```
Last login: Wed Feb  8 08:37:53 on ttys001
(base) oissan@cu-genvpn-comp-10 ~ % conda activate hux-rom
(hux-rom) oissan@cu-genvpn-comp-10 ~ % pip install line_profiler
Collecting line_profiler
  Downloading line_profiler-4.0.2-cp39-cp39-macosx_10_9_x86_64.whl (96 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 96.4/96.4 kB 851.7 kB/s eta 0:00:00
Installing collected packages: line_profiler
Successfully installed line_profiler-4.0.2
(hux-rom) oissan@cu-genvpn-comp-10 ~ % cd pycharmprojects
(hux-rom) oissan@cu-genvpn-comp-10 pycharmprojects % ls
3D_HD_SW_Enrico                 Space-Weather-ROM-Revised
Exasim                          UQ_notebook
Macintosh HD                    data
Parameter_Estimation_Solar_Wind mcmc_with_prior_samples
Probabilistic_ML                mfgsa
ROM--Maxwell                    nsf
ROM--Maxwell2                   opalissan
SVD                             shift-lift-learn
(hux-rom) oissan@cu-genvpn-comp-10 pycharmprojects % cd Parameter_Estimation_Solar_Wind
(hux-rom) oissan@cu-genvpn-comp-10 Parameter_Estimation_Solar_Wind % ls
ACE_measurement_analysis_CR2048_to_CR2058.ipynb
GONG
HUX
MCMC_results
MCMC_seven_params.ipynb
MCMC_seven_params.py
README.md
SA_CR2058_multi_A.py
SA_CR2058_multi_B.py
SA_CR2058_multi_C.py
SA_analysis
SA_evaluate_samples
SA_results
SA_tools
__pycache__
figs
lprof0
model_chain.py
model_chain_results
multi_processing
nohup.out
profiling_mcmc.ipynb
rip_samples.ipynb
rip_samples_time_dependent_QoIs.ipynb
tools
(hux-rom) oissan@cu-genvpn-comp-10 Parameter_Estimation_Solar_Wind %  kernprof -l model_chain
Could not find script model_chain
(hux-rom) oissan@cu-genvpn-comp-10 Parameter_Estimation_Solar_Wind %  kernprof -l model_chain.py
Wrote profile results to model_chain.py.lprof
(hux-rom) oissan@cu-genvpn-comp-10 Parameter_Estimation_Solar_Wind % python -m line_profiler model_chain.py.lprof
Timer unit: 1e-06 s

Total time: 5e-06 s
File: model_chain.py
Function: convert_vector_to_dict at line 23

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    23                                           @profile
    24                                           def convert_vector_to_dict(samples):
    25                                               """convert an array/list of coefficients to dictionary
    26
    27                                               :param samples: 1d array or list of size 11 (uncertain input parameters). Note: order matters...
    28                                               The order: {r_ss, v0, v1, alpha, beta, w, gamma, delta, psi, alpha_acc, rh}
    29                                               :return: dictionary of size 11 (uncertain input parameters).
    30                                               """
    31         1          2.0      2.0     40.0      return {"r_ss": samples[0],
    32         1          0.0      0.0      0.0              "v0": samples[1],
    33         1          0.0      0.0      0.0              "v1": samples[2],
    34         1          0.0      0.0      0.0              "alpha": samples[3],
    35         1          0.0      0.0      0.0              "beta": samples[4],
    36         1          1.0      1.0     20.0              "w": samples[5],
    37         1          1.0      1.0     20.0              "gamma": samples[6],
    38         1          1.0      1.0     20.0              "delta": samples[7],
    39         1          0.0      0.0      0.0              "psi": samples[8],
    40         1          0.0      0.0      0.0              "alpha_acc": samples[9],
    41         1          0.0      0.0      0.0              "rh": samples[10]}

Total time: 31.4764 s
File: model_chain.py
Function: run_chain_of_models_mcmc at line 269

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
   269                                           @profile
   270                                           def run_chain_of_models_mcmc(ACE_longitude,
   271                                                                        ACE_latitude,
   272                                                                        ACE_r,
   273                                                                        gong_map,
   274                                                                        coefficients_vec,
   275                                                                        n_r_pfss=100,
   276                                                                        n_r_hux=300,
   277                                                                        n_theta_ch=180,
   278                                                                        n_phi_ch=360):
   279                                               """functionality to run a chain of empirical and reduced-physics models:
   280
   281                                                                           [PFSS] -----> [WSA] -----> [HUX]
   282
   283                                               :param ACE_r: ACE simulation_output distance from the Sun.
   284                                               :param ACE_latitude: ACE simulation_output latitude [-90, 90] in degrees.
   285                                               :param ACE_longitude: ACE simulation_output longitude [0, 360] in degrees.
   286                                               :param n_phi_ch: number of phi mesh grid points in tracing coronal hole maps.
   287                                               :param n_theta_ch: number of theta mesh grid points in tracing coronal hole maps.
   288                                               :param n_r_hux: number of radial mesh grid points in the HUX finite difference uniform mesh.
   289                                               :param coefficients_vec: 11 parameters of PSS, WSA, and HUX.
   290                                               :param n_r_pfss: number of radial cells in finite differencing in PFSS.
   291                                               :param gong_map: SunPy Map object.
   292                                               :return: QoI evaluated for a specific CR. (float)
   293                                               """
   294                                               # convert coefficients to dictionary for readability.
   295         1         29.0     29.0      0.0      coeff = convert_vector_to_dict(samples=coefficients_vec)
   296
   297                                               # PFSS parameters + simulate.
   298         1       5113.0   5113.0      0.0      pfss_in = pfsspy.Input(br=gong_map, nr=n_r_pfss, rss=coeff["r_ss"])
   299         1    6755966.0 6755966.0     21.5       pfss_out = pfsspy.pfss(input=pfss_in)
   300
   301                                               # trace the magnetic field lines for the ACE projection to obtain the magnetic expansion factor.
   302         1      10439.0  10439.0      0.0      tracer = tracing.FortranTracer()
   303         1       3244.0   3244.0      0.0      seeds = SkyCoord(ACE_longitude.to(u.rad),
   304         1         59.0     59.0      0.0                       ACE_latitude.to(u.rad),
   305         1        124.0    124.0      0.0                       coeff["r_ss"] * const.R_sun,
   306         1      17879.0  17879.0      0.1                       frame=pfss_out.coordinate_frame)
   307         1    1643920.0 1643920.0      5.2       field_lines_fp = tracer.trace(seeds=seeds, output=pfss_out)
   308         1     648784.0 648784.0      2.1      fp_ace_traj = field_lines_fp.expansion_factors
   309
```

```
   310                                          # coronal hole mapping
   311         1   14117087.0 14117087.0     44.8      topologies = pfss2flines(pfsspy_out=pfss_out, nth=n_theta_ch, nph=n_phi_ch)
   312         1    7683391.0 7683391.0     24.4      d_ace_traj = distance_to_coronal_hole_boundary(topologies=topologies, field_lines_fp=field_lines_fp)
   313
   314                                          # WSA empirical model.
   315         1         99.0     99.0      0.0      v_wsa = wsa(fp=fp_ace_traj, d=d_ace_traj, coeff=coeff)
   316
   317                                          # define HUX grid.
   318         1        334.0    334.0      0.0      r_hux = (np.linspace(coeff["r_ss"], np.max(ACE_r.to(u.solRad)).value, n_r_hux) * u.solRad).to(u.km).value
   319         1         45.0     45.0      0.0      p_hux = np.linspace(0, 2 * np.pi, len(ACE_longitude))
   320
   321                                          # interpolate WSA velocity results on HUX grid.
   322         1        202.0    202.0      0.0      v_wsa_interp = interpolate_ace_data(x=p_hux, xp=ACE_longitude.to(u.rad).value, fp=v_wsa, period=2 * np.pi)
   323
   324                                          # simulate HUX for the entire grid [phi, r].
   325         1     503834.0 503834.0      1.6      vr_hux_wsa = apply_hux_f_model(initial_condition=v_wsa_interp,
   326         1          3.0      3.0      0.0                                     dr_vec=r_hux[1:] - r_hux[:-1],
   327         1          2.0      2.0      0.0                                     dp_vec=p_hux[1:] - p_hux[:-1],
   328         1          1.0      1.0      0.0                                     alpha=coeff["alpha_acc"],
   329         1          0.0      0.0      0.0                                     rh=coeff["rh"],
   330         1          1.0      1.0      0.0                                     r0=coeff["r_ss"],
   331         1        103.0    103.0      0.0                                     theta=np.mean(np.pi / 2 - ACE_latitude.to(u.rad).value))
   332
   333                                          # interpolate back to ACE longitude and radial trajectory..
   334         1      85705.0  85705.0      0.3      vr_hux_wsa_interp = interp_2d_ace_hux(p_hux=p_hux,
   335         1          0.0      0.0      0.0                                            r_hux=r_hux,
   336         1          0.0      0.0      0.0                                            vr_hux=vr_hux_wsa,
   337         1          0.0      0.0      0.0                                            ACE_r=ACE_r,
   338         1          1.0      1.0      0.0                                            ACE_longitude=ACE_longitude)
   339         1          0.0      0.0      0.0      return vr_hux_wsa_interp
```

```
(hux-rom) oissan@cu-genvpn-comp-10 Parameter_Estimation_Solar_Wind %
```