```
(***Input identifying information***)

date = ToString[Evaluate[Input["Input the date of the experiment"]]]

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

sessionNum = Evaluate[Input["Input the session number"]]

(***Import the raw Spike2 wheel trace***)

rawWheelTrace =
  Drop[Drop[(Import[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/",
       mouse, "/Session", ToString[sessionNum], "/", date, "_", mouse, "_",
       "Session", ToString[sessionNum], "_Wheel.txt"], "List"]), 18], -1];

(***Convert the raw voltage trace to wheel phase angle in radians***)

wheelScaleFactor = (2 * Pi) / (Max[rawWheelTrace] - Min[rawWheelTrace]);

scaledWheelTrace = rawWheelTrace * wheelScaleFactor;

wheelBias = Min[scaledWheelTrace];

wheelPhaseAngle = scaledWheelTrace - wheelBias;

wheelRad = (15.9) / 2;(**cm**)

initAngle = wheelPhaseAngle[[1]];

(**********************************************************)

Unwrap[a_] := a - initAngle Prepend[Accumulate[Round[Differences[a] / initAngle]], 0];
(***Function for unwrapping the wheel phase angle***)

unwrappedWheelPhaseAngle = Unwrap[wheelPhaseAngle];

distTrav = unwrappedWheelPhaseAngle * wheelRad;(***Calculate distance traveled
  (in cm) by multiplying the unwrapped phase angle by the wheel radius***)

smoothWindow = 500;(***100 ms at 5000 Hz sampling frequency***)

distTravSmooth = MovingMap[Mean, distTrav, smoothWindow, "Reflected"];
(***Smooth the distance traveled trace***)

timeVals = Table[n, {n, 0, (Length[rawWheelTrace] - 1) / 5000, N@(1 / 5000)}];
(***Time points at a 5000 Hz sampling frequency***)

distTravSmoothTimeSeries = Partition[Riffle[timeVals, distTravSmooth], 2];

distTravRawTimeSeries = Partition[Riffle[timeVals, distTrav], 2];

distTravSmoothInterp = Interpolation[distTravSmoothTimeSeries];

yData = Abs[Differences[Partition[Riffle[timeVals, distTravSmooth], 2]] /. {x_, y_} → y / x];

wheelSpeedTS = Partition[Riffle[timeVals, yData], 2];

wheelSpeedInterp = Interpolation[wheelSpeedTS];
```

```
wheelSpeedTSDownSamp = Table[{t, wheelSpeedInterp[t]}, {t, 0, Last[timeVals], 0.001}];
(***1 kHZ***)

yDataDownSamp = wheelSpeedTSDownSamp[[All, 2]];

xDataDownSamp = wheelSpeedTSDownSamp[[All, 1]];

fastData = ExponentialMovingAverage[yDataDownSamp, 5 * 10^-3];

slowData = ExponentialMovingAverage[yDataDownSamp, 5 * 10^-6];

fastDataTS = Partition[Riffle[xDataDownSamp, fastData], 2];

slowDataTS = Partition[Riffle[xDataDownSamp, slowData], 2];

fastInt = Interpolation[fastDataTS];

slowInt = Interpolation[slowDataTS];

candChangeTimes = x /. FindRoot[fastInt[x] - slowInt[x], {x, #[[1, 1]], #[[2, 1]]}] & /@
    Select[Partition[Sort@Last@Last@Reap[Plot[fastInt[x] - slowInt[x],
            {x, First[xDataDownSamp], Last[xDataDownSamp]}, EvaluationMonitor :>
             Sow[{x, fastInt[x] - slowInt[x]}]]], 2, 1], #[[2, 2]] #[[1, 2]] ≤ 0 &];

candChangePoints = Round[((# * 1000) + 1) & /@ candChangeTimes];

candLocOnsetPoints = {};

candLocOffsetPoints = {};

(***A candidate change point is a candidate locomotion onset point if the average speed
 100 ms after the change point is greater than the average speed 100 ms before
 the change point.  The inverse for a candidate locomotion offset point.***)
Table[
  If[Mean[Table[fastData[[i]], {i, candChangePoints[[n]] - 100, candChangePoints[[n]]}]] <
    Mean[Table[fastData[[i]], {i, candChangePoints[[n]], candChangePoints[[n]] + 100}]],
   candLocOnsetPoints = Append[candLocOnsetPoints, candChangePoints[[n]]];,
   candLocOffsetPoints = Append[candLocOffsetPoints, candChangePoints[[n]]];],
  {n, 1, Length[candChangePoints]}];

(***Accept a candidate locomotion onset point if
 the previous change point was a locomotion offset point,
unless it's the first candidate locomotion onset point AND there
 were no candidate locomotion offset points before it***)
acceptedLocOnsetPoints =
  DeleteCases[Table[Which[n == 1 && First[candLocOffsetPoints] > candLocOnsetPoints[[n]],
     candLocOnsetPoints[[n]], MemberQ[candLocOffsetPoints,
      Nearest[DeleteCases[candChangePoints, x_ /; x ≥ candLocOnsetPoints[[n]]],
       candLocOnsetPoints[[n]]][[1]]], candLocOnsetPoints[[n]],
     True, Null], {n, 1, Length[candLocOnsetPoints]}], Null];
```

```
In[•]:= (***Accept a candidate locomotion offset point
         if the previous change point was a locomotion onset point,
        unless it's the first candidate locomotion offset point AND there
         were no candidate locomotion onset points before it***)
        acceptedLocOffsetPoints =
          DeleteCases[Table[Which[n == 1 && First[candLocOnsetPoints] > candLocOffsetPoints[[n]],
              candLocOffsetPoints[[n]], MemberQ[candLocOnsetPoints,
               Nearest[DeleteCases[candChangePoints, x_ /; x ≥ candLocOffsetPoints[[n]]],
                 candLocOffsetPoints[[n]]][[1]]], candLocOffsetPoints[[n]],
              True, Null], {n, 1, Length[candLocOffsetPoints]}], Null];

In[•]:= acceptedLocOnsetTimes = ((# * 0.001) - 0.001) & /@ acceptedLocOnsetPoints;

In[•]:= acceptedLocOffsetTimes = ((# * 0.001) - 0.001) & /@ acceptedLocOffsetPoints;

In[•]:= candLocOnsetTimes = ((# * 0.001) - 0.001) & /@ candLocOnsetPoints;

In[•]:= candLocOffsetTimes = ((# * 0.001) - 0.001) & /@ candLocOffsetPoints;

        (***Examine candidate locomotion onset times and
         populate the list "locOnsetTimesWalkBouts" as the final
         list of locomotion onset times for each walk bout***)

In[•]:= Manipulate[{Quiet@Show[Plot[{fastInt[t], slowInt[t], wheelSpeedInterp[t]},
              {t, candLocOnsetTimes[[n]] - 30, candLocOnsetTimes[[n]] + 30},
              PlotStyle → {Directive[Green, Opacity[0.5]], Directive[Red, Opacity[0.5]],
                Directive[Black, Opacity[0.5]]}, PlotRange → {0, 40}],
             ListLinePlot[{{candLocOnsetTimes[[n]], 0}, {candLocOnsetTimes[[n]], 50}},
              PlotStyle → {Orange, Dotted, Thick}]],
          candLocOnsetTimes[[n]]}, {n, 1, Length[candLocOnsetTimes], 1}]

        locOnsetTimesWalkBouts = {};

        (***Examine candidate locomotion offset times
         and populate the list "locOffsetTimesWalkBouts" as the
         final list of locomotion offset times for each walk bout***)

In[•]:= Quiet[Manipulate[{Quiet@Show[Plot[{fastInt[t], slowInt[t], wheelSpeedInterp[t]},
               {t, candLocOffsetTimes[[n]] - 30, candLocOffsetTimes[[n]] + 30},
               PlotStyle → {Directive[Green, Opacity[0.5]], Directive[Red, Opacity[0.5]],
                 Directive[Black, Opacity[0.5]]}, PlotRange → {0, 50}],
              ListLinePlot[{{candLocOffsetTimes[[n]], 0}, {candLocOffsetTimes[[n]], 50}},
               PlotStyle → {Orange, Dotted}], ListLinePlot[
               Table[{{locOnsetTimesWalkBouts[[n]], 0}, {locOnsetTimesWalkBouts[[n]], 50}},
                {n, 1, Length[locOnsetTimesWalkBouts]}],
               PlotStyle → Table[{Blue, Dotted, Thick}, {Length[locOnsetTimesWalkBouts]}]]],
            candLocOffsetTimes[[n]]}, {n, 1, Length[candLocOffsetTimes], 1}]]

        locOffsetTimesWalkBouts = {};

In[•]:= (***Check that length of true locomotion onset
         times equals length of true locomotion offset times***)
```

```
In[•]:= {{Length[locOnsetTimesWalkBouts], Length[locOffsetTimesWalkBouts]},
         Length[locOnsetTimesWalkBouts] == Length[locOffsetTimesWalkBouts]}
Out[•]= {{11, 11}, True}

In[•]:= (***Enumerate intervals of walk bouts***)

In[•]:= walkBouts = Partition[Riffle[locOnsetTimesWalkBouts, locOffsetTimesWalkBouts], 2];

        (***Visualize the locomotion onset/offset points around each walk bout***)

In[•]:= Quiet[Manipulate[
          Quiet@Show[Plot[wheelSpeedInterp[t], {t, walkBouts[[n, 1]] - 30, walkBouts[[n, 2]] + 30},
              PlotStyle → Black, PlotRange → {0, 50}], ListLinePlot[
              {{walkBouts[[n, 1]], 0}, {walkBouts[[n, 1]], 50}}, PlotStyle → {Blue, Dotted}],
            ListLinePlot[{{walkBouts[[n, 2]], 0}, {walkBouts[[n, 2]], 50}},
              PlotStyle → {Blue, Dotted}]], {n, 1, Length[walkBouts], 1}]]

In[•]:= (***Get wheel speed traces 10 s before and 10 s after
         locomotion onset/offset.  Downsample wheel speed traces to 500 Hz***)

In[•]:= periLocOnsetWheelSpeeds = Table[Table[wheelSpeedInterp[t],
            {t, walkBouts[[n, 1]] - 10, walkBouts[[n, 1]] + 10, 0.002}], {n, 1, Length[walkBouts]}];

In[•]:= periLocOffsetWheelSpeeds = Table[Table[wheelSpeedInterp[t],
            {t, walkBouts[[n, 2]] - 10, walkBouts[[n, 2]] + 10, 0.002}], {n, 1, Length[walkBouts]}];

In[•]:= (***Determine candidate quiescent periods.  These are periods 2 s after locomotion
         offset of one walk bout and 2 s before locomotion onset of the next walk bout***)

In[•]:= candQuietPeriods =
          Table[{walkBouts[[n, 2]] + 2, walkBouts[[n + 1, 1]] - 2}, {n, 1, Length[walkBouts] - 1}];

In[•]:= trueQuietPeriods =
          DeleteCases[Table[If[candQuietPeriods[[n, 2]] - candQuietPeriods[[n, 1]] ≥ 5 &&
               Mean[Table[wheelSpeedInterp[t],
                  {t, candQuietPeriods[[n, 1]], candQuietPeriods[[n, 2]], 0.002}]] < 1,
             candQuietPeriods[[n]], Null], {n, 1, Length[candQuietPeriods]}], Null];

In[•]:= (**************Export data***************)

In[•]:= (***Export walk bouts***)

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

In[•]:= (***Export incomplete walk bouts bouts***)

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

In[•]:= (***Export quiescent bouts***)

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

```
In[ ]:= (***Export peri-locomotion-onset wheel speeds***)

In[ ]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/",
    date, "/", mouse, "/Session", ToString[sessionNum], "/",
    "LocomotionData/", date, "_", mouse, "_", "Session", ToString[sessionNum],
    "_PeriLocOnsetWheelSpeeds_20s_500Hz.txt"], periLocOnsetWheelSpeeds];

In[ ]:= (***Export peri-locomotion-offset wheel speeds***)

In[ ]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/",
    date, "/", mouse, "/Session", ToString[sessionNum], "/",
    "LocomotionData/", date, "_", mouse, "_", "Session", ToString[sessionNum],
    "_PeriLocOffsetWheelSpeeds_20s_500Hz.txt"], periLocOffsetWheelSpeeds];

In[ ]:= (*********************************************************)

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[ ]:= interWalkBoutInts =
  Append[Table[walkBouts[[n + 1, 1]] - walkBouts[[n, 2]], {n, 1, Length[walkBouts] - 1}],
   Last[tpFrameTimes] - walkBouts[[Length[walkBouts], 2]]];

In[ ]:= shortInts = Position[interWalkBoutInts, _?(# < 15 &)];

In[ ]:= newWalkBouts = Delete[walkBouts, shortInts];

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