**Agama Lizards**

**Foot Analysis Description**

**Joshua Levy**

* Application constructs foot kinematics relative to upper and lower body
* Turns the raw relative foot kinematics into a processed image of a vector field, depicting the directions that the foot will likely travel relative to the front/rear segments, superimposed on top of an image of the flux probability distribution, where it is most likely to find the foot over many trials
* Two coordinate systems
  + Absolute Cartesian coordinates- coordinate system that is relative to the ground and is unmoving, the lizard moves atop it
  + Lizard front/rear xb/yb coordinate system- coordinate system constructed from xb and yb vectors
    - yb is a unit vector that either points in the direction of the body to the shoulders (front segment, origin is COM of body and shoulder segments), or pointing from the hips to the body (rear segment, origin is COM of hip and body segments)
    - xb is a unit vector that is a 90 degree CCW rotation of the yb unit vector, found using a transformation matrix T=[0 1; -1 0]
    - both coordinate systems move as the body moves, so the fixed foot positions on the absolute plane will move relative to the xb/yb coordinates

**Steps:**

* Build lizard data for all trials
  + Construct relative foot data, that is, finding coordinates of feet relative to the moving front or rear body segments’ center of mass
  + This assumes that during the time interval between each touchdown and lift off of each foot and stance, that the foot does not move in absolute coordinates, so only the absolute Cartesian coordinates of the foot touchdowns are used in the analysis; some of the feet that slip to much will be removed during the Threshold phase
* Remove all trials for all foot and stances labeled as “Slip” trials from analysis
* Threshold: The foot moves/slips a little while it is in the down on the ground, if this slip is measured to be greater than one standard deviation above the mean slip, then the particular trial is thrown out for a given foot and stance
* Trials that start left of the fore aft axis are flipped to right of the fore aft axis
* Copy the relative foot data structure into a new data structure that will be used for further analysis and will allow each foot stance to be analyzed separately for each foot
* Plot the relative foot positions for each stance (HO1,2,3 HI1,2,3 FO1,2,3 FI1,2,3); this is the raw data being plotted
* Each foot and stance are then analyzed separately, trial per trial for each foot and stance:
  + Trials are first flagged if anything suspicious is suspected, eg. errors in data, contains hooks etc.
  + The criteria for flagging trials: Truncation: if data points of a trial must be cut off in order to remove hook features; Removal: if the data spiked at a certain point, were not flipped properly, exceed the dimensions/domain of all other trials for particular foot and stance
  + Flagged trials are revisited and are removed or truncated accordingly
  + For the truncation, the user is allowed to decide which data point to use as the starting one, all previous data points are removed
* Data/field dimensions for each foot and stance are found, the minimum and maximum xb and yb values are found of all trials for a given foot and stance, this will decide the size and other aspects of the probability distribution and vector field; allows for creation of two xb-yb grids from which to construct the distribution and fields; overall minimum data was rounded down to the nearest integer and overall maximum data was rounded up
* Flux probability distribution (For each foot and stance)
  + Created a mesh of xy values from the field dimensions; horizontal and vertical separation between points was 0.25 cm, more mesh points in order to collect more data and have well-defined/telling contour plots
  + For each xy value in the mesh, found “Flux” values, operationalized as the number of trials that pass through a circle of a certain radius around each xy mesh point
  + The circles overlap; radius chosen was 0.75\*0.5\*sqrt(2) cm, this is 75% of the distance of the diagonal between two diagonal data points; the overlap in circles smooths the data
  + The X, Y, and flux data are plotted as a contour plot for each foot and stance
* Vector field (For each foot and stance)
  + Created a different mesh of xy values with larger spacing values (0.5 cm); spacing larger in order to create more spacing between vectors, too many vectors in vector field may not be very telling of general flow trends
  + Created squares of side length 0.5 that are centered at each of the meshpoints (the squares do not overlap, but touch each other)
  + Within each square (for a given xy meshpoint), for a given trial, added up all of the differential path vectors that pass through square and averaged them out as an average slope vector for the trial for the given xy meshpoint; recorded number of datapoints that pass inside of square as a weight for the weighted average; NOTE: This is the same as taking the vector difference between the end and start points of the path of lizard inside each square but takes into account that perhaps the lizards foot left the square for a few datapoints and then returned to the square
  + Found the weighted averages of all of the trials’ average slope vectors for a particular meshpoint; summed these vectors up and normalized in order to find the mean weighted slope vector for a given square/meshpoints
  + If less than three trials passed through a square, record no weighted slope vector
  + Plotted the weighted slope vectors over the given meshpoints; these images are superimposed on top of the contour diagrams of the flux probability distribution

**Functions/Scripts**

1. **FootAnalysis** 
   1. Parent function of all other functions
   2. inputs the lizard build B structure and optionally the FootAnalyze structure which contains all of the data from a prior or in progress foot analysis
   3. Outputs a completed or saved FootAnalyze structure
   4. If run all the way through, should save and display 24 figures
      1. 12 of the figures are raw data plots of the relative foot data for each foot (four different feet) and each stance (three total stances)
      2. 12 of the figures are vector fields of the direction of the direction where each foot tends to travel superimposed on top of a contour plot of the flux probability distribution of where each foot tends to be, all relative to the front/rear body segment center of masses using the new coordinate system
   5. Modulated in order of particular operations/processes
   6. Order of Operations
      1. Checks if there exists an existing FootAnalyze data structure, if there is, it skips all subsequent processes until the truncation/removal process through the use of “trialsAddressed”
2. Initializes part of the FootAnalyze structure that will be used after truncation/removal process; **Catstruct** aids in this initialization and is borrowed from an outside source
   * 1. If FootAnalyze data structure does not exist, then go through following process:
        1. Build relative kinematic foot data (relfootdat) using **RelativeFootData** and **RelativePosition** functions
        2. Cut all slip trials from the analysis
        3. Store the B structure in the raw part of the FootAnalyze data structure
        4. Store stance positions for each foot
        5. Cut particular feet and stances that fail the slip threshold limits using **StanceSlipDist**
        6. Save remaining relfootdat structure as raw part in FootAnalyze
        7. Restructure relfootdat data and flip the data about the fore-aft axis such that each foot and stance will be analyzed on the right side of the body using **raw2processed**
        8. Plot this “raw data” using **plotFootData**; these are the first 12 figures produced
        9. Initialize variables and structures to be used during the truncation/removal process
     2. After those processes are complete, Truncation/removal process begins using **errorCutoff** function for each foot and stance
        1. After this process is done for a particular footstance, the **saveProgess** script is employed to allow you to save progress on the analysis so that you can return to it at a later time by inputting the partially complete FootAnalyze structure at the beginning of the **FootAnalysis** program
     3. FootAnalyze structure is saved following the truncation/removal process
     4. Vector fields (created with help from **avgSlopeSquare**) and probability flux distributions are plotted using **probFluxAndVectorFields** function; the remaining 12 figures are plotted
     5. FootAnalyze is saved once more
     6. End analysis
3. **Catstruct**
   1. Concatenates two structures together
   2. In this case, the existing 4 structures for the feet inside of the processed part of the FootAnalyze structure had similar particular foot structures added on to them
   3. These added foot structures contained initialized data useful for the flux/vector analysis
4. **RelativeFootData**
   1. Inputs the Cartesian foot kinematic data and uses the data to build the foot kinematic data with respect to front and rear body segment’s center of mass using a new coordinate system constructed from these segments using the relativePosition function for all stances in a given foot and trial; output is data structure containing this relative data
   2. Done for each foot (four feet, different numbers of foot stances, cut to three or less in the final analysis) for a given trial and then iterated over all trials
5. **relativePosition**
   1. Constructs relative foot kinematic data for all stances a given foot and trial
   2. Process
      1. Inputs B, the number in the B.feet structure corresponding to a particular foot, and the segment name (“front or rear”)
      2. Grabs frame intervals when the foot is between touch down and lift off, the number of intervals indicates the number of foot stances
      3. For each frame in each stance, a new basis of coordinates is created
         1. yb is a unit vector that either points in the direction of the body to the shoulders (front segment, origin is COM of body and shoulder segments), or pointing from the hips to the body (rear segment, origin is COM of hip and body segments)
         2. xb is a unit vector that is a 90 degree CCW rotation of the yb unit vector, found using a transformation matrix T=[0 1; -1 0]
         3. both coordinate systems move as the body moves, so the fixed foot positions on the absolute plane will move relative to the xb/yb coordinates
      4. The absolute Cartesian foot positions over the stance interval are dotted into the xb and yb coordinates to construct the foot data
      5. Data is converted into centimeters
      6. Output is relative kinematic data, done for all stances for a particular foot and trial, and the Cartesian COM positions of rear and front body segments and Cartesian foot positions
6. **StanceSlipDist**
   1. Takes Cartesian foot positions for each trial; foot positions are the Cartesian coordinates used for each touchdown and lift off
   2. Finds the distance between touchdown and lift offs for particular foot stances
   3. If the distance for a particular trial is one standard deviation above the mean slip of all trials for the particular footstance, then throw out trial for particular footstance
7. **raw2processed**
   1. takes relative foot data in raw structure inside FootAnalyze structure and moves it into the processed structure inside FootAnalyze for further analysis
      1. This processed relative foot data is “restructured” as it is moved into the processed part of FootAnalyze
      2. Please take a look around line 18 in order to see how restructuring is done
      3. Each foot is now a standalone structure containing a cell that contains three cells (one for each stance)
      4. Each of those stance cells may have a variable amount of trials in it, depending on what has been previously cut in **errorCutoff** or the truncation/removal process that is done subsequently to this program
   2. For each foot and stance, relative foot data for particular trials are flipped across the fore-aft axis if over 50% of their data points lie on the left side of that axis; this mostly ensures that the data being analyzed assumes that each foot and stance is to the right of the fore-aft axis in the analysis
   3. Output is the processed data structure
8. **plotFootData**
   1. All trials for each foot and stance from the processed relative foot data are plotted before the truncation/removal process
   2. Figures are saved, 12 figures total
9. **errorCutoff**
   1. Truncation/Removal process
   2. Trials are first flagged if anything suspicious is suspected, eg. errors in data, contains hooks etc.
   3. The criteria for flagging trials: Truncation: if data points of a trial must be cut off in order to remove hook features; Removal: if the data spiked at a certain point, were not flipped properly, exceed the dimensions/domain of all other trials for particular foot and stance
   4. Flagged trials are revisited and are removed or truncated accordingly
   5. For the truncation, the user is allowed to decide which data point to use as the starting one, all previous data points are removed
10. **saveProgress**
    1. Saves FootAnalyze structure between errorCutoff usage for foot and stance; can choose to not save or save or save and quit the program and return to it later
11. **probFluxAndVectorFields**
    1. Data/field dimensions for each foot and stance are found, the minimum and maximum xb and yb values are found of all trials for a given foot and stance, this will decide the size and other aspects of the probability distribution and vector field; allows for creation of two xb-yb grids from which to construct the distribution and fields; overall minimum data was rounded down to the nearest integer and overall maximum data was rounded up
    2. Flux probability distribution (For each foot and stance)
       1. Created a mesh of xy values from the field dimensions; horizontal and vertical separation between points was 0.25 cm, more mesh points in order to collect more data and have well-defined/telling contour plots
       2. For each xy value in the mesh, found “Flux” values, operationalized as the number of trials that pass through a circle of a certain radius around each xy mesh point
       3. The circles overlap; radius chosen was 0.75\*0.5\*sqrt(2) cm, this is 75% of the distance of the diagonal between two diagonal data points; the overlap in circles smooths the data
       4. The X, Y, and flux data are plotted as a contour plot for each foot and stance
    3. Vector field (For each foot and stance)
       1. Created a different mesh of xy values with larger spacing values (0.5 cm); spacing larger in order to create more spacing between vectors, too many vectors in vector field may not be very telling of general flow trends
       2. Created squares of side length 0.5 that are centered at each of the meshpoints (the squares do not overlap, but touch each other)
       3. Within each square (for a given xy meshpoint), for a given trial, added up all of the differential path vectors that pass through square and averaged them out as an average slope vector for the trial for the given xy meshpoint; recorded number of datapoints that pass inside of square as a weight for the weighted average; NOTE: This is the same as taking the vector difference between the end and start points of the path of lizard inside each square but takes into account that perhaps the lizards foot left the square for a few datapoints and then returned to the square; this process is essentially **avgSlopeSquare**
       4. Found the weighted averages of all of the trials’ average slope vectors for a particular meshpoint; summed these vectors up and normalized in order to find the mean weighted slope vector for a given square/meshpoints
       5. If less than three trials passed through a square, record no weighted slope vector
       6. Plotted the weighted slope vectors over the given meshpoints; these images are superimposed on top of the contour diagrams of the flux probability distribution
    4. 12 processed figures are produced
12. **avgSlopeSquare** (used inside **probFluxAndVectorFields** for vector field)
    1. Within each square (for a given xy meshpoint), for a given trial, added up all of the differential path vectors that pass through square
    2. Averaged them out as an average slope vector for the trial for the given xy meshpoint
    3. Recorded number of datapoints that pass inside of square as a weight for the weighted average
    4. NOTE: This is the same as taking the vector difference between the end and start points of the path of lizard inside each square but takes into account that perhaps the lizards foot left the square for a few datapoints and then returned to the square