```
In[•]:= (***Input identifying information***)

In[•]:= date = ToString[Evaluate[Input["Input the date of the experiment"]]]

In[•]:= mouse = ToString[Evaluate[Input["Input the mouse identity (e.g. Mouse123)"]]]

In[•]:= sessionNum = Evaluate[Input["Input the session number"]]

In[•]:= (***Import the frame times for the 2P images and calculate the frame rate***)

In[•]:= tpFrameTimes =
          Drop[Drop[(Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/",
                mouse, "/Session", ToString[sessionNum], "/", date, "_", mouse, "_",
                "Session", ToString[sessionNum], "_2PFrameTimes.txt"], "List"]), 16], -1];

In[•]:= tpFrameRate = Round[Length[tpFrameTimes] / (Last[tpFrameTimes] - First[tpFrameTimes])];

In[•]:= (***For each ROI picked for the session, upload the extracted dF/F0 time series***)

In[•]:= numROIs =
          Length[FileNames["*", File[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/",
              mouse, "/Session", ToString[sessionNum], "/dFOverF0TimeSeries/"]]]];

In[•]:= Table[Evaluate@ToExpression[StringJoin["dFFts", ToString[n]]] =
            ToExpression /@ Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date,
                "/", mouse, "/Session", ToString[sessionNum], "/dFoverF0TimeSeries/",
                date, "_", mouse, "_Session", ToString[sessionNum], "_",
                "dFoverF0ts_ROI", ToString[n], ".txt"], "List"];, {n, 1, numROIs}];

In[•]:= (*******************************)
        (***Data for whisker time series***)
        (*******************************)

In[•]:= (**Import the raw pixel intensities from the whisker pad**)
        whiskPixIntensities =
          Part[#, 2] & /@ (Drop[Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/",
                mouse, "/Session", ToString[sessionNum], "/", date, "_", mouse, "_", "Session",
                ToString[sessionNum], "_Whiskers/WhiskerPadPixelIntensity/", "Results.csv"]], 1]);

In[•]:= (***Import the frame times from the camera***)

In[•]:= frameTimes =
          Drop[Drop[(Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
                "/Session", ToString[sessionNum], "/", date, "_", mouse, "_",
                "Session", ToString[sessionNum], "_CamSync.txt"], "List"]), 16], -1];

        (***Calculate camera frame rate***)
        frameRate = Round[Length[frameTimes] / (Last[frameTimes] - First[frameTimes])];

        (***Verify that length of frameTimes = length of whiskPixIntensities, +/- 2***)

In[•]:= Length[whiskPixIntensities]

In[•]:= Length[frameTimes]

In[•]:= (***Match up the length of the cam sync pulses with the length of the image frames***)
```

```
In[•]:= If[Length[frameTimes] > Length[whiskPixIntensities],
          frameTimes = Take[frameTimes, Length[whiskPixIntensities]];,
          whiskPixIntensities = Take[whiskPixIntensities, Length[frameTimes]];];

In[•]:= (***Calculate the motion energy of the whisker pad***)

In[•]:= whiskMotionEnergy = Abs /@ Differences[whiskPixIntensities];

In[•]:= whiskMotionEnergyTS = Partition[Riffle[Drop[frameTimes, 1], whiskMotionEnergy], 2];

In[•]:= whiskMotionInterp = Interpolation[whiskMotionEnergyTS];

In[•]:= (***Import the whisk onset and offset times calculated from the MATLAB code***)

In[•]:= whiskOnsets =
         Import[StringJoin["F:/", date, "/", mouse, "/Session", ToString[sessionNum], "/", date,
           "_", mouse, "_", "Session", ToString[sessionNum], "_whiskOnsetTimes.txt"], "List"];

In[•]:= whiskOffsets =
         Import[StringJoin["F:/", date, "/", mouse, "/Session", ToString[sessionNum], "/", date,
           "_", mouse, "_", "Session", ToString[sessionNum], "_whiskOffsetTimes.txt"], "List"];

In[•]:= onsetEvokedWhiskTraces = Table[Table[whiskMotionInterp[i], {i, whiskOnsets[[x]] - 3,
            whiskOnsets[[x]] + 3, 1/tpFrameRate}], {x, 1, Length[whiskOnsets]}];

In[•]:= offsetEvokedWhiskTraces =
         Table[Table[whiskMotionInterp[i], {i, whiskOffsets[[x]] - 3, whiskOffsets[[x]] + 3,
            1/tpFrameRate}], {x, 1, Length[whiskOnsets]}];

In[•]:= (***Import the "low whisk" states for use as a baseline for z-scored DFF***)

In[•]:= (***lowFaceStateOnsets=
         ToExpression/@StringSplit[Import[StringJoin["F:/",date,"/",mouse,"/Session",
              ToString[sessionNum],"/",date,"_",mouse,"_","Session",ToString[sessionNum],
              "_noWhiskStateOnsetTimes.txt"],"List"][[1]],","];***)

In[•]:= (***lowFaceStateOffsets=
         ToExpression/@StringSplit[Import[StringJoin["F:/",date,"/",mouse,
              "/Session",ToString[sessionNum],"/",date,"_",mouse,"_","Session",
              ToString[sessionNum],"_noWhiskStateOffsetTimes.txt"],"List"][[1]],","];***)

In[•]:= lowFaceStateOnsets = Import[
          StringJoin["F:/", date, "/", mouse, "/Session", ToString[sessionNum], "/", date, "_",
           mouse, "_", "Session", ToString[sessionNum], "_noWhiskStateOnsetTimes.txt"], "List"];

In[•]:= lowFaceStateOffsets = Import[
          StringJoin["F:/", date, "/", mouse, "/Session", ToString[sessionNum], "/", date, "_",
           mouse, "_", "Session", ToString[sessionNum], "_noWhiskStateOffsetTimes.txt"], "List"];

In[•]:= lowFaceStates = Partition[Riffle[lowFaceStateOnsets, lowFaceStateOffsets], 2];

In[•]:= (***Get interpolation functions for the dF/F0 time series for each ROI***)

In[•]:= Table[Evaluate@ToExpression[StringJoin["dFFInterpFunc", ToString[n]]] =
           Interpolation[ToExpression[StringJoin["dFFts", ToString[n]]]];, {n, 1, numROIs}];
```

```
In[∘]:= (***For each ROI,
        calculate the raw evoked dF/F0 for around each stimulus presentation onset,
        which includes 1 s prior stimulus onset and 3 s post stimulus offset***)
        Table[Evaluate@ToExpression[StringJoin["onsetEvokedRawDFFs", ToString[n]]] =
            Table[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]]) [i],
              {i, whiskOnsets[[x]] - 3, whiskOnsets[[x]] + 3, 1 / tpFrameRate}],
             {x, 1, Length[whiskOnsets]}];, {n, 1, numROIs}];
```

```
In[∘]:= (***For each ROI,
        calculate the z-scored dF/F0 for around each stimulus presentation onset,
        which includes 1 s prior stimulus onset and 3 s post stimulus offset***)
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["baselineMean", ToString[n]]] =
            Mean@Table[Mean[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]]) [k],
                {k, lowFaceStates[[x, 1]], lowFaceStates[[x, 2]], 1 / tpFrameRate}]],
              {x, 1, Length[lowFaceStates]}];, {n, 1, numROIs}];
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["baselineSD", ToString[n]]] =
            Mean@Table[StandardDeviation[
                Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]]) [k],
                 {k, lowFaceStates[[x, 1]], lowFaceStates[[x, 2]], 1 / tpFrameRate}]],
              {x, 1, Length[lowFaceStates]}];, {n, 1, numROIs}];
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["onsetEvokedZscoredDFFs", ToString[n]]] =
            Table[Table[(((ToExpression[StringJoin["dFFInterpFunc", ToString[n]]]) [i]) -
                  (ToExpression[StringJoin["baselineMean", ToString[n]]])) /
                ((ToExpression[StringJoin["baselineSD", ToString[n]]])),
              {i, whiskOnsets[[x]] - 3, whiskOnsets[[x]] + 3, 1 / tpFrameRate}],
             {x, 1, Length[whiskOnsets]}];, {n, 1, numROIs}];
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["offsetEvokedZscoredDFFs", ToString[n]]] =
            Table[Table[(((ToExpression[StringJoin["dFFInterpFunc", ToString[n]]]) [i]) -
                  (ToExpression[StringJoin["baselineMean", ToString[n]]])) /
                ((ToExpression[StringJoin["baselineSD", ToString[n]]])),
              {i, whiskOffsets[[x]] - 3, whiskOffsets[[x]] + 3, 1 / tpFrameRate}],
             {x, 1, Length[whiskOffsets]}];, {n, 1, numROIs}];
```

```
In[∘]:= (************)
```

```
In[∘]:= evokedTimeVals = Table[N@n, {n, -3, 3, 1 / tpFrameRate}];
```

```
In[∘]:= (***For each ROI, make a time series of z-scored dF/F0 and mean z-scored dF/F0,***)
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanZscoredOnsetDFFsTS", ToString[n]]] =
          Partition[Riffle[evokedTimeVals, Mean[ToExpression[
              StringJoin["onsetEvokedZscoredDFFs", ToString[n]]]]], 2], {n, 1, numROIs}];
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanZscoredOffsetDFFsTS", ToString[n]]] =
          Partition[Riffle[evokedTimeVals, Mean[ToExpression[
              StringJoin["offsetEvokedZscoredDFFs", ToString[n]]]]], 2], {n, 1, numROIs}];
```

```
In[•]:= Manipulate[{ListLinePlot[
           ToExpression[ToExpression[StringJoin["meanZscoredOnsetDFFsTS", ToString[n]]]]],
         ListLinePlot[ToExpression[ToExpression[
             StringJoin["meanZscoredOffsetDFFsTS", ToString[n]]]]]}, {n, 1, numROIs, 1}]

In[•]:= CreateDirectory[StringJoin["F:/", date, "/", mouse,
           "/Session", ToString[sessionNum], "/WhiskChangePointData/"]];

In[•]:= (***Export data***)

In[•]:= Table[Export[StringJoin["F:/", date, "/", mouse, "/Session",
            ToString[sessionNum], "/WhiskChangePointData/", date, "_", mouse, "_", "Session",
            ToString[sessionNum], "_meanZDFFts_whiskOn_ROI", ToString[n], ".txt"],
           ToExpression[StringJoin["meanZscoredOnsetDFFsTS", ToString[n]]]], {n, 1, numROIs}];

In[•]:= Table[Export[StringJoin["F:/", date, "/", mouse, "/Session",
            ToString[sessionNum], "/WhiskChangePointData/", date, "_", mouse, "_", "Session",
            ToString[sessionNum], "_meanZDFFts_whiskOff_ROI", ToString[n], ".txt"],
           ToExpression[StringJoin["meanZscoredOffsetDFFsTS", ToString[n]]]], {n, 1, numROIs}];

In[•]:= Export[StringJoin["F:/", date, "/", mouse, "/Session",
           ToString[sessionNum], "/WhiskChangePointData/", date, "_", mouse, "_",
           "Session", ToString[sessionNum], "_onsetEvokedWhiskTrace", ".txt"],
         Partition[Riffle[evokedTimeVals, Mean@onsetEvokedWhiskTraces], 2]];

In[•]:= Export[StringJoin["F:/", date, "/", mouse, "/Session",
           ToString[sessionNum], "/WhiskChangePointData/", date, "_", mouse, "_",
           "Session", ToString[sessionNum], "_offsetEvokedWhiskTrace", ".txt"],
         Partition[Riffle[evokedTimeVals, Mean@offsetEvokedWhiskTraces], 2]];
```