DUNE Sensitivity Analysis with GLoBES

Paolo Minhas

GLoBES

- A software package used to simulate long baseline neutrino oscillation experiments
- Used to simulate 3 neutrino and new physics models in this project
- Provides a C library which is used to simulate the DUNE experiment

GLoBES

Definitions

```
/* Systematics Definitions */
include "definitions.inc"
include "syst_list.inc"


/* Baseline */
$profiletype = 3
$densitytab = {2.848}
$lengthtab = {1284.9}
```

Channels

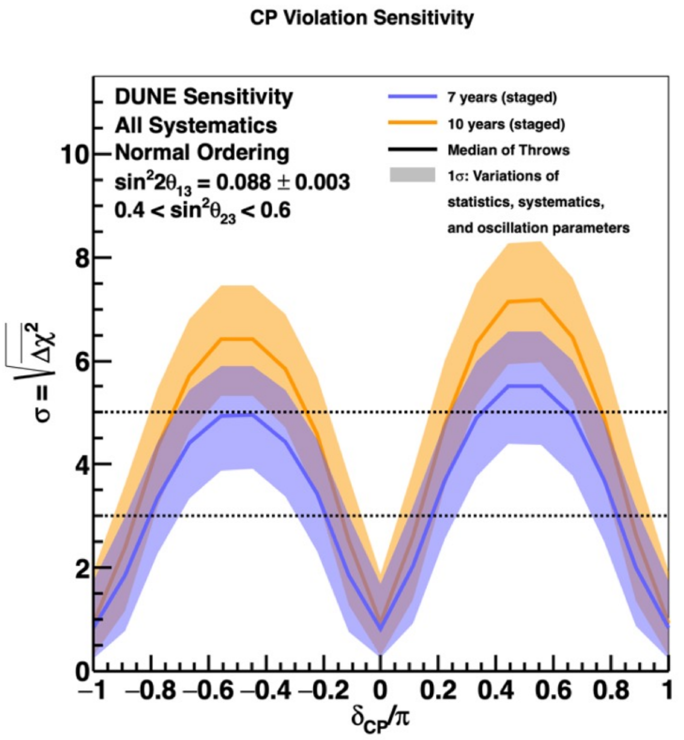
```
/* NUE APP */
channel(#FHC_app_osc_nue)<
        @channel =      #flux_FHC:     +:      m:      e:      #CC:    #app_nue_sig
        @post_smearing_efficiencies = copy(%post_app_FHC_nue_sig)
>
```

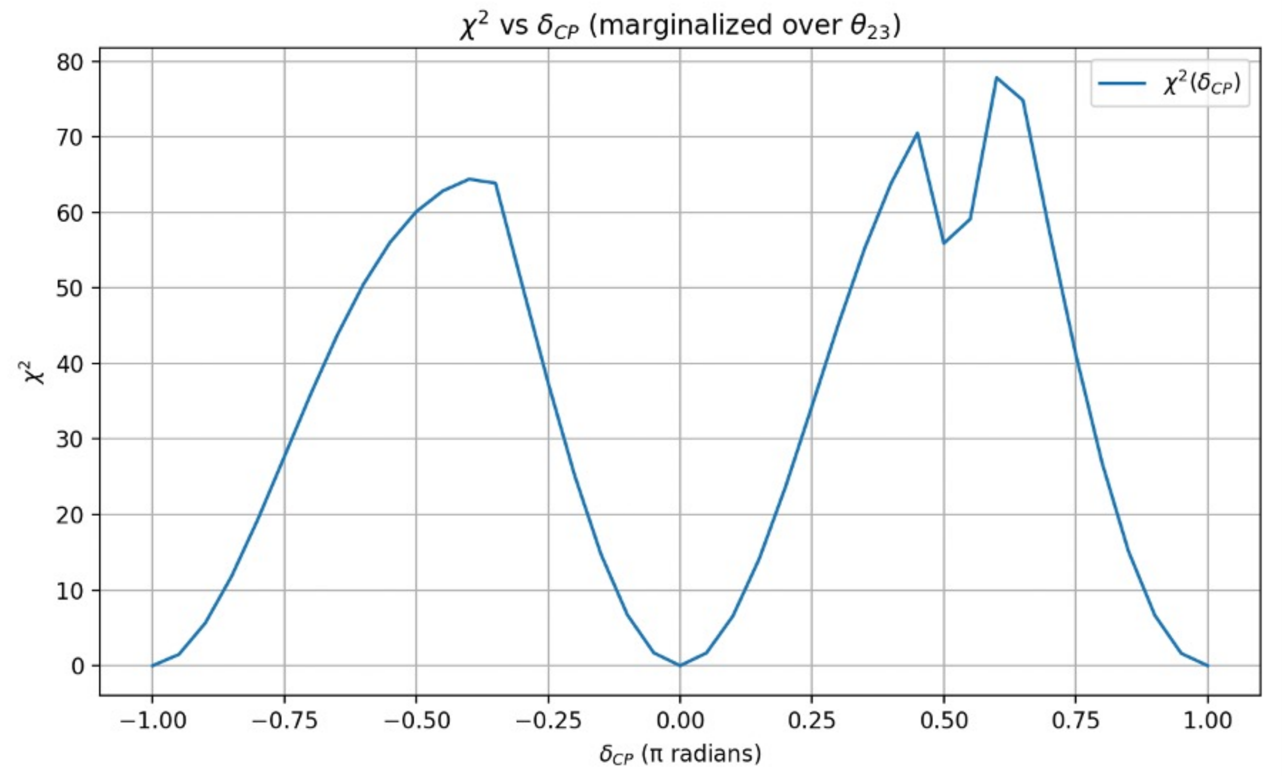
Rules

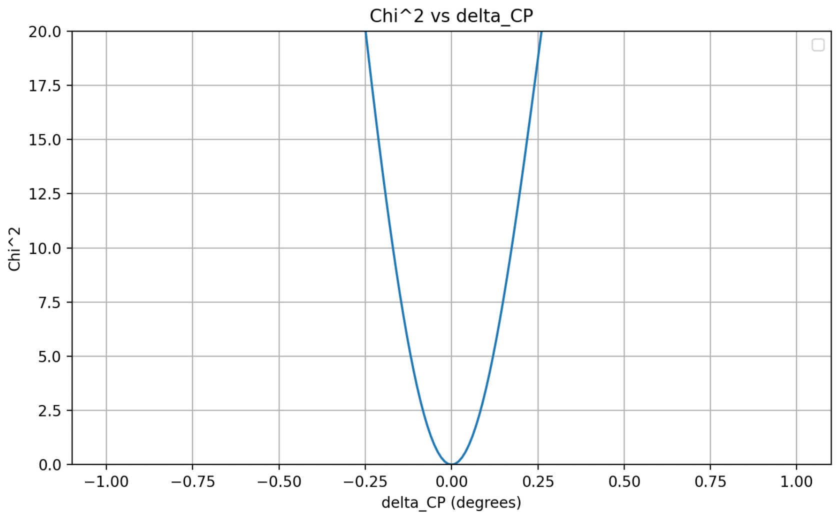
```
rule(#nue_app)<
        @signal = 1.0@#FHC_app_osc_nue : 1.0@#FHC_app_osc_nuebar
        @sys_on_multiex_errors_sig = { #err_nue_sig } : {#err_nue_sig}
        @background =  1.0@#FHC_app_bkg_nue : 1.0@#FHC_app_bkg_nuebar : 1.0@#FHC_app_bkg_numu : 1.0@#
        @sys_on_multiex_errors_bg = {#err_nue_bg} : {#err_nue_bg} : {#err_numu_bg} : {#err_numu_bg} :
        @errordim_sys_on = 0
        @errordim_sys_off = 2
        @sys_on_function = "chiMultiExp"
        @sys_off_function = "chiNoSysSpectrum"
        @energy_window = 0.5 : 18.0
>
```

χ^2 Tests

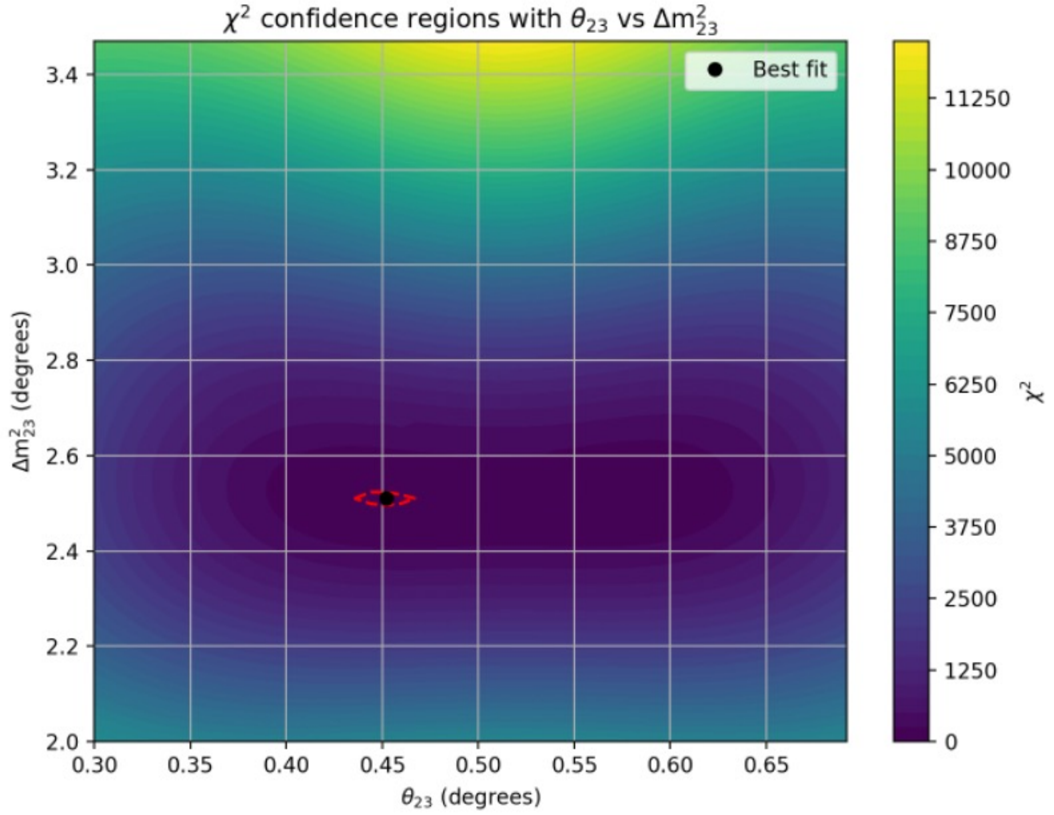
- Compares observed data to expected data
- How well does the observed data fits the expected distribution


CP Violation Sensitivity
DUNE Sensitivity
All Systematics
Normal Ordering
$\sin^2 2\theta_{13} = 0.088 \pm 0.003$
$0.4 < \sin^2\theta_{23} < 0.6$
7 years (staged)
10 years (staged)
Median of Throws
1σ: Variations of
statistics, systematics,
and oscillation parameters
$\sigma = \sqrt{\Delta\chi^2}$
δ_{CP}/π

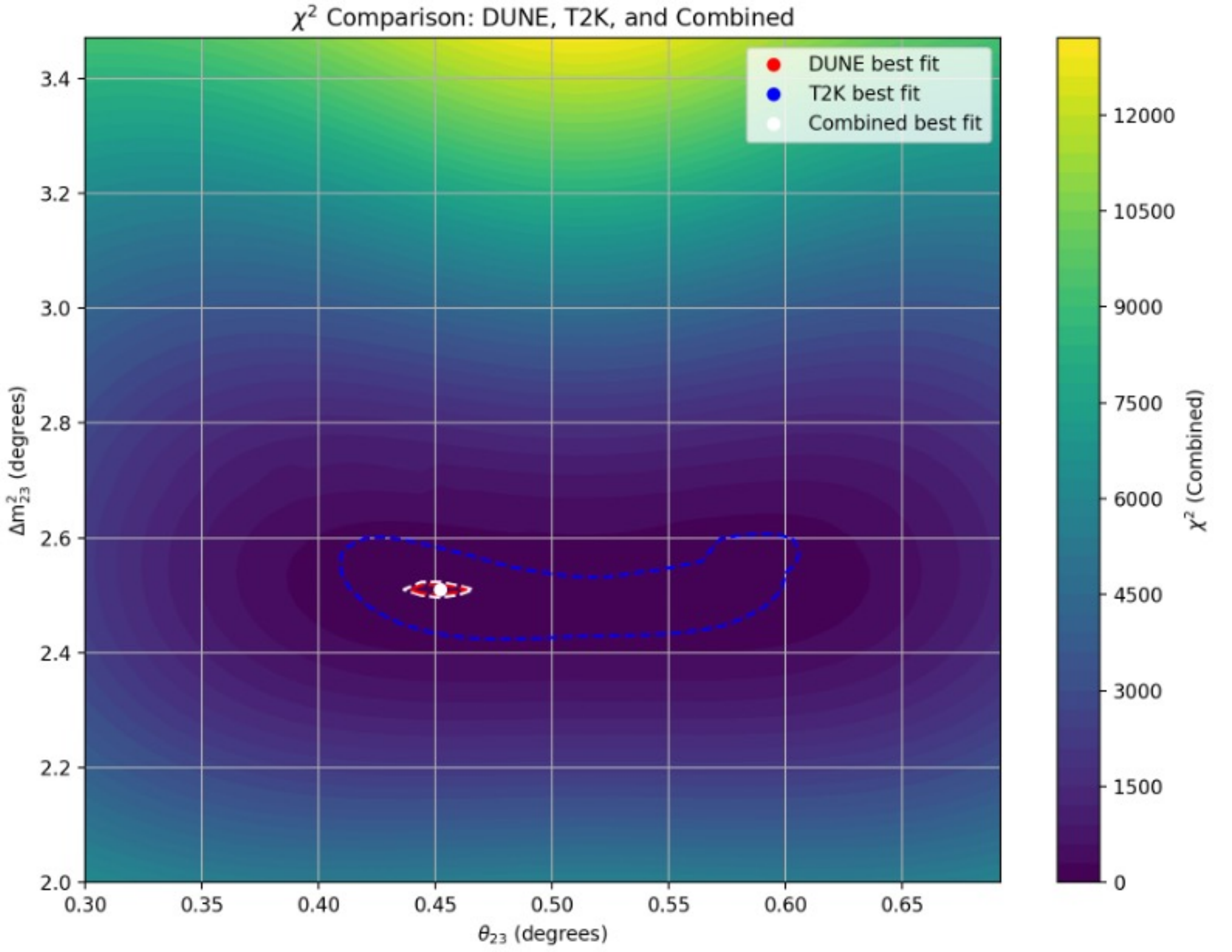

χ^2 vs δ_{CP} (marginalized over θ_{23})
$\chi^2(\delta_{CP})$
χ^2
δ_{CP} (π radians)


Chi^2 vs delta_CP
Chi^2
delta_CP (degrees)

χ^2 Tests

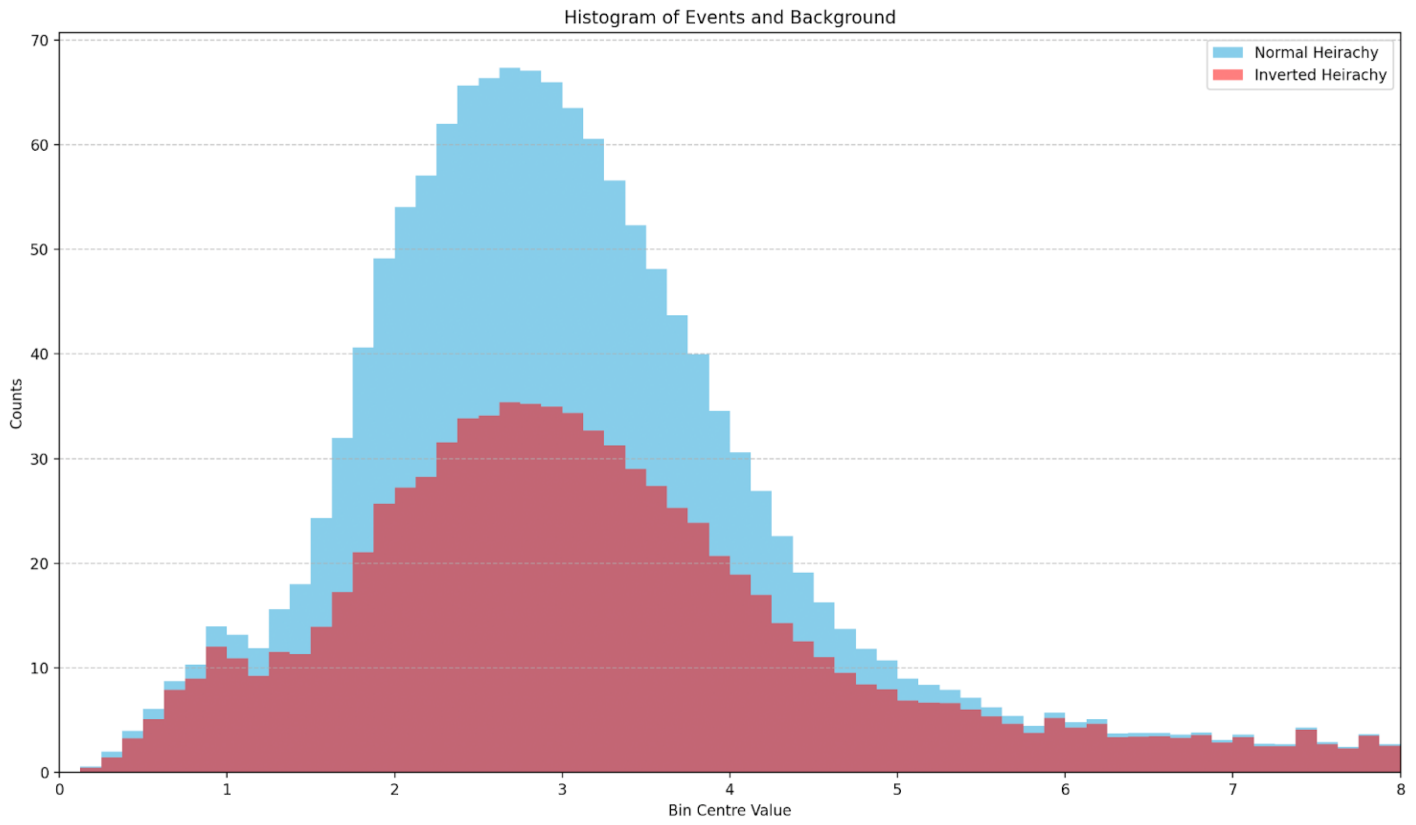


χ² confidence regions with θ23 vs Δm²23
Best fit
Δm²23 (degrees)
θ23 (degrees)
χ²
3.4
3.2
3.0
2.8
2.6
2.4
2.2
2.0
0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65
11250
10000
8750
7500
6250
5000
3750
2500
1250
0



χ² Comparison: DUNE, T2K, and Combined
DUNE best fit
T2K best fit
Combined best fit
Δm²23 (degrees)
θ23 (degrees)
χ² (Combined)
3.4
3.2
3.0
2.8
2.6
2.4
2.2
2.0
0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65
12000
10500
9000
7500
6000
4500
3000
1500
0

Events Spectra (Electron Neutrino Appearance or Rule 1)



Sterile Neutrino

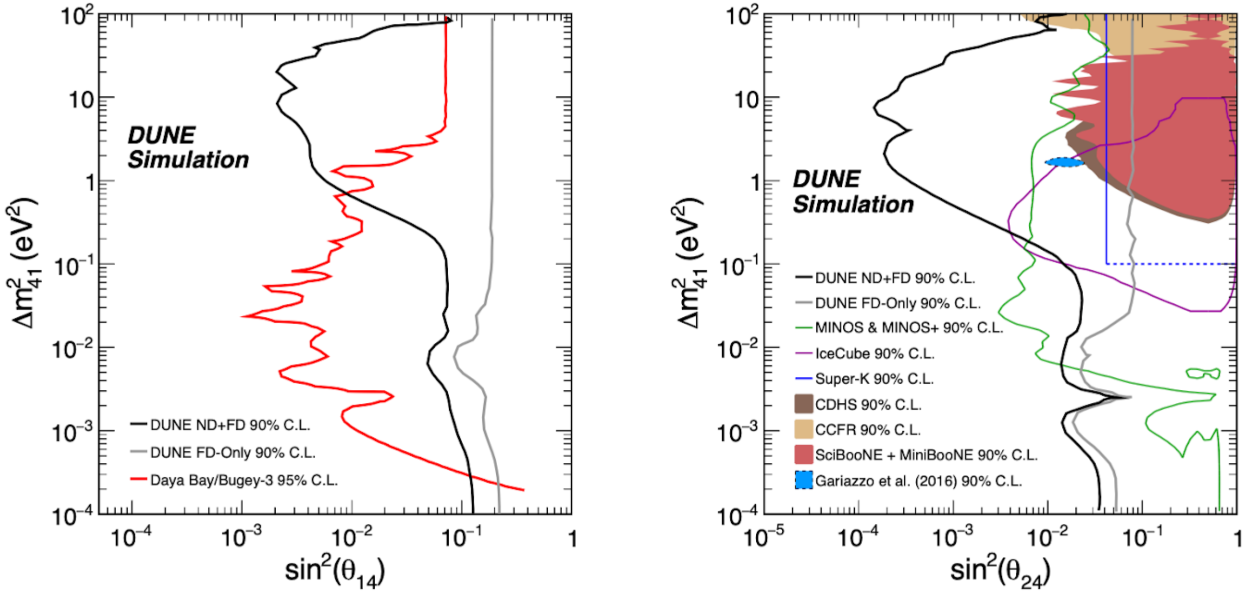


Figure 8.2: The left-hand plot shows the DUNE sensitivities to θ_{14} from the ν_e CC samples at the ND and FD, along with a comparison with the combined reactor result from Daya Bay and Bugey-3. The right-hand plot displays sensitivities to θ_{24} using the ν_μ CC and NC samples at both detectors, along with a comparison with previous and existing experiments. In both cases, regions to the right of the contours are excluded.

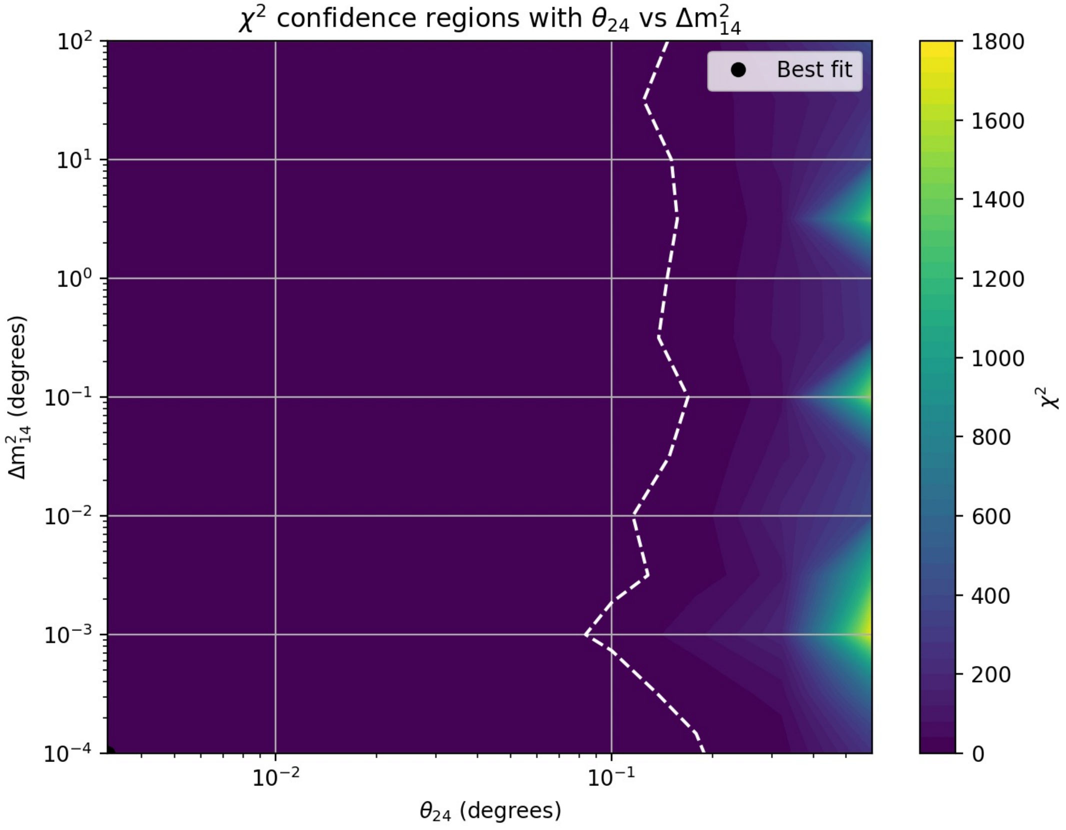
Extra parameters

```
/* Compute P_ee */
s12 = sin(th12);
c12 = cos(th12);
s13 = sin(th13);
c13 = cos(th13);
t = L / (4.0 * E);
Delta21 = sdm * t;
Delta31 = ldm * t;
Delta32 = Delta31 - Delta21;
t = M_SQRT2 * sigma_E / E;
D21 = exp(-square( Delta21 * t ));
D31 = exp(-square( Delta31 * t ));
D32 = exp(-square( Delta32 * t ));
P[0][0] = square(square(c13)) * ( 1 - 2.0*square(s12*c12)*(1 - D21*cos(2.0*Delta21)) )
            + 2.0*square(s13*c13) * ( D31*square(c12)*cos(2.0*Delta31)
                                     + D32*square(s12)*cos(2.0*Delta32) )
           + square(square(s13));

return 0;
```

$$P^{\mathrm{SB}}_{\overset{(-)}{\nu}_\mu \to \overset{(-)}{\nu}_e}(E, L) = \underbrace{\boxed{4|U'_{\mu 4}|^2|U'_{e4}|^2}}_{\sin^2 2\theta_{\mu e}} \sin^2\left(1.27\Delta m^2_{41} L/E\right)$$

Current work




```
// Sterile χ² Output //
  double this_theta14, this_theta24, this_stdm, en14, en24, enstdm;
  double chi_now, min_chi = 1e30;

  //glbSetOscParams(test_values, this_theta14, 7); // Set θ14 SUP
  for(en24 = -5; en24 <= -0.0001; en24 += 0.05) {
    this_theta24 = asin(sqrt(pow(10, en24)));
    //glbSetOscParams(test_values, this_theta24, 8); // Set θ24

    for(enstdm = -4; enstdm <= 2; enstdm += 0.05) {
        this_stdm = pow(10, enstdm);
        //glbSetOscParams(test_values, this_stdm, 6); // Set Δm²41

        for(en14= -4; en14 <= -0.0001; en14 += 0.05) {
          min_chi = 1e30; // Reset min_chi for each en14
          this_theta14 = asin(sqrt(pow(10, en14)));
          glbSetOscParams(test_values, this_theta24, 8); // Set θ24
          glbSetOscParams(test_values, this_theta14, 7); // Set θ14
          glbSetOscParams(test_values, this_stdm, 6); // Set Δm²41
          chi_now = glbChiNP(test_values, NULL, 0); // Compute chi^2 with the test values
          if (chi_now < min_chi) {min_chi = chi_now;}

        }
        fprintf(outfile3, "%g %g %g \n",  this_theta24, this_stdm, chi_now);
  }
  }
  fclose(outfile3);
```

```c
double param_values[12] = { // Normal ordering values
  asin(sqrt(0.307)),     // asin(sqrt(0.307)),     // θ₁₂ test
  asin(sqrt(0.02195)),   // asin(sqrt(0.02195)),   // θ₁₂ test
  asin(sqrt(0.561)),     // asin(sqrt(0.561)),     // θ₂₃ test
  177/180*M_PI,    // asin(sqrt(0.561)),     // δCP test
  7.49e-5,               // 7.49e-5,               // Δm²₂₁ test
  2.534e-3,              // 2.534e-3,              // Δm²₃₁ test
  0.0 ,                   // 0.0,                   // Δm²₄₁ test
  0.0,                   // 0.0,                   // θ₁₄ test
  0.0,                   // 0.0,                   // θ₂₄ test
  0.0,                   // 0.0,                   // θ₃₄ test
  0.0,                   // 0.0,                   // δCP1 test
  0.0                    // 1.0                    // δCP2 test
};
```

```c
double this_theta14, this_theta24, this_stdm, en14, en24, enstdm;
double chi_now, min_chi = 1e30;

/ glbSetOscParams(test_values, this_theta14, 7); // Set θ₁₄ SUP
for(en24 = -5; en24 <= -0.0001; en24 += 0.1) {
  this_theta24 = asin(sqrt(pow(10, en24)));
  //glbSetOscParams(test_values, this_theta24, 8); // Set θ₂₄

  for(enstdm = -4; enstdm <= 2; enstdm += 0.1) {
    this_stdm = pow(10, enstdm);
    //glbSetOscParams(test_values, this_stdm, 6); // Set Δm²₄₁


    //for(en14= -4; en14 <= -0.0001; en14 += 0.05) {
      min_chi = 1e30; // Reset min_chi for each en14
      //this_theta14 = asin(sqrt(pow(10, en14)));
      glbSetOscParams(test_values, this_theta24, 8); // Set θ₂₄
      //glbSetOscParams(test_values, this_theta14, 7); // Set θ₁₄
      glbSetOscParams(test_values, this_stdm, 6); // Set Δm²₄₁
      chi_now = glbChiNP(test_values, NULL, 0); // Compute chi^2 with the
      //if (chi_now < min_chi) {min_chi = chi_now;}

    // }
    fprintf(outfile3, "%g %g %g \n",  pow(10, en24), this_stdm, chi_now);
}
}
```

```c
for (int i = 0; i < 12; ++i) {
  glbSetOscParams(true_values, param_values[i], i);
  printf("%g \n", glbGetOscParams(true_values, i));
}
for (int i = 12; i < 92; ++i) {
  glbSetOscParams(true_values, 0.0, i);
}
glbSetDensityParams(true_values,1.0,GLB_ALL);
```

```c
snu_init_probability_engine(n_flavors, rotation_order, phase_order);
glbRegisterProbabilityEngine(92, //6*sqrt(n_flavors)
&snu_probability_matrix,
&snu_set_oscillation_parameters,
&snu_get_oscillation_parameters,
NULL);
```

Current Work