Statistical Plots

Created by Alanah Cardenas-O'Toole
Summer 2025
Latest update: 08/07/2025
Email alanahco@umich.edu

Walkthrough of how to use functions from Swarm_Stats.py

In [1]:

```
import numpy as np
import pandas as pd
from datetime import datetime,timedelta


# Statistical codes
from Swarm_Stats import states_report_swarm, LSS_plot_Swarm, map_hist_
from Swarm_Stats import plot_hist_quad_maps, Liemohn_Skill_Scores
from Swarm_Stats import decision_table_sat, style_df_table, HMFC_perce
from Swarm_Stats import style_LSS_table, LSS_table_sat, one_model_LSS_
```

Getting dataframes that include the basic state and H, M, F, C

Swarm_stats.states_report_swarm This code requires that both NIMO and PyIRI daily files have been created
and returns 3 dataframes
that will be used for future statistics
Note: if you just want H, M, F ,C for one model,
state_check(obs_type, mod_type, state='eia') is useful

Requred Parameters

> date_range : pandas daterange
>
> > Date range of desired states files
>
> daily_dir : str
>
> > directory of daily files

Key Word Arguments

> typ: str
>
> > desired type to check against

> > for state orientations
> >
> > 'eia'(default), 'peak', 'flat', 'trough'
> >
> > for direction orientations
> >
> > 'north', 'south', 'neither'
>
> NIMO_alt: str
>
> > specifies which altitude to use
> >
> > 'swarm'(default),'hmf2','100'

Returns

> NiSw : DataFrame
>
> > NIMO states, directions, and types (original full name)
> >
> > also includes longitude, local times, and sat list
>
> Sw : DataFrame
>
> > Swarm States, direction, and types
> >
> > also includes longitude, local times, and sat list
>
> Py : DataFrame
>
> > PyIRI states, directions, and types
> >
> > also includes longitude, local times, and sat list

In [2]:

```
date_range = pd.date_range(start='2020-04-01',  end='2020-04-30')
daily_files = '~/Type_Files/Daily'
NiSw, Sw, PyI = states_report_swarm(date_range, daily_files, typ='eia'
print(NiSw) # Nimo Swarm comparison
```

```
     state direction                 type   GLon         LT Sat skill
0     peak     north           peak_north  -40.0  21.950833   A     M
1     peak     south           peak_south  128.0   9.901974   A     C
2     peak     north           peak_north  -64.0  21.903889   A     C
3     peak   neither                 peak  104.0   9.897939   A     M
4      eia     south  eia_saddle_peak_south  -88.0  21.890833   A     H
...    ...       ...                  ...    ...        ...  ..   ...
2735   eia     north  eia_saddle_peak_north  160.0   7.292343   C     F
2736  peak   neither                 peak  -32.0  19.345556   C     M
2737   eia     north  eia_saddle_peak_north  136.0   7.284213   C     F
2738  peak     north           peak_north  -56.0  19.299444   C     M
2739  peak   neither                 peak  112.0   7.281541   C     C

[2740 rows x 7 columns]
```

Creating Liemohn Skill Score plots

Swarm_Stats.LSS_plot_Swarm
Created using Liemohn Skill Scores 1-4 from
"Leaving Heidke behind: Defining an independent reference model
for event detection skill scores" Liemohn et al. (in preparation 2025)
This requires 2 models for compasion becuase LSS is valuable as a comparison tool.
If you only want 1, then use Swarm_Stats.one_model_LSS_plot_Swarm
NOTE: LSS can range outside of +/-1

Plot LSS vs CSI or PC 4 panels (one for each LSS) Required Parameters

> model1 : dataframe
>
> > first model dataframe built by states_report_swarm
>
> model2 : dataframe
>
> > second model dataframe built by states_report_swarm
>
> eia_type : str
>
> > desired eia type for fig title
>
> date_range : datetime range
>
> > For plotting title purposes

Key Word Arguments

> model1_name : str kwarg
>
> > first model name for labelling purposes
>
> model2_name : str kwarg
>
> > second model name for labelling purposes
>
> PorC : str kwarg
>
> > Percent correct or Critical success index for x axes
>
> DayNight : bool kwarg

> > True (default) if panels should have separate markers
> > for day and night
>
> otherwise (false) all are plotted together

LT_range : list kwarg

> Range of day night local time, Default is 7 LT to 19 LT for day and
> 19 LT to 7 LT for Night

coin : bool kwarg

> If True, coin LSS will be plotted for comparison (default)
> if false, coin LSS will not be plotted

Returns

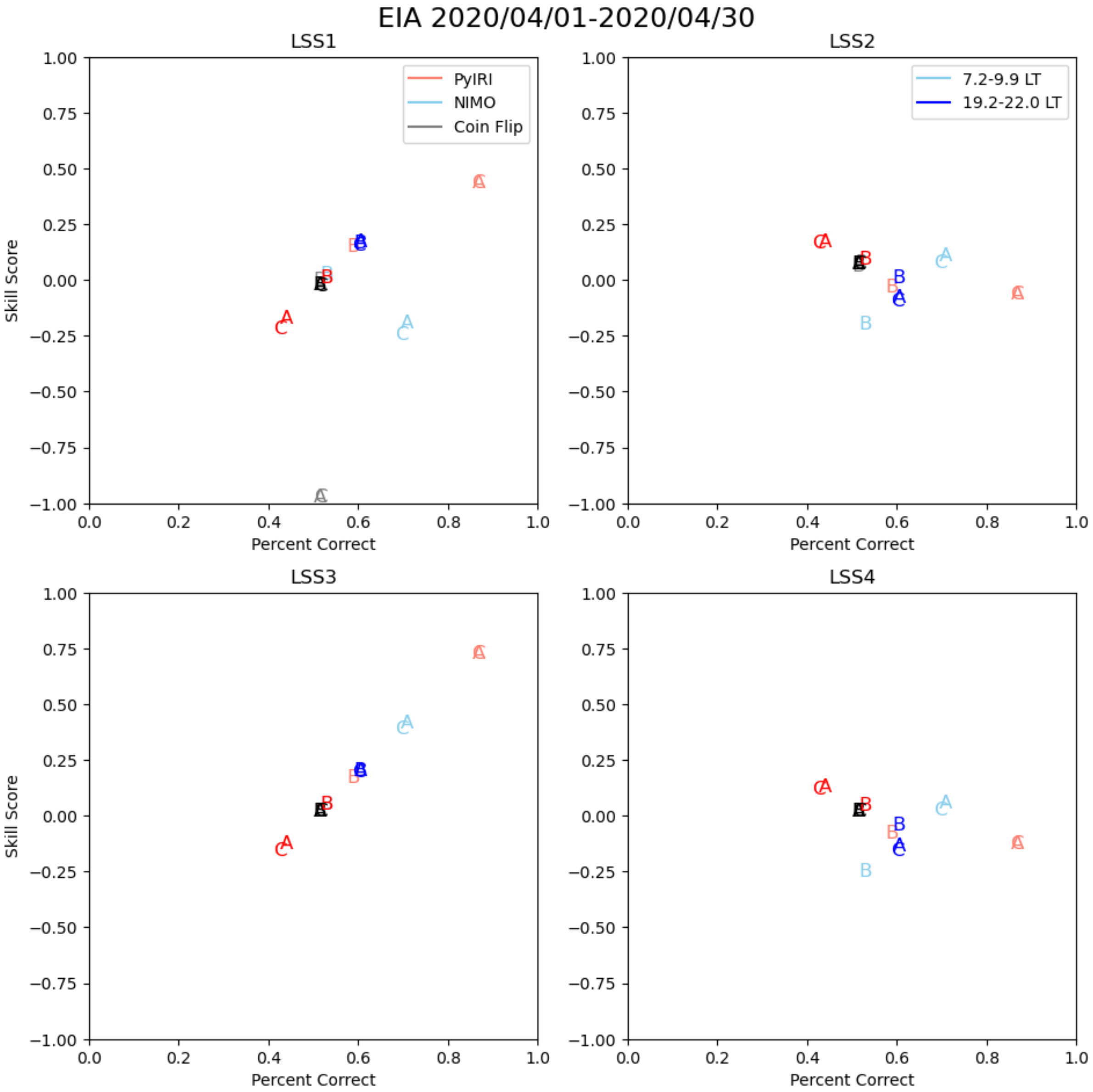
> fig : figure handle
>
> > 4 panel figure that includes LSS for the 2 models
> > and a coin toss if coin

Note: Since we care about Correct Negatives,
Percent Correct is more useful than Critical Success Index

> PC = (H + C)/T
> CSI = H/(H + M + F)
>
> > according to Liemohn et al. pg 8

In [3]:

```
fig = LSS_plot_Swarm(NiSw, PyI, 'EIA', date_range, model1_name='NIMO',
```



Creating Liemohn Skill Score plots continued

Swarm_Stats.one_model_LSS_plot_Swarm
Created using Liemohn Skill Scores 1-4 from
"Leaving Heidke behind: Defining an independent reference model
for event detection skill scores" Liemohn et al. (in preparation 2025)
If you want to compare 2 models, then use Swarm_Stats.LSS_plot_Swarm

Plot LSS vs CSI or PC 4 panels (one for each LSS) Required Parameters

> model1 : dataframe
>
> > first model dataframe built by states_report_swarm
>
> eia_type : str

> > desired eia type for fig title
>
> date_range : datetime range
>
> > For plotting title purposes

Key Word Arguments

> model_name : str kwarg
>
> > first model name for labelling purposes
>
> PorC : str kwarg
>
> > Percent correct or Critical success index for x axes
>
> DayNight : bool kwarg
>
> > True (default) if panels should have separate markers for day and night
>
> otherwise (false) all are plotted together

LT_range : list kwarg

> Range of day night local time, Default is 7 LT to 19 LT for day and 19 LT to 7 LT for Night

coin : bool kwarg

> If True, coin LSS will be plotted for comparison (default) if false, coin LSS will not be plotted

Returns

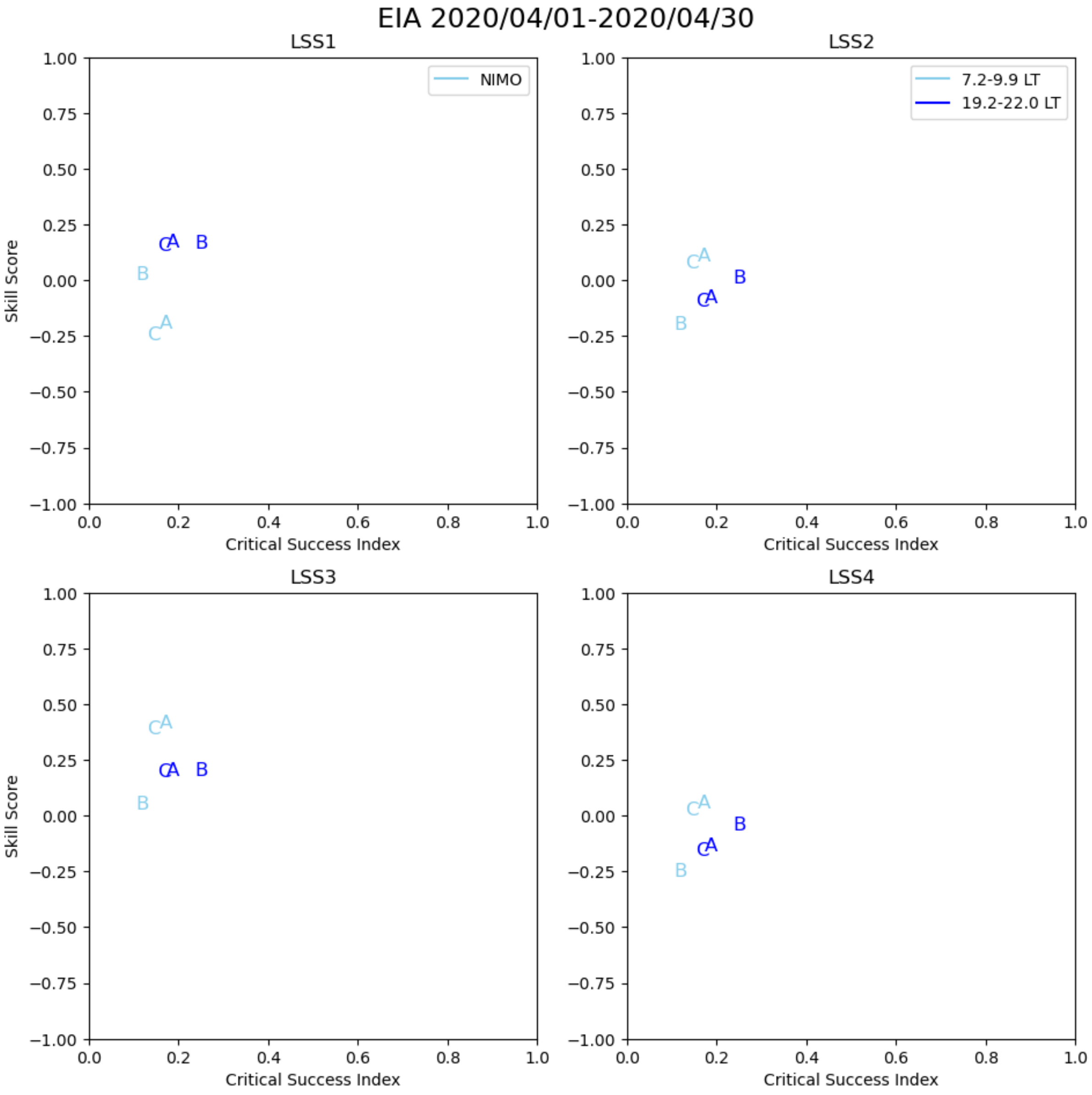
> fig : figure handle
>
> > 4 panel figure that includes LSS for the 2 models and a coin toss if coin

Note: Warning will be printed if coin is specified as False

In [4]:

```
fig = one_model_LSS_plot_Swarm(NiSw, 'EIA', date_range, model_name='NI
                               LT_range=[7, 19], coin=False)
```

```
/Users/aotoole/Documents/Python_Code/EIA_Update/Swarm_Stats.py:575: Use
rWarning: Warning: Coin is False! LSS is a comparison tool!
  warnings.warn("Warning: Coin is False! LSS is a comparison tool!")
```



Plotting Histogram Maps

Function Swarm_Stats.plot_hist_quad_maps
plot histogram maps on a 4 panel figure for each score: Hit, Miss, False positive, and Correct Negative
This function calls
Swarm_Stats.map_hist_panel(ax, model, bin_lons=37, DayNight=True, LT_range=[7, 19])
Which will make just 1 panel

Required Parameters

> model_states : dataframe
>
> > dataframe of model data including skill and local times
> > built by states_report_swarm
>
> sat : str
>
> > swarm satellite 'A', 'B', or 'C'
>
> eia_type : str
>
> > eia state e.g. EIA, Peak, etc.
> > depending on what is considered a hit
>
> date_range : pandas daterange
>
> > range of dates for title purposes

Key Word Arguments

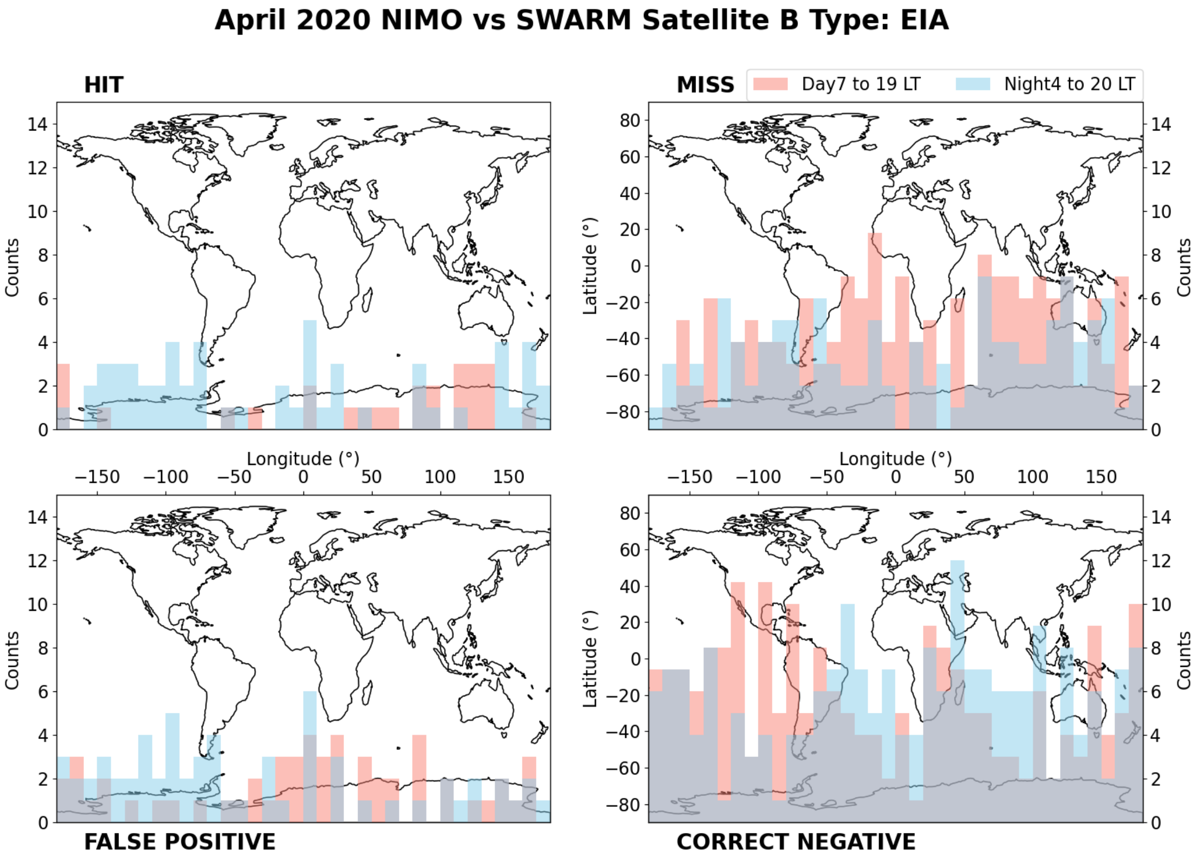
> bin_lons : int kwarg
>
> > number of bins between -180 and 180 deg geo lon
> > np.linspace(-180, 180, bin_lons)
> > default 37
>
> model_name : str kwarg
>
> > name of model for title purposes
> > default 'Model'
>
> fosi : int kwarg
>
> > font size for plot
> > default 16
>
> hist_ylim : list kwarg
>
> > y range (counts) for hist plot
> > default [0,15]
>
> LT_range : list kwarg
>
> > Range of day night local time
> > Default is 7 LT to 19 LT for day and
> > 19 LT to 7 LT for Night

Returns

> fig : figure handle
>
> > fig with 4 panels of hist maps

Note: A side thought is to have only 2 panels:
one with HIT and total in state (H + M)
and another with Correct Negatives and total out of state(C + F)

```
In [5]: fig = plot_hist_quad_maps(NiSw, 'B', 'eia', date_range, bin_lons=37, m
```



Making Decision Tables

Swarm_Stats.decision_table_sat

Takes in dataframe created by Swarm_Stats.states_report_swarm
Neat decision table summing up the hits, misses,
correct negatives, and false positives per satellite

Required Parameters

> states: dataframe

> > dataframe of model data including skill and local
> > times
> > built by states_report_swarm
>
> eia_type : str
>
> > eia state e.g. EIA, Peak, etc. depending on what is
> > considered a hit

Key Word Arguments

> sats : list of strings kwarg
>
> > swarm satellites 'A', 'B', and 'C' as default
> > can specify just 1 or 2
>
> model_name : str kwarg
>
> > Model name for decision table label
> > default 'Model'

Returns

> df : dataframe
>
> > dataframe in table format separated by satellite
> > and event state (state, non-state)
> > index using
> > df.loc[(f'Swarm {satellite}', eia_type), (model_name,
> > eia_type)]

Swarm_Stats.style_df_table

This function styles the table created by Swarm_Stats.decision_table_sat
This will only be for all satellites because I spent too much time
Trying to figure out how to make it more general.
The issue is from 941 where I specify the colors

Required Parameters:

> df_table : dataframe
>
> > dataframe created by decision_table_sat
>
> eia_type : str

> > string designating which eia type is being reported

Returns

> Styled dataframe with colors indicating successes and failures and table spearators by satelltie

In [6]:

```
df_table = decision_table_sat(NiSw)
df_table
```

Out[6]:

		Model	
		eia	Non-eia
Swarm A	eia	65.0	184.0
	Non-eia	142.0	522.0
Swarm B	eia	83.0	287.0
	Non-eia	118.0	416.0
Swarm C	eia	56.0	191.0
	Non-eia	146.0	530.0

In [7]:

```
styled_table = style_df_table(df_table, 'eia')
styled_table
```

Out[7]:

		Model	
		eia	Non-eia
Swarm A	eia	65	184
	Non-eia	142	522
Swarm B	eia	83	287
	Non-eia	118	416
Swarm C	eia	56	191
	Non-eia	146	530

Making Liemohn Skill Score Tables

Swarm_Stats.LSS_table_sat

Neat table including the Liemohn Skill Scores 1-4 separated by satellite

Required Parameters

model1: dataframe

> dataframe of 1st model data including skill and local times built by states_report_swarm

model2 : dataframe

> dataframe of 2nd model data including skill and local times built by states_report_swarm

Key Word Arugments

> model1_name : str kwarg
>
> > string of name of model1
>
> model2_name : str kwarg
>
> > string of name for model2
>
> sats : list of strings kwarg
>
> > swarm satellites 'A', 'B', and 'C' as default can specify just 1 or 2

Returns

> LSS_df : dataframe
>
> > dataframe in table format separated by satellite and Liemohn skill score

Swarm_Stats.style_LSS_table

This function styles LSS_df by adding lines in between each satellite
All satellites are not required for this one

Required Parameters

> LSS_df : dataframe
>
> > dataframe created by LSS_table_sat

Key word Arguments

> sat_list: list of strings kwarg
>
> > satellite list for LSS_df

Returns

> LSS table with dividers between satellites
> This can be further edited in pyhton and
> by copying and pasting it to a document

In [8]:

```
LSS_df = LSS_table_sat(NiSw, PyI, model1_name='NIMO', model2_name='PyI
LSS_df
```

Out[8]:

		NIMO	PyIRI
Swarm A	**LSS1**	0.099900	0.091616
	LSS2	0.009910	0.179228
	LSS3	0.285871	0.279299
	LSS4	-0.061149	0.120321
Swarm B	**LSS1**	0.073489	0.057475
	LSS2	-0.117436	0.014682
	LSS3	0.103982	0.088496
	LSS4	-0.169597	-0.031311
Swarm C	**LSS1**	0.068553	0.063026
	LSS2	-0.014166	0.173335
	LSS3	0.269772	0.265439
	LSS4	-0.086980	0.113983

In [9]:

```
styled_df = style_LSS_table(LSS_df)
styled_df
```

Out[9]:

		NIMO	PyIRI
Swarm A	**LSS1**	0.099900	0.091616
	LSS2	0.009910	0.179228
	LSS3	0.285871	0.279299
	LSS4	-0.061149	0.120321
Swarm B	**LSS1**	0.073489	0.057475
	LSS2	-0.117436	0.014682
	LSS3	0.103982	0.088496
	LSS4	-0.169597	-0.031311
Swarm C	**LSS1**	0.068553	0.063026
	LSS2	-0.014166	0.173335
	LSS3	0.269772	0.265439
	LSS4	-0.086980	0.113983

Plotting HM percents and FC percents

Plot full figure using HMFC_percent_panel
2 Models required e.g. Py IRI and NIMO

> This figure has a lot going on. When you look at it, think of each quadrant as a separate plot defined by Hit, Miss, Correct Negative, and False Positive as labelled. The percentages are the percent the model got correct or incorrect based on event states
> For example, for Hits, ther percentage is Hit/(Hit + Miss) where Hit+Miss
> is the total in the event states, the panel below that Miss/(Hit+Miss) is
> equivalent to 100% - Hit/(Hit + Miss), so those sectors are conjugate to
> each other
> For quick viewing, there are 4 shaded regions. These represent when a
> model is doing better than a coin toss. Ideally, False positives and Misses
> would have a low % and Hits and Correct Negatives have a higher

> percentage

Required Parameters

> model1 : dataframe
>
> > first model dataframe built by states_report_swarm
>
> model2 : dataframe
>
> > second model dataframe built by states_report_swarm
>
> eia_type : str
>
> > desired eia type for fig title

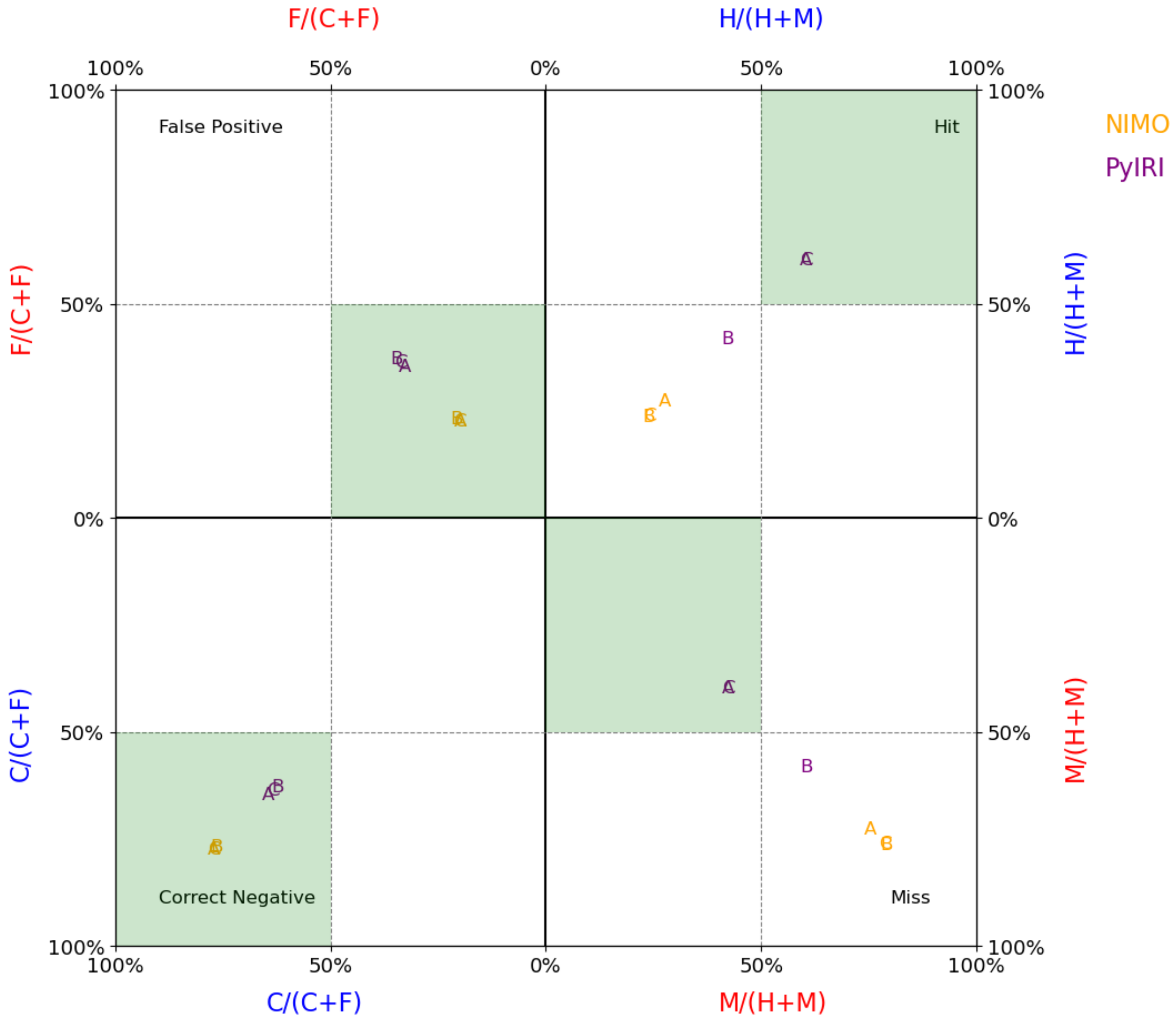
Key Word Arguments

> model1_name : str kwarg
>
> > first model name for labelling purposes
> > default Model1
>
> model2_name : str kwarg
>
> > second model name for labelling purposes
> > default Model2
>
> col1 : str
>
> > plotting color for Model1
> > defualt orange
>
> col2 : str
>
> > plotting color for Model 2
> > default purple
>
> fosi : int
>
> > font size for plot

Returns

> fig : figure handle as desribed above

```
In [10]: fig = HMFC_percent_figure(NiSw, PyI, 'eia', model1_name='NIMO', model2
```



```
In [ ]:
```