RelocDD-py is a software that contains three double-difference relocation methods:

(1) event-pair relocation [Waldhauser and Ellsworth, *BSSA*, 2000, https://doi.org/10.1785/0120000006],
(2) station-pair relocation [Zhang et al., *GRL*, 2010, https://doi.org/10.1029/2010GL043577], and
(3) double-pair relocation [Guo and Zhang, *GJI*, 2017, https://doi.org/10.1093/gji/ggw397].

Method 1 is applied most widely and can be used to improve relative event locations in most datasets. Methods 2 and Methods 3 require dense station networks. Generally, Method 2 only provides minor absolute location improvements. I recommend Method 3 for datasets with dense networks and with strong velocity effects (like directional anisotropy) in the source region. It can improve relative and absolute locations (per the Guo and Zhang paper). However, in practice, I have found no improvements in absolute locations in my work – although, in some cases, you can improve the relative locations over Method 1. Per Guo and Zhang 2017, for the best absolute and relative locations you want a two-step approach by first relocating using station-pair and then relocate using double-pair.

Some case studies showing results from this program include:

(1) Vasyura-Bathke, *Communications Earth & Env.*, 2023, https://doi.org/10.1038/s43247-023-01064-1.
(2) Biegel et al., *Tectonics*, 2024, https://doi.org/10.1029/2023TC008140.
(3) Biegel et al., *SRL*, 2024, https://doi.org/10.1785/0220240194.

The format of the code is based on the HypoDD format. This includes the variables in the Ph2DT and HypoDD input files. I recommend looking through the HypoDD manual for questions about specific variable values (https://www.ldeo.columbia.edu/~felixw/papers/Waldhauser_OFR2001.pdf).

Because relocDD-py is in Python, it is slower than the original hypoDD code in Fortran. I recommend using it only for small to medium-sized problems (less than about 5,000 events depending on the station array size).

I've uploaded example input files here:
https://www.dropbox.com/scl/fo/h8ax0rc63hk2hj1ccn5cs/h?rlkey=78bofcxuy5jr3is1oob7hvn8l&dl=0
For these files:

- Run.inp file controls the general run process.
- Ph2dt.inp is the updated ph2dt input file. You will notice an extra value for station-pair separation compared to the standard input file.
- HypoDD.inp is the hypoDD input file and is unchanged from that used in running standard hypoDD.

There are several commands you can run:

1. To run the equivalent of ph2dt or hypoDD from the hypoDD code, you should use:

   python /path/to/run.py /path/to/run.inp [ph2dt switch (0 to skip, 1 to run)] [hypoDD switch (0 to skip, 1 to run)] 0 0

   The last two switches are depreciated and should just be set to 0.
2. To run the damping selection (to automate the damping parameter selection), use:

   python /path/to/dampingtest.py /path/to/run.inp

   You will then have to enter the damping value range to test over. I usually do 0 to 3 (log-scale, so 1 to 1000) and then the number of the damping values (I typically do 20). You will then have to plot the damping values using an L-curve test. I have this as a Jupyter Notebook, which I also added to the Dropbox folder.

3. To run the bootstrap or the jackknife tests requires mpi4py. You want to run those from the main relocation folder since the paths are hardcoded. These are post-processing tests to access the uncertainty and robustness of the relocation solution.

Other people have tested the phd2dt and hypoDD functions (in the run.py script), so they should run. However, the damping test and the statistical analysis (bootstrapping and jackknifing) are untested, so you may encounter issues. If you happen to have any issues, please let me know so I can update those in the main code base – katherine.biegel@ucalgary.ca or kbiegel@uoregon.edu

I do ask that if you end up using the relocation software in any publications to please make sure to reference the software doi which is https://doi.org/10.5281/zenodo.10607406.