* This folder contains the code to compute the **DDM probability distribution function** for *any space- and time- varying drift (mu), noise (sigma), and bounds (B)*.
* **DDM\_main.py** is the actual code to run, loading DDM\_parameters.py and DDM\_functions.py.
* The **actual DDM model**, contained in **DDM\_functions.py**, solves for the *Fokker-Planck Equation* of the DDM model of interest, using the *implicit* numerical method
  + (with Dirchlet boundary conditions and staggered-mesh grids)
* DDM\_main\_load\_fit.py is the same as DDM\_main.py, but also contains, at the start, some sample codes to **load data for fitting**.
* DDM\_sim\_compare\_pdf.pycontains a simple code to **simulate** a DDM modelw with *space- and time- varying drift (mu), noise (sigma), and bounds (B)*, in order to **verify the code**
  + Results are shown in the folder **Compare\_num\_sim\_varying\_mu\_sig\_B**.
* **Few Final things to note:**
  + The simulation code is **verified** by comparing to analytical solutions (see code) for constant mu,sigma, and constant or linearly collapsing bounds.
    - It is also compared to space- and time- varying, linear and non-linear, mu/sig/B.
    - Results are shown in the folder **Compare\_num\_sim\_varying\_mu\_sig\_B**.
  + The end of the **main** code involves particular “tasks” which perturb the systems in different ways. The codes for such tasks were structured in a way to allow maximal flexibility.
    - Not sure if it is useful to you guys so I left it there. But feel free to delete it.
    - Similarly, the **functions** code contain, at the end, functions that are no longer of use but might prove useful. Feel free to delete it.
    - The simulation code also contains at the end codes which can be deleted (median reaction time trajectories).
  + The original code is designed for **Fixed Time task**, which can be easily, but still has to be, converted to reaction time tasks
    - The **fitting** is different too, we’re just fitting to win/lose/undec, while they should fit to correct and incorrect probability simultaneously.
      * I’ve primarily set this up in the function **MSE\_model\_fit\_RT**, but further work is needed to make sure it works and to optimize it, ito fitting metric & data structure.
      * The main code is not **structured** in the best possible way (summer school…so whatever was the fastest and easiest…)…**perhaps you’d want to totally redo**.