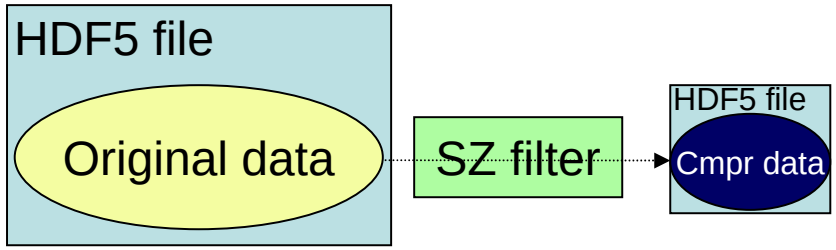
# H5Z-SZ: Handling HDF5 with SZ filter

- 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 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
  - Use-case B 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)

# H5repack with plugin

As demonstrated in the first slide, you can use h5repack command with plugin to compress the data in a hdf5 file and then store the data into another hdf5 file. There are two ways to do so:

**Option 1:** load the error bound information from sz.config (which should be stored in the current local directory where you are operating the h5repack command)

**Option 2:** you can use h5repack cd_values parameters. To this end, you can use example/print_h5repack_args.c to generate the parameters based on the error bound information. (you need to modify example/Makefile before building the codes in examples/.

Example: print_h5repack_args -M ABS -A 1E-3 (output: -f UD=32017,0,0,981668463,0,0,0)

# 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.
    - void **SZ_errConfigToCdArray**(size_t* cd_nelmts, unsigned int **cd_values, 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)

    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 want to set a particular error bound for each field/variable (a.k.a., one dataset) in one HDF5 file, you can use **SZ_errConfigToCdArray()**.

- 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 90) for details:
  - SZ_errConfigToCdArray(&cd_nelmts, &cd_values, 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 fines 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

or sdi1@anl.gov