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

In[•]:= rois =
          Range[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, rois}];

In[•]:= (***Import the walk bout start and end times***)

In[•]:= walkBouts =
          ToExpression[Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
              "/Session", ToString[sessionNum], "/", "LocomotionData/", date, "_", mouse, "_",
              "Session", ToString[sessionNum], "_isolatedWalkBouts.txt"], "List"]];

In[•]:= (***)

In[•]:= numAdditionalMovies =
          Length[FileNames["*", File[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/",
                date, "/", mouse, "/Session", ToString[sessionNum], "/Ftraces/"]]]] - 2;
         (*Subtract 2 because the first movie was already imported and the
        other file in the directory is an ROI list*)

In[•]:= (***)

In[•]:= numFramesPerAcq = Round[Length[tpFrameTimes] / (numAdditionalMovies + 1)];

In[•]:= tpFrameRate = Round[Mean[numFramesPerAcq / Flatten[Differences /@
              ({First[#], Last[#]} & /@ (Partition[tpFrameTimes, numFramesPerAcq]))]]];

In[•]:= acqBoutIntervals =
          Interval /@ ({First[#], Last[#]} & /@ (Partition[tpFrameTimes, numFramesPerAcq]));

In[•]:= (****Test which locomotion onset times are within at least 15 s after the start
          of an acquisition bout and 6 s before the end of an acquisition bout****)
```

```
In[∘]:= acceptedLocOnsetTest = Table[AnyTrue[acqBoutIntervals,
             IntervalMemberQ[#, Interval[{walkBouts[[n, 1]] - 15, walkBouts[[n, 1]] + 6}]] &],
           {n, 1, Length[walkBouts]}];

In[∘]:= acceptedLocOnsetPositions = Position[acceptedLocOnsetTest, True];

In[∘]:= (****Test which locomotion offset times are within at least 6 s after the start
          of an acquisition bout and 15 s before the end of an acquisition bout****)

In[∘]:= acceptedLocOffsetTest = Table[AnyTrue[acqBoutIntervals,
             IntervalMemberQ[#, Interval[{walkBouts[[n, 2]] - 6, walkBouts[[n, 2]] + 15}]] &],
           {n, 1, Length[walkBouts]}];

In[∘]:= acceptedLocOffsetPositions = Position[acceptedLocOffsetTest, True];

In[∘]:= (***Only consider walk bouts that fulfill the above criteria***)

In[∘]:= walkBoutsForOnset = Table[walkBouts[[(Flatten[acceptedLocOnsetPositions][[n]])]],
           {n, 1, Length[acceptedLocOnsetPositions]}];

In[∘]:= walkBoutsForOffset = Table[walkBouts[[(Flatten[acceptedLocOffsetPositions][[n]])]],
           {n, 1, Length[acceptedLocOffsetPositions]}];

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, rois}];

In[∘]:= (***For each ROI, calculate the dF/F0 15 to 10 s prior to walk onset***)
        Table[Evaluate@ToExpression[StringJoin["quiescentPriorRawDFFs", ToString[n]]] =
             Table[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]])[i],
                {i, walkBoutsForOnset[[x, 1]] - 15, walkBoutsForOnset[[x, 1]] - 10, 1/tpFrameRate}],
               {x, 1, Length[walkBoutsForOnset]}];, {n, rois}];

In[∘]:= (***For each ROI, calculate the dF/F0 10 to 15 s post walk offset***)
        Table[Evaluate@ToExpression[StringJoin["quiescentPostRawDFFs", ToString[n]]] =
             Table[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]])[i],
                {i, walkBoutsForOffset[[x, 2]] + 10, walkBoutsForOffset[[x, 2]] + 15,
                 1/tpFrameRate}], {x, 1, Length[walkBoutsForOffset]}];, {n, rois}];

In[∘]:= (***For each ROI, calculate the raw evoked dF/F0 for around each walk bout onset,
        which includes 15 s prior to walk onset and 6 s post walk onset***)
        Table[Evaluate@ToExpression[StringJoin["onsetEvokedRawDFFs", ToString[n]]] =
             Table[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]])[i],
                {i, walkBoutsForOnset[[x, 1]] - 15, walkBoutsForOnset[[x, 1]] + 6, 1/tpFrameRate}],
               {x, 1, Length[walkBoutsForOnset]}];, {n, rois}];

In[∘]:= (***For each ROI, calculate the raw evoked dF/F0 for around each walk bout offset,
        which includes 6 s prior to walk offset and 15 s post walk offset***)
        Table[Evaluate@ToExpression[StringJoin["offsetEvokedRawDFFs", ToString[n]]] =
             Table[Table[(ToExpression[StringJoin["dFFInterpFunc", ToString[n]]])[i],
                {i, walkBoutsForOffset[[x, 2]] - 6, walkBoutsForOffset[[x, 2]] + 15, 1/tpFrameRate}],
               {x, 1, Length[walkBoutsForOffset]}];, {n, rois}];
```

```
In[∘]:= (***Compute the mean quiescent dF/F0 trace for each ROI***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanQuiescentPriorRawDFFs", ToString[n]]] =
            Mean[ToExpression[StringJoin["quiescentPriorRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanQuiescentPostRawDFFs", ToString[n]]] =
            Mean[ToExpression[StringJoin["quiescentPostRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= (***Compute the mean peri-walk onset dF/F0 trace for each ROI***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanOnsetEvokedRawDFFs", ToString[n]]] =
            Mean[ToExpression[StringJoin["onsetEvokedRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= (***Compute the mean peri-walk offset dF/F0 trace for each ROI***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanOffsetEvokedRawDFFs", ToString[n]]] =
            Mean[ToExpression[StringJoin["offsetEvokedRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= (***Compute the mean dF/F0 15 to 10 s prior to walk onset,
        which will serve as baseline for calculation of modulation index around walk onset***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["baselinePriorMean", ToString[n]]] = Mean[
             ToExpression[StringJoin["meanQuiescentPriorRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= (***Compute the mean dF/F0 10 to 15 s post walk offset,
        which will serve as baseline for calculation of modulation index around walk offset***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["baselinePostMean", ToString[n]]] =
            Mean[ToExpression[StringJoin["meanQuiescentPostRawDFFs", ToString[n]]]];, {n, rois}];

In[∘]:= (***Compute the mean dF/F0 3 s after
         walk onset (for overall mod. index calculation)***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["initWalkMean", ToString[n]]] =
            Mean[Take[ToExpression[StringJoin["meanOnsetEvokedRawDFFs", ToString[n]]],
               {tpFrameRate * 15, (tpFrameRate * 15) + tpFrameRate * 3}]];, {n, rois}];

In[∘]:= (***For each ROI, calculate the time-dependent modulation index around walk onset,
        which is (dF/F[t]-baselineMean)/(dF/F[t]+baselineMean)***)
        (***Note: was originally using pre walk onset as a baseline mean,
        but now using post walk offset as a baseline mean.  So,
        now the baseline is the same for both peri-
         walk onset and peri-walk offset calculations ***)
        Table[Evaluate@ToExpression[StringJoin["onsetEvokedModIndex", ToString[n]]] =
            Table[((ToExpression[StringJoin["meanOnsetEvokedRawDFFs", ToString[n]]])[[i]] -
                ToExpression[StringJoin["baselinePostMean", ToString[n]]]) /
              ((ToExpression[StringJoin["meanOnsetEvokedRawDFFs", ToString[n]]])[[i]] +
                ToExpression[StringJoin["baselinePostMean", ToString[n]]]), {i, 1, Length[
               (ToExpression[StringJoin["meanOnsetEvokedRawDFFs", ToString[n]]])]}];, {n, rois}];
```

```
In[∘]:= (***For each ROI, calculate the summary modulation index around walk onset,
        which is (walkOnsetMean-baselineMean)/(walkOnsetMean+baselineMean)***)
        Table[Evaluate@ToExpression[StringJoin["locModIndex", ToString[n]]] =
            (ToExpression[StringJoin["initWalkMean", ToString[n]]] -
               ToExpression[StringJoin["baselinePostMean", ToString[n]]]) /
             (ToExpression[StringJoin["initWalkMean", ToString[n]]] +
               ToExpression[StringJoin["baselinePostMean", ToString[n]]]);, {n, rois}];
```

```
In[∘]:= (***For each ROI, calculate the time-dependent modulation index around walk offset,
        which is (dF/F[t]-baselineMean)/(dF/F[t]+baselineMean)***)
        Table[Evaluate@ToExpression[StringJoin["offsetEvokedModIndex", ToString[n]]] =
            Table[((ToExpression[StringJoin["meanOffsetEvokedRawDFFs", ToString[n]]])[[i]] -
                 ToExpression[StringJoin["baselinePostMean", ToString[n]]]) /
               ((ToExpression[StringJoin["meanOffsetEvokedRawDFFs", ToString[n]]])[[i]] +
                 ToExpression[StringJoin["baselinePostMean", ToString[n]]]),
             {i, 1, Length[(ToExpression[StringJoin["meanOffsetEvokedRawDFFs",
                   ToString[n]]])]}];, {n, rois}];
```

```
In[∘]:= timeValsOnset = Table[N@n, {n, -15, 6, 1/tpFrameRate}];
```

```
In[∘]:= timeValsOffset = Table[N@n, {n, -6, 15, 1/tpFrameRate}];
```

```
In[∘]:= (***For each ROI,
        make a time series of the time-dependent modulation index around walk onset***)
        Table[Evaluate@ToExpression[StringJoin["onsetEvokedModIndexTS", ToString[n]]] =
            Partition[Riffle[timeValsOnset,
               (ToExpression[StringJoin["onsetEvokedModIndex", ToString[n]]])], 2];, {n, rois}];
```

```
In[∘]:= (***For each ROI,
        make a time series of the time-dependent modulation index around walk offset***)
        Table[Evaluate@ToExpression[StringJoin["offsetEvokedModIndexTS", ToString[n]]] =
            Partition[Riffle[timeValsOffset,
               (ToExpression[StringJoin["offsetEvokedModIndex", ToString[n]]])], 2];, {n, rois}];

        locModROIs = rois;
```

```
In[∘]:= (***Export results***)
```

```
In[∘]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse, "/Session",
            ToString[sessionNum], "/", "LocomotionData/", date, "_", mouse, "_", "Session",
            ToString[sessionNum], "_SummaryLocModIndex_Baseline15sAway_ROI", ToString[n], ".txt"],
          ToExpression[StringJoin["locModIndex", ToString[n]]]], {n, rois}];
```

```
In[∘]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/",
            date, "/", mouse, "/Session", ToString[sessionNum], "/",
            "LocomotionData/", date, "_", mouse, "_", "Session", ToString[sessionNum],
            "_PeriOnsetLocModIndexTimeSeries_Baseline15sAway_ROI", ToString[n], ".txt"],
          ToExpression[StringJoin["onsetEvokedModIndexTS", ToString[n]]]], {n, rois}];
```

```
In[•]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/",
            date, "/", mouse, "/Session", ToString[sessionNum], "/",
            "LocomotionData/", date, "_", mouse, "_", "Session", ToString[sessionNum],
            "_PeriOffsetLocModIndexTimeSeries_Baseline15sAway_ROI", ToString[n], ".txt"],
          ToExpression[StringJoin["offsetEvokedModIndexTS", ToString[n]]]], {n, rois}];
```

```
In[•]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
          "/Session", ToString[sessionNum], "/", "LocomotionData/", date, "_", mouse,
          "_", "Session", ToString[sessionNum], "_locModROIs", ".txt"], locModROIs];
```