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

In[•]:= (***Import pupil diameter traces***)

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 pupil diameter time series***)

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

In[•]:= timeVals = Part[#, 1] & /@ pupilTimeSeriesBef;

In[•]:= (***Create pupil diameter time series,
        excluding NaN elements from edge detection abnormalities***)

In[•]:= pupilTimeSeries =
          DeleteCases[Table[If[NumberQ@pupilTimeSeriesBef[[n, 2]], {pupilTimeSeriesBef[[n, 1]],
              pupilTimeSeriesBef[[n, 2]]}, Null], {n, 1, Length[pupilTimeSeriesBef]}], Null];

In[•]:= (***Interpolate the pupil diameter time series and resample at 10 Hz***)

In[•]:= pupilInterp = Interpolation[pupilTimeSeries];

In[•]:= pupil10Hz = Table[pupilInterp[t],
           {t, First[pupilTimeSeries][[1]], Last[pupilTimeSeries][[1]], 0.1}];

In[•]:= pupil10HzTimeSeries = Partition[Riffle[Table[n,
             {n, First[pupilTimeSeries][[1]], Last[pupilTimeSeries][[1]], 0.1}], pupil10Hz], 2];

In[•]:= (***Band-pass filter the resampled pupil diameter trace from 0.1 to 1 Hz***)

In[•]:= fftFilter[data_, sR_, band_] := Module[{dur, fRes, f1, f2, rSpan, f}, dur = Length@data / sR;
          (*Total duration of data.*)fRes = 1 / dur;
          (*Resolution of FFT is 1/duration.*)
          (*frequencies in terms of Fourier part.1=DC*){f1, f2} = 1 + Quotient[#, fRes] & /@ band;
          rSpan = If[f1 == 1, Span @@ {2, -1}, Span @@ {1, -1}];
          f = Fourier[data, List /@ Range[f1, f2]];
          f = PadLeft[f, f2];
          f = Join[PadRight[f, (Length@data) - f2 + 1], Conjugate@Reverse@(f[[rSpan]])];
          f = PadRight[f, Length@data];
          Re@InverseFourier[f]]

In[•]:= bandPassPupil = fftFilter[pupil10Hz, 10, {0.1, 1}];

In[•]:= bandPassPupilTimeSeries = Partition[
           Riffle[Table[n, {n, First[pupilTimeSeries][[1]], Last[pupilTimeSeries][[1]], 0.1}],
            bandPassPupil], 2];

In[•]:= bandPassPupilInt = Interpolation[bandPassPupilTimeSeries];
```

```
In[∘]:= (***Function for Hilbert transform***)

In[∘]:= (***hilbert[data_?VectorQ]:=
         Module[{fopts=FourierParameters→{1,-1},e,n},e=Boole[EvenQ[n=Length[data]]];
           Im[InverseFourier[Fourier[data,fopts]*
              PadRight[ArrayPad[ConstantArray[2,Quotient[n,2]-e],{1,e},1],n],
            fopts]]]/;And@@Thread[Im[data]⩵0]***)

In[∘]:= (***Take the Hilbert transform of the low-pass-filtered pupil diameter data***)

In[∘]:= (***hilbertPupilBPF=hilbert[bandPassPupil];***)

In[∘]:= (***Get the phases of the low-pass-filtered pupil diameter data***)

In[∘]:= (***phasesPupilBPF=Table[
           ArcTan[bandPassPupil[[n]],hilbertPupilBPF[[n]]],{n,1,Length[bandPassPupil]}];***)

In[∘]:= (***phasesPupilBPFTimeSeries=Partition[
           Riffle[Table[n,{n,First[pupilTimeSeries][[1]],Last[pupilTimeSeries][[1]],0.1}],
            phasesPupilBPF],2];***)

In[∘]:= (***phasesPupilInt=Interpolation[phasesPupilBPFTimeSeries];***)

In[∘]:= (*********************)

In[∘]:= pupilDerivTimeSeries = Table[{t, bandPassPupilInt'[t]},
           {t, First[pupilTimeSeries][[1]], Last[pupilTimeSeries][[1]], 0.01}];

In[∘]:= pupilDerivInt = Interpolation[pupilDerivTimeSeries];

In[∘]:= zeroCrossings[l_] :=
         Module[{z, nz}, z[v_] := Complement[Range[Length[v]], Flatten@Position[v, 0]];
          nz[[{#, # + 1}]] & /@ z[Differences[Sign@l[[(nz = z[l])]]]]]

In[∘]:= zerCrossParts = Part[#, 1] & /@ (zeroCrossings[Part[#, 2] & /@ pupilDerivTimeSeries]);

In[∘]:= pupilDerivTimeVals =
          Table[t, {t, First[pupilTimeSeries][[1]], Last[pupilTimeSeries][[1]], 0.01}];

In[∘]:= zerCrossTimes = Table[pupilDerivTimeVals[[n]], {n, zerCrossParts}];

In[∘]:= dilOnsets = DeleteCases[Table[
            If[bandPassPupilInt[zerCrossTimes[[n]]] < bandPassPupilInt[zerCrossTimes[[n]] + 0.1],
             zerCrossTimes[[n]], Null], {n, 1, Length[zerCrossTimes]}], Null];

In[∘]:= consOnsets = Complement[zerCrossTimes, dilOnsets];

In[∘]:= (***Get dilation-constriction onset pairs***)

In[∘]:= dilConsOnsetPairsAll =
          Table[Sort@{dilOnsets[[n]], Nearest[DeleteCases[consOnsets, x_ /; x ≤ dilOnsets[[n]]],
               dilOnsets[[n]]][[1]]}, {n, 1, Length[dilOnsets]}];

In[∘]:= (***Get constriction-dilation onset pairs***)

In[∘]:= consDilOnsetPairsAll =
          Table[Sort@{consOnsets[[n]], Nearest[DeleteCases[dilOnsets, x_ /; x ≤ consOnsets[[n]]],
               consOnsets[[n]]][[1]]}, {n, 1, Length[consOnsets]}];
```

```
        (***Dilations or constrictions must be larger than 1.5x the
          interquartile range of the bandpass-filtered pupil trace for analysis***)

In[∘]:= pupChangeThresh = InterquartileRange[bandPassPupil] * 1.5;

In[∘]:= (***For each dilation and constriction, calculate the size of the event***)

In[∘]:= dilationSizes = Table[bandPassPupilInt[dilConsOnsetPairsAll[[n, 2]]] -
            bandPassPupilInt[dilConsOnsetPairsAll[[n, 1]]], {n, 1, Length[dilConsOnsetPairsAll]}];

In[∘]:= constrictionSizes = Table[bandPassPupilInt[consDilOnsetPairsAll[[n, 1]]] -
            bandPassPupilInt[consDilOnsetPairsAll[[n, 2]]], {n, 1, Length[consDilOnsetPairsAll]}];

In[∘]:= (***Only keep dilations and constrictions larger than the threshold cut-off***)

In[∘]:= dilConsOnsetPairsLarge = DeleteCases[
          Table[If[dilationSizes[[n]] > pupChangeThresh, dilConsOnsetPairsAll[[n]], Null],
           {n, 1, Length[dilationSizes]}], Null];

In[∘]:= consDilOnsetPairsLarge = DeleteCases[
          Table[If[constrictionSizes[[n]] > pupChangeThresh, consDilOnsetPairsAll[[n]], Null],
           {n, 1, Length[constrictionSizes]}], Null];

In[∘]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
          "/Session", ToString[sessionNum], "/Pupil/", date, "_", mouse, "_", "Session",
          ToString[sessionNum], "_dilationConstrictionTimePairs.txt"], dilConsOnsetPairsLarge];

In[∘]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
          "/Session", ToString[sessionNum], "/Pupil/", date, "_", mouse, "_", "Session",
          ToString[sessionNum], "_constrictionDilationTimePairs.txt"], consDilOnsetPairsLarge];

In[∘]:= (***Further,
        only keep the dilations and constrictions that occur during non-locomotion,
        non-whisking periods***)

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

In[∘]:= (***If a dilation-constriction interval is contained within a quiescent period,
        keep it.  Otherwise, discard it***)

In[∘]:= dilConsOnsetPairsLargeQui = DeleteCases[
          Table[If[Length@(Cases[IntervalMemberQ[#, Interval[dilConsOnsetPairsLarge[[n]]]] & /@
                 (Interval /@ quiescentPeriods), True]) > 0,
             dilConsOnsetPairsLarge[[n]], Null], {n, 1, Length[dilConsOnsetPairsLarge]}], Null];

In[∘]:= (***If a constriction-dilation interval is contained within a quiescent period,
        keep it.  Otherwise, discard it***)

In[∘]:= consDilOnsetPairsLargeQui = DeleteCases[
          Table[If[Length@(Cases[IntervalMemberQ[#, Interval[consDilOnsetPairsLarge[[n]]]] & /@
                 (Interval /@ quiescentPeriods), True]) > 0,
             consDilOnsetPairsLarge[[n]], Null], {n, 1, Length[consDilOnsetPairsLarge]}], Null];
```

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

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

In[•]:= (***Import calcium fluorescence traces***)

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["dFFtimeseries", 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[•]:= Table[Evaluate@ToExpression[StringJoin["dFFts", ToString[n]]] =
            Partition[Riffle[(ToExpression[StringJoin["dFFtimeseries", ToString[n]]])[[All, 1]],
              Standardize[(ToExpression[StringJoin["dFFtimeseries", ToString[n]]])[[All, 2]]]],
             2];, {n, 1, numROIs}];

In[•]:= (***Bandpass filter the dF/F0 traces from 0.1 to 1 Hz***)

In[•]:= Table[Evaluate@ToExpression[StringJoin["dFFbpf", ToString[n]]] =
            fftFilter[Part[#, 2] & /@ ToExpression[StringJoin["dFFts", ToString[n]]],
             tpFrameRate, {0.1, 1}];, {n, 1, numROIs}];

In[•]:= Table[Evaluate@ToExpression[StringJoin["dFFbpfTS", ToString[n]]] =
            Partition[Riffle[Part[#, 1] & /@ ToExpression[StringJoin["dFFts", ToString[n]]],
              ToExpression[StringJoin["dFFbpf", ToString[n]]]], 2];, {n, 1, numROIs}];

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

In[•]:= (***For each dilation-constriction interval for each ROI,
        extract the bandpass-filtered dF/F0 trace in that interval.  Assign phase values -
         Pi to 0 for these intervals***)
```

```
In[•]:= Table[
          Evaluate@ToExpression[StringJoin["dFFvsDilConsPhase", ToString[n]]] = Flatten[Table[
              Partition[Riffle[Table[n, {n, -Pi, 0, N@Pi / (Length[Table[(ToExpression[StringJoin[
                               "dFFbpfInt", ToString[n]]]) [t], {t, dilConsOnsetPairsLargeQui[[i,
                            1]], dilConsOnsetPairsLargeQui[[i, 2]], N@1 / tpFrameRate}]] - 1)}],
                 Table[(ToExpression[StringJoin["dFFbpfInt", ToString[n]]]) [t],
                  {t, dilConsOnsetPairsLargeQui[[i, 1]],
                   dilConsOnsetPairsLargeQui[[i, 2]], N@1 / tpFrameRate}]], 2],
              {i, 1, Length[dilConsOnsetPairsLargeQui]}], 1];, {n, 1, numROIs}];
```

```
In[•]:= (***For each ROI, bin dF/F0 values into 32 bins from -Pi to 0***)
```

```
In[•]:= dilPhaseBins = Partition[Table[n, {n, -Pi, 0, Pi / 32}], 2, 1];
```

```
In[•]:= dilPhaseBinInts = Interval /@ dilPhaseBins;
```

```
In[•]:= Table[Evaluate@ToExpression[StringJoin["meanDFFvsDilConsPhase", ToString[roi]]] =
            Mean /@ GatherBy[
              DeleteCases[Table[If[Length[Flatten[Position[IntervalMemberQ[#, (ToExpression[
                               StringJoin["dFFvsDilConsPhase", ToString[roi]]]) [[n, 1]]] & /@
                         dilPhaseBinInts, True]]] == 0, Null, {Take[Flatten[Position[
                         IntervalMemberQ[#, (ToExpression[StringJoin["dFFvsDilConsPhase",
                                ToString[roi]]]) [[n, 1]]] & /@ dilPhaseBinInts, True]], 1] [[1]],
                    (ToExpression[StringJoin["dFFvsDilConsPhase", ToString[roi]]]) [[n, 2]]}],
                 {n, 1, Length[(ToExpression[StringJoin["dFFvsDilConsPhase", ToString[roi]]])]}],
               Null], First];, {roi, 1, numROIs}];
```

```
In[•]:= (***For each constriction-dilation interval for each ROI,
        extract the bandpass-filtered dF/F0 trace in that
          interval.  Assign phase values 0 to Pi for these intervals***)
```

```
In[•]:= Table[
          Evaluate@ToExpression[StringJoin["dFFvsConsDilPhase", ToString[n]]] = Flatten[Table[
              Partition[Riffle[Table[n, {n, 0, Pi, N@Pi / (Length[Table[(ToExpression[StringJoin[
                               "dFFbpfInt", ToString[n]]]) [t], {t, consDilOnsetPairsLargeQui[[i,
                            1]], consDilOnsetPairsLargeQui[[i, 2]], N@1 / tpFrameRate}]] - 1)}],
                 Table[(ToExpression[StringJoin["dFFbpfInt", ToString[n]]]) [t],
                  {t, consDilOnsetPairsLargeQui[[i, 1]],
                   consDilOnsetPairsLargeQui[[i, 2]], N@1 / tpFrameRate}]], 2],
              {i, 1, Length[consDilOnsetPairsLargeQui]}], 1];, {n, 1, numROIs}];
```

```
In[•]:= (***For each ROI, bin dF/F0 values into 32 bins from 0 to Pi***)
```

```
In[•]:= consPhaseBins = Partition[Table[n, {n, 0, Pi, Pi / 32}], 2, 1];
```

```
In[•]:= consPhaseBinInts = Interval /@ consPhaseBins;
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["meanDFFvsConsDilPhase", ToString[roi]]] =
           Mean /@ GatherBy[
             DeleteCases[Table[If[Length[Flatten[Position[IntervalMemberQ[#, (ToExpression[
                           StringJoin["dFFvsConsDilPhase", ToString[roi]]]) [[n, 1]]] & /@
                       consPhaseBinInts, True]]] == 0, Null, {Take[Flatten[Position[
                       IntervalMemberQ[#, (ToExpression[StringJoin["dFFvsConsDilPhase",
                              ToString[roi]]]) [[n, 1]]] & /@ consPhaseBinInts, True]], 1] [[1]],
                   (ToExpression[StringJoin["dFFvsConsDilPhase", ToString[roi]]]) [[n, 2]]}],
                 {n, 1, Length[(ToExpression[StringJoin["dFFvsConsDilPhase", ToString[roi]]])]}],
               Null], First];, {roi, 1, numROIs}];

In[∘]:= (***Finally, assign the phase values the
         bins to generate the final dF/F0 vs pupil phase data***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFvsPupilPhase", ToString[roi]]] =
           Partition[Riffle[Join[First /@ dilPhaseBins, First /@ consPhaseBins], Join[
                Part[#, 2] & /@ ToExpression[StringJoin["meanDFFvsDilConsPhase", ToString[roi]]],
                Part[#, 2] & /@ ToExpression[StringJoin["meanDFFvsConsDilPhase", ToString[roi]]]]],
              2];, {roi, 1, numROIs}];

        (***Visualize dFF aligned to pupil dilation-constriction phase for each ROI***)

In[∘]:= Manipulate[ListLinePlot[ToExpression[StringJoin["dFFvsPupilPhase", ToString[roi]]]],
         {roi, 1, numROIs, 1}]

In[∘]:= (***Export dFF-pupil phase data***)

In[∘]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
            "/Session", ToString[sessionNum], "/", "Pupil/", date, "_", mouse, "_",
            "Session", ToString[sessionNum], "_dFFvsPupilPhase_ROI", ToString[n], ".txt"],
          ToExpression[StringJoin["dFFvsPupilPhase", ToString[n]]]], {n, 1, numROIs}];

In[∘]:= (*******************************************************************************
         ****************************************)
        (***In this part, calculate cross-correlograms between
          dF/F0 and either pupil diameter or derivative of pupil diameter***)
        (*******************************************************************************
         ****************************************)

In[∘]:= (***Interpolate the 10 Hz sampled pupil time series***)

In[∘]:= pupil10HzInt = Interpolation[pupil10HzTimeSeries];

In[∘]:= (***Lowpass filter the dF/F0 traces at 10 Hz***)

In[∘]:= Table[
          Evaluate@ToExpression[StringJoin["dFFlpfTS", ToString[n]]] = LowpassFilter[TimeSeries@
              ToExpression[StringJoin["dFFts", ToString[n]]], 10 Hz ];, {n, 1, numROIs}];

In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFlpfInt", ToString[n]]] =
            Interpolation[ToExpression[StringJoin["dFFlpfTS", ToString[n]]]];, {n, 1, numROIs}];

In[∘]:= (***Compute cross-correlations between pupil diameter and dF/F0 during quiescent
          periods.  Compute in a time window of 8 seconds with 100 ms resolution***)
```

```
In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFpupilCrossCorr", ToString[roi]]] =
          Mean[DeleteCases[Table[pupil = Table[pupil10HzInt[t],
                 {t, quiescentPeriods[[i, 1]], quiescentPeriods[[i, 2]], 0.1}];
               neuron = Table[(ToExpression[StringJoin["dFFlpfInt", ToString[roi]]])[t],
                 {t, quiescentPeriods[[i, 1]], quiescentPeriods[[i, 2]], 0.1}];
               If[quiescentPeriods[[i, 2]] - quiescentPeriods[[i, 1]] > 8,
                Join[Reverse[Table[Correlation[pupil, RotateRight[neuron, -n]], {n, 0, 80, 1}]],
                 Drop[Table[Correlation[pupil, RotateRight[neuron, n]], {n, 0, 80, 1}], 1]], Null],
               {i, 1, Length[quiescentPeriods]}], Null]];, {roi, 1, numROIs}];

In[∘]:= (***Compute cross-correlations between derivative of pupil diameter and dF/F0 during
          quiescent periods.  Compute in a time window of 8 seconds with 100 ms resolution***)

In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFpupilDerivCrossCorr", ToString[roi]]] =
          Mean[DeleteCases[Table[pupil = Table[pupil10HzInt'[t],
                 {t, quiescentPeriods[[i, 1]], quiescentPeriods[[i, 2]], 0.1}];
               neuron = Table[(ToExpression[StringJoin["dFFlpfInt", ToString[roi]]])[t],
                 {t, quiescentPeriods[[i, 1]], quiescentPeriods[[i, 2]], 0.1}];
               If[quiescentPeriods[[i, 2]] - quiescentPeriods[[i, 1]] > 8,
                Join[Reverse[Table[Correlation[pupil, RotateRight[neuron, -n]], {n, 0, 80, 1}]],
                 Drop[Table[Correlation[pupil, RotateRight[neuron, n]], {n, 0, 80, 1}], 1]], Null],
               {i, 1, Length[quiescentPeriods]}], Null]];, {roi, 1, numROIs}];

In[∘]:= (***For each ROI,
        make a time series of the time-dependent modulation index around dilation onset***)

In[∘]:= crossCorrTimeVals = Table[n, {n, -8, 8, 0.1}];

In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFpupilCrossCorrTS", ToString[n]]] =
          Partition[Riffle[crossCorrTimeVals, (ToExpression[
               StringJoin["dFFpupilCrossCorr", ToString[n]]])], 2];, {n, 1, numROIs}];

In[∘]:= Table[Evaluate@ToExpression[StringJoin["dFFpupilDerivCrossCorrTS", ToString[n]]] =
          Partition[Riffle[crossCorrTimeVals, (ToExpression[
               StringJoin["dFFpupilDerivCrossCorr", ToString[n]]])], 2];, {n, 1, numROIs}];

        (***Visualize pupil-dFF or pupil derivative-dFF cross correlograms***)

In[∘]:= Manipulate[{ListLinePlot[ToExpression[StringJoin["dFFpupilCrossCorrTS", ToString[roi]]]],
          ListLinePlot[ToExpression[StringJoin["dFFpupilDerivCrossCorrTS", ToString[roi]]]]},
         {roi, 1, numROIs, 1}]

In[∘]:= (***Export dFF-pupil cross-correlation data data***)

In[∘]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
          "/Session", ToString[sessionNum], "/", "Pupil/", date, "_", mouse, "_",
          "Session", ToString[sessionNum], "_dFFpupilCrossCorr_ROI", ToString[n], ".txt"],
         ToExpression[StringJoin["dFFpupilCrossCorrTS", ToString[n]]]], {n, 1, numROIs}];

In[∘]:= Table[Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse, "/Session",
          ToString[sessionNum], "/", "Pupil/", date, "_", mouse, "_", "Session",
          ToString[sessionNum], "_dFFpupilDerivCrossCorr_ROI", ToString[n], ".txt"],
         ToExpression[StringJoin["dFFpupilDerivCrossCorrTS", ToString[n]]]], {n, 1, numROIs}];
```

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

        pupilModROIs = Range[numROIs];

In[∘]:= Export[StringJoin["S:/Imaging/Garrett/FMB208_2PRig/", date, "/", mouse,
           "/Session", ToString[sessionNum], "/", "Pupil/", date, "_", mouse, "_",
           "Session", ToString[sessionNum], "_pupilModROIs", ".txt"], pupilModROIs];
```