Author: Zach Philip
Date: June 2025
Property of Whisper Aero

Documentation and Instructions for Pak Interface Program (Python)

Purpose:

This program was designed to create an interface to the C program provided by MBBM so that Python packages could be applied to the data. A Python interface allows for easy, flexible, and reliable modification of the data. The main.py file reads in the binary data from the input file, processes it using the desired function in my_function.py and writes it to the output file.

The program makes the following assumptions:

1. nz = 1
2. n_data_arrays = 1
3. n_data_sets = 1
4. The ydata is the only data to process

Each of these limitations can easily be accounted for by slight manipulation of the python files. The c code should not need to be edited.

User Instructions:

1: Installs (Required)

Python:

Ensure the latest version of Python is installed. If not, download here and follow these instructions (for Windows).

1. Click download and run the installer once it's downloaded.
2. On the first screen, check the box that says: “Add Python to PATH.”
3. Click “Install Now” and follow the prompts.
4. After installation, verify it worked by opening a terminal window and typing:
   ```
   Python --version
   ```
5. You should see something like: Python 3.13.0
6. Then, verify pip is installed properly by running
   ```
   pip --version
   ```
7. You should see something like: pip 24.0 from < your directory >

VS Code Extension:

This guide assumes that VS Code is properly installed and configured. Be sure the Python extension by Microsoft is installed. Do this by navigating to extensions on the left tool bar (4 square blocks) and typing in “Python.”

Author: Zach Philip
Date: June 2025
Property of Whisper Aero

C Code: Not Required

If you're not looking to use the C code (most use cases), skip to step 2.

The instructions below allow a user to edit, run, and compile the C code provided by MBBM. If the DLL ever needs to be changed, instructions below will explain how to do it. As of June 2025, the DLL is working properly and grants all desired functionality to the Python user. Note: the C files have been slightly edited by Zach Philip.

These instructions will give a rough outline of what to download and install to use the C code. More specific instructions, if errors occur, can be found online.

Zip File:

The zip file with C code includes the rw files and the main.c file. There is also a provided sin_wave.pak52 file to use as an example input.

VS Code extensions:

Install C/C++ by Microsoft, found under extensions on left tool bar.

C Compiler:

MinGW-w64: Download it here.

During the install, follow these instructions:

1. During install: Architecture: x86_64
2. Threads: posix
3. Exception: seh
4. Build revision: latest
5. Add the bin directory to your system PATH:
   Example: C:\Program Files\mingw-w64\x86_64-posix-seh\mingw64\bin
6. Open a new terminal and check:
   ```
   gcc –version
   ```
7. Confirm output looks something like:
   ```
   gcc (x86_64-posix-seh, Built by MinGW-W64 project) 13.2.0
   Copyright (C) 2023 Free Software Foundation, Inc.
   This is free software; see the source for copying conditions.
   There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
   PURPOSE.
   ```

Running The Code:

Once the compiler and extensions are set up in VS code, the code can be run. First, the user must compile the code into an executable that can be run. Do this by running the following command into the terminal:

```
gcc main.c rw_data.c -o main
```

Author: Zach Philip
Date: June 2025
Property of Whisper Aero

This will create an executable and place it into your working directory. Any errors are likely due to directories, so make sure you run this command from your working directory. Note: if you want to run in debug mode, add "-g" in front of main.c and you can step through the program.

Once the main.exe file is made, run the code using a launch.json file. Here is an example launch.json file:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Debug main.exe",
      "type": "cppdbg",
      "request": "launch",
      "program": " ", // Path to and including your main.exe
      "args": ["input.bin", "output.bin"], // ifile and ofile
      "stopAtEntry": false,
      "cwd": " ", // Working directory one level higher than executable
      "environment": [],
      "externalConsole": true,
      "MIMode": "gdb",
      "miDebuggerPath": "C:/msys64/ucrt64/bin/gdb.exe", // may be different on user system
      "setupCommands": [
        {
          "description": "Enable pretty-printing",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ]
    }
  ]
}
```

Be sure to edit the file with the correct directories and file names. Then use the run button to execute the main.c. Check that it worked by verifying your input file matches your output file. Do this by opening the files in the "Hex Editor." Get there by clicking the file, clicking "Open Anyway" and selecting "Hex Editor."

Creating a DLL:

A DLL allows the python code to access the C functions. To create one, go to the rw_data.h file and uncomment the statements commented out by Zach Philip. Then type the following command into the command line to create the DLL:

```
gcc -shared -o pak_lib.dll rw_data.c
```

The DLL should be created, and you can copy it over to your working directory in Python.

2: Edit Files

Author: Zach Philip
Date: June 2025
Property of Whisper Aero

The user should only edit main.py and my_function.py.

In main.py, there are only two places to edit. Comments are in place to guide the user to where to edit.

First, edit the file names. Assign the variable "ifile" to your input file's name as a string, as shown in the program. Set the name of the output file to your desired string, as shown.

Second, the user must call their processing function that is written in my_function.py. Assign the variable "filtered_data" to the return value of your processing function. This variable is of type NumPy array. The user must ensure they're passing the proper parameters and returning the proper value type.

In my_function.py the user must define and implement their scaling function, paying close attention to parameters and return type. An example using the Gaussian filter is provided. Note: user will likely have to add additional import statements based on what packages they're using.

Input files:

Ensure the input file is in the same directory as main.py. The input file should be in the PAK binary format. Its name should not contain spaces or non-ASCII characters.

Imports:

All third-party libraries are documented in requirements.txt. Other imports are in the standard Python library or are local modules and do not need to be installed separately. After the user has Python installed, run the following command in the terminal from the project folder to install all packages from the requirements.txt file:

```
pip install -r requirements.txt
```

If the user needs to install additional packages for their my_function.py function, use the following command:

```
pip install package_name
```

3: Using the Python code

Running the code:

The user can run the code from the terminal or from the text editor's built-in run button. From the terminal, navigate to your project folder and use the following command to run the main.py file (assuming Python version 3):

```
python3 main.py
```