**Steps to validate any pull request for REMOLL**

REMOLL is the GEANT-based software dedicated for physics and geometry optimization simulation for MOLLER experiment The MOLLER collaboration uses GitHub for the version control of the REMOLL. If you are not familiar with the GitHub functionalities, then you can take help from this link: <https://docs.github.com/en/get-started.> Pull Requests (PR) contain specific modifications of some specific directory or file in the simulation, e.g., realistic material budget implementation, realistic active and passive volume implementation etc. It is particularly important to understand the effects of the newly made changes in the simulation on the result. Therefore, it is advised to do a validation study whenever some significant number of changes are requested to be merged with the main(/develop) branch (which is the up-to-date branch and should be used for any simulation studies).

To perform a PR validation study, you need to have a clear understanding of what PR version needs to be checked and what are the modifications in there. First login to the ifarm. Go to your working directory (**cd** **/w/halla-scshelf2102/moller12gev**). After login into your work directory, you need to source the required software/dependencies to run your simulation. To do that use the command,

**source /site/12gev\_phys/softenv.csh 2.5.**

The latest REMOLL version can be downloaded from GitHub using the command:

**git clone** [**https://github.com/JeffersonLab/remoll.git**](https://github.com/JeffersonLab/remoll.git) **remoll**.

After that if you go inside the remoll directory (**cd remoll**) and do git status it should show the following lines;

**On branch develop**

**Your branch is up to date with 'origin/develop'.**

**nothing to commit, working tree clean**

Here you need to checkout to the branch (branch\_name) that you need to validate using the command;

**git checkout branch\_name.**

After that to make sure you can use the **git status** command to cross-check that you are in the desired branch. After that, you need to build REMOLL using the instructions here <https://github.com/JeffersonLab/remoll/blob/develop/README.Compiling.md.> If you need to work with several PRs at a time, it is advisable to use separate build directories inside remoll just to avoid confusions. Once you are ready with your REMOLL installation, you need to proceed to submit simulation jobs at ifarm. To validate the PRs, you may need to do different analysis, but the common thing that you need to do is to perform the deconvolution analysis and for that you need to run your simulation with different physics event generators (**elastic, inelastic, moller, elasticAl, inelasticAl, quasielasticAl, pion**). If you need to study the backgrounds, then we will need to use the **beam** generator. The details for simulating files with different event generators are discussed below.

A job submission script is available in the following path at ifarm which can be accessed by anyone;

**/w/halla-scshelf2102/moller12gev/commonFiles/deconvolutionStudyMarch2023/remoll-job-submission/simulation/macro.py**

To get an idea about the parameters needed to run the macro can be obtained by using the command,

**python macro.py --help**

and it should give you the following output,

**usage: macro.py [-h] -a ACCOUNT [-s SRC] [-v VERSION] -j JSUB\_DIR -t TMP\_DIR -o OUT\_DIR**

**[-g GEN] [-d DET\_LIST [DET\_LIST ...]] [--bhd BHD\_LIST [BHD\_LIST ...]] [-r RUN\_RANGE]**

**[-n N\_EVENTS] [--time TIME] [--umap UMAP] [--dmap DMAP] [--targ TARG]**

**[--pion PION] [-w WORK\_DIR] --voff VOFF --scale SCALE**

**Submit array jobs to ifarm.**

**optional arguments:**

**-h, --help show this help message and exit**

**-a ACCOUNT Enter the Jefferson Lab account. Example: halla**

**-s SRC source folder where simulation directory exists**

**-v VERSION choose the version of simulation to use.**

**-j JSUB\_DIR choose directory to write the slurm submission scripts**

**-t TMP\_DIR choose directory to write the slurm output logs**

**-o OUT\_DIR choose where to write the output root files**

**-g GEN choose generator to use. Options are moller, elastic, inelastic, beam, etc.**

**-d DET\_LIST [DET\_LIST ...] provide list of sensitive detectors separated by space. Example: 28 29.**   **By default, all detectors detect low energy neutrals and secondaries.**

**--bhd BHD\_LIST [BHD\_LIST ...] provide list of boundary hit detectors. By default only detector 28 is**   **boundary hit detector**

**-r RUN\_RANGE provide run range. Example: "2-5"**

**-n N\_EVENTS provide number of events per job in the array**

**--time TIME provide the estimated run time. Ex: "00:25:00". Usually it is 10 minutes for 1000**   **moller events.**

**--umap UMAP provide the map name for upstream magnet. Ex: V2U.1a.50cm.parallel.txt**

**--dmap DMAP provide the map name for downstream magnet. Ex: V2DSg.9.75cm.parallel.txt**

**--targ TARG provide the target name.**

**--pion PION provide the pion name.**

**-w WORK\_DIR Enter location where analysis takes place. Choose:/scratch/slurm.**

**--voff VOFF provide offset value. Options: -2 to 2 for xoffsets, -1 to 1 for angle offsets**

**--scale SCALE provide magnetic field scale factor.**

Just to give an example, a sample job submission using macro.py script should look like this, (similar for **moller, elastic, inelastic generators.** Only need to change the generator name after **-g**)

python macro.py -a halla -s ../../ -v remoll -j /volatile/halla/moller12gev/sayak/remoll/moller/jsub -t /farm\_out/sayak/remoll/moller/tmp -o /volatile/halla/moller12gev/sayak/remoll/moller/out -g moller -d 28 --bhd 28 -r 1-100 -n 75000 --time 10:00:00 --umap V2U.1a.50cm.parallel.txt --dmap subcoil\_2\_3\_3mm\_full.txt -w /scratch/slurm/ --voff 0.0 --scale 1.0

**NOTE: -a s**hould always be halla**, -s** should always point to the directory where you have remoll**, -v** should reflect the remoll version you want to use and that should be in your –s directory, **-j** should point to someplace in /volatile/halla/moller12gev, **-t** should point to someplace in /farm\_out/, **-o** should point to someplace in /volatile/halla/moller12gev, **-g** should be the generator you want to use for your simulation, **-d** the detector list should contain 28 and if you need anything else you will be informed, **--bhd** 28 has to be registered as boundary hit detector and if anyother detector needs to be included here, you’ll be advised accordingly, **-r** run range which maximum can go upto 1000 files at a time, **-n** specifies the number of events in each run, **-time** estimated time to execute each run, **--umap and –damp** are the upstream and downstream magnetic field maps. If it is not clear what file you should use then ask any expert or the persons created the specific PR, **--targ** this needs to be set only for elasticAl, inelasticAl and quasielasticAl, **--pion** only needs to be set while using pion generator, **-w** should be /scratch/slurm always, --voff 0.0 and –scale 1.0

Now if you want to use **pion** generator, then you need to add this one extra line while submitting jobs at ifarm;

python macro.py -a halla -s ../../ -v remoll -j /volatile/halla/moller12gev/sayak/remoll/pion/jsub -t /farm\_out/sayak/remoll/pion/tmp -o /volatile/halla/moller12gev/sayak/remoll/pion/out -g pion -d 28 --bhd 28 -r 1-100 -n 50000 --time 10:00:00 --umap V2U.1a.50cm.parallel.txt --dmap subcoil\_2\_3\_3mm\_full.txt --pion pi- -w /scratch/slurm/ --voff 0.0 --scale 1.0

Now if you want to use **elasticAl, inelasticAl and quasielasticAl** generator, then you need to add target information while submitting jobs at ifarm;

python macro.py -a halla -s ../../ -v remoll -j /volatile/halla/moller12gev/sayak/remoll/elasticAl/usal/jsub -t /farm\_out/sayak/remoll/ elasticAl/usal//tmp -o /volatile/halla/moller12gev/sayak/remoll/ elasticAl/usal//out -g elasticAl -d 28 --bhd 28 -r 1-100 -n 75000 --time 10:00:00 --umap V2U.1a.50cm.parallel.txt --dmap subcoil\_2\_3\_3mm\_full.txt --targ USAl -w /scratch/slurm/ --voff 0.0 --scale 1.0

python macro.py -a halla -s ../../ -v remoll -j /volatile/halla/moller12gev/sayak/remoll/elasticAl/dsal/jsub -t /farm\_out/sayak/remoll/ elasticAl/dsal//tmp -o /volatile/halla/moller12gev/sayak/remoll/ elasticAl/dsal//out -g elasticAl -d 28 --bhd 28 -r 1-100 -n 75000 --time 10:00:00 --umap V2U.1a.50cm.parallel.txt --dmap subcoil\_2\_3\_3mm\_full.txt --targ DSAl -w /scratch/slurm/ --voff 0.0 --scale 1.0

**NOTE: For elasticAl, inelasticAl and quasielasticAl, you will need to submit jobs with –-targ USAl and –-targ DSAl separately.**

Once all the jobs are submitted you can check the job progress using the following command,

**squeue -u username**

Once all the simulation files are ready then you need to perform analysis. To do that, go to the following directory,

**cd /w/halla-scshelf2102/moller12gev/commonFiles/deconvolutionStudyMarch2023/remoll-job-submission/analysis**

There you will see a directory deconvolution inside which there is a macro named **shieldingAna.C** which is used for the deconvolution analysis. To run the analysis you will find a python script named analyse.py. To see the usage of the script just type **python analyse.py --help.** It will give you the following output,

**usage: analyse.py [-h] [--home HOME] -s SRC\_DIR -j JSUB\_DIR -t TMP\_DIR -o OUT\_DIR [-g GEN] [--time TIME] [-r RUN\_RANGE] [-w WORK\_DIR]**

**Submit array analysis jobs on ifarm. This script can be further improved.**

**Optional arguments:**

**-h, --help show this help message and exit**

**--home HOME**

**-s SRC\_DIR source folder where simulation directory exists**

**-j JSUB\_DIR choose directory to write the slurm submission scripts**

**-t TMP\_DIR choose directory to write the slurm output logs**

**-o OUT\_DIR choose where to write the output root files**

**-g GEN choose generator to use. Options are moller, elastic, inelastic, beam, etc.**

**--time TIME provide the estimated run time. Ex: "00:25:00". Usually it is 10 minutes for 1000 moller events.**

**-r RUN\_RANGE provide run range. Example: "2-5"**

**-w WORK\_DIR Enter location where analysis takes place. Choose: /scratch/slurm**

After having all the simulated files analyzed, you need to check how many files are good or corrupted. To do that, you need to use the macro correct.py at (**/w/halla-scshelf2102/moller12gev/commonFiles/deconvolutionStudyMarch2023/remoll-job-submission/simulation/**)

To run the macro, do the following command,

python –c path of the simulated output files (set in the earlier step using **-o**) -g generator

This will give you the number of good and corrupted files in the respective directories containing the simulated output files. (Thess numbers are important and will be required during deconvolution analysis)

**NOTE: After the analysis, you need to add the analyzed root files into a single root file using the hadd command.**

**hadd** **outputfile inputfiles …**

**In case of elasticAl, inelasticAl and quasielasticAl you have to hadd seperately for the target setting USAl and DSAl. After that you need to hadd the root files from USAl and DSAl. Make sure you have same number of analyzed files in the USAl and DSAl before doing the hadd (if say USAl has 100 files and DSAl has 99 then delete one file from USAl before doing the hadd). So at the end, for each generators you will have one analyzed root file (output file after hadd)**

Now to do the deconvolution analysis, you must use the script at

**/w/halla-scshelf2102/moller12gev/commonFiles/deconvolutionStudyMarch2023/remoll-job-submission/plot/deconvolution/deconvolution.C**

**NOTE: Do not change the ordering of the input files in the deconvolution.C script. Depending on the number of files you have, the rate factor needs to be modified, for example it can be done in the script as following,**

**double rateFactor;**

**if(proc==0){ rateFactor = 70.0/85.0\*1.0/99.0; }**

**if(proc==1){ rateFactor = 70.0/85.0\*1.0/98.0;}**

**if(proc==2||proc==3||proc==4){ rateFactor = 70.0/85.0\*1.0/99.0; }**

**if(proc==5) { rateFactor = 70.0/85.0\*1.0/99.0; }**

**if(proc==6) { rateFactor = 70.0/85.0\*1.0/99.0;}**

**if(proc==7) { rateFactor = 70.0/85.0\*1.0/99.0; }**

**if(proc==8) { rateFactor = 70.0/85.0\*1.0/100.0;}**

**Where, proc=0 => moller, proc=1=>elastic, proc2,3,4=>inelastic, proc5 => elasticAl,**

**proc6 => quasielasticAl, proc7 =>inelasticAl, proc8 => pion and the no of files should go as 1.0/no of files in the rateFactor defination.**

By default, the deconvolution script uses the primary particles but if you need to do the deconvolution analysis with all particles then you need to change the **eP1** histograms to **e1** histograms. [So just go to the deconvolution.C script and search for eP1 and replace them all by e1]

Once you run the deconvolution.C script, it will generate some output like this,

Overall

Name Asymmetry uncert[ppb] relative uncer[ppb]

moller -34.8475 3.0925 -0.0887

epElastic -29.4329 19.9494 -0.6778

epInelasticW1 -522.3339 869.3186 -1.6643

epInelasticW2 -547.2616 520.2827 -0.9507

epInelasticW3 -443.8245 1097.4528 -2.4727

This table gives you the required asymmetry and uncertainties for respective physics generators. In the output you also get 23 columns of output and that you need to paste in this excel sheet ([sheet](https://docs.google.com/spreadsheets/d/19fOYKwEMWfIdIej8VpnlWuLScMMthF9TKQfzSBPDLU4/edit#gid=0)) to get the total rate at the respective sectors of the main detector. For details of the deconvolution analysis, you can take help from this writeup: [**A introduction to the contamination background study for Moller, Yuxiang Zhao**](http://chrome-extension://oemmndcbldboiebfnladdacbdfmadadm/https://moller.jlab.org/DocDB/0006/000691/001/Contamination_study_note.pdf)

If you need to study the background effects due to some material implementation, the sample code can be found in [here.](https://github.com/JeffersonLab/remoll-job-submission/blob/beluga/analysis/rate_beam_vert_spike/analyse.C) This script basically plots the Z-vertex distribution of different particles and the radial distributions for different sectors at the main detector plain. You need to use the simulated files as input to this script. Keep in mind while plotting the histograms should be weighted by the number of files you are using to generate the respective plots.

Once you are done with all the results, present it in one of the simulation meetings and try to explain whatever you are seeing and why!!!

----------------------------------------------------------------THE END-----------------------------------------------------------