Kaon LT Status Update

April 30th, 2019

Richard Trotta

How should I analyze data?

- Doing things locally will always be your best option for actual analysis.
  - Fork a repo of hallc_replay_kaonlt **and** UTIL_KAONLT for your own custom version that you can play with

How should I analyze data?

- Once you have forked the repo, clone hallc_replay_kaonlt to a local directory

```
trottar-VB ~> git clone https://github.com/JeffersonLab/hallc_replay_kaonlt.git
```

- Now the tricky part, UTIL_KAONLT is a submodule of hallc_replay_kaonlt so some intermediate steps will need to be made

```
Branch-[local]
trottar-VB ~/Analysis/hallc_replay> git submodule --init --recursive
```

  - Check .gitmodules to make sure submod is listed

```
[submodule "UTIL_KAONLT"]
        path = UTIL_KAONLT
        url = https://github.com/JeffersonLab/UTIL_KAONLT
        branch = develop
```

```
Branch-[local]
trottar-VB ~/Analysis/hallc_replay> git submodule update --recursive --remote
```

How should I analyze data?

- If HEAD is detached (check with git branch -a)

```
Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> cd UTIL_KAONLT/
```

```
Branch-[HEAD]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git branch -a
* (HEAD detached at a0d1e24)
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/develop
  remotes/origin/fall-2018-ver2
  remotes/origin/master
  remotes/origin/offline
  remotes/origin/spring-2019-ver1
  remotes/origin/spring-2019-ver2
  remotes/origin/spring-2019-ver3
  remotes/origin/spring-2019-ver4
```

How should I analyze data?

- Check if head is really detached

```
Branch-[HEAD]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git symbolic-ref HEAD
fatal: ref HEAD is not a symbolic ref

Branch-[HEAD]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git remote update
Fetching origin
```

- Change to master branch

```
Branch-[HEAD]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

How should I analyze data?

- Pull and check branch again, everything should be set!

```
Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git pull
Already up-to-date.

Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test/UTIL_KAONLT> git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/develop
  remotes/origin/fall-2018-ver2
  remotes/origin/master
  remotes/origin/offline
  remotes/origin/spring-2019-ver1
  remotes/origin/spring-2019-ver2
  remotes/origin/spring-2019-ver3
  remotes/origin/spring-2019-ver4
```

How should I analyze data?

- Now that we have our repo locally we should set it up to pull from the “main” JeffersonLab version
- First check your remote “origin” repo (this is where you will push to)

```
Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> git remote -v
origin  https://github.com/trottar/hallc_replay_kaonlt.git (fetch)
origin  https://github.com/trottar/hallc_replay_kaonlt.git (push)
```

- Next lets set up the “upstream” which is the JeffersonLab repo (DO NOT push HERE)

```
Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> git remote add upstream https://github.c
om/JeffersonLab/hallc_replay_kaonlt.git

Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> git remote -v
origin  https://github.com/trottar/hallc_replay_kaonlt.git (fetch)
origin  https://github.com/trottar/hallc_replay_kaonlt.git (push)
upstream        https://github.com/JeffersonLab/hallc_replay_kaonlt.git (fetch)
upstream        https://github.com/JeffersonLab/hallc_replay_kaonlt.git (push)
```

How should I analyze data?

- You will not be able to push to upstream unless you're Stephen or me so don't worry too much. Just be cautious.
- A similar procedure can be performed with UTIL_KAONLT
- Finally, let's talk about branches. Let's add the develop branch to our local system…
    - First create a branch locally called develop and change to it

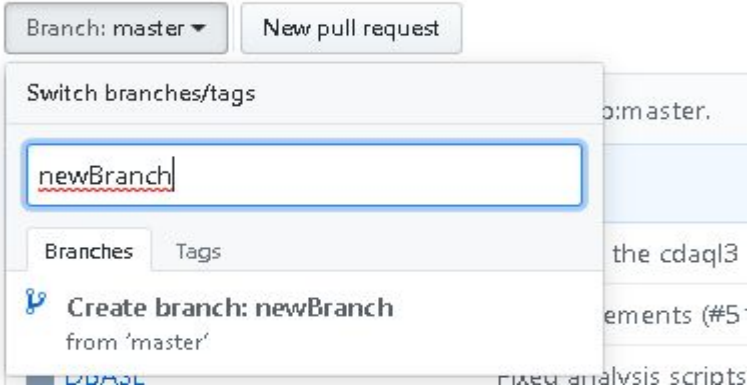
```
Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> git branch develop

Branch-[master]
trottar-VB ~/Analysis/hallc_replay_test> git checkout develop
M       UTIL_KAONLT
M       UTIL_OL
Switched to branch 'develop'
```

- Now simply pull develop

```
Branch-[develop]
trottar-VB ~/Analysis/hallc_replay_test> git pull origin develop
```

How should I analyze data?

- To create a new branch you must first create it in github



- Then simply repeat the steps for setting up a branch from the previous slide

Replaying

- Before we can analyze we must replay. This should be done in the farm to save yourself time and local cpu effort. The easiest way is to do a batch job submission, but this comes with some prep work.
- Before a batch submission, I highly encourage two preliminary steps
    1. Do all debugging of replays locally, once this works move to the farm
    2. Once on the farm you have two options; your ifarm version or our group (discussed soon). This is for final debugging purposes to assure everything works in the farm, then you can submit a batch job. Save the root files in /volatile/hallc/c-kaonlt/<USER> (note: volatile is NOT backed up)
- There is a batch script I have created and Stephen as changed with the help of Brad to assure it will not mess things up. Again, I highly recommend the two above steps before moving onto this script or you will be wasting time and resources.
- Your final batch job submissions can be saved directly to tape.

Group environment

- You can do replays under your farm directory or you can use our group environment.
- We have set up a group environment with a version of our repo that currently mimics the cdaq as close as possible (although an updated hcana is used).
  - This group environment is under /u/group/c-kaonlt
  - I have made a directory USERS which you can use for person replay scripts and environments. DO NOT change any replays that are not under USERS without contacting Stephen or me first.
  - There is an hcana already set up here, use this for any group replays. If you would like to use a different version of hcana please use your farm directory. If I find a hcana in USERS I will destroy it.
- You may have issues with hcana, make sure you are in the JLab software environment version 2.1
  - source /site/12gev_phys/softenv.csh 2.1 (or .sh if using bash)

Group environment

- The group environment has a 100 gb quota and is backed up. This means two important things…
  1. DO NOT save root files here! Ever!
  2. It's backed up so its good for important calibration work (*wink *wink)
- Upon the request of Stephen, any improper use of this environment will incur a penalty of one beer/bottle of single malt or an owl shift (depending upon severity).

Writing to tape

- Writing to tape info, read - https://scicomp.jlab.org/docs/write-through-cache.
- In your batch script, specify OUTPUT_FILE:/cache/hallc/kaonlt/USER/ROOTfiles/FILE.
  - Material in /cache is automatically copied to tape after some time if it is static
  - Small files (~1 MB) will not be backed up on tape
  - Once copied to tape, you can view the tape stub (NOT the file itself) under /mss/hallc/kaonlt/…
  - The tape does not handle overwriting well so if submit a job you must create a new "pass" directory…
    - ➢ jput ... file.root /mss/hallc/kaonlt/USER/ROOTfiles/pass1/
  - The tape has FAR more space than we could get through so do not worry about "filling" it
  - Write to tape once you're happy with your code... just do it correctly

Few more words of warning

- Do not write analysis to tape unless you are 100% certain it works correctly (and you don't want to repeat it very soon).
- For farm jobs some info is included below -
  - See https://scicomp.jlab.org/docs/text_command_file for info on commands
  - Do not set CPU above 1 (it will slow your job down in the queue and hcana is single threaded anyway so you gain nothing)
  - Farm/Auger project: c-kaonlt
  - For TEMPORARY output, write to volatile - /volatile/hallc/c-kaonlt/USER, this space is NOT backed up!
  - Specify the FULL path to this in your symbolic link
  - Make sure relvant directories are created
- You can use our work environment (/work/hallc/kaon), but this is not backed up and I will no be setting up an environment similar to group there. It’s a good place to put personal scripts if you don’t want to take up space in your farm directory.