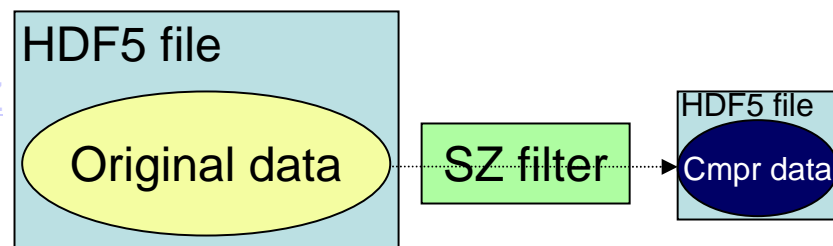


# H5Z-SZ: Handling HDF5 with SZ filter

- Download and install HDF5
- Download SZ and install SZ
  - git clone <https://github.com/szcompressor/SZ>
- Compile [SZ-package]/H5Z-SZ
  - Set SZPATH and HDF5PATH in Makefile
  - cd H5Z-SZ; make; make install
  - Add \$SZPATH/lib and \$HDF5PATH/lib in LD\_LIBRARY\_PATH
  - export HDF5\_PLUGIN\_PATH=\${SZ\_INSTALL\_PATH}/lib
- Quick start:
  - Use-case A with library:
    - (1) cd [SZ-package]/H5Z-SZ/test;
    - (2) Set SZPATH and HDF5PATH in Makefile
    - (3) make (Two executables will be generated: szToHDF5 and dszFromHDF5)
    - (4) szToHDF5 will load a 3d array and then write the compressed bytes in a HDF5 file. (See test\_compress.sh for details)
    - (5) dszFromHDF5 will read the HDF5 file generated by test\_compress.h and then decompress the data inside it. (See test\_decompress.sh for details)
  - Use-case B with plugin:
    - (1) Put the sz.config configuration in the current directory. (Please see README in SZ to understand the configuration sz.config)
    - (2) h5repack.sh [input\_hdf5\_file] [compressed\_hdf5\_file] or  
h5repack -f UD=32017,0 [input\_hdf5\_file] [compressed\_hdf5\_file]
    - (3) Read the compressed HDF5 file:  
h5dump [compressed\_hdf5\_file] > data.txt
    - (4) Decompress the data and dump them to a HDF5 file.  
h5repack -f NONE compressed.h5 decompressed.h5



# H5Z-SZ: API

- C programming API:

- In order to set the HDF5 SZ filter in C/C++, the key is generating `cd_values[]` for the filter. There are two ways to do so:

- `void SZ_metaDataToCdArray(size_t* cd_nelmts, unsigned int** cd_values, int dataType, size_t r5, size_t r4, size_t r3, size_t r2, size_t r1);`
- `void SZ_metaDataErrToCdArray(size_t* cd_nelmts, unsigned int **cd_values, int dataType, size_t r5, size_t r4, size_t r3, size_t r2, size_t r1, int error_bound_mode, float abs_error, float rel_error, float pw_rel_error, float psnr);`

**Input:** `dataType`: SZ\_FLOAT, SZ\_DOUBLE, SZ\_INT16, SZ\_INT32, etc. (The complete data type definitions can be found in SZ/sz/include/defines.h: line 43~52)

`r5`, `r4`, `r3`, `r2`, `r1`: the size of each dimension. If the dataset is a 3D array (100x200x300 in C style), the 5 dimension sizes should be 0, 0, 100, 200, 300.

`error_bound_mode`: ABS, REL, PW\_REL, or PSNR. (More modes can be found in SZ/sz/include/defines.h: line 29~41)

`abs_error`: absolute error bound

`rel_error`: value-range based relative error bound

`pw_rel_error`: point-wise relative error bound

`psnr`: peak signal to noise ratio (PSNR).

**Output:** `cd_nelmts`: the number of elements in `cd_values[]`

`cd_values`: the elements of `cd_values[]`

# H5Z SZ: API (Cont'd)

**Example:** Please check the example `./test/szToHDF5.c` to understand how to set hdf5 filter appropriately. It also demonstrates how to set different error bounds for specific fields/variables.

If you use **SZ\_metadataToCdArray()**, the compression errors will be loaded from `sz.config` (which should be put in the current directory). In this situation, you need to set the global variable **load\_conf\_file\_flag** to 1 before hand.

If you want to set particular error bound for each field/variable (a.k.a., one dataset) in one HDF5 file, you can use **SZ\_metadataErrToCdArray()**. Correspondingly, you should set **load\_conf\_file\_flag** to 0 before hand. (`load_conf_file_flag` is set to 1 by default)

- After you specify the `error_bound_mode`, only the corresponding error bounding parameter will be valid and all others will be ignored.
- For example, if `error_bound_mode` is set to `ABS`, only the `abs_error` argument is valid, and other arguments such as `rel_error`, `pw_rel_error` and `psnr` could be set to 0 for simplicity.
- Please check `./test/szToHDF5.c` (line 94) for details:
  - `SZ_metadataErrToCdArray(&cd_nelmts, &cd_values, dataType, r5, r4, r3, r2, r1, ABS, 0.01, 0, 0, 0);`

# Trouble Shooting

1. Core-dump segmentation fault issue when trying to compress data by h5repack.

**Answer:** I noted that if I use hdf5-1.12.0 on Fedora Linux, it may encounter the core-dump issue. But after installing hdf5-1.10.1 and compiling by it, everything works well.

2. The compressed data has the same size with the original file size.

**Answer:** There might be two reasons:

- (1) The first thing you need to check is HDF5\_PLUGIN\_PATH and make sure libhdf5sz.so could be found there. If there is anything incorrect about setting the environment variable HDF5\_PLUGIN\_PATH, it will have this 'same-size' issue.
- (2) If you are sure the HDF5\_PLUGIN\_PATH is set correctly, please change your hdf5 version. My experience is that hdf5-1.10.1 works fine but hdf5-1.12.0 doesn't. As for this reason, you can simply change the filter to any other filters such as lossless compression filter embedded in hdf5, and you will observe the same issue.

Any questions?

Please contact Sheng Di

at [szlossycompressor@gmail.com](mailto:szlossycompressor@gmail.com)

or [sdi1@anl.gov](mailto:sdi1@anl.gov)