README file for the data release of "Search for heavy neutrinos with the T2K near detector ND280"

November 18, 2018

1 General

This data release corresponds to the article "Search for heavy neutrinos with the T2K near detector ND280" published by the T2K collaboration in Physical Review D.

Flux and efficiencies files contain several vector and matrices, one for each masses from 140 to 490 MeV/c^2 by step of 10 MeV/c^2

2 Flux

- `flux.root` contains, for each mass m, a TVectorD object called *vector_flux_m*, each row corresponds to a different heavy neutrino production and decay mode:

BEAM in NEUTRINO MODE		BEAM in ANTINEUTRINO MODE	
row 0	$K \to \mu N \to \mu(\mu^-\pi^+)$	row 24	$K \to \mu N \to \mu(\mu^-\pi^+)$
1	$K \to eN \to e(\mu^-\pi^+)$	25	$K \to eN \to e(\mu^-\pi^+)$
2	$K \to \mu N \to \mu(e^-\pi^+)$	26	$K \to \mu N \to \mu(e^-\pi^+)$
3	$K \to eN \to e(e^-\pi^+)$	27	$K \to eN \to e(e^-\pi^+)$
4	$K \to \mu N \to \mu(\mu^+\pi^-)$	28	$K \to \mu N \to \mu(\mu^+\pi^-)$
5	$K \to eN \to e(\mu^+\pi^-)$	29	$K \to eN \to e(\mu^+\pi^-)$
6	$K \to \mu N \to \mu(e^+\pi^-)$	30	$K \to \mu N \to \mu(e^+\pi^-)$
7	$K \to eN \to e(e^+\pi^-)$	31	$K \to eN \to e(e^+\pi^-)$
8	$K \to \mu N \to \mu(\mu^+\mu^-\nu(\bar{\nu})_{e,\tau})$	32	$K \to \mu N \to \mu(\mu^+\mu^-\nu(\bar{\nu})_{e,\tau})$
9	$K \to eN \to e(\mu^+\mu^-\nu(\bar{\nu})_{e,\tau})$	33	$K \to eN \to e(\mu^+\mu^-\nu(\bar{\nu})_{e,\tau})$
10	$K \to \mu N \to \mu(e^+e^-\nu(\bar{\nu})_{\mu,\tau})$	34	$K \to \mu N \to \mu(e^+e^-\nu(\bar{\nu})_{\mu,\tau})$
11	$K \to eN \to e(e^+e^-\nu(\bar{\nu})_{\mu,\tau})$	35	$K \to eN \to e(e^+e^-\nu(\bar{\nu})_{\mu,\tau})$
12	$K \to \mu N \to \mu(\mu^+\mu^-\nu(\bar{\nu})_{\mu})$	36	$K \to \mu N \to \mu(\mu^+\mu^-\nu(\bar{\nu})_{\mu})$
13	$K \to eN \to e(\mu^+\mu^-\nu(\bar{\nu})_{\mu})$	37	$K \to eN \to e(\mu^+\mu^-\nu(\bar{\nu})_{\mu})$
14	$K \to \mu N \to \mu(e^+e^-\nu(\bar{\nu})_{e})$	38	$K \to \mu N \to \mu(e^+e^-\nu(\bar{\nu})_{e})$
15	$K \to eN \to e(e^+e^-\nu(\bar{\nu})_{e})$	39	$K \to eN \to e(e^+e^-\nu(\bar{\nu})_{e})$
16	$K \to \mu N \to \mu(\mu^-e^+\nu_e)$	40	$K \to \mu N \to \mu(\mu^-e^+\nu_e)$
17	$K \to eN \to e(\mu^-e^+\nu_e)$	41	$K \to eN \to e(\mu^-e^+\nu_e)$
18	$K \to \mu N \to \mu(\mu^-e^+\bar{\nu}_\mu)$	42	$K \to \mu N \to \mu(\mu^-e^+\bar{\nu}_\mu)$
19	$K \to eN \to e(\mu^-e^+\bar{\nu}_\mu)$	43	$K \to eN \to e(\mu^-e^+\bar{\nu}_\mu)$
20	$K \to \mu N \to \mu(\mu^+e^-\bar{\nu}_e)$	44	$K \to \mu N \to \mu(\mu^+e^-\bar{\nu}_e)$
21	$K \to eN \to e(\mu^+e^-\bar{\nu}_e)$	45	$K \to eN \to e(\mu^+e^-\bar{\nu}_e)$
22	$K \to \mu N \to \mu(\mu^+e^-\nu_\mu)$	46	$K \to \mu N \to \mu(\mu^+e^-\nu_\mu)$
23	$K \to eN \to e(\mu^+e^-\nu_\mu)$	47	$K \to eN \to e(\mu^+e^-\nu_\mu)$

The first part indicates the production mode of the heavy neutrino (in this analysis, we consider only heavy neutrinos producted in a kaon decay, jointly either with an electron or a muon). Even numbers corresponds to $K \to \mu N$ and odd numbers to $K \to eN$. The second part indicates the decay mode, where $\bar{\nu}$ is the antineutrino and $\nu(\bar{\nu})$ signifies that the contribution of neutrino and antineutrino are summed over.

The value of the vector for a given row gives the expected number of decays in ND280 TPCs in the corresponding production/decay mode combination with run 2-8 T2K statistics and assuming $U_e^2 = U_\mu^2 = U_\tau^2 = 1$, before applying any detection efficiency.

- `flux.root` also contains *covmat_flux_m*, a TMatrixDSym object containing the relative covariance matrix (rcov) that is used to cover flux uncertainties. As explained in the paper, it was assumed to be 15% and fully correlated. The statistical uncertainties related to the size of the simulated sample are added on the diagonal.

  The absolute covariance matrix is obtained by simply multiplying by the absolute flux: $cov(i,j) = rcov(i,j) \times flux(i) \times flux(j)$.

3 Efficiencies

- `efficiency.root` contains, for each mass m, a TMatrixD object called *effmat_eff_m*, its dimensions are 10(rows)*24(columns), each row corresponds to a different analysis channel, each column to a different production/decay mode:

BEAM in NEUTRINO MODE		BEAM in ANTINEUTRINO MODE	
row 0	$\mu\pi$	row 5	$\mu\pi$
row 1	$e^-\pi^+$	row 6	$e^-\pi^+$
row 2	$e^+\pi^-$	row 7	$e^+\pi^-$
row 3	$\mu+\mu-$	row 8	$\mu+\mu-$
row 4	e^+e^-	row 9	e^+e^-

  Columns (0 to 23) are the same as for the `flux.root` files

- `cov_efficiency.root` contains, for each mass m, a TMatrixD object called *covmat_eff_m* which is the relative covariance matrix of efficiencies, its dimensions are 240×240, each row/column corresponds to a different analysis channel + production/decay mode combination.

  The numbering is: row/col$k = 10 \times i + j$
  i is the production/decay mode from 0 to 23
  j is the analysis channel from 0 to 9

  The absolute covariance matrix is obtained by simply multiplying by the absolute efficiency: $cov(a,b) = rcov(a,b) \times eff(a) \times eff(b)$.

4 Number of expected signal

This can be easily obtained by combining the information from flux files and efficiencies files. For instance, the number of expected signal events after selection in channel A ($A = 0...4$) in neutrino mode for $U_e^2 = U_\mu^2 = U_\tau^2 = 1$ and for the considered T2K statistic is simply:

$$\sum_{i=0}^{23} \Phi_i \epsilon_{A,i} \tag{1}$$

5 Number of expected background and data

- `background.root` contains a TVectorD *vector_bkg* where each row is the nominal expected background from NEUT 5.3.2 with run 2-8 T2K statistics for a given analysis channel, with the same numbering as the rows in `efficiency.root`. It also contains a TVectorD *error_bkg* containing the total uncertainty on this background, for each analysis channel and as discuss in the article.

- `data.root` contains a TVectorD *vector_data* where each row is the observed number of events in the signal region from NEUT 5.3.2 in runs 2 to 8 for a given analysis channel, with the same numbering as the rows in efficiency.root.

6 Final limits

- The subdirectory `limits_single` contains .dat files corresponding to the upper limits obtained with the signle channel approach and presented in Fig.5 of the paper. Each file contains two columns, the first one being the scanned mass (in MeV/c^2) and the second the limits on the mixing element:
  - Uee => limit on U_e^2 using the mode $K \to eN, N \to e\pi$

- Uemu => limit on $U_e U_\mu$ using the mode $K \to eN, N \to \mu\pi$
- Umue => limit on $U_e U_\mu$ using the mode $K \to \mu N, N \to e\pi$
- Uee => limit on $U_e\sqrt{(U_e^2 + U_\tau^2)}$ using the mode $K \to eN, N \to \mu^+\mu^-\nu_{e,\tau}$
- methods A,B,C are numbered as described in detail in the paper

- The subdirectory `limits_combined` contains `limits_marginalisation.dat` file with 4 columns, the first one being the scanned mass (in MeV/c^2) and the second to fourth are respectively the limits on the mixing elements U_e^2, U_μ^2 and U_τ^2. They are obtained after marginalising over the two other mixing elements. Similarly, `limits_profiling.dat` contains the limits obtained after profiling the two other mixing elements (setting them to 0); it is not possible to follow this procedure for the limit on U_τ^2 for all masses and for the limit on U_μ^2 for masses above 388 MeV/c^2 as there are no production/decay mode combination directly sensitive to them in these ranges.

- The subdirectory `limits_combined/2D` contains the 2D limits at 90% as presented in Figure 7, after profiling over U_τ^2. First column is the value of U_e^2, second column is the value of U_μ^2. The list of points in a given file forms a contour in the $U_e^2 - U_\mu^2$ plane.

- The subdirectory `histograms_combined` contains a list of root files (one for each considered mass). Each root file contains simply one 3D histogram filled with the MCMC steps in the 3D space (U_e^2, U_μ^2, U_τ^2). The chosen binning is regular in log, from 10^{-15} to 1

  Recommendations:

  - You can easily obtain 2D histograms by using Project3D (corresponds to marginalisation over the last one). For instance, `h->Project3D("yx")` will create a 2D histogram called "h_yx" of y versus x (integrating over z).
  - You can then use the draw option "colz" to check the distribution: `h_yx->Draw("colz")`
  - It is best to visualize the plot in log-scale: `canvas->SetLogx()` , `canvas->SetLogy()`
  - You can similarly project in 1D with `h->Project3D("y")` to obtain the distribution as a function of U_μ^2, in this case.

  WARNING

  This is not a posterior probability function as each bin contains the total number of steps of the MCMC falling in this bin. It is however very easy to obtain the posterior by simply dividing the content of each bin by the size of the bin (1D, 2D or 3D) and then simply normalize the obtained histogram to have integral=1.

  Other tips:

  - You can change the prior on the mixing elements U_α^2 ($\alpha = e, \mu, \tau$), by simply multiplying the pdf by a factor (new prior/old prior). The limits can then be recomputed easily

7 Scripts

They are provided in the folder `scripts`.

7.1 Re-run the Markov Chain Monte Carlo

The python script `heavyNu_combined_MCMC.py` can be used to run the Markov Chain Monte Carlo using the flux/effiencies/background/data inputs (to be put in a subfolder `input`.

The model, as presented in the paper, is fully implemented in `heavyNu_combined_model.py`.

It used PyMC package, to be installed by the user for instance with pip. The output is an HDF5 file with 3 columns, one for each U_α^2 ($\alpha = e, \mu, \tau$), there are stored in a subdirectory `db`.

7.2 Compute the limits

The output of the MCMC can further be used to extract limits using the script `read_results.py`. This example script computes the limits on each U_α^2 ($\alpha = e, \mu, \tau$) after marginalisation over the others. It also writes 3D histograms as the ones presented in section 6.