reduction strategy

In this example, we solve the same problem as before, but partitioned for 4 processes. Here, we showcase a more advanced setup of *MGR* involving multiple options.

1. Prepare your linear system files.
2. Use the YAML configuration file `ex4.yml`:

```yaml
linear_system:
  rhs_filename: data/compflow6k/np4/IJ.out.b
  matrix_filename: data/compflow6k/np4/IJ.out.A
  dofmap_filename: data/compflow6k/np4/dofmap.out

solver:
  gmres:
    min_iter: 0
    max_iter: 100
    skip_real_res_check: no
    krylov_dim: 30
    rel_change: no
    print_level: 2
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    conv_fac_tol: 0.0

preconditioner:
  mgr:
    tolerance: 0.0
    max_iter: 1
    print_level: 1
    coarse_th: 0.0
    level:
      0:
        f_dofs: [2]
        f_relaxation: single
```

(continues on next page)

(continued from previous page)

```yaml
      g_relaxation: none
      restriction_type: injection
      prolongation_type: jacobi
      coarse_level_type: rap

    1:
      f_dofs: [1]
      f_relaxation: single
      g_relaxation: ilu
      restriction_type: columped
      prolongation_type: injection
      coarse_level_type: rai

  coarsest_level:
    amg:
      tolerance: 0.0
      max_iter: 1
      print_level: 0 # Turn off printout from AMG since it's done from MGR
      interpolation:
        prolongation_type: 18 # Use MM-ext+e interpolation
        restriction_type: 0
        trunc_factor: 0.0
        max_nnz_row: 4
      coarsening:
        type: 8 # Use PMIS coarsening
        strong_th: 0.3
        seq_amg_th: 0
        max_coarse_size: 64
        min_coarse_size: 0
        max_levels: 25
        num_functions: 1
        rap2: 0
        mod_rap2: 0
        keep_transpose: 0
        max_row_sum: 0.9
      aggressive:
        num_levels: 0
        num_paths: 1
        prolongation_type: 4
        trunc_factor: 0
        max_nnz_row: 0
        P12_trunc_factor: 0.0
        P12_max_elements: 0
      relaxation:
        down_type: 13
        up_type: 14
        coarse_type: 9
        down_sweeps: -1
        up_sweeps: -1
        coarse_sweeps: -1
        num_sweeps: 1
        order: 0
```

(continues on next page)

(continued from previous page)

```
        weight: 1.0
        outer_weight: 1.0
      smoother:
        type: 5
        num_levels: 0
        num_sweeps: 1
```

3. Run *hypredrive* with the configuration file:

```
mpirun -np 4 ./hypredrive ex4.yml
```

4. Your output should look like:

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

Using HYPRE_DEVELOP_STRING: HYPRE_VERSION_GOES_HERE

------------------------------------------------------------------------------------
linear_system:
  rhs_filename: data/compflow6k/np4/IJ.out.b
  matrix_filename: data/compflow6k/np4/IJ.out.A
  dofmap_filename: data/compflow6k/np4/dofmap.out
solver:
  gmres:
    min_iter: 0
    max_iter: 100
    skip_real_res_check: no
    krylov_dim: 30
    rel_change: no
    print_level: 2
    relative_tol: 1.0e-6
    absolute_tol: 0.0
    conv_fac_tol: 0.0
preconditioner:
  mgr:
    tolerance: 0.0
    max_iter: 1
    print_level: 1
    coarse_th: 0.0
    level:
      0:
        f_dofs: [2]
        f_relaxation: single
        g_relaxation: none
        restriction_type: injection
        prolongation_type: jacobi
        coarse_level_type: rap
      1:
        f_dofs: [1]
        f_relaxation: single
        g_relaxation: ilu
        restriction_type: columped
```

(continues on next page)

(continued from previous page)

```
        prolongation_type: injection
        coarse_level_type: rai
    coarsest_level:
      amg:
        tolerance: 0.0
        max_iter: 1
        print_level: 0
        interpolation:
          prolongation_type: 18
          restriction_type: 0
          trunc_factor: 0.0
          max_nnz_row: 4
        coarsening:
          type: 8
          strong_th: 0.3
          seq_amg_th: 0
          max_coarse_size: 64
          min_coarse_size: 0
          max_levels: 25
          num_functions: 1
          rap2: 0
          mod_rap2: 0
          keep_transpose: 0
          max_row_sum: 0.9
        aggressive:
          num_levels: 0
          num_paths: 1
          prolongation_type: 4
          trunc_factor: 0
          max_nnz_row: 0
          P12_trunc_factor: 0.0
          P12_max_elements: 0
        relaxation:
          down_type: 13
          up_type: 14
          coarse_type: 9
          down_sweeps: -1
          up_sweeps: -1
          coarse_sweeps: -1
          num_sweeps: 1
          order: 0
          weight: 1.0
          outer_weight: 1.0
        smoother:
          type: 5
          num_levels: 0
          num_sweeps: 1
------------------------------------------------------------------------------------



 Num MPI tasks = 4
 Num OpenMP threads = 1
```

(continues on next page)

(continued from previous page)

```
 Execution policy = Host


MGR SETUP PARAMETERS:

        MGR num levels = 2
 coarse AMG num levels = 4
      Total num levels = 6

            Global           Fine           Coarse
lev     relaxation     relaxation      grid method     Prolongation      Restriction
====================================================================================
  0             --         Jacobi          Glk-RAP         Diag Inv        Injection
  1        BJ-ILU0         Jacobi          Glk-RAI        Injection    Blk-ColLumped



Full Operator Matrix Hierarchy Information:

                     nonzero    actual          |       entries/row       |            ␣
↪rowsums
lev     rows     fine  entries   entries  sparse |  min  max    avg  stdev |       min     ␣
↪max       avg     stdev
==============================================+=========================+==============================
  0    5625     1875    77475     51023  99.755 |    3   21   13.8  14.19 | -7.1e+00  6.
↪3e+03  1.1e+03  1.6e+03         ( MGR )
  1    3750     1875    47900     38184  99.659 |    8   14   12.8  11.09 | -8.2e+02  3.
↪1e+03  4.2e+02  9.2e+02
----------------------------------------------+-------------------------+----------------
↪--------------------- MGR's coarsest level
  2    1875     1168    11975     11975  99.659 |    4    7    6.4   5.54 | -6.1e-04  2.0e-
↪03  3.4e-04  3.5e-04
  3     707      536    14669     14669  97.065 |    5   44   20.7  19.56 |  6.0e-05  4.1e-
↪03  8.9e-04  9.7e-04         ( AMG )
  4     171      126     4913      4913  83.198 |    7   65   28.7  27.70 |  1.2e-04  2.0e-
↪02  3.6e-03  4.7e-03
  5      45       45      999       999  50.667 |   11   38   22.2  19.72 |  1.7e-04  9.2e-
↪02  1.3e-02  2.1e-02



Full Prolongation Matrix Hierarchy Information:

                       nonzero    actual          |      entries/row      |            ␣
↪rowsums
lev     rows x cols      entries   entries  sparse |  min  max    avg stdev |       min     ␣
↪max       avg     stdev
================================================+=======================+==============================
  0    5625 x 3750         7500      7500  99.976 |    1    2    1.3   1.24 | -1.3e+00  1.
↪0e+00  2.5e-01  1.1e+00
  1    3750 x 1875         1875      1875  99.987 |    0    1    0.5   0.66 |  0.0e+00  1.
↪0e+00  5.0e-01  6.6e-01         ( MGR )
```

(continues on next page)

(continued from previous page)

```
------------------------------------------------+-----------------------+----------------
↪--------------------- MGR's coarsest level
  2    1875 x 707         4463     4463  99.873 |    1    4   2.4  2.42 |  2.8e-01  1.
↪0e+00  9.8e-01  8.4e-01
  3     707 x 171         1614     1614  99.677 |    1    4   2.3  2.27 |  2.0e-01  1.
↪0e+00  9.4e-01  8.1e-01        ( AMG )
  4     171 x 45           329      329  98.875 |    1    4   1.9  1.93 |  9.8e-02  1.
↪0e+00  9.1e-01  8.0e-01


At coarsest MGR level --> Solver parameters:

   Solver Type = BoomerAMG
   Strength Threshold = 0.300000
   Interpolation Truncation Factor = 0.000000
   Maximum Row Sum Threshold for Dependency Weakening = 0.900000
   Number of functions = 1
   Coarsening type = PMIS
   Prolongation type = MM-extended+e
   Cycle type = V(1,1)

MGR complexities:

 lev       grid   operator     memory
=====================================
   0      1.000      1.000      1.000
   1      1.000      1.000      1.000
   2      1.492      2.719      3.705
-------------------------------------
 All      2.164      2.038      2.191
-------------------------------------



L2 norm of b: 1.000000e+00
Initial L2 norm of residual: 1.000000e+00
=============================================

Iters     resid.norm     conv.rate  rel.res.norm
-----    ------------    ---------- ------------
    1    3.122166e-01    0.312217   3.122166e-01
    2    7.150505e-02    0.229024   7.150505e-02
    3    1.408777e-02    0.197018   1.408777e-02
    4    4.375035e-03    0.310556   4.375035e-03
    5    1.139532e-03    0.260462   1.139532e-03
    6    1.947384e-04    0.170893   1.947384e-04
    7    4.609606e-05    0.236708   4.609606e-05
    8    9.785908e-06    0.212294   9.785908e-06
    9    1.400809e-06    0.143146   1.400809e-06
   10    1.885831e-07    0.134624   1.885831e-07


Final L2 norm of residual: 1.885831e-07
```

(continues on next page)

(continued from previous page)

```
================================================================================

STATISTICS SUMMARY:

+------------+------------+------------+------------+------------+------------+
|            |   LS build |      setup |      solve |   relative |            |
|      Entry |      times |      times |      times |  res. norm |      iters |
+------------+------------+------------+------------+------------+------------+
|          0 |   3.93e-03 |   1.39e-02 |   1.15e-02 |   1.89e-07 |         10 |
+------------+------------+------------+------------+------------+------------+

Date and time: YYYY-MM-DD HH:MM:SS
${HYPREDRIVE_PATH}/hypredrive done!
```

CHAPTER

FIVE

FREQUENTLY ASKED QUESTIONS (FAQ)

This section provides answers to some of the most commonly asked questions about *hypredrive*.

5.1 What is *hypredrive*?

hypredrive is a high-level interface for solving linear systems with hypre.

5.2 How do I install *hypredrive*?

You can install *hypredrive* by downloading and compiling its source file. Please refer to *Installation* for detailed installation instructions.

5.3 Which linear system types can *hypredrive* solve?

hypredrive is capable of solving both symmetric and non-symmetric sparse linear systems. The specific capabilities depend on the underlying HYPRE library and the configuration of *hypredrive*.

5.4 How do I configure *hypredrive* for my specific problem?

You can configure *hypredrive* by creating a YAML configuration file. This file specifies all necessary settings, including the linear system, solver, and preconditioner configurations. For more information, see the *Input File Structure*. For examples of input files, see *Input File Examples*.

5.5 How can I contribute to *hypredrive*?

Contributions to *hypredrive* are welcome! You can contribute by filing issues, submitting pull requests or improving its documentation. Please refer to *Contributing* for guidelines.

5.6 Can I use *hypredrive* on GPU-accelerated systems?

Yes, *hypredrive* supports GPU acceleration. Note that *hypre* also needs to be compiled with GPU support and the keyword `exec_policy` under `general` must be set to `device`.

5.7 What should I do if I encounter an issue with *hypredrive*?

If you encounter an issue, you can open a GitHub issue. Providing detailed information about your problem, including configuration files, system details, and error messages, will help in resolving issues more quickly.

5.8 How is *hypredrive* licensed?

hypredrive is licensed under the MIT License. For more details, see the LICENSE file in the source distribution.