



DEC 01, 2022

WORKS FOR ME

1

Shape and color analysis at wing-size and wing-pattern levels

DOI

dx.doi.org/10.17504/protocols.io.81wgb783qvpk/v1Wei-Ping Chan^{1,2}¹Department of Organismic and Evolutionary Biology, Harvard University;²Museum of Comparative Zoology, Harvard University

Wei-Ping Chan

COMMENTS 0

ABSTRACT

Details could be found in the Supplementary Information of "[A high-throughput multispectral imaging system for museum specimens](#)"

DOI

dx.doi.org/10.17504/protocols.io.81wgb783qvpk/v1

PROTOCOL CITATION

Wei-Ping Chan 2022. Shape and color analysis at wing-size and wing-pattern levels. **protocols.io**
<https://dx.doi.org/10.17504/protocols.io.81wgb783qvpk/v1>

LICENSE

This is an open access protocol distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

CREATED

Aug 15, 2021

LAST MODIFIED

Dec 01, 2022

PROTOCOL INTEGER ID

52357

- 1 Inspect all specimens before running the dorsal-ventral analysis
- 1.1 Download all images in **Shape_analysis/wing_segmentation** to your local disk. Make sure all segmented images are in a local folder. The images should all contain the suffix: 'wing_segmentation.jpg'
- 1.2 Sometimes a specimen may have an incomplete set of images (i.e. is missing the dorsal or ventral side result). We can use the following script to detect those incomplete specimens, and further operations (e.g. manual redo or checking the source) can be launched to regenerate missing files.

Here, we provide an example script written in Mathematica.

Note

```

refdir = "Path\\to\\wing_segmentation\\directory";
(*Check the inspection images*)
imglist = FileNames["*.jpg", refdir];
imgnamelist =
  Table[StringSplit[FileNameTake[imglist[[k]],
    "_wing_segmentation.jpg"][[1]], {k, 1, Length[imglist]}];
barcodelist =
  Table[StringSplit[imgnamelist[[k]], "_"][[1]], {k, 1,
    Length[imgnamelist]}];
singleSideMat = Select[Tally[barcodelist], #[[2]] != 2 &];
singleSideList =
  Table[imgnamelist[[

```

Citation: Wei-Ping Chan Shape and color analysis at wing-size and wing-pattern levels

<https://dx.doi.org/10.17504/protocols.io.81wgb783qvpk/v1>This is an open access protocol distributed under the terms of the **Creative Commons Attribution License** (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium,

```
Position[barcodelist, singleSideMat[[k, 1]]][[1, 1]]], {k, 1,
Length[singleSideMat]}];
MatrixForm[Transpose[{singleSideMat[[All, 1]], singleSideList}]]
```

Note

```
matdir = "Path\\to\\wing_shape_matrices-seg\\directory";
(*Check the morph-seg matrices*)
imglist2 = FileNames["*.mat", matdir];
imgnamelist2 =
Table[StringSplit[FileNameTake[imglist2[[k]],
 "_morph-seg.mat"]][[-1]], {k, 1, Length[imglist2]}];
barcodelist2 =
Table[StringSplit[imgnamelist2[[k]], "_"]][[1]], {k, 1,
Length[imgnamelist2]}];
singleSideMat2 = Select[Tally[barcodelist2], #[[2]] != 2 &];
singleSideList2 =
Table[imgnamelist2[[
Position[barcodelist2, singleSideMat2[[k, 1]]][[1, 1]]], {k, 1,
Length[singleSideMat2]}];
MatrixForm[Transpose[{singleSideMat2[[All, 1]], singleSideList2}]]
```

2 General color pattern and shape analysis based on universal wing grid after dorsal-ventral mapping

- 2.1 Identify a folder of morph-seg files on the cluster. Usually, these will reside in a folder ***/Shape_analysis/wing_shape_matrices-seg** and the files within will end with the suffix: 'morph-seg.mat'
- 2.2 Download MatLab scripts from [the GitHub repository](#) to a **MATLAB_CODE_LIBRARY**. Run the following command on the cluster to generate wing grids

Command

batch_dorsal_ventral_mapping

```
sbatch batch_dorsal_ventral_mapping.sh MORPH-SEG_DIR SPP_MATRICES_DIR/ MANUAL_GRID_DIR MATLAB_CODE_LIBRARY RESULT_DIR
GRID_RESOLUTION
```

MORPH-SEG_DIR: Directory described above, containing morph-seg.mat files for all specimens in the dataset

SPP_MATRICES_DIR: Directory containing MatLab specimen boundary parameters for every specimen imaged. Located and created in the RESULT_DIRECTORY during all_in_one_image_analysis (see **3.5**) [note: the directory name *must* be followed by a '/' in order for the files to be correctly sourced from the folder

MANUAL_GRID_DIR: A folder contains manually defined grid files. It can be set to a random directory during the first run (i.e. no manually-defined grids)

MATLAB_CODE_LIBRARY

RESULT_DIR

GRID_RESOLUTION: default value is 5, specifying that $2^5 \times 2^5$ grids be mapped onto each piece of wing. All scripts were developed under this resolution with the flexibility of using higher or lower values.

- 3 Body part quality inspection (This can be done parallel with Step 2).
This allows one to tailor the dataset to exclude unwanted segments of specimens in downstream analyses (i.e., broken wings, missing antennae, segmentation anomalies)
- 3.1 Run a script like the example Mathematica script provided below to generate a list of all specimens for inspection. If the user decides to generate the table manually or by other methods, please do include the following column names to prevent potential problems in the future processing steps: "barcode", "side", "template", "LF_wing_d", "RF_wing_d", "LH_wing_d", "RH_wing_d", "Body", "L_antennae_d", "R_antennae_d".

Concatenating "barcode" and "side" with '_' can form "template". For example 'MCZ-ENT000000001' (barcode) and 'dorsal' (side) can be joined to form 'MCZ-ENT000000001_dorsal' (template).

Note

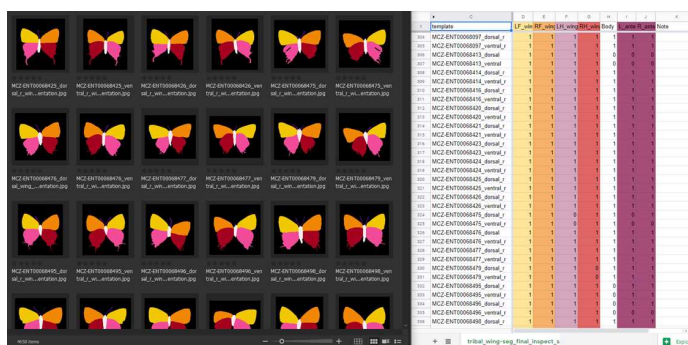
```
refdir = "Path\\to\\wing_segmentation\\directory";
outdir = "Path\\to\\output\\directory";
(*Generate a list for wing part checking*)
label = ("LF_wing_d", "RF_wing_d", "LH_wing_d", "RH_wing_d", "Body",
"L_antennae_d",
"R_antennae_d");(*_d: the L/R parts are defined by looking from the \
dorsal side, it's definite*)
imglist = FileNames["*.jpg", refdir];
imgnamelist =
Table[StringSplit[FileNameTake[imglist[[k]],
"_wing_segmentation.jpg"][[1]], {k, 1, Length[imglist]}];
barcodelist =
Table[StringSplit[imgnamelist[[k]], "_"][[1]], {k, 1,
Length[imgnamelist]}];
sidelist =
Table[StringSplit[imgnamelist[[k]], "_"][[2]], {k, 1,
Length[imgnamelist]}];
emptyList =
Table[Join[{barcodelist[[k]], sidelist[[k]], imgnamelist[[k]],
ConstantArray[1, Length[label]]}, {k, 1, Length[imgnamelist]}];
outEmptyList =
Join[{Flatten[{"barcode", "side", "template", label}], emptyList};
outname = FileNameTake[outDir] <> "_spp_preference_table.csv";
out = outDir <> "\\ " <> outname;
Export[out, outEmptyList];
```

- 3.2 Upload the empty preference table to Google Drive and color each column as in the example below for your convenience. The colors below correspond to the colors of each segmented body-part in specimen images.

	C	D	E	F	G	H	I	J	K
1	template	LF_wing	RF_wing	LH_wing	RH_wing	Body	L_ante	R_ante	Note
304	MCZ-ENT00068097_dorsal_r	1	1	1	1	1	1	1	
305	MCZ-ENT00068097_ventral_r	1	1	1	1	1	1	1	
306	MCZ-ENT00068413_dorsal	1	1	1	1	0	0	0	
307	MCZ-ENT00068413_ventral	1	1	1	1	0	0	0	
308	MCZ-ENT00068414_dorsal_r	1	1	1	1	1	1	1	
309	MCZ-ENT00068414_ventral_r	1	1	1	1	1	1	1	
310	MCZ-ENT00068416_dorsal_r	1	1	1	1	1	1	1	
311	MCZ-ENT00068416_ventral_r	1	1	1	1	1	1	1	
312	MCZ-ENT00068420_dorsal_r	1	1	1	1	1	1	1	
313	MCZ-ENT00068420_ventral_r	1	1	1	1	1	1	1	
314	MCZ-ENT00068421_dorsal_r	1	1	1	1	1	1	1	
315	MCZ-ENT00068421_ventral_r	1	1	1	1	1	1	1	
316	MCZ-ENT00068423_dorsal_r	1	1	1	1	1	1	1	
317	MCZ-ENT00068423_ventral_r	1	1	1	1	1	1	1	
318	MCZ-ENT00068424_dorsal_r	1	1	1	1	1	1	1	
319	MCZ-ENT00068424_ventral_r	1	1	1	1	1	1	1	
320	MCZ-ENT00068425_dorsal_r	1	1	1	1	1	1	1	
321	MCZ-ENT00068425_ventral_r	1	1	1	1	1	1	1	
322	MCZ-ENT00068426_dorsal_r	1	1	1	1	1	1	1	
323	MCZ-ENT00068426_ventral_r	1	1	1	1	1	1	1	
324	MCZ-ENT00068475_dorsal_r	1	1	0	1	1	0	1	
325	MCZ-ENT00068475_ventral_r	1	1	0	1	1	0	0	
326	MCZ-ENT00068476_dorsal	1	1	1	1	1	1	1	
327	MCZ-ENT00068476_ventral_r	1	1	1	1	1	1	1	
328	MCZ-ENT00068477_dorsal_r	1	1	1	1	1	1	1	
329	MCZ-ENT00068477_ventral_r	1	1	1	1	1	1	1	
330	MCZ-ENT00068479_dorsal_r	1	1	1	0	1	1	1	
331	MCZ-ENT00068479_ventral_r	1	1	1	0	1	1	1	
332	MCZ-ENT00068495_dorsal_r	1	1	1	1	0	1	1	
333	MCZ-ENT00068495_ventral_r	1	1	1	1	0	1	1	
334	MCZ-ENT00068496_dorsal_r	1	1	1	1	0	1	1	
335	MCZ-ENT00068496_ventral_r	1	1	1	1	0	0	0	
336	MCZ-ENT00068498_dorsal_r	1	1	1	1	1	1	1	

The default preference table has all cells filled with 1.

- 3.3 Open the folder of segmented images and the Google Drive preference table side by side for inspection. Change the value in a cell to 0 if the corresponding body part is not ideal (biased) for further analysis, e.g., a wing is significantly damaged, an antenna is broken, etc. The terms "left" and "right" (i.e. left fore/hindwing, right fore/hindwing) as used in the table are all with respect to the dorsal image; 'right' wing pieces will appear on the left side of ventral segmentation images and 'left' wing pieces on the ventral right side. Thus, it may be easier to rely on the color codes in spreadsheet's columns to remind yourself which wing piece is 'left' or 'right.'



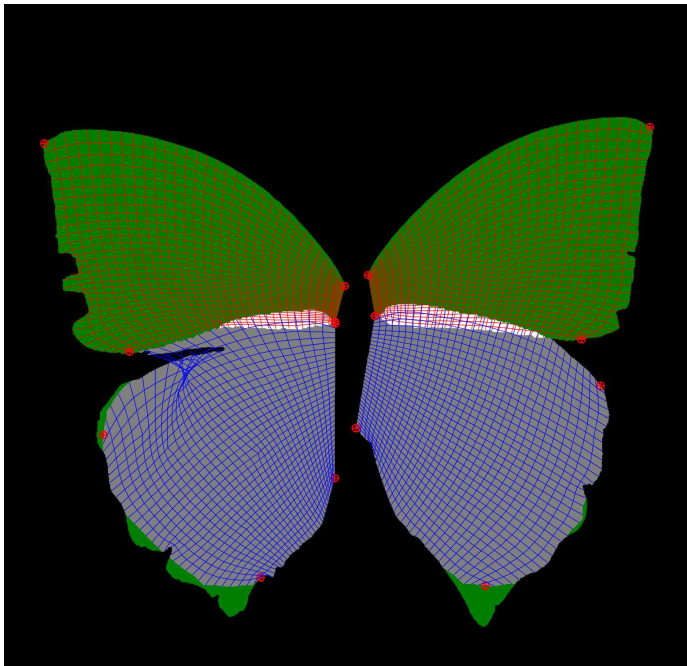
Screenshot showing a divided desktop workspace with two side-by-side windows: one with segmented images in Adobe bridge (left) and another with the corresponding preference table in Google spreadsheet format (right).

4 Wing grid inspection

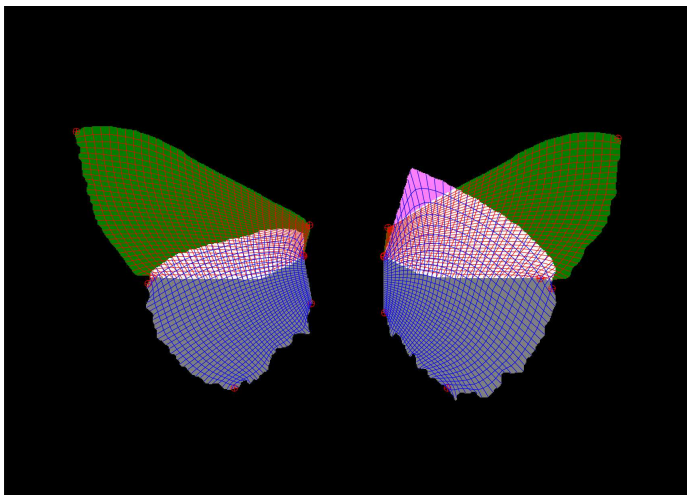
- 4.1 After Step 2 is done, download the images from **inspect_imgs** to the local disk for inspection.

4.2 Use Adobe Bridge to do the inspection.

If there are obviously incorrect wing segmentations (see the image attached), they should be mainly due to flawed data in .json files. Thus, rerun every step from **Step 6.2 in the Primary Processing Protocol** for those images as needed.

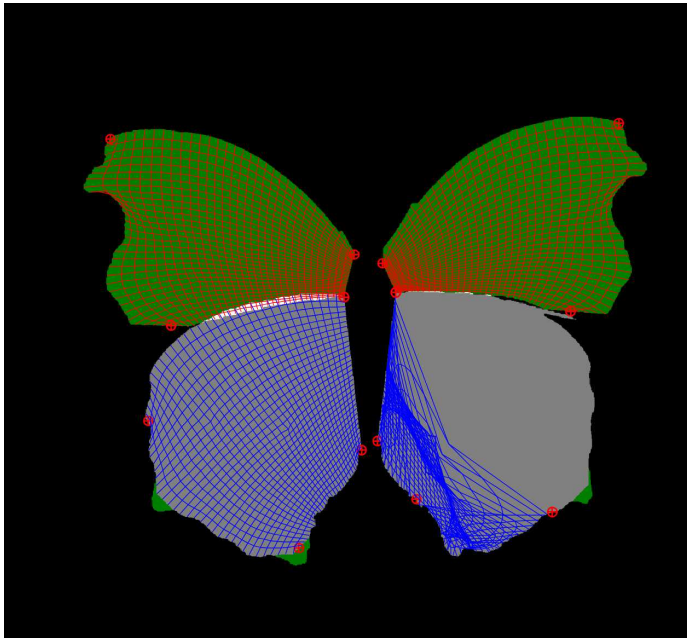


Example 1: incorrect wing segmentation. Notice how the left-side forewing is artificially drawn too large, overlapping the left hindwing and the joint where they intersect, and thus results in grid-drawing issues in that region.

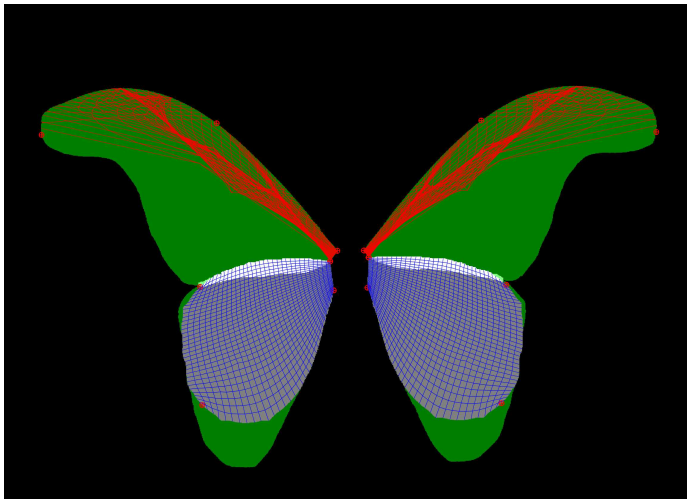


Example 2: incorrect right hindwing. This is induced by the incorrect fore-hindwing segmenting line at the ventral side of the specimen, so a new segmented line in .json format is needed to make it correct.

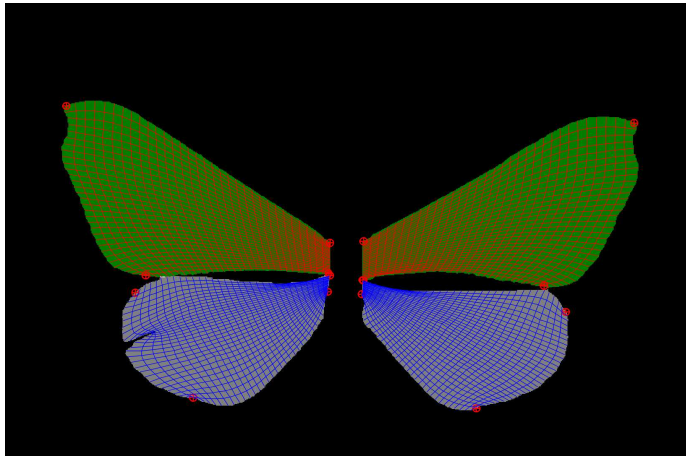
If there are some images showing correct segmentation but unideal wing grids (see the images attached), put those problematic ones into a subfolder named "manual". This folder will be referred as **manual_grid_redo_directory** in the following steps.



Example 3: incorrect wing grid for the right hindwing



Example 4: incorrect wing grids on forewings. This is due to the curved forewing in jumbo specimens, and it can be fixed in the manual correction part.




Example 5: incorrect overlapping area reconstruction. Generally, this is caused by incorrect specimen placement during imaging (i.e. the ventral side faced up in the dorsal image; dorsal side faced up in the ventral image). Though it could be solved by systematically changing the file names (switch "dorsal" and "ventral") from the upstream to downstream, this approach is not recommended since human interaction easily messes up the systematic file names and causes even bigger problems.

4.3 Make sure all morph-seg files are in a folder on your local disk. This folder will be referred to as **morph_mat_directory** in the following steps.

Check, also, that all .json files are also stored in a folder on your local disk. If segmentation corrections were necessary in the previous step, please replace/overwrite the original erroneous segmentation files with the corrected ones.

4.4 Download the required script here ([manual_wing_grid_correction](#)), and check if all necessary MatLab toolboxes are installed properly according to the list below:

 required_toolboxes.txt

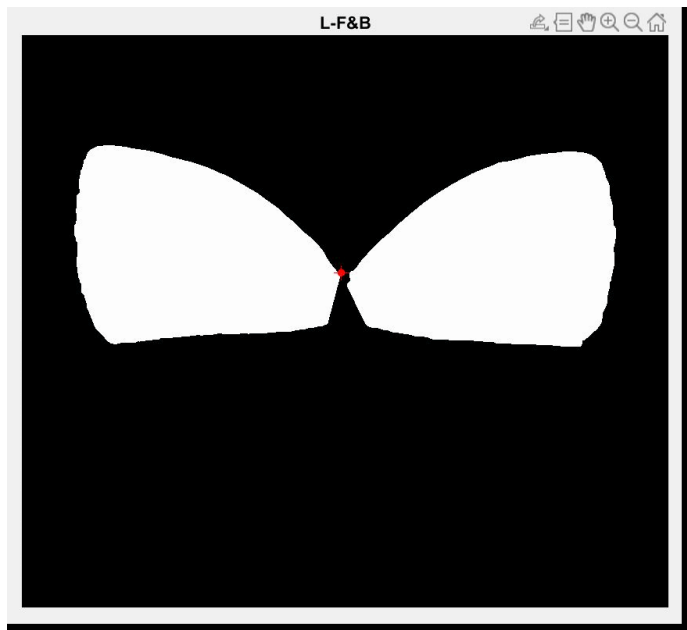
4.5 Run the MatLab script 'manually_define_grid.m', specify paths and parameters in the heading, and follow the tutorial in the following step to manually correct problematic grids in **manual_grid_redo_directory**.

4.6 Tutorial for manual wing grid correction

Using the cursor, click and drag misidentified landmarks (red/green points) to the desired location.

Plot order moves in a clockwise manner around the specimen image, identifying morphometric landmarks in the following order:

1. Left corner dividing Fore & Hing Wings
2. Left corner dividing Forewing & Body
3. Right corner dividing Forewing & Body
4. Right corner dividing Fore & Hing Wings
5. Right corner dividing Hindwing & Body
6. Left corner dividing Hindwing & Body
7. Left forewing tip
8. Right forewing tip



Order in which landmarks appear for correction: begins in forewings and moves to hindwings; cycles through 4 joint landmarks between the wing and the body (red dots) as well as 4 wing "corner" landmarks (green dots).

Press ENTER to confirm using the point shown on the screen. This will also move the correction process forward to the next landmark.

Press 'r' to redo the manual process

Press 'f' to force-exit the entire manual process

Note: once ENTER is pressed, the point will move to the nearest detected white/black edge boundary; for precise and rapid point placement, merely re-position landmark points just close enough to the desired location to enable the program to automatically detect the correct and nearest edge at which to place the landmark.

- 4.7 Inspect and upload the manually defined grids to the cluster. These will have been output to the folder **spp_manual_grid_parameters** generated within the **manual_wing_grid_correction** code directory

- 4.8 Relaunch the dorsal-ventral mapping script (as in 2.2), this time making sure to specify the MANUAL_GRID_DIR variable with the directory containing manually defined grids (**spp_manual_grid_parameters**).

Command

batch_dorsal_ventral_mapping

```
sbatch batch_dorsal_ventral_mapping.sh MORPH-SEG_DIR SPP_MATRICES_DIR/ MANUAL_GRID_DIR MATLAB_CODE_LIBRARY RESULT_DIR
GRID_RESOLUTION
```

MORPH-SEG_DIR: Directory containing morph-seg.mat files for all specimens in the dataset

SPP_MATRICES_DIR: Directory containing MATLAB specimen boundary parameters for every specimen imaged. Located and created in the RESULT_DIRECTORY during all_in_one_image_analysis (see 3.5) [note: the directory name *must* be followed by a '/' in order for the files to be correctly sourced from the folder

MANUAL_GRID_DIR: A folder that contains manually defined grid files, in the automatically generated folder **spp_manual_grid_parameters** generated as a result of the manual_grid_correction process. It was set to a random directory during the first run (i.e. no manually-defined grids)

MATLAB_CODE_LIBRARY


RESULT_DIR (same result directory as in step 2.2 - previously problematic grid files will be overwritten)

GRID_RESOLUTION: default value is 5, turning into 2^5 * 2^5 grids on each piece of wing. All scripts were developed under this resolution with the flexibility of using higher or lower values.

- 5 Download group summary materials to local machine

5.1 Download the final results in **spp_wing_parameter** to the local disk.

5.2 Download the required [dorsal-ventral_summary_script](#), and check if all necessary MatLab toolboxes are installed properly according to the following list:

 required_toolboxes.txt

6 Prepare the list of groups by which to summarize the data.

6.1 Create a dataset indicates different groups and members. The format should read as follows, with specimen IDs in the first 'barcode' column. Specimens may belong to multiple groups, but individual columns may not specify more than one group at a time.

A	B	C	D	E
barcode	Group_1	Group_2	Group_3	Group_4
MCZ-ENT00113098			Heliconius	All
MCZ-ENT00116621			Heliconius	All
MCZ-ENT00068296	Lycaenidae			All
MCZ-ENT00069063	Lycaenidae			All
MCZ-ENT00086787	Lycaenidae			All
MCZ-ENT00086886	Lycaenidae			All
MCZ-ENT00103895		NymPap		All
MCZ-ENT00134839		NymPap		All
MCZ-ENT00173775		NymPap		All
MCZ-ENT00183049		NymPap		All
MCZ-ENT00185721	Lycaenidae			All
MCZ-ENT00185723	Lycaenidae			All
MCZ-ENT00193718	Lycaenidae			All
MCZ-ENT00212762		NymPap		All
MCZ-ENT00733306				All
MCZ-ENT00753857				All
MGCL-295034				All

Each column indicates a group, and the cells of those group members are filled with the same group name.

6.2 Convert the .csv file to .json format

An exemplar script written in Mathematica is provided. Users can also manually prepare the .json format document without relying on the script.

Note

```
indir = "...";
infinalist = "specimen_groups.csv";
(*Identify the import file*)
(*-----\
*)
raw = Drop[Import[FileNameJoin[{indir, infinalist}]], 1];
nGroup = Length[raw[[1]]] - 1;
(*create a sublist based on Group*)
groupBarcodes = ConstantArray[0, nGroup];
For[groupID = 1, groupID <= nGroup, groupID++, {
  groupName =
    Select[raw[[All, groupID + 1]], StringLength[#] > 0 &][[1]];
  groupBarcodes[[groupID]] = groupName;
}
```

```

Join[{groupName},
  Select[raw, #[[groupId + 1]] == groupName &][[All, 1]]];
groupBarcodes =
  ReplacePart[groupBarcodes, groupId -> groupBarcodes0];
}];
groupBarcodeList = Select[groupBarcodes, Length[#] > 1 &];

(*Convert the data into json format*)
allGroupBarcodesJson =
  Table[groupBarcodeList[[k, 1]] ->
    groupBarcodeList[[k, 2 ;; -1]], {k, 1, Length[groupBarcodeList]};
outGroupListName =
  FileNameJoin[{indir, outGroupListName}, allGroupBarcodesJson];
Export[FileNameJoin[{indir, outGroupListName}, allGroupBarcodesJson];

```

The .json format should be as follows:

Note

```

{
  "Lycaenidae":[
    "MCZ-ENT00068296",
    "MCZ-ENT00069063",
    "MCZ-ENT00086787",
    "MCZ-ENT00086886",
    "MCZ-ENT00185721",
    "MCZ-ENT00185723",
    "MCZ-ENT00193718"
  ],
  "NymPap":[
    "MCZ-ENT00103895",
    "MCZ-ENT00134839",
    "MCZ-ENT00173775",
    "MCZ-ENT00183049",
    "MCZ-ENT00212762"
  ],
  "Heliconius":[
    "MCZ-ENT00113098",
    "MCZ-ENT00116621"
  ],
  "All":[
    "MCZ-ENT00068296",
    "MCZ-ENT00069063",
    "MCZ-ENT00086787",
    "MCZ-ENT00086886",
    "MCZ-ENT00103895",
    "MCZ-ENT00113098",
    "MCZ-ENT00116621",
    "MCZ-ENT00134839",
    "MCZ-ENT00173775",
    "MCZ-ENT00183049",
    "MCZ-ENT00185721",
    "MCZ-ENT00185723",
    "MCZ-ENT00193718",
    "MCZ-ENT00212762",
    "MCZ-ENT00733306",
    "MCZ-ENT00753857",
    "MGCL-295034"
  ]
}

```

- 7.1 Knowing the locations of (1) the preference table that was generated in step 3 and (2) the group_list.json generated in step 6.2, specify all required information (i.e. paths, parameters) in the header of the script "specimen_group_summary.m" in MatLab.
- 7.2 Run the script "specimen_group_summary.m". It is possible to generate statistical summaries for all groups listed in the 'group_list.json' or to target groups one-by-one for summarization based either on group number (e.g. "group_4") or group name (e.g. "Lycaenidae").

8 Multispectral reflectance at the wing-size level

- 8.1 Make sure all specimen matrices (*AllBandsMask.mat) are complete and error-free in their own directory.
- Make sure all corresponding segmentation matrices (*morph-seg.mat) are complete and error-free in their own directory.

- 8.2 Download MatLab scripts from [the GitHub repository](#) to a **MATLAB_CODE_LIBRARY**. Calculate multispectral reflectance at the wing-size level.

Command

Summarize the reflectance of multispectral wavelength bands

```
sbatch batch_BW_bands_analysis.sh SPP_MATRICES_DIR MORPH-SEG_DIR MATLAB_CODE_LIBRARY RESULT_DIR REF_THRESHOLD
```

SPP_MATRICES_DIR: Directory containing all finalized/manually checked and corrected specimen image matrices

MORPH-SEG_DIR: Directory containing all morph-seg.mat matrices, which describe detailed segmentation information

MATLAB_CODE_LIBRARY: Directory containing MatLab program files

RESULT_DIR: Directory in which the output tables (*.csv) should be placed

REF_THRESHOLD: The threshold used to identify signaling features. Default value is 0.1, which means a feature having reflectance over 0.1 will be measured, and statistical summaries such as the mean reflectance and the size of the reflective area will be provided.

N.B.: it is highly recommended that the user run this script/analysis multiple times, each time specifying a different reflectance threshold. The reason for this is that different wavelength bands, by default, tend to have very low or very high signal. (UV reflectance, for instance, is usually very low. Setting a reflectance threshold that is too high will wipe all apparent UV signal from the data. On the other hand, 940nm band reflectance is normally very high. A higher reflectance threshold will be ideal to capture the variability in this data which would otherwise be 'oversaturated' at a lower value of ref_threshold) Running multispectral analysis with multiple thresholds will allow the user to pinpoint which sets of data/threshold parameters to use for each wavelength band (i.e., the ones that yield the highest signal-to-noise ratio). Sets of thresholds customarily run are {0.01, 0.02, 0.05, 0.1, 0.2, 0.5}.

- 8.3 Result tables will contain the name of a wavelength band and the reflectance threshold specified. For example, 'done_31-10-21_15-49-28_UV_refThreshold-0.1_summary.csv' indicates the date, UV band, and the threshold at 0.1.

The schema of table is provided as follows.

- Specimen_Barcode: The barcode of specimens
- Side: Dorsal or ventral side
- body_part: all, lf_wing (left forewing), rf_wing (right forewing), lh_wing (left hindwing), rh_wing (right hindwing), body, antennae
- Area_Mask_cm2: The size of each body part according to the background removed mask
- Area_F_cm2: The size of the reflective region (where reflectance is higher than the threshold) in cm2
- b_F_reflectance_perCm2_mean: The mean reflectance in the reflective region
- b_F_reflectance_perCm2_cv: The coefficient of variation in the reflective region
- Area_F_pct: The size of the reflective region as a percentage of the area_mask

Note that the name of the wavelength band has been included in the variable names. For example, 'Area_F_cm2' represents the reflective area of fluorescence. In many cases, results will be further divided by RGB channel. For example, the following columns represent reflectance data in the white-light band and are segregated by RGB channel:

A	B	C	D	E	F	G
Area_Mask_cm2	Area_white_R_cm2	b_white_R_reflectance_perCm2_me	b_white_R_reflectance_perCm2_c	Area_white_R_pct	Area_white_G_cm2	b_white_G_refl

Example column names for the complement of data generated for the white-light spectral band

A cell with no data is flagged by -9999

9 Replot the visualization with customized settings

- 9.1 Download scripts for [replotting customized visualizations](#) to local machine from the linked GitHub repository. Check that the required toolboxes listed in 'required_toolboxes.txt' are installed.
- 9.2 Knowing the folder that has the summary matrices for replotting, run "replot_group_mean_shapes" with specified parameters.
- 9.3 Try different parameters to find the most suitable one for the group of interest.