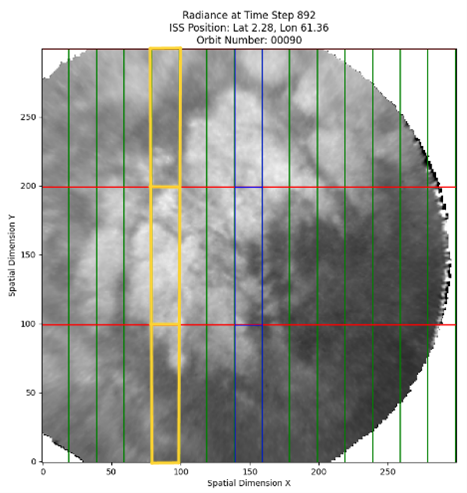
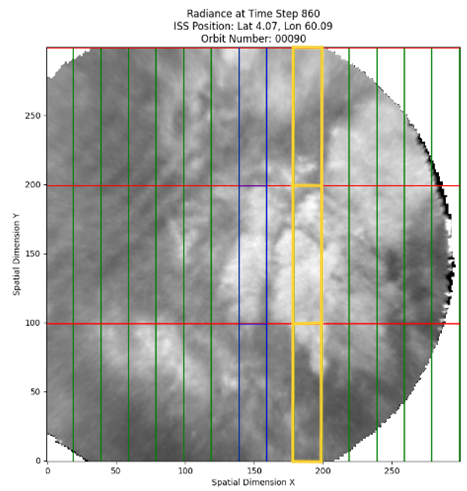
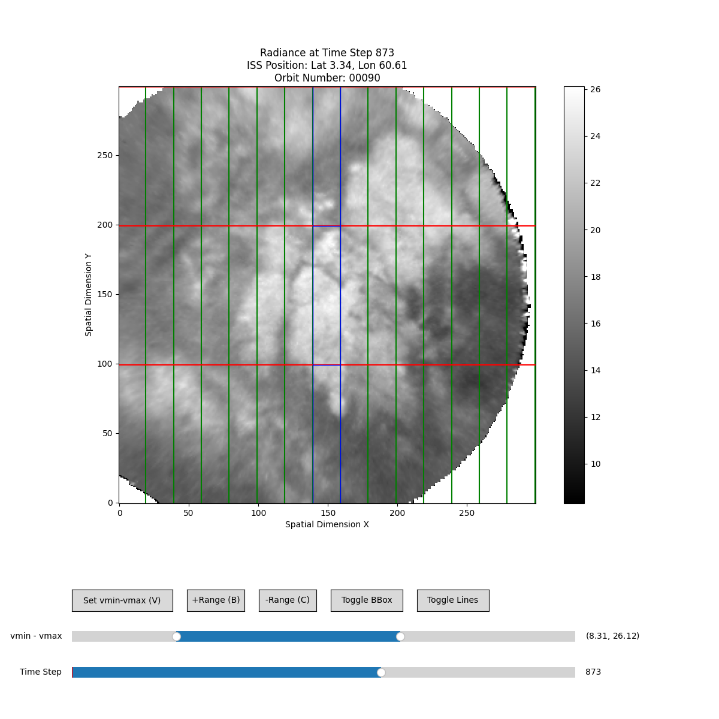
## Labelling:



I checked the intervals. Turns out the logic was wrong!

Fixed it.

Orbit 75, Box 0: Intervals = [(898, 935), (950, 972), (1091, 1124), (1147, 1147), (1186, 1204)]

Orbit 75, Box 1: Intervals = [(922, 952), (967, 992), (1054, 1059), (1110, 1123), (1135, 1201), (1210, 1221)]

Orbit 75, Box 2: Intervals = [(519, 529), (909, 991), (1039, 1060), (1090, 1176), (1187, 1206)]

Orbit 90, Box 0: Intervals = [(676, 812), (826, 941), (972, 996), (1248, 1256)]

Orbit 90, Box 1: Intervals = [(676, 813), (842, 925), (975, 986), (1000, 1010), (1248, 1276)]

Orbit 90, Box 2: Intervals = [(694, 812), (835, 954), (997, 997), (1263, 1305)]

Orbit 105, Box 0: Intervals = [(638, 667), (715, 715), (791, 862), (898, 965)]

Orbit 105, Box 1: Intervals = [(645, 686), (759, 760), (773, 860), (894, 939), (950, 955), (966, 1051)]

Orbit 105, Box 2: Intervals = [(679, 698), (750, 760), (791, 824), (844, 850), (922, 1014), (1039, 1058)]

Orbit 120, Box 0: Intervals = [(191, 193), (240, 289), (676, 681), (691, 778), (809, 810), (837, 849)]

Orbit 120, Box 1: Intervals = [(217, 295), (645, 652), (679, 695), (705, 762), (808, 808), (819, 862)]

Orbit 120, Box 2: Intervals = [(221, 298), (645, 652), (665, 670), (707, 710), (757, 779), (799, 803), (816, 871)]

Orbit 135, Box 0: Intervals = [(733, 772), (791, 800), (814, 817)]

Orbit 135, Box 1: Intervals = [(710, 730), (751, 838)]

Orbit 135, Box 2: Intervals = [(705, 705), (720, 724), (740, 809), (821, 857)]

Check the labelling: wrong box indices for the boxes on the left!

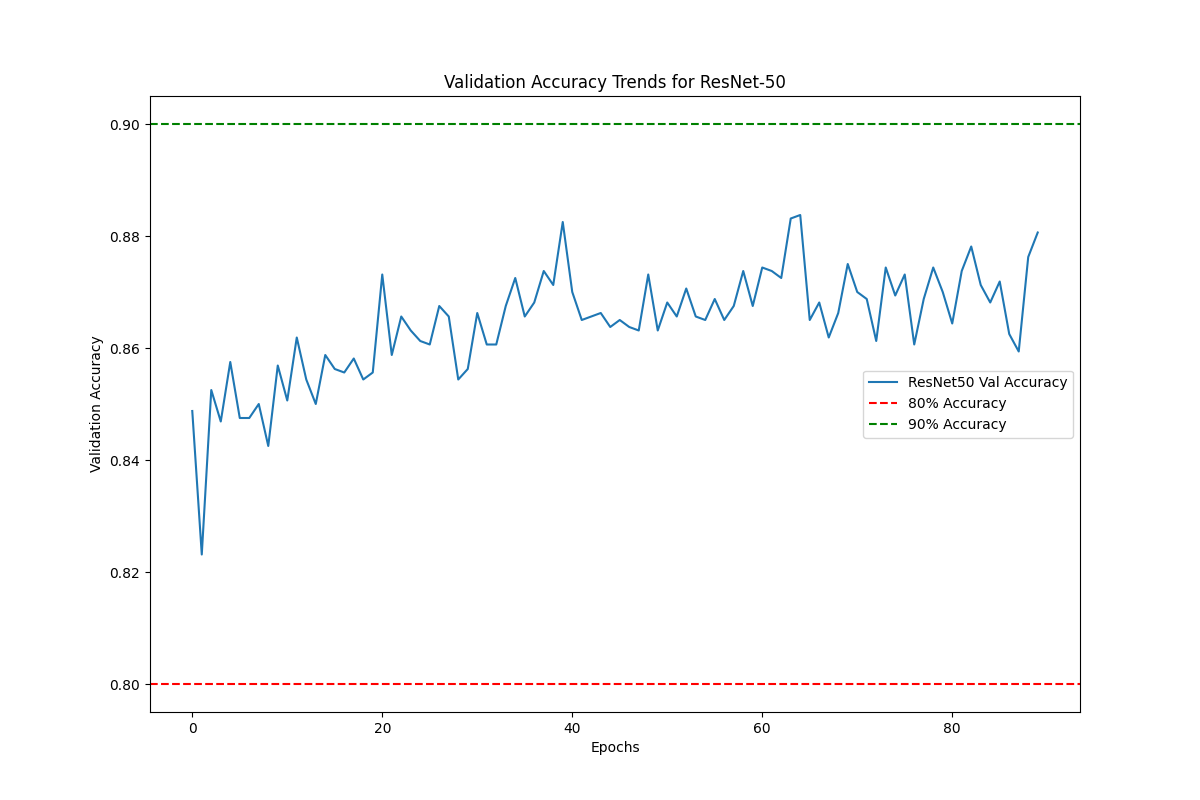
* Always good to double/triple check.

Go through 1 set of label: Center, right, and left to check.

orbit75\_box1\_1193.png

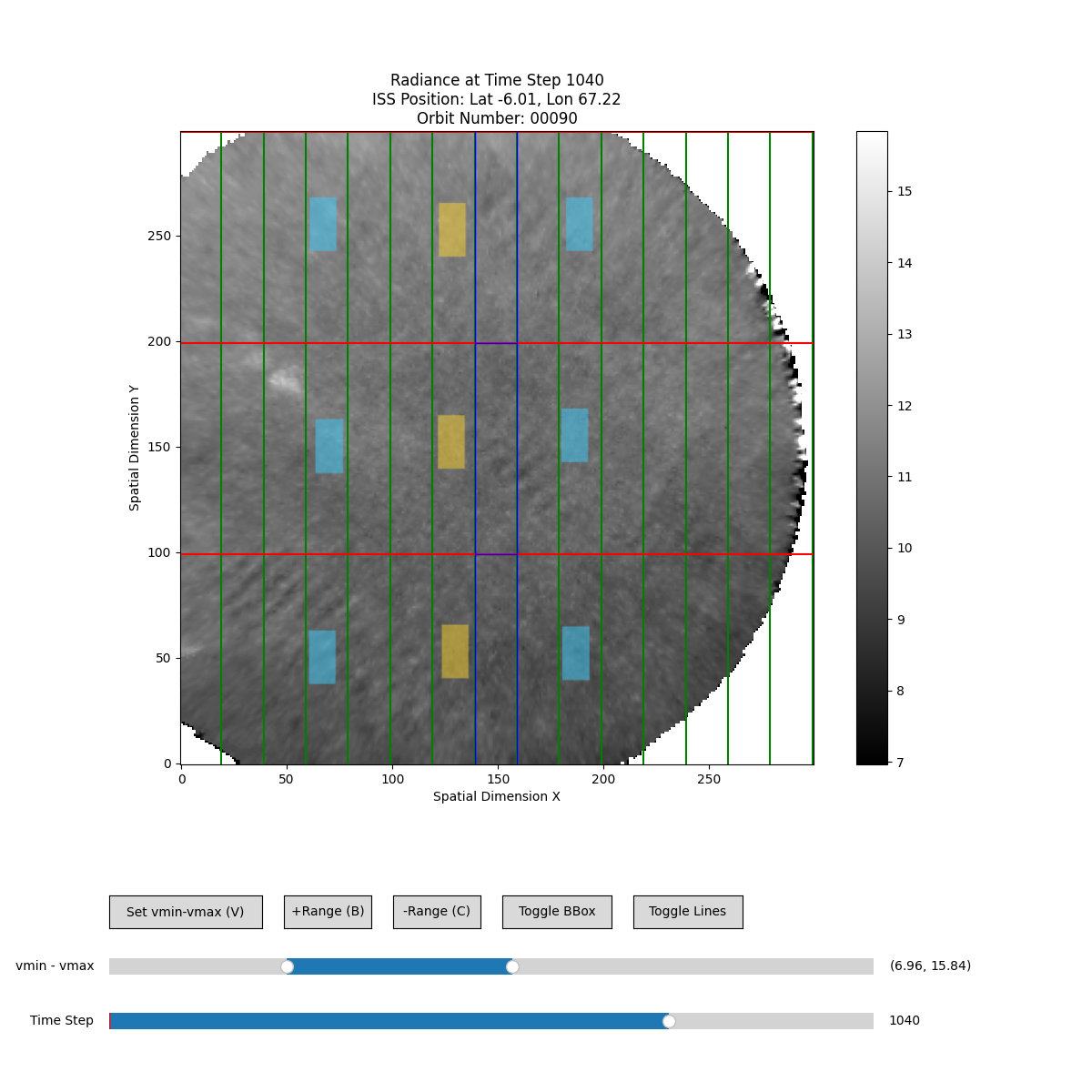
orbit75\_box4\_1180.png

orbit75\_box7\_1212.png

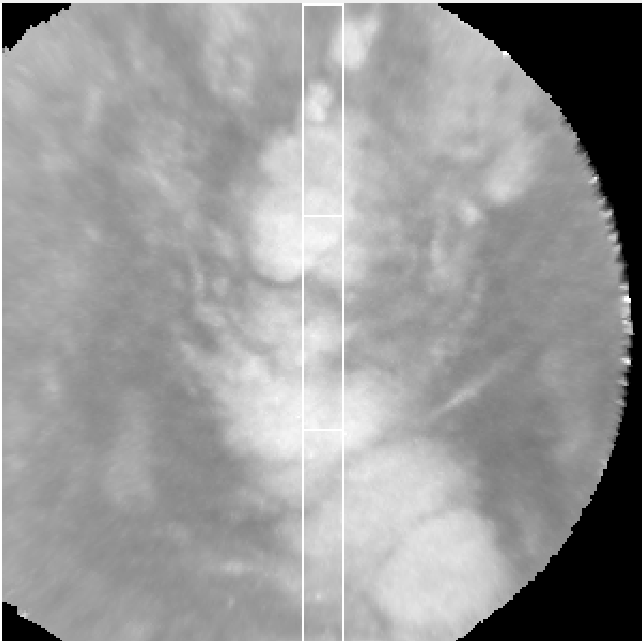


## Predicting

Generate images to predict



Check the generated images.

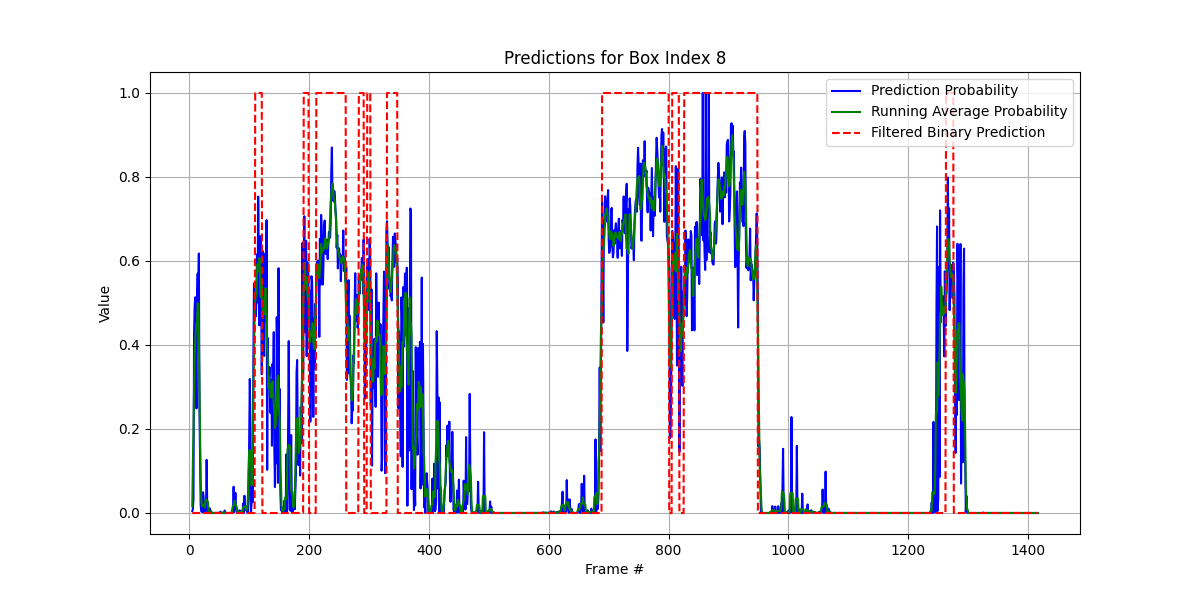
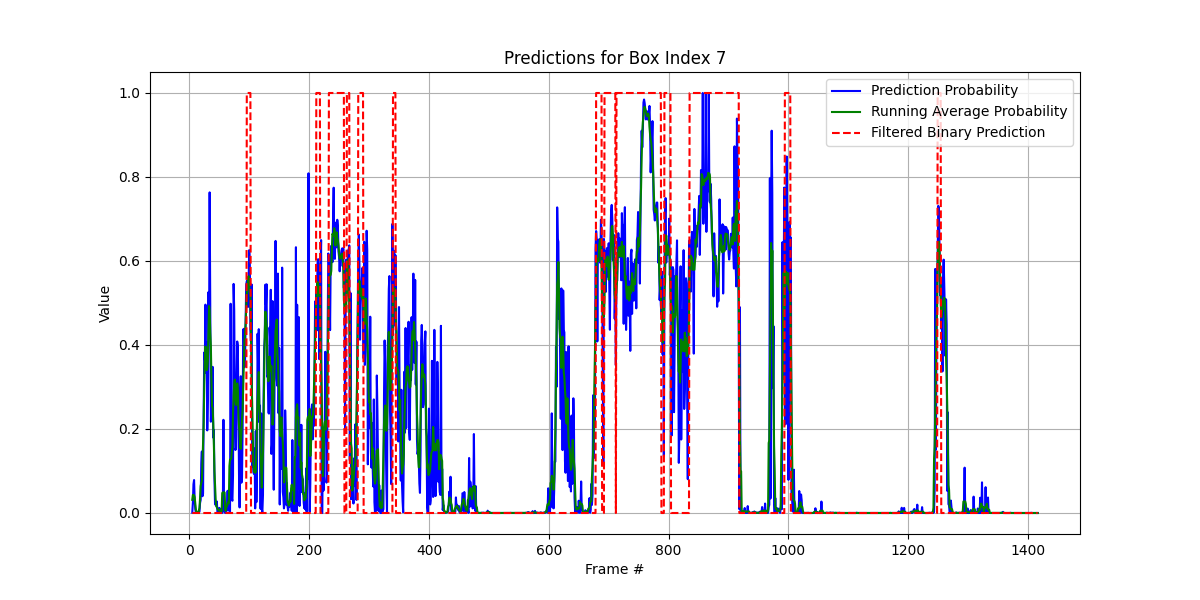
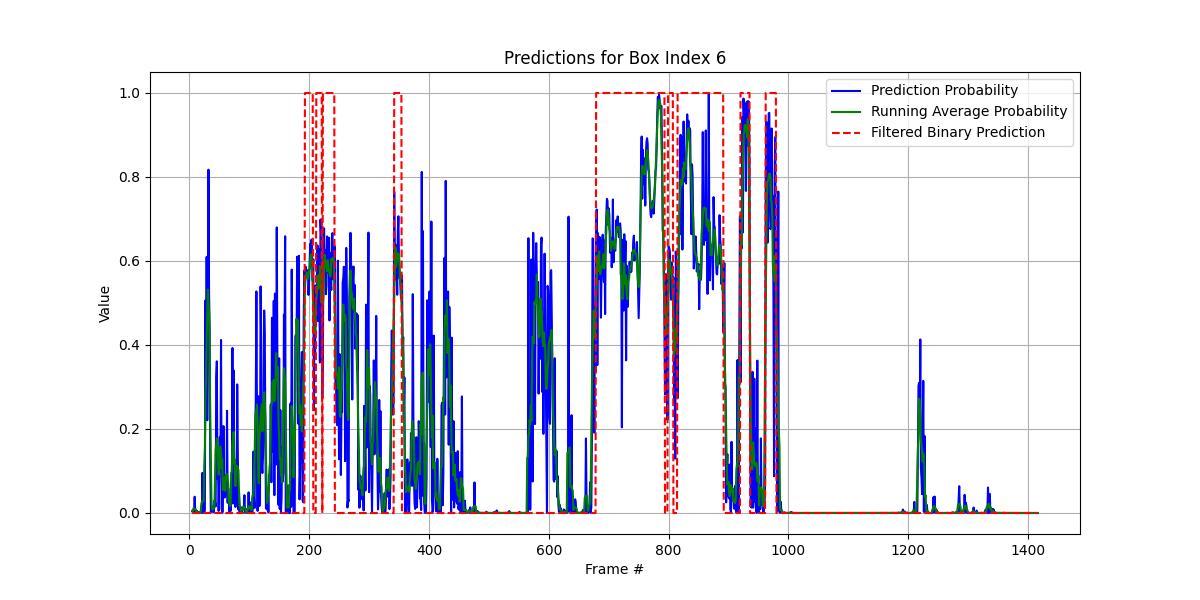
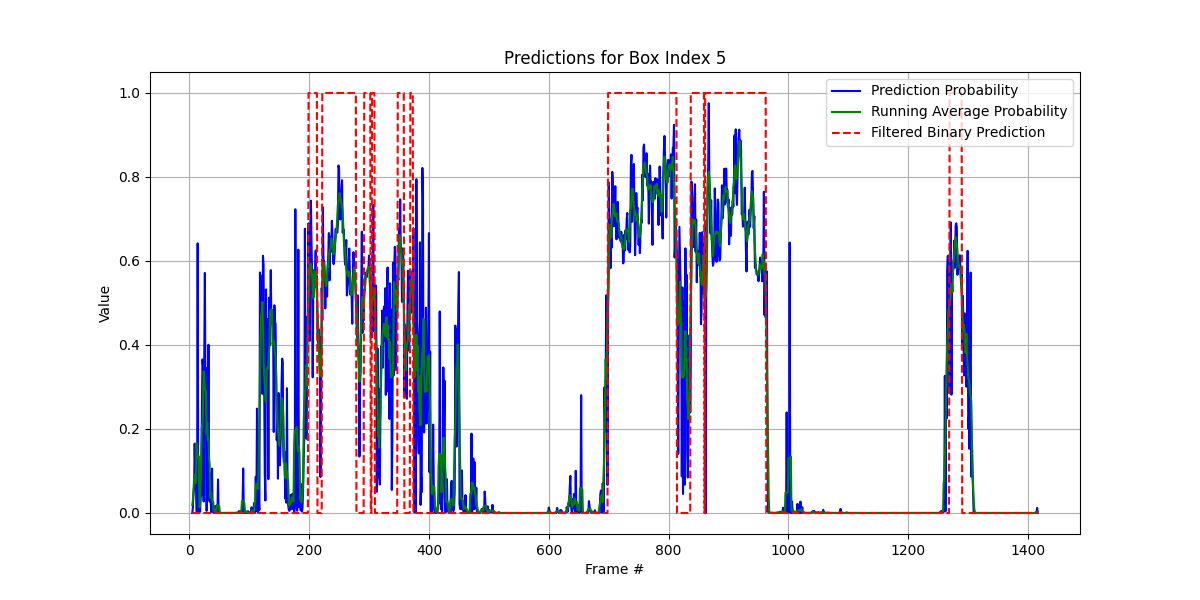
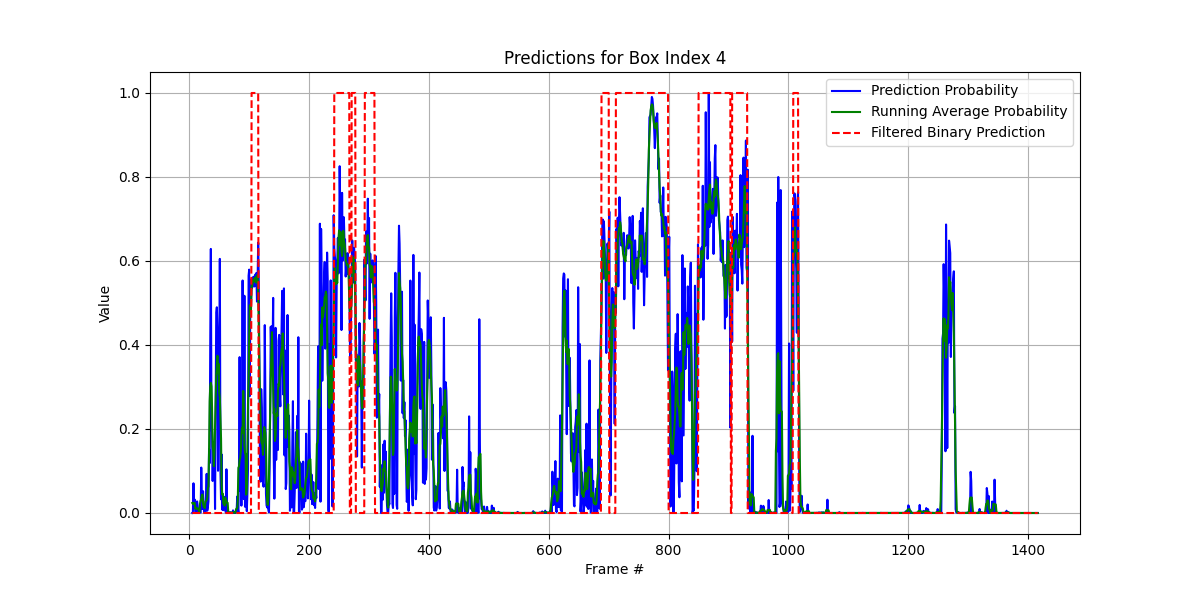
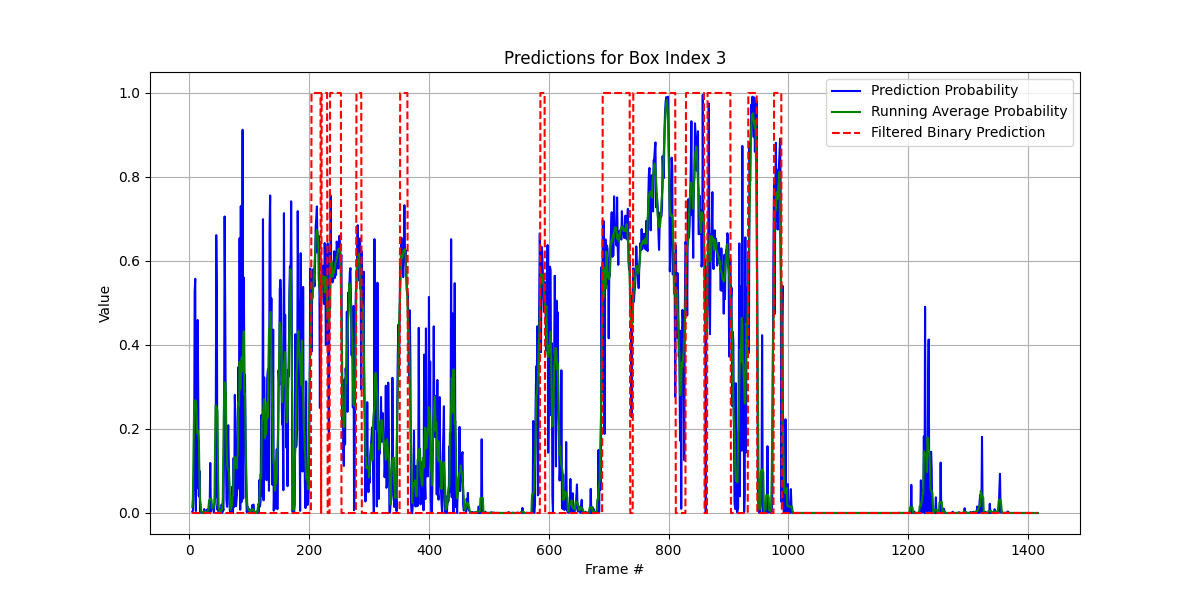
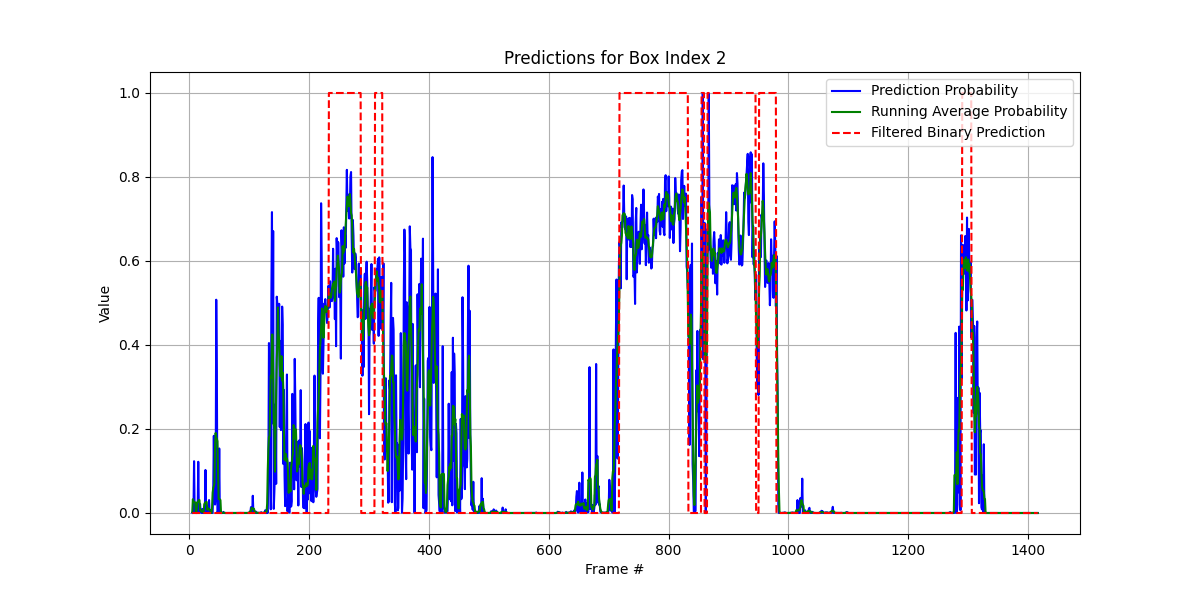
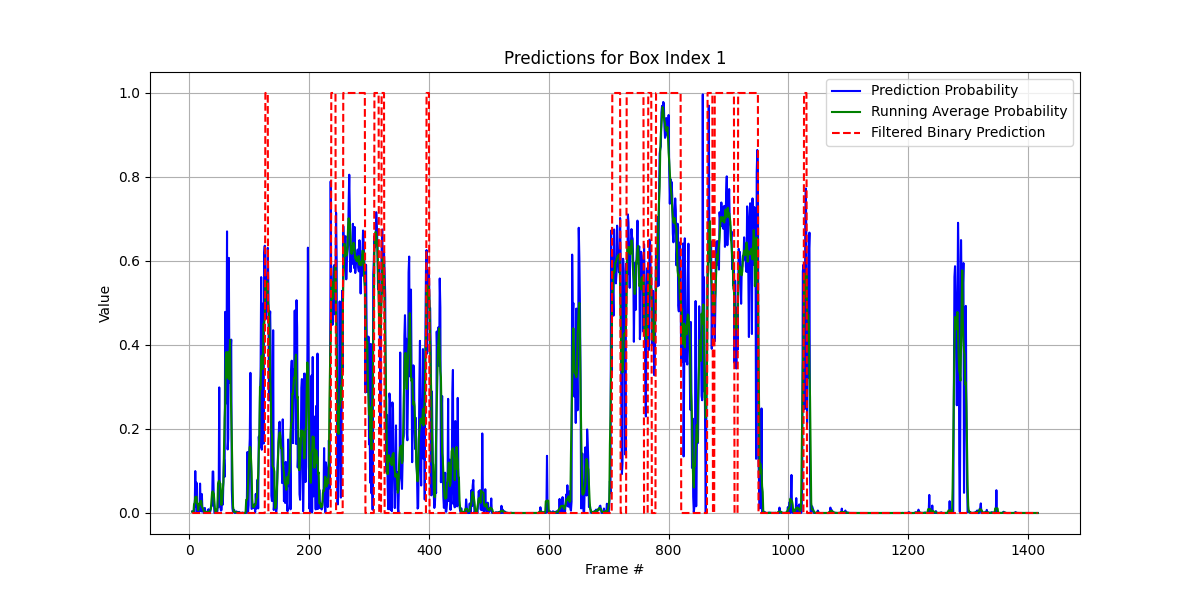
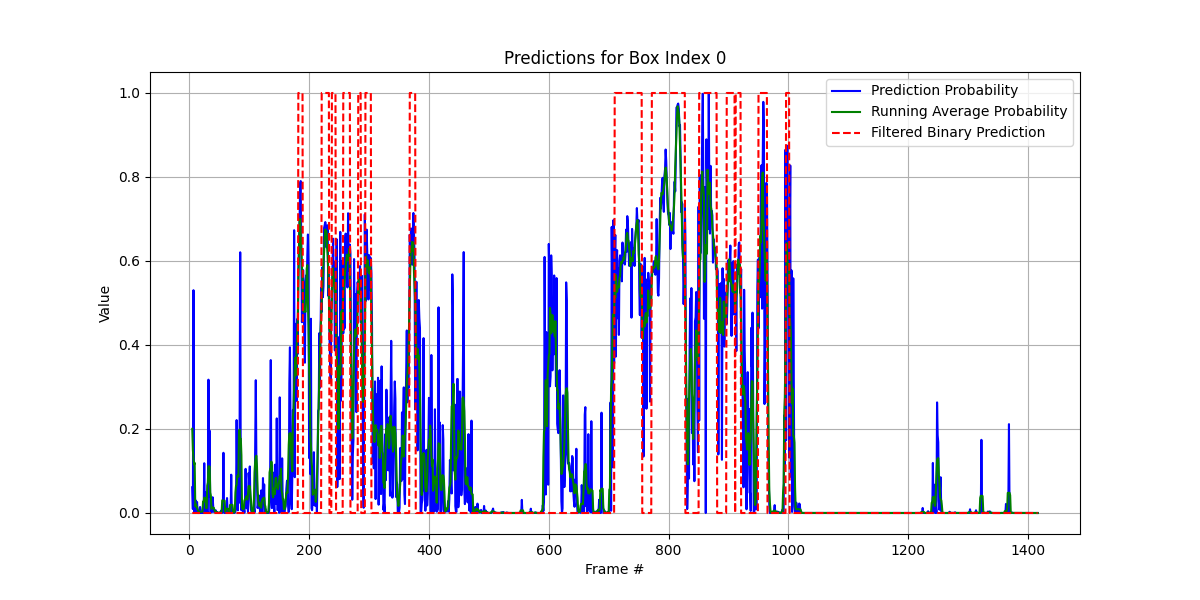


Frame 772, box 0  (Bottom center)

Frame 790, box 3(Bottom left)

Frame 759, box 6(Bottom right)

Predict orbit 90. Time needed to predict: 553.52 seconds



Modify the code so that the MLCloud variable is saved as a 2D array with 9 slices corresponding to the 9 boxes:

* Reshape the MLCloud data appropriately and store it in a new 2D array.
* Each row in this array will correspond to a specific time frame, and each column will represent one of the 9 boxes.

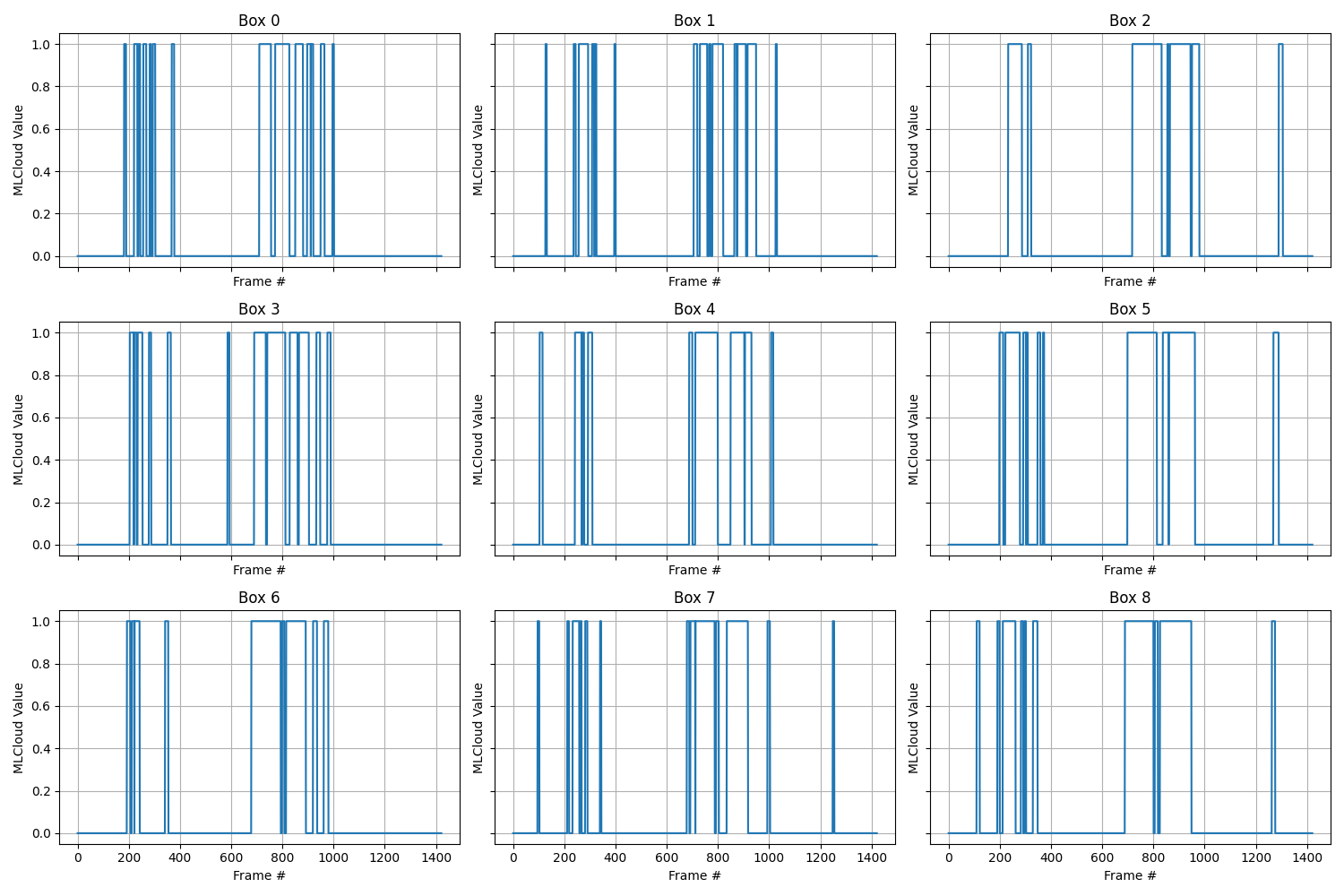
 **Creating the boxes Dimension:**

* The line dst\_nc.createDimension('boxes', 9) creates a new dimension called boxes with a size of 9 (since you have 9 boxes). This dimension is then used when creating the MLCloud variable.

 **MLCloud Variable Creation:**

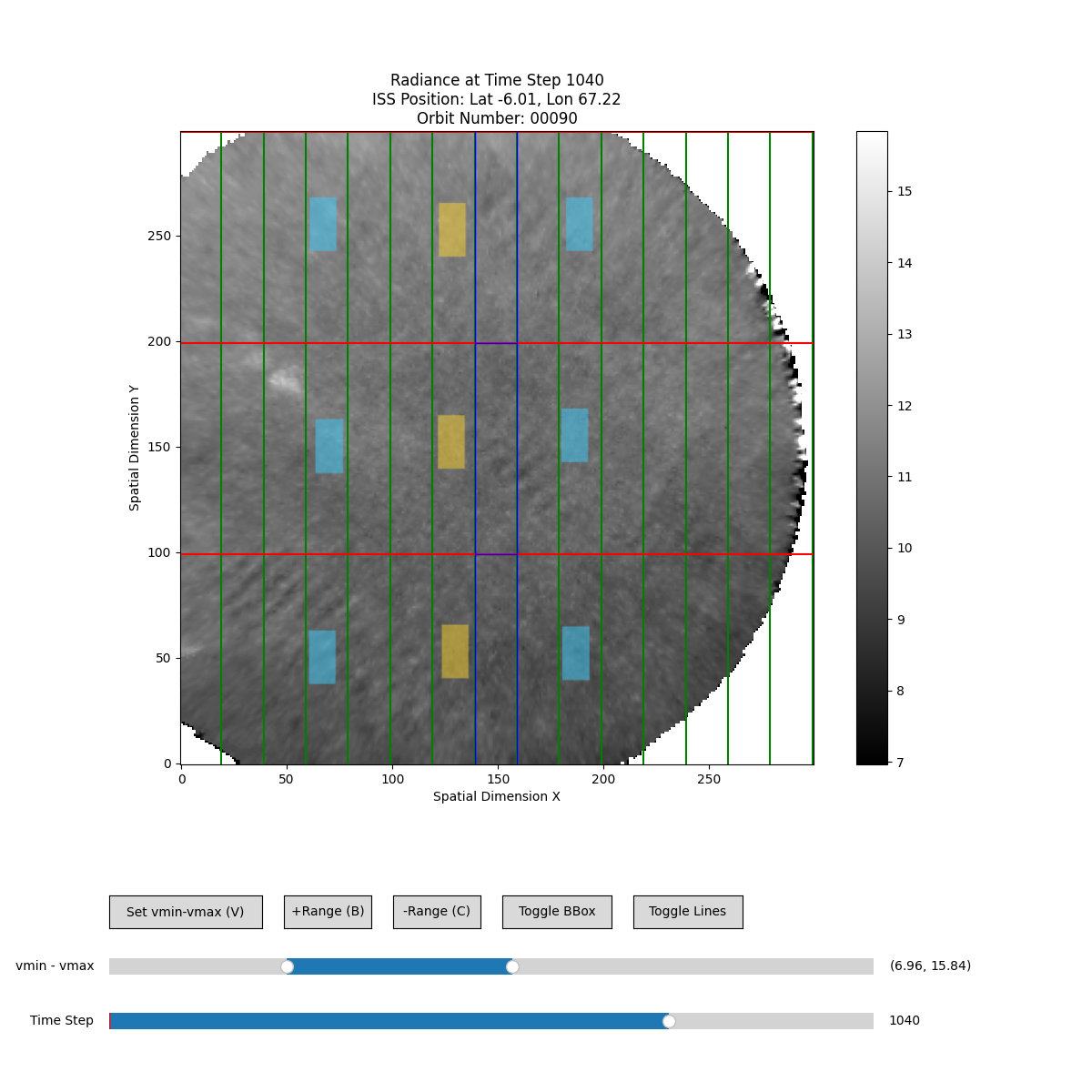
* The MLCloud variable is created with the dimensions ('time', 'boxes'), corresponding to the time dimension (frames) and the new boxes dimension.

Read the netCDF file and plot MLCloud



We can see the clouds moving to the left.

Projecting the prediction to other boxes.



boxes = [

(60, 80, 0, 100), #0

(60, 80, 100, 200), #1

(60, 80, 200, 300), #2

(120, 140, 0, 100), #3

(120, 140, 100, 200), #4

(120, 140, 200, 300), #5

(180, 200, 0, 100), #6

(180, 200, 100, 200), #7

(180, 200, 200, 300), #8

]

(0, 20, 0, 100), #9

(0, 20, 100, 200), #10

(0, 20, 200, 300), #11

Use boxes 0, 1, 2 by +19 frames

(20, 40, 0, 100), #12

(20, 40, 100, 200), #13

(20, 40, 200, 300), #14

Use boxes 0, 1, 2 by +13 frames

(40, 60, 0, 100), #15

(40, 60, 100, 200), #16

(40, 60, 200, 300), #17

Use boxes 0, 1, 2 by +6 frames

(80, 100, 0, 100), #18

(80, 100, 100, 200), #19

(80, 100, 200, 300), #20

Use boxes 0, 1, 2, by -6 frames

(100, 120, 0, 100), #21

(100, 120, 100, 200), #22

(100, 120, 200, 300), #23

Use boxes 3, 4, 5, by +6 frames

(140, 160, 0, 100), #24

(140, 160, 100, 200), #25

(140, 160, 200, 300), #26

Use boxes 3, 4, 5, by -6 frames

(160, 180, 0, 100), #27

(160, 180, 100, 200), #28

(160, 180, 200, 300), #29

Use boxes 6, 7, 8, by +6 frames

(200, 220, 0, 100), #30

(200, 220, 100, 200), #31

(200, 220, 200, 300), #32

Use boxes 6, 7, 8, by -6 frames

(220, 240, 0, 100), #33

(220, 240, 100, 200), #34

(220, 240, 200, 300), #35

Use boxes 6, 7, 8, by -13 frames

(240, 260, 0, 100), #36

(240, 260, 100, 200), #37

(240, 260, 200, 300), #38

Use boxes 6, 7, 8, by -19 frames

(260, 280, 100, 200), #39

Use box 7, by -25 frames

### Explanation of Changes:

1. **Extended Boxes:**
   * The script now includes additional boxes 9 through 39, which are derived by shifting the existing boxes by specified frame offsets.
2. **extend\_mlcloud\_with\_shifts Function:**
   * This function handles the creation of new boxes by applying the specified shifts to the original 9 boxes. The shifted values are extended with edge values to fill any gaps created by the shifts.
3. **Shift Handling:**
   * For each box with a shift, the corresponding original box is rolled by the specified number of frames (np.roll), and any resulting gaps are filled with the nearest available value.
4. **Dimension for 40 Boxes:**
   * The boxes dimension is updated to accommodate all 40 boxes in the output NetCDF file.

### Usage:

1. **Run the Script:**
   * Save the script and run it in your Python environment. The script processes the NetCDF files and generates new ones with the extended MLCloud variable including all 40 boxes.
2. **Result:**
   * You will get NetCDF files with the MLCloud variable containing 40 columns, each representing a different box with the appropriate frame shifts applied.

Consideration: Because edge-extension, the labels of the edge frames are from the nearest available value.

19+5 on the left, 25+5 on the right.