pMSSM McMC code

Malte Mrowietz

11.11.2020

Overview

- https://github.com/mmrowietz/pMSSM_McMC
- Completely python2 based (if print statements are rewritten, it should work in python3)
- Output stored in ROOT trees
- (very) simplistic interfaces to external tools *SPheno, superiso, FeynHiggs*

How to run

- Get the code and numpy
- Install *SPheno, superiso, FeynHiggs* and compile the executables
- in mcmc.py, set homedir to code location

```
25  homedir = "/nfs/dust/cms/user/mrowietm/python_scan/pMSSM_McMC/"
26  packagedir = homedir+"packages/"
27  spnexe = packagedir+"SPheno-4.0.4/bin/SPheno"
28  fhexe = packagedir+"FeynHiggs-2.16.1/bin/FeynHiggs"
29  sisoexe = packagedir+"superiso_v4.0/slha.x"
30  #sisochi2exe = packagedir+"superiso_v4.0/slha_chi2.x" #use all of the non-controversial low-energy resul
31  sisochi2exe = packagedir+"superiso_v4.0/slha_chi2_reduced.x"#use only branching ratios in superiso chi2.
```

- execute mcmc.py with following required inputs:
  - "-m <runmode>": choices=["new","resume"]
  - “-o <output directory>”
  - “-i <input root file>” if run mode is resume, give path to input root file with chain to resume
- optional inputs:
  - “-n <number of points to run>” (default =1000): Number of points that the McMC runs for.
  - “-c <chain index>” (default=1): Index given to chain to uniquely identify it.
  - “-s <save interval>” (default=300): Save progress and move it to output directory after every save interval worth of points. Prevent complete loss if job crashes

Scan ranges

- Scan ranges are set at the top of mcmc.py

```
12  #set up the parameter ranges
13  parameter_ranges ={}
14  for parameter in ["mu","M1","M2"]:
15      parameter_ranges[parameter] = (-4000,4000)
16  for parameter in ["M3","Mq1","Mq3","Mu1","Mu3","Md1","Md3"]:
17      parameter_ranges[parameter] = (0,10000)
18  for parameter in ["Ml1","Mr1","Ml3","Mr3","Mh3",]:
19      parameter_ranges[parameter] = (0,4000)
20  for parameter in ["At","Ab","Al"]:
21      parameter_ranges[parameter] = (-7000,7000)
22  parameter_ranges["tb"] = (2,60)
23
24  width_coefficient = 0.1 #the width coefficient of the gaussian for the mcmc step.
```

- Do our tools work up to high scales needed for 100 TeV collider?

Output branches

- Tree branches are defined in tree_branches dictionary
- numpy is used to interface python types to ROOT types
- numpy container value needs to be set in McMC loop

```
37  #containers for the tree branches. Numpy arrays are used as an interface between python types and the root branches
38  tree_branches = {}
39  tree_branches["slha_file"]={"container":TString(),"dtype":"TString"}
40  tree_branches["likelihood"]= {"container":np.zeros(1,dtype = float),"dtype":"D"}
41  tree_branches["iteration_index"] = {"container":np.zeros(1,dtype = int),"dtype":"I"}
42  tree_branches["accepted_index"] = {"container":np.zeros(1,dtype = int),"dtype":"I"}
43  tree_branches["chain_index"] = {"container":np.zeros(1,dtype = int),"dtype":"I"}
44  tree_branches["mtop"] = {"container":np.zeros(1,dtype = float),"dtype":"D"}
45  tree_branches["mbottom"] = {"container":np.zeros(1,dtype = float),"dtype":"D"}
46  tree_branches["alpha_s"] = {"container":np.zeros(1,dtype = float),"dtype":"D"}
47  tree_branches["mhiggs"] = {"container":np.zeros(1,dtype = float),"dtype":"D"}
48  for param in parameter_ranges.keys():
49      tree_branches[param] = {"container":np.zeros(1,dtype=float),"dtype":"D"}
50  tree_branches["superiso_chi2_stdout"]={"container":TString(),"dtype":"TString"}
51  tree_branches["superiso_stdout"]={"container":TString(),"dtype":"TString"}
52  tree_branches["chi2"]={"container":np.zeros(1,dtype=float),"dtype":"D"}
53  tree_branches["chi2_ndf"]={"container":np.zeros(1,dtype=int),"dtype":"I"}
```

Likelihood calculation: standard cases

- Experimental value and uncertainties are inserted into dictionary in likelihood.py

```
likelihood_contributions ={}
likelihood_contributions["mtop"] = {"value":173.1,"uncertainty":0.9}
likelihood_contributions["mbottom"] = {"value":4.18,"uncertainty":[0.03,0.04]}
likelihood_contributions["alpha_s"] = {"value":0.1181,"uncertainty":0.0011}
likelihood_contributions["mhiggs"] = {"value":125.26}#good practise would be to cite from where
#superiso gives only the SUSY contribution to amu, thus it must be compared against delta amu
likelihood_contributions["a_muon"]={"value":26.8E-10,"uncertainty":4.3E-10}#http://pdg.lbl.gov/
likelihood_contributions["BR_B_to_tau_nu"]={"value":1.44E-04,"uncertainty":0.31E-04}#http://www
likelihood_contributions["BR_Ds_to_tau_nu"]={"value":5.48E-02,"uncertainty":0.23E-02}#http://pd
likelihood_contributions["BR_Ds_to_mu_nu"]={"value":5.5E-03,"uncertainty":0.23E-03}#http://pdg.
```

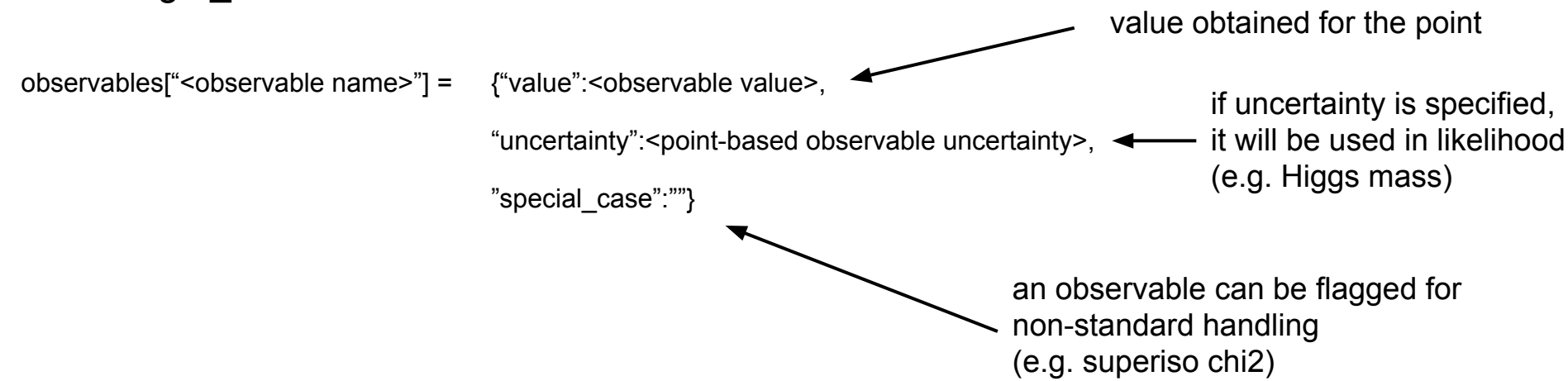
symmetric uncertainty

asymmetric uncertainty

- likelihood is calculated in `def get_likelihood(observables):`
- observables is dictionary containing observable values keyed by observable name (which must be the same as in likelihood_contributions dictionary)

Likelihood calculation: standard cases

- observables is dictionary containing observable values keyed by observable name (which must be the same as in likelihood_contributions dictionary)
- If observables entry contains key “uncertainty”, it will be used instead of uncertainty in likelihood_contributions (e.g. for Higgs mass)
- if observables entry contains key “special_case”, its handling has to be implemented in get_likelihood function



Likelihood calculation: how to add a new one

1. Write an interface to required tool
2. In McMC loop, extract the value for the point and add the key-value pair to observables dictionary before these lines:

```
397             _l = likelihood.get_likelihood(observables)#get likelihood
458             _l = likelihood.make_decision(observables,lastaccepted["likelihood"])#get likelihood
```

3. If necessary, implemented the contribution to the likelihood in likelihood.py
4. (Add the likelihood to the tree_branches dictionary)

McMC run loop

Slightly different code depending on runmode=”new” or “resume”

```
438         finite_lh = False
439         while not finite_lh:
440             utils.clean()
441             spnerr = False
442             while not spnerr:#find a viable point
443                 utils.clean()
444                 candidate = generate_point(lastaccepted)#generate a point from the last point
445                 spnin = utils.write_spheno_input(candidate)#write the input for spheno
446                 spnerr = run_spheno(spnin,devnull) #run spheno, check if viable point
447             if not run_feynhiggs('>& /dev/null'):#run feynhiggs, replace higgs sector
448                 continue
449             observables = get_observables(slhapath = "SPheno.spc") #get observables for the likelihood
450             siso_obs = run_superiso("SPheno.spc")
451             if siso_obs == -1:
452                 continue
453             for obs in siso_obs:
454                 observables[obs] = siso_obs[obs]
455             siso_chi2_obs = run_superiso_chi2("SPheno.spc")
456             for obs in siso_chi2_obs:
457                 observables[obs] = siso_chi2_obs[obs]
458             _l = likelihood.make_decision(observables,lastaccepted["likelihood"])#get likelihood
459             finite_lh = _l != 0
```

Require a non-zero likelihood

generate candidate points until SPheno does not complain

Replace Higgs sector, try new candidate if FeynHiggs complains

Try new candidate if superiso complains

Collect information nessesary to likelihood calculation, add entries to observables dictionary

Find a valid pMSSM point and make McMC decision

McMC run loop

```
458             _l = likelihood.make_decision(observables,lastaccepted["likelihood"])#get likelihood
459             finite_lh = _l != 0
460         if _l<0:
461             move +=1
462             if iter_ix == start+tend:
463                 print "Made all "+str(tend)+" iterations, moving "+outname+" to storage"
464                 outtree.BuildIndex("chain_index","iteration_index")
465                 outtree.Write()
466                 outroot.Close()
467                 os.system(" ".join(["mv",outname,outdir]))
468             continue #point was not accepted
469         lastaccepted["likelihood"]=_l
470         lastaccepted["iteration_index"] = iter_ix
471         lastaccepted["accepted_index"] =lastaccepted["accepted_index"]+1
472         lastaccepted["chain_index"] = chainix
473         lastaccepted["superiso_chi2_stdout"] = observables["superiso_chi2_stdout"]["value"]
474         lastaccepted["superiso_stdout"] =observables["superiso_stdout"]["value"]
475         lastaccepted["chi2"] =observables["chi2"]["value"]
476         lastaccepted["chi2_ndf"]=observables["chi2_ndf"]["value"]
477
```

Not yet included and areas of improvement

- Run mode to start from non-random point
- Dark matter related constraints (costly in terms of computation time)
- Sampling from McMC (oversampling, undersampling)
- Code overhaul for efficiency
- Better interface to external tools? (in-memory I/O?)