5-2022

Dynamical Systems Analysis in Adaptive and Metapopulation Ecology with Applications to Conservation Management

Guenchik Grosklos
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/etd

Part of the Mathematics Commons

Recommended Citation

This Dissertation is brought to you for free and open access by the Graduate Studies at DigitalCommons@USU. It has been accepted for inclusion in All Graduate Theses and Dissertations by an authorized administrator of DigitalCommons@USU. For more information, please contact digitalcommons@usu.edu.
ABSTRACT

Dynamical systems analysis in adaptive and metapopulation ecology with applications to conservation management

by

Guenchik Grosklos, Doctor of Philosophy
Utah State University, 2022

Major Professor: Jia Zhao, Ph.D.
Department: Mathematics and Statistics

An important area of applied ecology is the study of population dynamics and species responses to environmental change using theoretical (e.g., math models) and empirical (e.g., field experiments) methods. When empirical efforts are too costly and outcomes of management actions uncertain, mathematical models are used to describe processes in ecology. This allows different management scenarios to be explored without the need for field work and identifies viable management options to promote objectives of interest. In this dissertation, we use methods from applied math to analyze theoretical models that describe organism behavior and population responses to habitat change.

The ability for a species to adapt to a changing environment has direct implications on its persistence. Adaptation, represented by measurable phenotypic traits, may be driven by plasticity or evolution. Plastic change typically occurs much more quickly than genetic change but both may occur fast enough to impact population densities. We measure and compare rates of change in population densities and phenotypes using fast-slow dynamical systems theory and scaling theory. We categorize our results into phenotypic change driven by plasticity or evolution. We find that phenotypes have the potential to change at similar rates to densities. Moreover, we did not find any systematic differences between systems
with different modes of adaptation. Therefore, the speed of adaptation of a species is not necessarily dependent on the type of adaptation and both modes of adaptation have the potential to change at rates comparable to density rates of change.

The ability for a species to persist relies not only on its potential to adapt but also on its interactions with proximal populations. Spatially separated populations, or patches, make up a metapopulation where interactions between populations occur through some dispersal mechanism. In metapopulation theory, patches are classified as sources and sinks based on their local demography and interactions with other patches. Source populations enable growth in the whole metapopulation whereas sink populations rely on immigration from other patches to avoid extinction. Correlations between patch fluctuations, or synchrony, also play a role in extinction risk for metapopulations, where highly correlated population fluctuations are more at risk of extinction when populations are especially low.

We analyze metapopulation dynamics by investigating key components that encourage persistence, such as population fluctuations, population synchrony, and patch contribution. We first explore the effects of a range of constant dispersal probabilities on oscillating populations and synchrony of a two-patch system with a species that has 1, 2, and 3 life history stages. Oscillations are classified into three attractor types based on their qualitative behavior (invariant cycles, k-cycles, and chaos), and synchrony is defined as the degree to which total populations in each patch fluctuate in the same direction. We also classify dispersal regions into zones that describe attractor types and synchrony patterns based on dispersal symmetry and probability, and use bifurcation diagrams and eigenvalue analysis to identify the bifurcations that occur when equilibria shift to oscillations, and vice-versa. We find that symmetric dispersal generally has less fluctuations and lower synchrony than asymmetric dispersal but high dispersal probabilities regardless of symmetry may induce population fluctuations and rapid decreases in synchrony. Where chaos is generally thought to have lower synchrony, our results show that chaotic dynamics reflect the synchrony of the cycle types that precede chaos. Additionally, we identify three routes to chaos (phase-locking, period-doubling, and boundary crisis) and the existence multiple fixed
points. Comparing the relationship between synchrony, oscillation type, and bifurcations has not been done before and shows the complexity of metapopulation dynamics even under simple dispersal assumptions.

Finally, we investigate a seven-patch metapopulation of Columbia spotted frogs located in Montana. We test different density-dependent dispersal rates and dispersal strategies that reflect the system’s landscape features. We use a patch contribution measure to determine patches that are most important to the system, then compare results across the different dispersal strategies. We develop a sensitivity analysis method that can be used for local sensitivity and elasticity analysis of any density-dependent matrix model. Applied to our system, we find that managing high survival patches produces the largest increases to the total population, but changes in patch contributions and patch sizes vary depending on the dispersal type. Where negative density-dependent dispersal decreases patch contributions, constant dispersal and positive density-dependent dispersal increase patch contribution. We use these results to offer potential management action that encourages patch heterogeneity in this system.
Dynamical systems analysis in adaptive and metapopulation ecology with applications to conservation management

Guenchik Grosklos

The ability for a species to persist largely relies on how well they adapt to the environment and their interactions with local and global communities. Specifically, if adaptation occurs quickly enough or nearby communities sufficiently promote growth rates, populations at risk of extinction may persist. In this dissertation, we first develop a method that estimates and compares rates of change in time series data of population densities and measurable traits (phenotypes). Additionally, we compare between genetic (evolutionary) and non-genetic (plastic) trait change to determine whether phenotypes change faster when driven by evolutionary or plastic change. We then focus on metapopulation models to understand system dynamics and viability metrics in amphibian populations. We start by investigating a two patch model with 1, 2, and 3 life history stages to understand how dispersal affects population dynamics and synchrony. We categorize dispersal based on the magnitude of dispersal probabilities and degree of symmetry to understand how different dispersal types affect population fluctuations and synchrony. Finally, we use habitat contribution metrics to investigate viability in a seven pond Columbia spotted frog population located in western Montana. We classify each pond based on their relative importance to the global community and use sensitivity analysis to measure how habitat management affects pond size, total population size, and the degree of habitat importance. These results provide a means to understand how species respond to environmental and anthropogenic disturbances for habitat management efforts.
Acknowledgments

I would like to thank my advisor, Dr. Jia Zhao, for guiding me through my PhD program and keeping me on track to finish. Thanks to my committee members, Dr. Noelle Beckman, Dr. Michael Cortez, Dr. Luis Gordillo, and Dr. Joe Koebbe, for their help and support throughout my research. A special thanks goes to Dr. Michael Cortez for his financial support, mentorship, and patience for me early in my PhD program. Also, I could not have navigated all of the administrative rigmarole without the help of the staff in the USU math department. A special thanks goes to Gary Tanner for being a great friend to every graduate student and organizing all of the events to make the math department friendly and inclusive.

I would also like to thank all of my mentors, colleagues, and friends for the variety of great experiences they gave me throughout my program. The numerous summer internships and short-term work experiences gave me diverse perspectives and research experiences I could not have found anywhere else. At Los Alamos National Lab, Dr. James Theiler, Dr. Chonggang Xu, and Dr. Devin Goodsman provided me research experience in machine learning algorithms for hyperspectral imaging, and integral projection models for plant demography. Thanks to the folks at the National Park Service (GrYN) for their financial and moral support, and motivation for much of my dissertation and research interests. Dr. Andrew Ray, Dr. David Thoma, Dr. Rebecca McCafferey, and everyone at the USGS office, it was a pleasure meeting you all, and thank you for showing me immense kindness and support. Finally, thanks to all the folks in the Climate Adaptation Science program for providing me with the interdisciplinary research experience and networking opportunities. The local subculture with my friends in this strange little town made my experience here that much better.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>PUBLIC ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xiii</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2 EVOLUTIONARY AND PLASTIC PHENOTYPIC CHANGE CAN BE JUST AS FAST AS</td>
<td>10</td>
</tr>
<tr>
<td>CHANGES IN POPULATION DENSITIES</td>
<td></td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>11</td>
</tr>
<tr>
<td>2.2 Data sets, methods and theory</td>
<td>14</td>
</tr>
<tr>
<td>2.2.1 Estimating rates of change from empirical time series</td>
<td>14</td>
</tr>
<tr>
<td>2.2.2 Distinguishing between fast and slow rates of change</td>
<td>16</td>
</tr>
<tr>
<td>2.2.3 Scaling to account for units and dimensions of data</td>
<td>18</td>
</tr>
<tr>
<td>2.3 Results</td>
<td>21</td>
</tr>
<tr>
<td>2.3.1 Application of method to simulated data</td>
<td>21</td>
</tr>
<tr>
<td>2.3.2 Application of method to empirical time series</td>
<td>22</td>
</tr>
<tr>
<td>2.4 Discussion</td>
<td>23</td>
</tr>
<tr>
<td>2.5 Acknowledgments</td>
<td>26</td>
</tr>
<tr>
<td>3 CHAOS DOES NOT DRIVE LOWER SYNCHRONY FOR INTRINSICALLY-INDUCED</td>
<td>30</td>
</tr>
<tr>
<td>POPULATION FLUCTUATIONS</td>
<td></td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>32</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.2 Mathematical model formulation</td>
<td>36</td>
</tr>
<tr>
<td>3.2.1 The two-patch model</td>
<td>36</td>
</tr>
<tr>
<td>3.2.2 Local patch demographics</td>
<td>38</td>
</tr>
<tr>
<td>3.3 Eigenvalue analysis and dynamics for the one-patch model</td>
<td>40</td>
</tr>
<tr>
<td>3.3.1 The Jacobian matrix</td>
<td>41</td>
</tr>
<tr>
<td>3.3.2 Analytic solutions of the Jacobian at equilibrium</td>
<td>42</td>
</tr>
<tr>
<td>3.3.3 Attractor types</td>
<td>43</td>
</tr>
<tr>
<td>3.3.4 Phase planes for the one-patch model</td>
<td>45</td>
</tr>
<tr>
<td>3.4 Two-patch model dynamics under constant dispersal probabilities</td>
<td>50</td>
</tr>
<tr>
<td>3.4.1 Dispersal symmetry</td>
<td>51</td>
</tr>
<tr>
<td>3.4.2 Patch synchrony</td>
<td>53</td>
</tr>
<tr>
<td>3.4.3 Fractal basins of attraction</td>
<td>54</td>
</tr>
<tr>
<td>3.4.4 Two-patch model attractor type and synchrony results</td>
<td>55</td>
</tr>
<tr>
<td>3.4.5 Bifurcation analysis</td>
<td>59</td>
</tr>
<tr>
<td>3.5 Conclusions</td>
<td>63</td>
</tr>
<tr>
<td>3.5.1 Dispersal effects on attractor types</td>
<td>64</td>
</tr>
<tr>
<td>3.5.2 Dispersal effects on synchrony</td>
<td>65</td>
</tr>
<tr>
<td>3.5.3 Implications for amphibian populations</td>
<td>66</td>
</tr>
<tr>
<td>4 SENSITIVITY ANALYSIS OF HABITAT CONTRIBUTIONS IN A DENSITY-DEPENDENT COLUMBIA SPOTTED FROG METAPOPULATION MODEL</td>
<td>68</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>70</td>
</tr>
<tr>
<td>4.2 Methods</td>
<td>74</td>
</tr>
<tr>
<td>4.2.1 The matrix model</td>
<td>74</td>
</tr>
<tr>
<td>4.2.2 The contribution metric</td>
<td>76</td>
</tr>
<tr>
<td>4.2.3 Habitat contribution metric</td>
<td>76</td>
</tr>
<tr>
<td>4.2.4 Sensitivity of population vectors</td>
<td>78</td>
</tr>
<tr>
<td>4.2.5 Sensitivity of patch contributions</td>
<td>79</td>
</tr>
<tr>
<td>4.2.6 Elasticity</td>
<td>80</td>
</tr>
</tbody>
</table>
4.3 Application to a Columbia spotted frog population ........................................ 80
  4.3.1 Study area ............................................................... 80
  4.3.2 Within pond demographics ............................................. 82
  4.3.3 Dispersal strategies .................................................... 83
  4.3.4 Fully vs partially connected systems ................................. 85
  4.3.5 Sensitivity analysis .................................................... 86

4.4 Results ................................................................. 89
  4.4.1 Fitting density-dependent larval survival to data ...................... 89
  4.4.2 Dispersal effects on pond dynamics .................................. 90
  4.4.3 Population size and contribution elasticity analyses .................. 92
  4.4.4 Implications of elasticity analysis .................................... 97
  4.4.5 Implications for conservation management ............................... 99

4.5 Discussion ............................................................. 100

5 CONCLUSION ...................................................................... 104

BIBLIOGRAPHY ................................................................... 106

APPENDICES ........................................................................ 127
  A Supplementary Information for ‘Evolution and plastic phenotypic change can
    be just as fast as changes in population densities’ .................................. 128
    A.1 The link between feedbacks and rates of change ............................ 128
    A.2 Details about empirical data sets ............................................. 133
    A.3 Model simulations ............................................................. 135
    A.4 Comparison of results from methods using alternative scalings and sum-
      mary statistics .................................................................. 136
  B Supplementary Information for ‘Chaos does not drive lower synchrony for
    intrinsically-induced population fluctuations’ ................................. 144
    B.1 Proof of vec-permutation matrix conversion .............................. 144
    B.2 Total maturation probability .................................................. 145
B.3 Bifurcation analysis in the one patch model ......................... 145
B.4 Calculating multiple end states ........................................ 152
B.5 Two-patch model results ................................................. 153
B.6 Multiple end state phase planes ....................................... 166
B.7 2-cycle saddle-node bifurcation ....................................... 168
B.8 Matlab code ............................................................. 170
C Supplementary Information for ‘Sensitivity analysis of habitat contributions in a density-dependent Columbia spotted frog metapopulation model’ ........ 259
C.1 Larval survival model fitting .............................................. 259
C.2 Local sensitivity analysis .................................................. 261
C.3 Global sensitivity analysis ............................................... 264
C.4 Matlab code ............................................................. 267
CURRICULUM VITAE ......................................................... 321
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Estimated scaled maximum rates of density and phenotypic change from empirical studies</td>
<td>27</td>
</tr>
<tr>
<td>3.1</td>
<td>Symbols used in the model and analysis.</td>
<td>40</td>
</tr>
<tr>
<td>3.2</td>
<td>Default vital rates and values used in the model. Vital rates are based on ranges used in different amphibian systems.</td>
<td>46</td>
</tr>
<tr>
<td>3.3</td>
<td>Adult survival used in the two-patch systems with local stability dynamics and Lyapunov exponent ($\lambda_e$).</td>
<td>52</td>
</tr>
<tr>
<td>4.1</td>
<td>Symbols used in the model and analysis.</td>
<td>81</td>
</tr>
<tr>
<td>4.2</td>
<td>Distances (m) between each pond in the study site. Measurements were taken by using the Google Maps linear distance measuring tool between center points of each pond.</td>
<td>86</td>
</tr>
<tr>
<td>4.3</td>
<td>Default vital rates and values used in the model. Vital rates are based on ranges defined in McCaffery and Maxell (2010).</td>
<td>89</td>
</tr>
<tr>
<td>4.4</td>
<td>Management strategies to maintain pond type heterogeneity in the metapopulation. Strategies may restore (up arrow), degrade (down arrow), do nothing to (NA) a pond to increase contributions in other ponds. Bolded numbers are largest contributions in each pond type.</td>
<td>100</td>
</tr>
<tr>
<td>A.1</td>
<td>Estimated scaled maximum rates of density and phenotypic change for each time series.</td>
<td>141</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Behavior of a fast-slow eco-evolutionary predator-prey model where evolution of prey defense is slow.</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>(A) Prey density (solid black) and mean prey defense (dashed black); predator density is not shown.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(B) Rates of change for prey density (solid black) and mean defense (dashed black). The vertical dashed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>line denotes where rates of change in prey density transition from being faster than (left) to similar</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to (right) phenotypic rates of change. (C) Solution (solid black line) plotted in phase space with</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the critical manifold (gray). Double and single arrows denote when rates of change in population density</td>
<td></td>
</tr>
<tr>
<td></td>
<td>are much faster than and similar to, respectively, rates of phenotypic change. Simulation is of the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Abrams and Matsuda (1997) model; see appendix for equations and parameter values.</td>
<td></td>
</tr>
</tbody>
</table>
2.2 With sufficient sampling, our method can accurately estimate maximum rates of density and phenotypic change from simulated time series generated. Time series were generated using the eco-evolutionary predator-prey model in Abrams and Matsuda (1997), where the rate of evolution was sped up or slowed down by factors of 2, 1, 0.5, 0.1, and 0.05. (A) Estimated scaled maximum rates of density and phenotypic change for different speeds of evolution. Diagonal lines denote when maximum rates of phenotypic change are ten times faster (upper dashed line), equal to (solid line), and ten times slower (lower dashed line) than maximum rates of change in population densities. Examples of scaled prey density (solid black) and mean trait (dashed black) time series when (B) $\epsilon_{ph} \approx 2\epsilon_{den}$, (C) $\epsilon_{ph} \approx \epsilon_{den}$, and (D) $\epsilon_{ph} \approx 0.1\epsilon_{den}$. (E) Estimated scaled maximum rates of density and phenotypic change for different speeds of evolution and sampling intervals. Sampling is every 1 (black shapes), 4, 10, 20, 40, 100 or 200 (lightest gray shapes) time units. (F-H) Demonstration of how increasing the sampling interval to every 40 time steps alters the time series in panels B-D; numerical values differ between pairs of panels because the sampling interval affects the range of the time series and consequently, how each time series is scaled. See appendix for model equations and parameter values.

2.3 Maximum rates of density and phenotypic change are similar in all 30 empirical studies. (A) Estimated scaled maximum rates with standard error bars; see Table 1 for legend. (B) Ratio of the maximum rate of phenotypic change to the maximum rate of density change. Red, blue, and brown correspond to studies where the mode of adaptation is genetic, plastic, or unknown, respectively. In both panels, lines denote when maximum phenotypic rates are two times faster (upper dashed line), equal to (solid line), and two times slower (lower dashed line) than maximum density rates.
3.1 Phase planes for juvenile and adult populations in the one-patch, 2-stage model with calculated Lyapunov exponent for an invariant cycle (A; \( \lambda_e = 0 \)), low-point k-cycle (B; \( \lambda_e < 0 \)), high-point k-cycle (C; \( \lambda_e < 0 \)), and chaos (D; \( \lambda_e > 0 \)). The k-cycles are formed by the phase-locked invariant cycles while retaining the shape of the closed loop. Chaos is developed as a wrinkle forms in the invariant cycle, creating sensitivity to initial conditions. Simulations are created by plotting the final 9000 juvenile and adult densities for \( R_H = 167.333 \), and \( S_a = 0.366(A), 0.228(B), 0.139(C), \) and \( 0.099(D) \).

3.2 Dominant eigenvalues (\( \lambda \)) for a range of adult survival (\( S_a \)), yearly recruitment rate (\( R_H \)), and density-dependent exponent (\( \gamma \)) for the one-patch model. The solid black lines represent bifurcations in the nontrivial fixed points. The 1-stage model has flip bifurcations, creating k-cycles whereas the 2- and 3-stage models undergo Hopf bifurcations, inducing invariant cycles. Additionally, under low adult survival and fecundity in the 3-stage model (C), the nontrivial fixed point experiences a transcritical bifurcation where it switches stability with the trivial fixed point.

3.3 Attractor planes for a range of adult survival (\( S_a \)), yearly recruitment rate (\( R_H \)), and density-dependent exponent (\( \gamma \)) for the one-patch model. Attractor type is determined with the Lyapunov exponent with subclassification of k-cycles into 2, 4, 8, 16, 16+ period cycles, and all other k-cycles with less than 16 points (other cycles). Flip bifurcations create k-cycles in the 1-stage model where chaos arises under period-doubling routes with periodic windows of low-point k-cycles. For the 2- and 3-stage models, invariant cycles are created through a Hopf bifurcation with phase-locked regions generating low-point k-cycles in the 3-stage model and a mix of low-point cycles, high-point cycles, and chaos in the 2-stage model.
3.4 One-patch bifurcation diagrams (A, B, C) with calculated Lyapunov exponents (D, E, F) and dominant eigenvalues (G, H, I) allow us to analyze bifurcations under a range of adult survival probabilities. In the 1-stage model, k-cycles occur through a flip bifurcation, creating chaos through a period-doubling route. In the 2-stage model, Hopf bifurcations create invariant cycles with phase-locked k-cycles whereas invariant cycles persist in the 3-stage model. Bifurcations and attractor types are verified with respective Lyapunov exponents (2nd row) and dominant eigenvalues (3rd row).

3.5 The 5 zones for the dispersal plane that represent different dispersal probabilities and symmetries. Zone 1 represents low symmetric dispersal, zone 2 represents intermediate symmetric dispersal, zone 3 represents high symmetric dispersal, zone 4 represents low asymmetric dispersal, and zone 5 represents high asymmetric dispersal.

3.6 Dispersal may create multiple end state dynamics under different initial conditions even though all vital rate parameters are unchanged. We vary initial conditions \( (a_1(0), a_2(0)) \) while fixing all parameters to produce fractal basins of attraction. In A, we use \( \sigma_1 = 0.87 \) and \( \sigma_2 = 0.96 \), and the system either experiences chaos (red) or stable dynamics (white). In B, we use \( \sigma_1 = 0.87 \) and \( \sigma_2 = 0.9 \), and the system either experiences k-cycles (black) or stable dynamics (white). In both scenarios, \( S_{a1} = 0.1 \) and \( S_{a2} = 0.3 \).

3.7 Attractor planes for homogeneous metapopulations \( (S_{a1} = S_{a2}) \) in the one stage model show that dispersal regions containing oscillations are similar under small vital rate perturbations. Attractor type is classified as equilibria (white), invariant cycles (blue), k-cycles (black), and chaos (red). K-cycles are further classified based on their periodicity. Dispersal-induced oscillations occur for high adult survival (B, C) in similar unstable regions as intermediate adult survival (A).
3.8 The relative frequency of per capita symmetric dispersal ($\alpha$) for each attractor type for the 1-stage (A), 2-stage (B), and 3-stage (C) models, and the fraction that each dispersal type appears in each zone (D). Unstable occurrences are classified as invariant cycles (blue), $k$-cycles (black), chaos (red), and stable equilibria (purple). The number of occurrences for each attractor type are noted in parentheses. Chaos is most prevalent under symmetric dispersal in the 1-stage model, invariant cycles occur under asymmetric dispersal in the 1-stage model and symmetric dispersal in the 2- and 3-stage models, and $k$-cycles occur for all dispersal symmetries in the 1-stage model, under high symmetric dispersal in the 2-stage model, and rarely in the 3-stage model. Stable equilibria are present for all symmetries across all models.

3.9 The relative frequencies for synchrony levels for each attractor type summarize the stability-synchrony relationships in the different dispersal zones. The number of occurrences that an attractor type appears in each zone is in parentheses and $\beta$ is the proportion of synchrony occurrences within each attractor type. In the 1-stage model, intermediate to high dispersal probabilities (zones 2, 3, and 5) produce lower synchrony ($\beta \approx 0.8$) for all attractor types whereas low dispersal probabilities (zones 1 and 4) have more results with near-perfect synchrony. In the 2-stage model, most results have near-perfect synchrony under sufficient dispersal (zones 2, 3, and 5) with the unique finding of near-perfect asynchronous $k$-cycles under very high symmetric dispersal (zone 3). In the 3-stage model, highly synchronous invariant cycles are consistently high across all zones.
3.10 Bifurcation and eigenvalue plots of all global attractors (black dots), fixed points (red circles), and dominant eigenvalues of fixed points for the two-patch, 1-stage model. As $\sigma_{a1}$ increases in plots A, D, k-cycles undergo a period-doubling route to chaos ($\sigma_{a1} \approx 0.45$) which then experiences a crisis at $\sigma_{a1} \approx 0.8$, creating a brief window of stability as it collides with an unstable orbit. At $\sigma_{a1} \approx 0.9$, the stable equilibrium undergoes a Hopf bifurcation, creating an invariant cycle. In plots B, E, a k-cycle appears at $\sigma_{a1} \approx 0.7$ and undergoes a period-doubling route to chaos. This persists until a new fixed point is created at $\sigma_{a1} \approx 0.87$. The new fixed point quickly undergoes a flip bifurcation to Hopf bifurcation, causing the alternate attractor to shift from a k-cycle to equilibrium to invariant cycle. In plots C, F, chaos persists under phase-locked k-cycles for $0.15 < \sigma_{a1} < 0.9$ with a period doubling route occurring at $\sigma_{a1} \approx 0.1$. Two unstable fixed points are created at $\sigma_{a1} \approx 0.75$ via a pitchfork bifurcation where only the chaotic attractor exists. The fixed points gain stability for a brief window before once again losing stability via Hopf bifurcations where invariant cycles are created that quickly phase-lock to chaos (see appendix). Parameters used are $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.6$ for plots A, D, $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.96$ for plots B, E, and $S_{a1} = S_{a2} = 0.3$, $\sigma_{a2} = \sigma_{a1}$ for plots C, F.
3.11 Bifurcation and eigenvalue plots of all global attractors (black dots), fixed points (red circles), and dominant eigenvalues of fixed points for the two-patch, 2-stage model (columns 1 and 2), and 3-stage model (column 3). In plots A, D, an invariant cycle persists until the loss of the imaginary component in the dominant eigenvalue shifts the invariant cycle to a 2-point k-cycle. In plots B, E, a Hopf bifurcation marks the creation of an invariant cycle for a brief window at $0.3 < \sigma_{j1} < 0.4$ with a stable equilibrium occurring elsewhere. In plots C, F, a saddle-node bifurcation creates two fixed points at $\sigma_{s1} \approx 0.8$, one of which is stable, resulting in two stable nontrivial fixed points. Parameters used are $S_{a1} = S_{a2} = 0.5$, $\sigma_{a2} = \sigma_{a1}$ for plots A, D, $S_{a1} = 0.5$, $S_{a2} = 0.9$, $\sigma_{j2} = 0.8$ for plots B, E, and $S_{a1} = S_{a2} = 0.9$, $\sigma_{s2} = 0.9$ for plots C, F.

4.1 Examples of how dispersal probabilities change as $\beta$ and $r$ change. Dispersal from a patch (A) is negative density-dependent (dash line), positive density-dependent (solid line), or constant (dash-dot line) when $\beta$ is negative, positive, or zero. As $r$ increases (B), the probability of dispersing to all ponds becomes more uniform (dash-dot line) whereas low $r$ values distribute individuals to nearby ponds (dash line). We set $\sigma_{\max} = 0.5$ in A and used $e^{-d/r}$ in B where $d$ is the distance between patches.

4.2 Individuals may disperse to ponds based on waterway networks. Our system consists of seven ponds with a river network connecting some of the ponds to each other (A). Individuals that choose to disperse may only be able to disperse to the nearest pond connected by a waterway (B).
4.3 We fit the Hassell competition model to egg count data for each pond type to define pond-specific larval survival. Permanent ponds (A) have high carrying capacities where larval survival is less affected by pond densities. Semi-permanent ponds (B) have the highest maximum survival but cannot sustain a large number of recruits. Ephemeral ponds (C) have consistently lower survival than permanent and semi-permanent ponds.

4.4 Pond populations (A, B) and contributions (C, D) for a range of density-dependent dispersal ($\beta$) averaged over all $r$ values (solid lines) with standard deviations (shaded area). Under strong negative density-dependent dispersal ($\beta < -0.1$), contributions are all approximately 1 while populations reflect local pond demographics. For weak negative density-dependent dispersal ($-0.1 < \beta < 0$), contributions and populations experience drastic changes with large (full) or small (partial) standard deviations. Under positive density-dependent dispersal ($\beta > 0$), contributions and populations vary based on pond demography and geographic locations, where centrally-located ponds have higher populations and lower contributions depending on feedbacks between connected ponds. In particular, $P_2$ has the highest population in both the full and partial systems but is a source in the full systems and sink in the partial system because of high immigration coming from ponds $P_1, S_2, E_1, \text{and } E_2$. 

90

93
4.5 The difference between pond contributions in the partial and full systems averaged across all \( r \) values for strong negative density-dependent dispersal \((\beta < -0.1; \text{open circle})\), weak negative density-dependent dispersal \((-0.1 < \beta < 0; \text{ex})\), weak positive density-dependent dispersal \((0 < \beta < 0.1; \text{star})\), and strong positive density-dependent dispersal \((\beta > 0.1; \text{closed circle})\) with standard deviations (black line). Little differences occur between the full and partial systems under strong negative density-dependent dispersal while the other dispersal strategies produce a variety of contributions with larger standard deviations. In general, contributions are higher in the partial system for \(P_1, S_2, E_1, \) and \(E_2\) and lower for \(P_2\) and \(S_1\). Even though strong negative density-dependent dispersal produces similar results between the full and partial systems, weak negative density-dependent dispersal has the largest standard deviations.

4.6 We validate our sensitivity analysis (\(\partial\)) by comparing results with perturbed simulations (\(\Delta\)) to perturbations in larval survival parameters. Plots A and C compare pond size sensitivities while plots B and D compare contribution sensitivities. The linear relationship between the simulated perturbations (\(x\)-axis) with the calculated sensitivities (\(y\)-axis) show that our sensitivity analysis accurately represents simulations. All results are scaled by the pseudo-logarithm to account for large-scale differences.

4.7 Elasticities of total population sizes to larval survival parameters averaged over all \( r, \beta \) simulations in the full system (A) and partial system (B). Each colored bar represents elasticity of the total population to pond-specific larval survival parameters with corresponding error bars. Permanent pond perturbations have the largest total population changes with pond size (\(\gamma_x\)) having greater elasticities compared to maximum larval survival (\(L_x\)) and carrying capacity (\(K_x\)).
4.8 Elasticities of pond contributions (A) and pond population sizes (B) to larval survival parameters averaged over $r = 50,1000$ and both the full and partial systems. Changes in pond-specific larval survival have both direct (colored symbols) and indirect (black dots) effects on each pond. Under negative density-dependent dispersal ($\beta = -0.2$), pond contributions show very little change under larval survival perturbations while pond sizes experience much larger direct changes. For constant dispersal ($\beta = 0$) and positive density-dependent dispersal ($\beta = 0.2$), all ponds experience noticeable direct and indirect changes under larval survival perturbations. Additionally, pond contributions experience a sign change as density-dependent dispersal shifts from negative to constant to positive. This sign change does not occur for pond size.

A.1 Ratios of maximum phenotypic rates and maximum density rates for all time series analyzed in this study. Replicates from the same study are connected by solid vertical gray lines. Studies are colored according to the mode of adaptation: genetic ($a-r$; red), plastic ($A-H$; blue), and unknown ($\alpha - \psi$; brown). Symbols correspond to studies listed in Tables 1 and S1. The horizontal lines denote when phenotypic rates are two times faster (upper dashed line), equal to (solid line), and two times slower (lower dashed line) than density rates of change.

A.2 The average number of samples collected per generation for all time series analyzed in this study. Replicates from the same study are connected by solid vertical gray lines. Symbols correspond to studies listed in Tables 1 and S1 and colored according to the mode of adaptation: genetic ($a-r$; red), plastic ($A-H$; blue), and unknown ($\alpha - \psi$; brown). The dashed line represent studies where the average number of samples collected were once per generation. Studies that lie below the dashed line indicate that less than one sample was collected per generation.
A.3 Comparison of the methods for estimating rates of density and phenotypic change used in the current study and in DeLong et al. (2016). Rates of density and phenotypic change estimated (left column) using maximum derivatives or (right column) average derivatives that are either (4th row) scaled by the standard deviation, (3rd row) unscaled, (2nd row) scaled by the magnitude of the values (i.e., per capita rates), or (1st row) scaled by the range of the data. Panel A shows the results from the method used in the current study. Panel F shows the results from the method used in DeLong et al. (2016). In all panels, the black line denotes the 1:1 line and the upper and lower dashed lines denote where phenotypic rates of change are two times faster or two times slower, respectively, than density rates of change. Symbols correspond to studies listed in Table 1.

B.1 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 1-stage, 1 patch model. We find that k-cycles appear through flip bifurcations at $S_a \approx 0.6$. From there, we find two different routes to chaos. For $R_H = 96.77$, chaos occurs in one section ($0.2 \leq S_a \leq 0.4$) through period doublings in the k-cycles. Period doublings also induce chaos for $R_H = 247.98$ and $R_H = 399.19$ on the left-most and right-most chaotic bounds, and for chaos found in the intermediate section of $R_H = 399.19$. However, for $S_a \approx 0.2$ and $S_a \approx 0.35$, a window of stable 3-cycles appear through a phase-locking route.

B.2 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 1-stage, 1 patch model. Similar to bifurcations in $R_H$, k-cycles form through flip bifurcations and transition to chaos through period doublings. We also see small windows of stable k-cycles within chaos regions.
B.3 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 2-stage, 1 patch model. Hopf bifurcations create invariant cycles for $S_a \approx 0.6$ which experience phase-locking that create k-cycles in different regions. The phase-locked k-cycles may then either return to invariant cycles (e.g., $0.3 < S_a < 0.35$ for $R_H = 96.77$) or is replaced with chaos (e.g., $0.05 < S_a < 0.1$ for $R_H = 247.98$). In these plots, the invariant cycles develop wrinkles that mix trajectories, creating sensitivities to initial conditions. 

B.4 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 2-stage, 1 patch model. Similar to the $R_H$ bifurcations, invariant cycles appear under Hopf bifurcations which are then phase-locked to k-cycles and in some cases chaos.

B.5 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 3-stage, 1 patch model. We find that invariant cycles occur through Hopf bifurcations with windows of phase-locked k-cycles.

B.6 Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 3-stage, 1 patch model. We find that invariant cycles occur through Hopf bifurcations.

B.7 Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 1-stage model.
B.8 Eigenvalue planes for adult dispersal in the two patch, 1-stage model. Solid black lines represent where bifurcations occur ($|\lambda| = 1$).

B.9 Attractor (left) and synchrony planes (right) for juvenile dispersal in the two patch, 2-stage model.

B.10 Eigenvalue planes for juvenile dispersal in the two patch, 2-stage model. The black regions in the top-right corners of the last two rows are a mix of eigenvalues of unstable steady states created under high dispersal probabilities.

B.11 Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 2-stage model.

B.12 Eigenvalue planes for adult dispersal in the two patch, 2-stage model.

B.13 Attractor (left) and synchrony planes (right) for juvenile dispersal in the two patch, 3-stage model.

B.14 Eigenvalue planes for juvenile dispersal in the two patch, 3-stage model. The small black region in the final plot is due to a saddle-node bifurcation that created a stable and unstable equilibrium.

B.15 Attractor (left) and synchrony planes (right) for subadult dispersal in the two patch, 3-stage model.

B.16 Eigenvalue planes for subadult dispersal in the two patch, 3-stage model. The small black region in the final plot is due to a saddle-node bifurcation that created a stable and unstable equilibrium.

B.17 Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 3-stage model.

B.18 Eigenvalue planes for adult dispersal in the two patch, 3-stage model.
B.19 Phase planes for the two patch, 1-stage model with parameters $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.96$, and patch 1 dispersal $\sigma_{a1} = 0.75$ (A), $\sigma_{a1} = 0.85$ (B), and $\sigma_{a1} = 0.95$ (C). We find that two stable fixed points exist in each of these cases. First, a 4-point k-cycle occurs in (A), then as $\sigma_{a1}$ increases, the chaos exists via a period-doubling route from a flip bifurcation. Note that at least 2 unstable fixed points exist among these phase planes (verified with numerical simulations initialized by multiple fixed points).

B.20 Phase planes for the two patch, 1-stage model with parameters $S_{a1} = 0.3$, $S_{a2} = 0.3$, $\sigma_{a2} = \sigma_{a1} = 0.75$ (A), $\sigma_{a2} = \sigma_{a1} = 0.85$ (B), $\sigma_{a2} = \sigma_{a1} = 0.95$ (C). What starts as chaos formed from k-cycles (A) turns into two unique fixed points (B) then finally three unique chaotic cycles identified by red dots, circles, or stars (C). Each cycle in (C) is found by closely examining end-state behavior of simulations under initial conditions that are slightly perturbed from the different fixed points. Among these cycles exist two unstable equilibria. Note that because the two patches are symmetric with symmetric dispersal, the two stable equilibria in (B) and two of the chaotic cycles in (C; star, dot) have symmetric fixed points.

B.21 First- and second-iterate maps of the two-patch, 1 life-history stage model for $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.96$ (Figure 3.10B in the main text). The upper intersections in all plots represent the persistent stable equilibrium for this scenario. In plot D, $f^2(a_1, a_2)$ lies near the one-to-one line but does not intersect. As $\sigma_{a1}$ increases, the graph crosses the one-to-one line, creating a stable and saddle 2-point k-cycle via 2-cycle saddle-node bifurcation.
C.1 Optimized root mean squared error estimates for a range of parameters sampled from the Latin hypercube sampling method shows that two local minima occur for each pond type (subfigures A, E, I). Plotting all lower-valued root mean squared errors for each parameter shows that absolute minima occur for permanent and semi-permanent ponds; however, ephemeral ponds never reach a local minimum. This is most likely due to the small sample size in ephemeral ponds.

C.2 Total population elasticities to pond-specific parameters for the six $r - \beta$ combinations in the full (A) and partial (B) systems. Each bar represents total population elasticity to the pond-specific parameter specified by color.

C.3 Elasticities of pond contribution to model parameters averaged over $r = 50, 1000$ and the full and partial systems. Each bar represents specific pond elasticity to the parameter defined on the $x$-axis.

C.4 Elasticities of pond size to model parameters averaged over $r = 50, 1000$ and the full and partial systems. Each bar represents specific pond elasticity to the parameter defined on the $x$-axis.

C.5 Global sensitivity of the total population to larval survival parameters, $r$, and $\beta$. We find that the magnitude of most sensitivities are less than 0.2, indicating that confidence in these estimates are similar.

C.6 Global sensitivity of pond contribution and size to larval survival parameters, $r$, and $\beta$. We find that the magnitude of some sensitivities are greater than 0.5, indicating that confidence in these parameter estimates are low. This emphasizes the importance of ensuring accurate parameter estimates.
CHAPTER 1

INTRODUCTION

One of the goals in ecology research is to understand the underlying mechanisms driving population dynamics and species responses to environmental change. Specifically, the ways in which an organism responds to environmental disturbances and the degree to which adaptation occurs determine their likelihood of persistence. This information is used for managing communities affected by environmental and anthropogenic changes. To understand and test different management scenarios, mathematical models are used as a cost-effective and non-invasive complement to field studies and conservation efforts.

The following chapters use mathematical models that cover a broad range of topics in theoretical ecology, including time series data analysis and population-level dynamics with applications to conservation management. First, we analyze time series data using scaling theory and fast-slow dynamical systems theory in order to estimate and compare relative rates of ecological and phenotypic change. Further, phenotypic change may be driven by either plasticity or evolution which we use to determine whether the relationship between ecological and phenotypic change depends on the mode of adaptation. For the remainder of the dissertation, we use matrix population models to analyze metapopulation dynamics and understand drivers of population persistence in two contexts: 1) we investigate two-patch metapopulations with 1, 2, and 3 life stages to understand dispersal effects on model dynamics and population synchrony, and 2) we explore a seven-patch metapopulation and how different dispersal behaviors affect the relationship between population abundances, patch contributions, and their sensitivities. We base our model and parameter estimates on general amphibian literature for the two-patch model and a Columbia spotted frog population in the seven-patch model. These results help land managers understand general metapopulation ecology and inform them of effective conservation management strategies.
Feedbacks exist between ecological (i.e., population density) and phenotypic (i.e., population-level traits) processes. Evolutionary and plastic adaptation can occur fast enough to alter population dynamics within a few generations. Empirical studies find that phenotypic rates of change are often comparable to ecological rates of change. For example, environmental changes that threaten species extinction can be prevented via evolutionary rescue (Bell 2017; Carlson et al. 2014) and predator-prey cycles can be altered by rapidly evolving prey defenses (Becks et al. 2010; Yoshida et al. 2007; 2003). These studies suggest that rates of phenotypic change have the potential to change as fast as rates of density change. One previous study (DeLong et al. 2016), performed a meta-analysis on eco-evolutionary data to compare how fast phenotypic rates of change are to density rates of change. They used averaged per capita rates of density and phenotypic change and found that phenotypic rates were always slower than density rates, with phenotypes being more than ten times slower in some cases. This result conflicts with the empirical studies cited above, suggesting that additional work is needed to explain the disagreement.

To estimate rates of change in ecology and phenotype, we use fast-slow dynamical systems theory (Arnold et al. 1995; Kuehn 2015) and scaling theory (Logan 2013). Fast-slow systems are characterized by some components changing at a much faster rate than other components. Additionally, the fast component typically changes at a rate comparable to the slow component for the majority of the time and experiences fast rates for only a short period of time. Fast-slow dynamical system models have been used to model plasticity (Cortez 2011) and eco-evolutionary dynamics (Cortez and Ellner 2010; Cortez 2015; Dercole et al. 2006; Dieckmann et al. 1995; Khibnik and Kondrashov 1997; Marrow et al. 1996). Scaling theory is used to account for units in time series data (Logan 2013). Specifically, we scale each data set by the range of the data and the species generation time to create a dimensionless variable, allowing for unbiased comparisons across estimations of phenotypic and density rates of change.

We analyze published time series from thirty empirical studies gathered from the seventeen studies analyzed in DeLong et al. (2016), searches using Web of Science and Google
Scholar, and other known studies. In each study, the phenotype, or trait, is either classified as plastic, genetic, or an unknown combination of the two. We find that the maximum rate of phenotypic change ranges between 0.5 and 2.5 times faster than the maximum rate of density change. Additionally, the relationships between maximum density rates and maximum phenotypic rates do not differ between the modes of adaptation. Our results suggest that population densities and population-level phenotypes are changing on similar time scales in nearly all of the systems and that the mode of adaptation does not systematically affect the rate of phenotypic change relative to the rate of change in population density. This result is at odds with the concept that traits driven by plasticity change at a faster rate than traits driven by genetics. This is important for understanding rates of adaptation in different organisms as well as model development in theoretical applications.

Habitat persistence relies not only on adaptation potential but also on the interactions between proximal habitats in a connected network. We use metapopulation models to explore population dynamics, synchrony, and patch contributions to understand how dispersal affects metapopulation viability and dynamics. We apply our models to amphibian literature with special attention paid to a Columbia spotted frog population.

Metapopulations are spatially structured landscapes with isolated habitats, or patches. Individuals in each patch undergo independent demographic processes while dispersal creates interactions between populations. Dynamics in a metapopulation rely on the connections between all habitats, making it important to understand both intrapatch and interpatch interactions. In particular, metapopulation dynamics and synchrony are affected by dispersal behavior and may depend on the type of dispersal, number of patches, and individual demography (Ylikarjula et al. 2000). Fluctuations may experience different qualitative behavior while synchrony is the degree to which fluctuations of multiple populations move in the same direction.

Dispersal between patches in a metapopulation can have both positive and negative effects on population viability. Where low dispersal can increase viability through local stabilizing effects, medium to high dispersal can synchronize populations, increasing the risk of
extinction in unfavorable environmental conditions (Abbott 2011; Wang et al. 2015). Alternatively, dispersal may induce rescue effects in endangered populations when immigration rates exceed emigration and mortality rates (Abbott 2011; Brown and Kodric-Brown 1977; Griffiths et al. 2010; Hudson and Cattadori 1999; Stevens and Baguette 2008; Ylikarjula et al. 2000). Dispersal is also known to stabilize cyclic and chaotic dynamics (Hastings 1993; Hellriegel 2000), but may also create fluctuations in equilibrium populations (Jang and Mitra 2000; Silva et al. 2001).

Many studies have investigated the stability-synchrony relationship in different metapopulation systems. Population fluctuations may be driven by environmental variation, intrinsic dynamics, or some combination of the two (Abbott 2011; Allen et al. 1993; Jansen 1999; Wang et al. 2015). However, a unifying theory connecting stability, synchrony, and dispersal does not exist. Generally speaking, low levels of dispersal may initially synchronize populations and reduce fluctuations, whereas moderate to high levels of dispersal may desynchronize and create fluctuations (Abbott 2011; Bjørnstad et al. 1999; Dey et al. 2014; Jansen 1999; Kendall and Fox 1998). Additionally, the mechanisms that drive oscillating behavior play a role in the effects of dispersal on synchrony and dynamics, highlighting the need to understand the relationship between dispersal, synchrony, and the causes of population fluctuations.

From a management perspective, it is important to identify patches that have the largest influence on metapopulation persistence, especially in large systems. The contribution metric achieves this by classifying patches as sources and sinks based on patch demography and dispersal (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). Sources produce sufficient recruits to sustain themselves while promoting metapopulation viability through emigration, whereas sinks cannot maintain viable populations without the support from immigrating individuals. By classifying each patch as a source or sink, land managers can readily distinguish between patches most vital for metapopulation viability from those that are less critical.

In population ecology, it is important to understand how changes in vital rates affect
population abundances. In the mathematical literature and applied ecology, this is called sensitivity analysis and quantifies the effects of a perturbed parameter on an output variable (Caswell 2001; Grant and Grant 2002; Hunter and Caswell 2005). We develop sensitivity analysis of habitat contribution of a generalized matrix metapopulation model. From an ecological context, understanding how patch contributions are affected by vital rate perturbations show the robustness of the metric to environmental and anthropogenic disturbances. Additionally, it provides a measure of error tolerance in parameter estimates. Combining sensitivities of patch contributions with sensitivities of other dependent variables such as population structures and growth rates brings additional confidence in error estimates and expected biological responses to environmental changes.

We focus our metapopulation analysis on general amphibian models and specifically to a Columbia spotted frog (*Rana luteiventris*) population located in the Selway-Bitterroot Wilderness, Montana. The Columbia spotted frog is a pond-breeding amphibian with a range that expands from the northwestern US to western Canada (Green et al. 1997; Patla and Keinath 2005a; b). Although spotted frogs are currently not considered under threat from extinction (Holthuijzen 2014; USFWS 2015), previous studies have shown population declines in Nevada (Reaser 2000), northern Great Basin (Wente et al. 2005), and western Montana (McCaffery et al. 2014; 2012). Reasons for these declines are generally unknown, but environmental factors such as annual runoff (Gould et al. 2019; Ray et al. 2016), snow water equivalency (McCaffery and Maxell 2010; McCaffery et al. 2012), and drought (Hossack et al. 2013; Patla and Keinath 2005a) all affect spotted frog growth rates and breeding success. Additionally, frog habitats may be negatively affected by grazing (Maxell 2000; Munger et al. 2002; Ross et al. 1999), fragmentation (Brown and Kodric-Brown 1977; Funk et al. 2005a; Patla and Keinath 2005a), and predation due to stocked fish (Arkle and Pilliod 2015; Hirner and Cox 2007; Pilliod and Peterson 2001; Reaser 2000). Even though spotted frogs are most not currently under threat from extinction, understanding the drivers that affect the dynamics in spotted frog populations can inform managers on proper actions necessary for future conservation efforts.
Columbia spotted frogs have complex life cycles consisting of 3 main life stages; juveniles, subadults, and adults (McCaffery et al. 2012). Juveniles are greater than 1 year old and too young to be sexed, subadults can be sexed but are smaller than a breeding adult (> 50 mm and < 62 mm snout-vent length), and adults are large enough to breed (> 62 mm snout vent length) (McCaffery et al. 2012). Spotted frogs can reach sexual maturity in 3-6 years with males typically reaching sexual maturity at a faster rate (Bull 2005; McCaffery et al. 2012; Patla and Keinath 2005a; Reaser 2000). Females lay their eggs in breeding sites each spring, though some sites do not show evidence of breeding every year (Patla and Keinath 2005a; Ray et al. 2016). The factors driving low breeding probabilities are mostly unknown, but may include environmental variables such as annual runoff and precipitation (Ray et al. 2016), or temporary migration where females leave the breeding site for a few years before returning to breed (Muths et al. 2018).

Columbia spotted frogs live near various waterbodies (ponds, lakes, springs, and/or streams) for breeding, foraging, and overwintering (Patla and Keinath 2005a; b). Each spring, mature adults gather at breeding sites characterized by slow-moving or stagnant water which includes slow-flowing streams, shallow ponds, and lake edges (Patla and Keinath 2005a; b). After breeding, individuals typically remain near breeding sites for foraging in non-breeding waterbodies, forests, meadows, and marshes (Patla and Keinath 2005a). Most spotted frogs stay within 600 meters of breeding sites (Patla and Keinath 2005a) but individuals have been known to travel up to 5700 meters in one season (Funk et al. 2005b). Spotted frogs seek shelter to avoid freezing or hypoxia in winter habitats which may include partially frozen ponds, undercut stream beds, or dug out water-filled holes (Bull and Hayes 2001; Patla and Keinath 2005a; Tattersall and Boutilier 1997).

Each Spring, a female spotted frog may lay up to one egg mass with egg counts ranging from 150 to 2400 eggs per egg mass depending on the region (Maxell 2000; Patla and Keinath 2005a). Once hatched, larvae use emergent vegetation for both food and shelter with pond hydroperiod (length of time that water remains in the pond) being a key factor for survival (Bull 2005; McCaffery et al. 2014). Ephemeral ponds, characterized by shallow
water no deeper than 1 meter, can dry out as early as August, killing all larvae before metamorphosing (Loman 2002; McCaffery et al. 2014). Semi-permanent ponds may dry out in particularly warm summers but also have the highest recruitment rates (McCaffery et al. 2014). Permanent ponds are characterized by waterbodies deeper than 1 meter, never dry out, and are able to support an abundance of egg masses (McCaffery et al. 2014). However, permanent ponds may have a higher predator presence, lowering amphibian recruitment rates due to predation (Knapp and Matthews 2000; Loman 2002; Meyer et al. 1998).

Individuals of all life stages may return to their natal pond for breeding or move to a different pond. Annual movement, or dispersal between breeding sites, is defined as the uni-directional movement from one breeding site to another over the course of the year (Semlitsch 2010), and can be classified as either natal dispersal (dispersal for pre-breeding amphibians) or breeding dispersal (dispersal for amphibians after first reproduction) (Cayuela et al. 2020; Smith and Green 2005). We define dispersal as the year-to-year movement from one breeding site to another for both pre- and post-breeding individuals. In general, juveniles tend to exhibit much more active movement and have higher dispersal rates than adults who show high site fidelity (Funk et al. 2005a; McCaffery et al. 2014; Smith and Green 2005). Reasons for dispersing are also unknown, but factors such as landscape type, social behavior, and proximity to other breeding sites could all play a role in dispersal rates and breeding site choice (Bowler and Benton 2005; Cayuela et al. 2020).

Mathematical models have been developed to understand amphibian life histories, growth rates, and processes driving population dynamics. Because of the discrete nature of amphibian life cycles, matrix models are typically used with body sizes discretized into multiple life stages (Biek et al. 2002; Halley et al. 1996; Hellriegel 2000; McCaffery et al. 2014; McCaffery and Maxell 2010; McCaffery et al. 2012; Vonesh and De la Cruz 2002). The effects of environmental factors on population viability and extinction rates are modeled using stochastic variables in one or more life stages (Halley et al. 1996; McCaffery et al. 2014; 2012; Willson et al. 2012). Competition for resources are addressed using intra-stage density-dependent survival rates (Halley et al. 1996; Hellriegel 2000; Vonesh and De la Cruz
Finally, populations with multiple, spatially separated water-bodies are modeled using metapopulation theory with interactions between habitats defined by dispersal mechanisms (Halley et al. 1996; Hellriegel 2000).

In chapter 3, we investigate the effects of dispersal on metapopulation stability and synchrony in two-patch systems with 1, 2, or 3 stages. We assume that all vital rates are constant except for larval survival which we use the Hassell competition model to regulate populations based on the number of eggs laid (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012). We classify attractor type into three categories (invariant cycles, k-cycles, chaos) using the Lyapunov exponent, and measure synchrony using summed temporal covariance matrices (Loreau and Mazancourt 2008; Thibaut and Conolly 2013; Wang et al. 2015; Wang and Loreau 2014). We test a range of constant dispersal in each life stage and divide dispersal probabilities into 5 zones based on dispersal degree and symmetry. In general, low levels of dispersal initially synchronize and stabilize dynamics while moderate to high levels of dispersal create oscillations and significantly desynchronize populations. Additionally, symmetric dispersal is less synchronizing and maintains fluctuations more than asymmetric dispersal. Using bifurcation and eigenvalue analysis, we find three different routes to chaos: phase-locking, period-doubling, and boundary crises. Where chaos is considered to have lower synchrony than other attractor types, we find that synchrony in chaos reflects the synchrony of the fluctuations preceding chaos. Thus, in many cases, synchrony does not necessarily rely on the qualitative behavior of fluctuations. The relationship between dispersal, population synchrony, and attractor type in metapopulations has not been done before, and we show that dispersal degree and symmetry have large effects on synchrony whereas attractor types generally have similar synchronies within the same dispersal regimes.

In chapter 4, we develop a model for sensitivity and elasticity analysis of patch contributions for density-dependent matrix metapopulation models. The analysis builds on sensitivity analysis of density-dependent populations and measures the change of habitat contribution to perturbed environments. We apply this analysis to the seven-patch
Columbia spotted frog population where dispersal is negative density-dependent, positive density-dependent, or density-independent. We analyze patch contributions and their relation to local and global abundances, then use elasticity analysis to identify how population size and contributions change under larval survival perturbations. We find that patch contribution elasticities are robust to landscape topography but depend on density-dependent dispersal type (positive versus negative) while patch population elasticities are robust to both density-dependent dispersal type and landscapes. Specifically, under negative density-dependent dispersal, vital rate perturbations decrease patch contributions while positive density-dependent dispersal and constant dispersal increase patch contributions. On the other hand, patch populations always increase when survival rates are increased. Additionally, conserving high survival patches always produces the largest population and contribution increases. We use these results to offer potential management action that encourages patch heterogeneity in this metapopulation.
CHAPTER 2

EVOLUTIONARY AND PLASTIC PHENOTYPIC CHANGE CAN BE JUST AS FAST AS CHANGES IN POPULATION DENSITIES

Abstract

Evolution and plasticity can drive population-level phenotypic change (e.g., changes in the mean phenotype) on time scales comparable to changes in population densities. However, it is unclear if phenotypic change has the potential to be just as fast as changes in densities, or if comparable rates of change only occur when densities are changing slow enough for phenotypes to keep pace. Moreover, it is unclear if this depends on the mode of adaptation. Using scaling theory and fast-slow dynamical systems theory, we develop a method for comparing maximum rates of density and phenotypic change estimated from population-level time series data. We apply our method to 30 published empirical studies where changes in morphological traits are caused by evolution, plasticity, or an unknown combination. For every study, the maximum rate of phenotypic change was 0.5 to 2.5 times faster than the maximum rate of change in density. Moreover, there were no systematic differences between systems with different modes of adaptation. Our results show that plasticity and evolution can drive phenotypic change just as fast as changes in densities. We discuss the implications of our results in terms of the strengths of feedbacks between population densities and traits.
2.1 Introduction

Populations change over time both in terms of their densities and in terms of the distributions of the phenotypes of individuals. The rate of phenotypic change at the population level (e.g., changes in the mean phenotype) and whether phenotypic change is driven by evolution versus plasticity can affect population-level dynamics, including species persistence following environmental change (Ghalambor et al. 2007; Reed et al. 2011) and invasion success (Lee 2002; Stockwell et al. 2003; Zenni et al. 2014). Empirical studies have shown that both evolution and plasticity have the potential to drive phenotypic change on the same time scale as changes in population densities. Studies on eco-evolutionary dynamics (Carrol et al. 2007; Fussman et al. 2007; Pelletier et al. 2009) have shown that evolution can occur fast enough to alter population dynamics over ecological time scales (e.g., a few generations). For example, rapid evolution during evolutionary rescue prevents species extinction driven by environmental change (Bell 2017; Carlson et al. 2014) and rapidly evolving prey defenses can cause and alter predator-prey cycles (Becks et al. 2010; Yoshida et al. 2007; 2003). Plastic behavioral and morphological response times can range between a few hours or a few weeks (Auld and Relyea 2011; Kuhlmann and Heckmann 1985; Kusch 1993; reviewed in Tollrian and Harvell 1999) and also can affect population dynamics (e.g., Lürling et al. 2005; Verschoor et al. 2004).

While these and other studies show that evolution and plasticity can drive population-level phenotypic change on the same time scale as changes in population densities, we might expect the relationships between density and phenotypic rates of change to systemically differ for plastic and evolving traits. In particular, it is reasonable to expect that phenotypic change is faster than changes in density (but still on the same time scale) for plastic responses that occur within the lifetime of an individual. In comparison, because evolutionary responses necessarily occur across generations, it is reasonable to expect that evolution would drive phenotypic change that is slower than changes in population densities (but also still on the same time scale). Thus, even though the changes in density and phenotypes are occurring on similar time scales in both cases, we might expect the population-level
rates of change for plastic traits to be faster than changes in population densities and the population-level rates of change for evolving traits to be slower than changes in population densities.

Understanding when phenotypic rates of change are faster than, equal to, or slower than changes in population densities is important because it, in part, determines the relative strengths of specific feedbacks, that in turn affect the dynamics of systems. Previous studies have focused on the roles ecological, evolutionary, and eco-evolutionary feedbacks play in driving eco-evolutionary dynamics (Cortez 2018; Cortez et al. 2020; Fleischer et al. 2018; Patel et al. 2018). These feedbacks can be naturally extended to density feedbacks (i.e., the effects species’ densities have on the dynamics of population densities), phenotype feedbacks (i.e., the effects species’ traits have on the dynamics of species traits), and density-phenotype feedbacks (i.e., the effects species’ population densities have on the dynamics of traits, and vice versa). Importantly, all of the feedbacks affect the dynamics of a system, but a given feedback has relatively stronger effects when the involved variables change at faster rates and when the coupling between the variables is stronger (Cortez 2018; Patel et al. 2018); see appendix S1.1 for mathematical details. For example, if there is sufficient coupling between all densities and all phenotypes and species’ densities change faster than phenotypes in a system, then theory predicts that effects of density feedbacks will be strongest, the effects of density-phenotype feedbacks will be intermediate, and the effects of phenotype feedbacks will be weakest. Thus, given sufficient coupling between densities and traits, feedbacks involving densities and traits with the fastest rates of change will have a stronger effect than feedbacks involving densities and traits with slower rates of change (Cortez 2018; Patel et al. 2018).

This suggests that if the relationships between density and phenotypic rates of change systemically differ for plastic and evolving traits, then different kinds of feedbacks have stronger effects on the dynamics of communities with plastic versus evolving traits. For example, if plastic traits change faster than population densities, then theory predicts that the effects of phenotype feedbacks are greater than the effects of density-phenotype feedbacks.
which are greater than the effects of density feedbacks in those systems. In comparison, if evolving traits change slower than population densities, then theory predicts that the ordering is reversed in those systems (i.e., density feedbacks are strongest and phenotype feedbacks are weakest). Previous theoretical work on eco-evolutionary dynamics predicts that differences in the strengths of feedbacks and differences in the rates of density and phenotypic change can lead to different population-level dynamics in predator-prey and other communities (Cortez 2016; 2018; Cortez and Patel 2017; Patel and Schreiber 2015; van Velzen and Gaedke 2017; Vasseur et al. 2011). Thus, systematic differences in the relationships between density and phenotypic rates of change for plastic and evolving traits could imply different population-level dynamics in those systems.

A previous meta-analysis of empirical studies (DeLong et al. 2016) explored whether the rates of change of ecologically important phenotypes were faster, slower, or comparable to rates of change in densities. DeLong et al. (2016) found that average per capita rates of phenotypic change for both evolving and plastic traits were always slower than average per capita changes in density, with evolution being more than ten times slower in some cases. Based on the above, these results suggest that in all of those systems, density feedbacks were stronger than density-phenotype or phenotype feedbacks. However, this conclusion seems at odds with current mathematical theory for two reasons. First, eco-evolutionary theory for predator-prey systems (Cortez and Ellner 2010; Cortez 2016; van Velzen and Gaedke 2017) predicts that the cycle periods and phase lags observed in empirical systems with prey evolution (Becks et al. 2012; 2010; Yoshida et al. 2007; 2003) can only arise if prey evolution is sufficiently fast (which implies density-phenotype and phenotype feedbacks are sufficiently strong). Second, the body of mathematical theory that deals with the analysis of models with fast and slow variables (known as fast-slow dynamical systems theory; Arnold et al. 1995; Kuehn 2015) does not define fast and slow variables in terms of average per capita rates of change. This suggests that additional work is needed to address the disagreement.

In this study, we explore how the mode of adaptation influences the relationships between density and phenotypic rates of change in empirical systems. To do this, we
develop a method for estimating and comparing rates of density and phenotypic change from empirical time series. Our method compares scaled maximum rates of change, a choice which is supported by two bodies of mathematical theory: fast-slow dynamical systems theory (Arnold et al. 1995; Kuehn 2015) and scaling theory (Logan 2013). We apply our method to thirty published empirical studies with population-level time series of densities and morphological phenotypes. For all studies, we find that the maximum phenotypic rates of change are between 0.5 and 2.5 times faster than the maximum rates of change in population densities. Moreover, the relationships between density and phenotypic rates of change did not systemically differ for evolving and plastic traits. Our results suggest that both evolution and plasticity have the potential to drive phenotypic change that is just as fast as changes in densities (even though they may not always do so). We discuss the implications of our results in terms of the feedbacks that may be driving the dynamics in these systems.

2.2 Data sets, methods and theory

2.2.1 Estimating rates of change from empirical time series

We analyzed published time series from thirty empirical studies (Table 2.1), gathered from the seventeen studies analyzed in DeLong et al. (2016), searches using Web of Science and Google Scholar, and other studies known to the authors. A study was included if it contained population-level times series for density and a morphological trait from the same population. Data was either extracted from PDFs or provided by authors of the original study. The thirty studies consist of seventeen field-based studies and thirteen constant condition laboratory experiments (FS and LE, respectively, in Table 2.1). In the field-based studies, temporal dynamics in densities and population-level phenotypes were driven by a combination of species interactions and (natural) environmental forcing. In the laboratory experiments, the communities were held under constant environmental conditions, the communities were seeded with populations not at equilibrium densities or trait distributions, and temporal dynamics were driven by processes internal to the system.
(e.g., oscillations due to predator-prey interactions); see appendix for additional details.

In most studies, the phenotype of interest was a quantitative variable (e.g., beak length) and the population-level trait was the mean phenotype for the population. In eight of the studies, phenotypes were one of two qualitative types, e.g., susceptible or resistant strains of bacteria. In these systems, the proportion of individuals with a particular phenotype, e.g., proportion resistant, was used as the population-level trait. This is appropriate, because when individuals can only have one of two phenotypes, there is a one-to-one mapping between the mean trait value and the frequency of one phenotype; see appendix for details.

The mode of phenotypic change for each trait was classified as genetic, plastic, or an unknown combination (hereafter, unknown), based on measures of heritability or descriptions of the mode of adaptation in the original studies. ‘Genetic’ means either (i) the narrow sense heritability \( h^2 \) of the trait measured from breeding experiments was estimated to be greater than 0.5 or (ii) the trait is for a clonal organism where plasticity was shown to be absent (e.g., prey expressing the undefended phenotype did not change expression in the presence of predators). ‘Plastic’ means that phenotypic plasticity was stated to be the known mode of phenotypic change (which could be shown, e.g., with reactions norms). The mode of adaptation for all other studies was classified as ‘unknown’; this category includes traits whose narrow sense heritability was estimated less than 0.5 and traits where the mode of adaptation was unknown to the authors of the original study.

To determine whether rates of phenotypic change were faster, slower, or comparable to rates of change in population density in each system, we computed rates of change for each time series using finite differences and compared their scaled maximum magnitudes. Specifically, given density values \( \{x_1, x_2, ..., x_n\} \) at time points \( \{t_1, t_2, ..., t_n\} \) and trait values \( \{z_1, z_2, ..., z_m\} \) at time points \( \{s_1, s_2, ..., s_m\} \), the scaled maximum rates of density and phenotypic change are
Here, \( \epsilon_{\text{den}} \) and \( \epsilon_{\text{ph}} \) are the unscaled magnitudes of the rates of density and phenotypic change computed using finite differences; \( R_{\text{den}} = \max \{ x_i \} - \min \{ x_i \} \) and \( R_{\text{ph}} = \max \{ z_i \} - \min \{ z_i \} \) are the ranges of the density and phenotypic time series, i.e., the differences between the minimum and maximum values; and \( G \) is the generation time of the species. For studies with multiple time series, we computed \( \epsilon_{\text{den}} \) and \( \epsilon_{\text{ph}} \) for each time series and then computed average \( \epsilon_{\text{den}} \) and \( \epsilon_{\text{ph}} \) values.

When \( \epsilon_{\text{den}} \) is larger than \( \epsilon_{\text{ph}} \) (\( \epsilon_{\text{den}} > \epsilon_{\text{ph}} \)), population density has the potential to change at rates faster than those possible for the population-level trait. Colloquially this means phenotypic change is slower than changes in densities. When \( \epsilon_{\text{den}} \) is smaller than \( \epsilon_{\text{ph}} \) (\( \epsilon_{\text{den}} < \epsilon_{\text{ph}} \)), the population-level trait has the potential to change at rates faster than those possible for population density. Colloquially this means phenotypic change is faster than changes in densities. The justifications for this interpretation and our particular choice of scaling and comparison are presented in the following sections.

### 2.2.2 Distinguishing between fast and slow rates of change

In order to compare rates of change in population densities and population-level traits, we first need a way to define and distinguish between faster and slower variables (i.e., densities or traits). The body of mathematical theory that does this is fast-slow dynamical systems theory (Arnold et al. 1995; Kuehn 2015). Below, we use that theory to show that (i) fast and slow variables are defined by maximum rates of changes, (ii) faster variables have the potential to change much faster than slower variables but do not always do so, and (iii) the time series of fast and slow variables are distinguished by faster variables having larger maximum rates of change.

Fast-slow dynamical systems are models where some variables change much faster than
others. They have been used to model plasticity (Cortez 2011) and eco-evolutionary
dynamics (Cortez and Ellner 2010; Dercole et al. 2006; Dieckmann et al. 1995; Khibnik and
Kondrashov 1997; Marrow et al. 1996). To introduce them, consider a minimal model for
changes in the density ($x$) and population-level trait ($z$) of a species,

$$\frac{dx}{dt} = \epsilon_x f(x, z)$$
$$\frac{dz}{dt} = \epsilon_z g(x, z).$$

(2.2)

The first equation defines how the population density changes due to all ecological processes,
including interactions with other unmodeled species. The second equation describes how the
population-level trait changes over time, e.g., due to selection or induction. The notation
in model (2.2) is adapted from fast-slow systems theory and has the following biological
interpretation. The values of $\epsilon_x$ and $\epsilon_z$ define the maximum magnitudes of the rates of
density and phenotypic change, respectively. The functions $f$ and $g$ define the direction
and the fraction of the maximum rate that the density or trait, respectively, is changing at.
For example, $f(x, z) = 0.5$ and $g(x, z) = -1$ mean that the species’ density is increasing at
half of its maximum rate and the population-level trait is decreasing at its maximum rate.
The magnitudes of $\epsilon_z$ and $\epsilon_x$ determine whether phenotypes have the potential to change at
rates faster than those possible densities ($\epsilon_z > \epsilon_x$) or not ($\epsilon_z < \epsilon_x$). For example, $\epsilon_z/\epsilon_x = 2$
means that the population-level trait can change up to two times faster than the maximum
rate of change in population density.

Fast-slow systems behave in the following way. The slow variable always changes at a
slow rate. In contrast, the fast variable has short periods of fast rates of change and long
periods of slow rates of change. Figure 2.1 shows an example where density dynamics are
fast and the phenotypic dynamics are much slower ($\epsilon_x = 1$ and $\epsilon_z = 0.01$). The population-
level trait ($z$) always changes at a slow rate (dashed line close to 0 in Figure 1B). In contrast,
the population density ($x$) changes at a fast rate whenever $f(x, z)$ is large in magnitude
(solid line far from zero for $t \approx 0$ in Figure 1B). However, if $f(x, z)$ is small in magnitude,
then $dx/dt$ will be small and the population density will change at a slow rate (solid line near zero for large $t$ in Figure 1B). The values of $x$ and $z$ that yield $f(x, z) = 0$ define the $x$-nullcline, which is known as the critical set in the fast-slow systems literature. When a solution to a fast-slow system is plotted in the phase portrait (Figure 1C), the fast variable has fast rates of change (double arrows) when away from the critical set (gray plane) and slow rates of change (single arrow) when near the critical set.

Importantly, because fast variables have time periods of both fast and slow rates of change, fast variables can be distinguished from slow variables only by comparing maximum rates of change. In particular, large and small maximum rates of change ($\epsilon_x$, $\epsilon_z$) define whether a variable is fast or slow, respectively. We make three points about this. First, a faster variable has the potential to change at much faster rates than a slower variable, but that does not mean that a faster variable is always changing at a fast rate. For example, prey density (the faster variable) in Figure 2.1B eventually changes as slowly as the prey trait (the slower variable). Second, a faster variable does not always change faster than a slower variable. In general, faster variables will change slower than slower variables whenever the rate of change in the faster variables is zero; this occurs whenever the variable passes through a local minimum or maximum value in the time series. For example, the phenotypic rate of change is faster at $t = 2$ in Figure 2.1B because prey density is at a trough.

Third, metrics that use different summary statistics can incorrectly classify fast variables as slow variables because fast variables spend much more time changing slowly, which ends up giving too little weight to the periods of time where fast changes are occurring. For example, the average rates of phenotypic and density change in Figure 2.1B are nearly identical despite the phenotype being one hundred times slower than the population density. Overall, this shows that in order to determine if densities or phenotypes change faster, we need to compare the magnitudes of their maximum rates of change.

### 2.2.3 Scaling to account for units and dimensions of data

Now that we have a way to define fast and slow variables, we need a fair way to compare values estimated from time series of variables that are measured in different quantities (e.g.,
Fig. 2.1: Behavior of a fast-slow eco-evolutionary predator-prey model where evolution of prey defense is slow. (A) Prey density (solid black) and mean prey defense (dashed black); predator density is not shown. (B) Rates of change for prey density (solid black) and mean defense (dashed black). The vertical dashed line denotes where rates of change in prey density transition from being faster than (left) to similar to (right) phenotypic rates of change. (C) Solution (solid black line) plotted in phase space with the critical manifold (gray). Double and single arrows denote when rates of change in population density are much faster than and similar to, respectively, rates of phenotypic change. Simulation is of the Abrams and Matsuda (1997) model; see appendix for equations and parameter values.

density versus length) and with different units (e.g., 1-2 cm versus 10-20 mm). This is done by scaling the variables such that they become dimensionless quantities; the body of mathematical theory that addresses this issue is known as scaling theory (Logan 2013). We note that scaling the species’ densities and trait values changes the magnitudes of their rates of change and that this is necessary in order to be able to compare rates of change of quantities with different dimensions in an unbiased way. Below, we present our specific scaling and explain why it is a better choice than other possible scaling choices.

We scale the densities, phenotypes, and their rates of change by the range of each time series, i.e., the difference between the maximum and minimum values. We also scale the rates by the species’ generation time, $G$, but because density and phenotypic rates for each species are scaled by the same value, our results are unchanged if a different value is used.

One key advantage of this scaling is that it removes biases caused by measuring the same quantity in different units. For example, if a species’ mean body size is modeled as $100 + 10 \sin(t)$ cm or, equivalently, $1 + 0.1 \sin(t)$ m, then the rates of change are $10 \cos(t)$ cm/yr and $0.1 \cos(t)$ m/yr, respectively. The coefficients 10 and 0.1 correspond to $\epsilon_{ph}$ in model (2.1). This is a problem because the phenotype is classified as one hundred times
faster in one case, even though the time series are identical after accounting for the different units. Dividing the rates of change by the range of the time series resolves this issue because both scaled rates become $0.5 \cos(t) \text{ yr}^{-1}$.

The second key advantage of scaling by the range is that it avoids inconsistencies that arise with other scalings. Three alternative choices are to scale rates of change by (i) the maximum value of a variable, (ii) the value of the variable, or (iii) the sample standard deviation of the time series. As an illustrative example, consider two different populations whose mean beak lengths in each year ($t$) can be modeled as $\sin(t) + 2$ cm and $\sin(t) + 10$ cm. The rates of change in mean beak length are equal for both populations (both are $\cos(t)$ cm/yr) and scaling each by the range maintains the equivalence; both scaled rates are $0.5 \cos(t) \text{ yr}^{-1}$. In contrast, scaling by the maximum value of a variable yields the unequal scaled rates $\cos(t)/3 \text{ yr}^{-1}$ and $\cos(t)/11 \text{ yr}^{-1}$. Similarly, scaling by the value of the variable, i.e., using per capita rates of change, yields the unequal scaled rates $\cos(t)/[\sin(t) + 2] \text{ yr}^{-1}$ and $\cos(t)/[\sin(t) + 10] \text{ yr}^{-1}$. Moreover, if a phenotype can take on a value of zero (e.g., spine length of zero), then the per capita rates of change can produce infinite values.

Scaling by the sample standard deviation avoids all of the above issues, but introduces additional issues. For example, consider a population whose mean beak length switches between 1 cm and 2 cm each year and a second population whose mean beak length is 2 cm every fifth year and 1 cm otherwise. When scaled by the range of the data, the scaled absolute maximum rates of change are correctly computed to be 1 for both populations. In contrast, when scaled by the standard deviation, the scaled maximum rates in the second population are always greater than the second due to the lower variation in the time series. In addition, in systems with fast and slow variables, the slow variable will often have lower variation than the fast variable because the slow variable can only change slowly. In these cases, scaling by the sample standard deviation ends up speeding up the slow variable and slowing down the fast variable, which then biases the results towards more equal rates of density and phenotypic change. In total, scaling the rates of change by the range of the variables avoids many inconsistencies introduced by alternative scalings.
2.3 Results

2.3.1 Application of method to simulated data

To demonstrate that our method can identify when population densities change at faster or slower rates than population-level traits, we applied our method to simulations of the eco-evolutionary predator-prey model in Abrams and Matsuda (1997). To do this, we first chose parameter values such that the estimated maximum rates of density and phenotypic change taken from simulated time series were equal (i.e., \( \epsilon_x \) and \( \epsilon_z \) in the model were chosen such that the values of \( \epsilon_{\text{den}} \) and \( \epsilon_{\text{ph}} \) estimated from the simulated time series were equal). We then sped up (or slowed down) the rate of phenotypic change in the model by multiplying the right hand side of the trait equation (\( \epsilon_z \)) by factors of 2, 0.5, 0.1, and 0.05, estimated the values of \( \epsilon_{\text{den}} \) and \( \epsilon_{\text{ph}} \) using our method, and compared the estimated values to the multiplicative factors set in the model. As shown in Figure 2.2A, our method captures how the phenotypic rates were sped up or slowed down. Specifically, as we decreased the maximum phenotypic rate in the model, our method showed a transition from maximum phenotypic rates being faster than (triangle; Figure 2.2B) to being equal to (asterisk; Figure 2.2C) to being slower than (x, square, and circle; Figure 2.2D) maximum rates of change in density. Additionally, the estimated maximum density rate remained unchanged, which is consistent with how the speed of the population dynamics was unaltered in the model.

We note that the estimated values of \( \epsilon_{\text{ph}} \) are not identical to the values used in the model. For example, \( \epsilon_z = 0.05\epsilon_x \) in the model for the simulation with the slowest rate of phenotypic change, but the estimated value of \( \epsilon_{\text{ph}} \) is 0.066\( \epsilon_{\text{den}} \). This difference is not surprising because each time series has a different range and is therefore scaled by a different numerical value. The ranges differ because the system is stabilized as the maximum rate of phenotypic change decreases (sustained cycles occur in Figure 2.2B while convergence to equilibrium occurs in Figure 2.2C,D). This is not an issue when applying our method to the empirical data sets because we are not comparing rates of density or phenotypic change between different empirical systems (where ranges of the variables differ). Instead, we are only comparing the rate of phenotypic change to the rate of density change for the same
empirical system.

One limitation of our method is that the estimated maximal rates of change \( (\epsilon_{\text{den}}, \epsilon_{\text{ph}}) \) depend on the sampling interval of the time series (i.e., the length of time between consecutive data points). In general, larger sampling intervals lead to scaled maximum rates of change that are close to 1. In addition, the magnitude of the effect is larger for variables with greater variation (which often means variables with faster rates of change). For example, Figure 2.2F-H shows how increasing the sampling interval affects the time series in panels B-D. For all of the time series in Figure 2.2A, increases in the sampling interval move the estimated scaled maximum rates of change closer to the one-to-one line (light gray shapes in Figure 2.2 are closer to one-to-one line than black shapes). The main implication for our results is that if the sampling is too sparse, our results will be biased towards concluding that the scaled maximum rates of density and phenotypic change are more similar than in reality.

2.3.2 Application of method to empirical time series

We applied our method for estimating maximum rates of density and phenotypic change to the 30 published studies (Table 2.1); see Table A.1 and the supplementary material for additional details. Across all studies, we found that the maximum rate of phenotypic change was between 0.5 and 2.5 times faster than the maximum rate of density change; the mean and median were 1.2 and the lower and upper quartiles were 0.85 and 1.5, respectively. Additionally, in only three studies did the maximum scaled rates differ by more than a factor of two (three studies above upper dashed lines in Figure 2.3A,B). The mode of adaptation did not affect the relationship between the maximum rates (Figure 2.3B); rates of phenotypic change were faster than rates of density change in approximately half of the studies in each category. Overall, our results suggest that population densities and population-level phenotypes could change at similar rates in all systems and that the mode of adaptation did not have an effect on the relationship between rates of density and phenotypic change.

Sparse sampling can potentially bias our results toward estimating equal maximum scaled rates of density and phenotypic change. Many of the study systems satisfy conditions
that suggest undersampling is less likely to be an issue; see appendix for additional details. For 17 of the 30 studies, the average sampling interval is less than or equal to the species’ generation time (Figure A.2). This suggests that for those 17 studies, undersampling is unlikely to bias our results. In 6 of the 13 studies where the sampling interval is greater than the species’ generation time, the adapting species is the prey species in an oscillating predator-prey system. Because the predator generation time imposes limits on the cycle period and the sampling interval is less than the predator generation time, undersampling is unlikely to bias our results for those 6 studies. Undersampling may be an issue for 3 studies with genetic traits (Bohannon and Lenski 1998; Sanchez and Gore 2013; Schrag and Mittler 1996) and 4 studies with plastic traits (Caron et al. 1985; DeLong et al. 2014; Gonzalez et al. 1993; Suzuki et al. 2017) where the sampling interval is greater than the species’ generation time.

2.4 Discussion

In this study, we compared scaled maximum rates of density and phenotypic change in thirty empirical studies and found that phenotypic rates range from 0.5 to 2.5 times faster than rates of change in population density. Moreover, we found that the relationship between rates of density and phenotypic change did not differ for systems with plastic versus evolving morphological traits. Overall, our results show that plasticity and evolution have the potential to drive changes in morphological traits that are just as fast as changes in population densities. On the one hand, our general findings are not surprising given that there are many empirical studies documenting evolution occurring on ecological time scales (e.g., Carrol et al. 2007; Fussman et al. 2007; Pelletier et al. 2009) and plastic responses in individuals can occur within hours or weeks (e.g., Auld and Relyea 2011; Kuhlmann and Heckmann 1985; Kusch 1993). On the other hand, our results are somewhat unexpected because evolutionary change necessarily requires multiple generations whereas all of the plastic responses in the studies we considered occur within the lifetime of the organism. Our results suggest that this difference did not have a strong effect on the relationships between population-level rates of change in densities and phenotypes for the studies we
analyzed.

Our main conclusion is that plasticity and evolution have the potential to drive population-level changes in morphological traits that are just as fast as changes in population densities. Our use of the word ‘potential’ is intentional, because our results do not mean that rates of density and phenotypic change are always comparable. Rapid phenotypic change occurs for evolving traits when there is strong selective pressure and high genetic variation and for plastic traits when rates of induction or loss of induction are high. Rapid changes in densities can occur when a species’ interactions with its environment (e.g., intraspecific and interspecific interactions with other species) are strong. In general, we do not expect the conditions for rapid phenotypic change and rapid changes in density to align in time. This means that we expect to observe periods of time where rates of phenotypic change are greater than rates of change in densities, and vice versa. Thus, overall our results support the claim that phenotypic change can be just as fast as evolutionary change.

One caveat to our conclusions is that we only focused on plastic morphological phenotypes that change within the lifetime of the organism. Our conclusions may extend to transgenerational plastic responses, wherein an offspring’s phenotype is determined by the parent’s environment (Agrawal et al. 1999; Shimada et al. 2010). This is because the generational lag in such plastic responses is similar to the lag for evolutionary responses. However, we do not expect that our results will extend to all behavioral responses. Behavioral plastic responses can occur over periods of time much shorter than the generation time of the organism (Tollrian and Harvell 1999). For such traits, it seems virtually guaranteed that population-level changes in phenotypes can be much faster than changes in population densities.

Our results differ markedly from DeLong et al. (2016), who found that evolutionary or plastic phenotypic change was no more than two-thirds as fast as changes in population densities. Our method and the method in DeLong et al. (2016) use different scalings (range of the data versus per capita rates) and compare different values (maximum rates versus average rates). Our comparison of these two approaches in Figure A.3 shows that the
difference in scaling is the key factor causing the differing results. Specifically, our results are nearly identical if we compare average rates of change scaled by the range of the time series. Our results are also nearly identical if we compare maximum or average rates of change scaled by the sample standard deviation of the time series; see Figure A.3. We have argued that our method is better than the alternatives because it is consistent with fast-slow dynamical systems theory and scaling theory and because it avoids inconsistencies that arise with other methods. Our comparison of methods suggests that the issues associated with using average rates of change or scaling by the sample standard deviation are not present in our data sets. However, it also suggests that the issues associated with using per capita rates are. This is not surprising since the trait values and densities take on small values in many of the studies, which can lead to errors when computing per capita rates. In total, while other methods can yield similar results, we recommend comparing maximum rates of change that are scaled by the range of the data in order to avoid potential errors.

Finally, our results suggest that multiple types of feedbacks were likely important for driving the dynamics in the systems we considered. The strengths of density feedbacks (i.e., feedbacks between population densities), phenotype feedbacks (i.e., feedbacks between species traits), and density-phenotype feedbacks (i.e., feedbacks between population densities and species traits) depend on the coupling between the variables involved in the feedback and by how fast the variables can change ($\epsilon_{den}$ and $\epsilon_{ph}$ in our study) (Cortez 2018; Patel et al. 2018). Because we focused on studies with changes in ecologically relevant traits, the coupling between densities and traits was likely high. Given this sufficient coupling, the similarity in density and phenotypic rates of change suggests that density, phenotype, and density-phenotype feedbacks were roughly equally strong in all systems. This relates to previous empirical (Becks et al. 2010; Steiner and Masse 2013; Verschoor et al. 2004; Yoshida et al. 2007; 2003) and theoretical (Cortez 2011) studies that suggest plasticity and evolution have different effects on the population dynamics of predator-prey systems. If our results about rates of density and phenotypic change are generally true for predator-prey systems, then this would suggest that dynamical differences between predator-prey systems
with plastic and evolving traits might not be due to which feedbacks are driving the dynamics, but instead due to differences between the properties of the feedbacks involving plastic versus evolving traits (e.g., stabilizing versus destabilizing feedbacks).

More generally, it is currently not known if and how the feedbacks involving plastic versus evolving traits differ. This points to a need for new theory that directly compares the feedbacks between evolving traits and population densities with the feedbacks between plastic traits and densities. Much of the current theoretical work on systems with evolving or plastic traits assumes phenotypic rates of change are either much slower (e.g., Dercole et al. 2006; Marrow et al. 1996; Patel et al. 2018) or much faster (e.g., Cortez and Ellner 2010; Cortez 2011; Patel et al. 2018) than density rates of change. In these systems, density feedbacks are much stronger than phenotype feedbacks, or vice versa. However, our results suggest that density feedbacks and phenotype feedbacks may be equally strong in many systems. Recent eco-evolutionary theory (Cortez 2016; 2018; Patel et al. 2018) shows that theory assuming very fast or very slow rates of phenotypic change may not accurately predict the dynamics of systems with comparable rates of density and phenotypic change. Thus, the results in our study point to the need for new theory that explains how feedbacks between densities and traits affect the dynamics of natural communities with commensurate rates of density and phenotypic change.

### 2.5 Acknowledgments

We thank L Becks, E Edeline, SP Ellner, M Kasada, and A Sanchez for sharing empirical data sets with us. GG received support from the National Science Foundation under Grant No. 1633756.
Table 2.1: Estimated scaled maximum rates of density and phenotypic change from empirical studies

<table>
<thead>
<tr>
<th>Code</th>
<th>Study</th>
<th>Type</th>
<th>Species</th>
<th>Trait</th>
<th>$\epsilon_{\text{den}}$</th>
<th>$\epsilon_{\text{ph}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Becks et al. (2010)</td>
<td>LE</td>
<td>Chlamydomonas reinhardtii</td>
<td>Clump size</td>
<td>0.37</td>
<td>0.63</td>
</tr>
<tr>
<td>b</td>
<td>Becks et al. (2012)</td>
<td>LE</td>
<td>Chlamydomonas reinhardtii</td>
<td>Clump size</td>
<td>2.5</td>
<td>4.8</td>
</tr>
<tr>
<td>c</td>
<td>Bohannon and Lenski (1998)</td>
<td>LE</td>
<td>Escherichia coli</td>
<td>Proportion susceptible</td>
<td>0.0097</td>
<td>0.012</td>
</tr>
<tr>
<td>d</td>
<td>Coltman et al. (2003)</td>
<td>FS</td>
<td>Ovis canadensis</td>
<td>Horn length (adult male)</td>
<td>1.8</td>
<td>2.4</td>
</tr>
<tr>
<td>e</td>
<td>Fussman et al. (2003)</td>
<td>LE</td>
<td>Brachionus calyciflorus</td>
<td>Proportion mictic (female)</td>
<td>0.97</td>
<td>1.8</td>
</tr>
<tr>
<td>f</td>
<td>Grant and Grant (2002)</td>
<td>FS</td>
<td>Geospiza foris</td>
<td>Beak length</td>
<td>2.9</td>
<td>2.1</td>
</tr>
<tr>
<td>g</td>
<td>Grant and Grant (2002)</td>
<td>FS</td>
<td>Geospiza scandens</td>
<td>Beak shape</td>
<td>2.9</td>
<td>1.9</td>
</tr>
<tr>
<td>h</td>
<td>Grant and Grant (2002)</td>
<td>FS</td>
<td>Geospiza scandens</td>
<td>Beak shape</td>
<td>2.1</td>
<td>1.1</td>
</tr>
<tr>
<td>i</td>
<td>Haafe et al. (2016)</td>
<td>LE</td>
<td>Chlamydomonas reinhardtii</td>
<td>Clump size</td>
<td>0.055</td>
<td>0.076</td>
</tr>
<tr>
<td>k</td>
<td>Hiltunen and Becks (2014)</td>
<td>LE</td>
<td>Pseudomonas flourescens</td>
<td>Proportion defended</td>
<td>0.43</td>
<td>0.23</td>
</tr>
<tr>
<td>l</td>
<td>Hiltunen et al. (2014)</td>
<td>LE</td>
<td>Chlorella autotrophica</td>
<td>Clump size</td>
<td>0.16</td>
<td>0.38</td>
</tr>
<tr>
<td>m</td>
<td>Kasada et al. (2014)</td>
<td>LE</td>
<td>Chlorella vulgaris</td>
<td>Proportion defended (costly tradeoff)</td>
<td>0.43</td>
<td>0.43</td>
</tr>
<tr>
<td>n</td>
<td>LE</td>
<td></td>
<td></td>
<td>Proportion defended (cheap tradeoff)</td>
<td>0.82</td>
<td>0.57</td>
</tr>
<tr>
<td>o</td>
<td>Milot et al. (2011)</td>
<td>FS</td>
<td>Homo sapiens</td>
<td>Age at first reproduction</td>
<td>0.58</td>
<td>0.77</td>
</tr>
<tr>
<td>p</td>
<td>Robinson et al. (2006)</td>
<td>FS</td>
<td>Ovis aries</td>
<td>Proportion normal horned</td>
<td>2.7</td>
<td>6.1</td>
</tr>
<tr>
<td>q</td>
<td>Sanchez and Gore (2013)</td>
<td>LE</td>
<td>Saccharomyces cerevisiae</td>
<td>Proportion cooperator</td>
<td>0.039</td>
<td>0.081</td>
</tr>
<tr>
<td>r</td>
<td>Schrag and Mittler (1996)</td>
<td>LE</td>
<td>Escherichia coli</td>
<td>Proportion resistant</td>
<td>0.28</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td>Plastic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Campbell and Echternacht (2003)</td>
<td>FS</td>
<td>Anolis sagrei</td>
<td>Snout-vent length</td>
<td>0.51</td>
<td>0.78</td>
</tr>
<tr>
<td>B</td>
<td>Caron et al. (1985)</td>
<td>LE</td>
<td>Paraphysomonas imperforata</td>
<td>Cell volume</td>
<td>0.22</td>
<td>0.21</td>
</tr>
<tr>
<td>C</td>
<td>DeLong et al. (2014)</td>
<td>LE</td>
<td>Didinium</td>
<td>Cell volume</td>
<td>0.11</td>
<td>0.14</td>
</tr>
<tr>
<td>D</td>
<td>Fenchel and Jonsson (1988)</td>
<td>LE</td>
<td>Strombidium sulcatum</td>
<td>Cell volume</td>
<td>1.7</td>
<td>2.5</td>
</tr>
<tr>
<td>E</td>
<td>Gonzalez et al. (1993)</td>
<td>LE</td>
<td>Cafeteria sp.</td>
<td>Cell volume</td>
<td>0.26</td>
<td>0.22</td>
</tr>
<tr>
<td>F</td>
<td>Lüring et al. (2005)</td>
<td>LE</td>
<td>Scenedesmus obliquus</td>
<td>Clump size</td>
<td>0.53</td>
<td>0.39</td>
</tr>
<tr>
<td>G</td>
<td>Ozgul et al. (2010)</td>
<td>FS</td>
<td>Marmota flaviventris</td>
<td>Weight</td>
<td>1.7</td>
<td>3.1</td>
</tr>
<tr>
<td>H</td>
<td>Suzuki et al. (2017)</td>
<td>LE</td>
<td>Flectobacillus</td>
<td>Cell volume</td>
<td>0.023</td>
<td>0.017</td>
</tr>
<tr>
<td></td>
<td>Unknown</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>α</td>
<td>Brown and Brown (2013)</td>
<td>FS</td>
<td>Petrochelidon pyrhonotanest</td>
<td>Wing length</td>
<td>0.69</td>
<td>0.59</td>
</tr>
<tr>
<td>β</td>
<td>Coltman et al. (2003)</td>
<td>FS</td>
<td>Ovis canadensis</td>
<td>Weight (adult male)</td>
<td>1.8</td>
<td>2.8</td>
</tr>
<tr>
<td>γ</td>
<td>Edeline et al. (2008)</td>
<td>FS</td>
<td>Perca fluviatilis</td>
<td>Body length</td>
<td>2.3</td>
<td>3.6</td>
</tr>
<tr>
<td>δ</td>
<td>Ezard et al. (2009)</td>
<td>FS</td>
<td>Ovis aries</td>
<td>Weight</td>
<td>4.3</td>
<td>5.6</td>
</tr>
<tr>
<td>η</td>
<td>Fischer et al. (2014)</td>
<td>LE</td>
<td>Chlamydomonas reinhardtii</td>
<td>Proportion strain 1</td>
<td>0.22</td>
<td>0.27</td>
</tr>
<tr>
<td>θ</td>
<td>Nakazawa et al. (2007)</td>
<td>FS</td>
<td>Gymnogobius isaza</td>
<td>Body length</td>
<td>0.72</td>
<td>0.69</td>
</tr>
<tr>
<td>λ</td>
<td>FS</td>
<td></td>
<td></td>
<td>Wet weight</td>
<td>0.72</td>
<td>0.79</td>
</tr>
<tr>
<td>μ</td>
<td>Schoener et al. (2002)</td>
<td>FS</td>
<td>Anolis sagrei</td>
<td>Hindlimb length</td>
<td>0.81</td>
<td>0.69</td>
</tr>
<tr>
<td>ξ</td>
<td>FS</td>
<td></td>
<td></td>
<td>Lamellae number</td>
<td>0.81</td>
<td>0.97</td>
</tr>
<tr>
<td>ρ</td>
<td>Sinervo et al. (2000)</td>
<td>FS</td>
<td>Uta stansburiana</td>
<td>Clutch size</td>
<td>0.62</td>
<td>0.72</td>
</tr>
<tr>
<td>σ</td>
<td>FS</td>
<td></td>
<td></td>
<td>Egg mass</td>
<td>0.62</td>
<td>0.87</td>
</tr>
<tr>
<td>τ</td>
<td>FS</td>
<td></td>
<td></td>
<td>Proportion orange-throated (female)</td>
<td>0.61</td>
<td>0.96</td>
</tr>
<tr>
<td>φ</td>
<td>Swain et al. (2007)</td>
<td>FS</td>
<td>Gadus morhua</td>
<td>Body length at age 6 years</td>
<td>1.5</td>
<td>1.4</td>
</tr>
<tr>
<td>ψ</td>
<td>Tessier et al. (1992)</td>
<td>FS</td>
<td>Daphnia galeata mendotae</td>
<td>Body size</td>
<td>3.7</td>
<td>3</td>
</tr>
</tbody>
</table>

Mode of adaptation is either genetic (a−r), plastic (A−H), or unknown (α − ψ). Study type is either field-based study (FS) or laboratory experiment under controlled conditions (LE); see text for details. $\epsilon_{\text{den}}$ and $\epsilon_{\text{ph}}$ are the scaled maximum rates of density and phenotypic change, respectively.
Fig. 2.2: With sufficient sampling, our method can accurately estimate maximum rates of density and phenotypic change from simulated time series generated. Time series were generated using the eco-evolutionary predator-prey model in Abrams and Matsuda (1997), where the rate of evolution was sped up or slowed down by factors of 2, 1, 0.5, 0.1, and 0.05. (A) Estimated scaled maximum rates of density and phenotypic change for different speeds of evolution. Diagonal lines denote when maximum rates of phenotypic change are ten times faster (upper dashed line), equal to (solid line), and ten times slower (lower dashed line) than maximum rates of change in population densities. Examples of scaled prey density (solid black) and mean trait (dashed black) time series when (B) $\epsilon_{ph} \approx 2\epsilon_{den}$, (C) $\epsilon_{ph} \approx \epsilon_{den}$, and (D) $\epsilon_{ph} \approx 0.1\epsilon_{den}$. (E) Estimated scaled maximum rates of density and phenotypic change for different speeds of evolution and sampling intervals. Sampling is every 1 (black shapes), 4, 10, 20, 40, 100 or 200 (lightest gray shapes) time units. (F-H) Demonstration of how increasing the sampling interval to every 40 time steps alters the time series in panels B-D; numerical values differ between pairs of panels because the sampling interval affects the range of the time series and consequently, how each time series is scaled. See appendix for model equations and parameter values.
Fig. 2.3: Maximum rates of density and phenotypic change are similar in all 30 empirical studies. (A) Estimated scaled maximum rates with standard error bars; see Table 1 for legend. (B) Ratio of the maximum rate of phenotypic change to the maximum rate of density change. Red, blue, and brown correspond to studies where the mode of adaptation is genetic, plastic, or unknown, respectively. In both panels, lines denote when maximum phenotypic rates are two times faster (upper dashed line), equal to (solid line), and two times slower (lower dashed line) than maximum density rates.
CHAPTER 3
CHAOS DOES NOT DRIVE LOWER SYNCHRONY FOR INTRINSICALLY-INDUCED POPULATION FLUCTUATIONS

Abstract

Synchrony among connected populations is an important indicator of extinction risk. Populations fluctuating in the same direction within a network are more prone to extinction, especially when population sizes are small. The qualitative behavior of fluctuations may be an important indicator for predicting synchrony. Specifically, chaotic oscillations have been found to amplify noise, thus lowering population synchrony; however, other oscillation types (invariant cycles, k-cycles) have not been explicitly explored in synchrony-stability research. In this paper, we investigate the relationship between synchrony and oscillations where we classify oscillations into three types: invariant cycles, k-cycles, and chaos. We use a mechanistic two-patch metapopulation model of a species with 1, 2, and 3 life-history stages. Relative dispersal between the two patches are categorized into zones based on dispersal strength and symmetry. We use bifurcation and eigenvalue analysis with multiple initial conditions to determine the mechanisms that induce oscillations while identifying alternate fixed points and asymptotic behaviors. We find that low dispersal in the 1-stage model generally produce k-cycles mixed with chaotic oscillations that experience increases in synchrony until flip bifurcations stabilize fluctuations to equilibria. In the 2- and 3-stage models, invariant cycles occur under low dispersal that once again experience increases in synchrony until Hopf bifurcations stabilize the fluctuations to equilibria. Under high dispersal, equilibrium points may destabilize while experiencing significantly lower population synchrony under Hopf bifurcations in the 1-stage model and flip bifurcations in the 2-stage model. Additionally, we find that chaos does not have noticeably lower synchrony than other attractor types but has synchrony levels similar to the oscillation types that precede chaos.
These results show that we may not expect chaos to create lower population synchrony; rather, sudden decreases in synchrony may occur with high dispersal under bifurcations that are different from those that occur in the low dispersal regions.
3.1 Introduction

Many species live in isolated habitats as part of a spatially structured population, or metapopulation. In these systems, the habitats, or patches, experience demographic processes (fecundity, growth, etc.) independent of proximal patches while being connected by dispersal. The consequences of these interactions vary and can range from inducing rescue effects in low quality patches to increasing global extinction risk through hyper-synchronization (Abbott 2011; Hudson and Cattadori 1999; Ylikarjula et al. 2000). Additionally, dispersal has different effects on population dynamics, and can shift steady states from equilibria to oscillations, or vice-versa (Abbott 2011; Doebeli 1995; Gyllenberg et al. 1992; Hastings 1993; Ives et al. 2003; Wang et al. 2015; Wang and Loreau 2014; Ylikarjula et al. 2000). Understanding how movement between patches affects metapopulations provides insight into the mechanisms driving population dynamics and can be used to assess population viability and extinction risk across the metapopulation.

Stability, in the general sense, may refer to equilibrium dynamics, low-amplitude cycles, and populations sufficiently large enough to sustain themselves, whereas instability may refer to high-amplitude cycles and small populations at risk of extinction (Abbott 2011). Previous studies have shown that dispersal generally has a stabilizing effect on population dynamics, shifting asymptotic behavior from oscillations to equilibria or decreasing population variance (Abbott 2011; Davis 1992; Doebeli 1995; Doebeli and Ruxton 1998; Gyllenberg et al. 1992; Hastings 1993; Hellriegel 2000; Ives et al. 2003; Tromeur et al. 2016; Wang et al. 2015; Wang and Loreau 2014; Ylikarjula et al. 2000). However, there exist counter-examples where dispersal induces oscillations or has a U-shaped relationship with population variance (Doebeli 1995; Gyllenberg et al. 1992; Hastings 1993; Jang and Mitra 2000; Silva et al. 2001; Tromeur et al. 2016; Ylikarjula et al. 2000). The effects of dispersal on metapopulation dynamics depend on the type of dispersal (e.g., density-dependent, constant), the number of patches, and the degree to which populations are connected (Ylikarjula et al. 2000). For example, asymmetric dispersal (i.e., unequal dispersal between patches) is typically more stabilizing than symmetric dispersal (Dey et al. 2014; Doebeli 1995; Ylikarjula et al. 2000),
though very high asymmetric dispersal may induce oscillations (Ylikarjula et al. 2000). This means that oscillations tend to persist when the number of dispersing individuals is similar for all patches whereas populations at equilibrium may shift to oscillations if the number of dispersing individuals between patches is very different. Other factors that affect population dynamics are local demography (e.g., survival probabilities and fecundity rates) (Abbott 2011; Udwadia and Raju 1998), and dispersal strategies (e.g., density-dependent, constant, and dispersal symmetry) (Hellriegel 2000; Ylikarjula et al. 2000).

Limit cycles are isolated closed trajectories that occur in nonlinear systems (Strogatz 2015). A limit cycle is stable if all nearby trajectories converge to the limit cycle, unstable if all nearby trajectories are repelled from the limit cycle, or half-stable if a nearby trajectory approaches the limit cycle as time goes to positive infinity and another trajectory approaches the limit cycle as time goes to negative infinity (Strogatz 2015). We classify stable limit cycles into three types based on their qualitative behavior: invariant cycles, k-cycles, and chaos (Caswell 2001). Invariant cycles have quasi-periodic dynamics, characterized by iterative cycles around a smooth curve with slightly shifting rotations around a closed loop. K-cycles are classified by the number of points that occur in each period. Chaos is deterministic but has seemingly random dynamics that are sensitive to initial conditions. Attractors, which we define as stable equilibria and stable limit cycles, are classified using the Lyapunov exponent which averages over the distance between two nearby trajectories in time (Caswell 2001).

The onset of chaos has important implications for understanding the nature of population dynamics under parameter perturbations. For our purposes, we define chaos as aperiodic oscillations with sensitivity to initial conditions, the latter of which is identified by a positive Lyapunov exponent. Chaos may arise under a number of circumstances, typically occurring through a series of bifurcations. Two common routes to chaos are period-doubling k-cycles where a flip bifurcation creates k-cycles that lead to chaos, and frequency-locking (or phase-locking) invariant cycles marked by the development of cusps or ‘wrinkles’ in the phase plane, creating sensitivity to initial conditions due to mixed trajectories (Caswell
Chaos may also suddenly be created or destroyed in a crisis when a stable cycle collides with an unstable fixed point or periodic orbit (Grebogi et al. 1983; Ugarcovici and Weiss 2004). This may be further classified as a boundary crisis (collision of an unstable orbit with the chaotic attractor on the boundary of the basin of attraction), interior crisis (collision of an unstable orbit with the chaotic attractor within the basin of attraction), or merging crisis (collision of two chaotic attractors to form one chaotic attractor) (Ugarcovici and Weiss 2004).

In addition to oscillation type, dispersal has a variety of effects on population synchrony. Synchrony refers to the degree to which populations fluctuate in the same direction such that synchronous populations fluctuate in the same direction and asynchronous populations fluctuate in opposite directions. Two well-known drivers of population synchrony are dispersal and environmental variability (Abbott 2011; Hudson and Cattadori 1999). In other words, patches within a network may fluctuate in the same direction when the connection between them is strong, or, for less connected systems, fluctuations may be driven by changes in the environment, i.e., the Moran effect (Moran 1953). Because the two drivers are not mutually exclusive, identifying the mechanisms that drive synchrony can be difficult, especially in empirical studies (Hudson and Cattadori 1999). Therefore, theoretical models are used to isolate dispersal and environmental variables to understand their independent effects on synchrony.

There are two main methods for calculating synchrony; 1) temporal correlations between local populations (Heino et al. 1997; Ranta et al. 1998; Ylikarjula et al. 2000), and 2) the ratio of the summed temporal covariances between populations to the summed variances within populations (Loreau and Mazancourt 2008; Thibaut and Conolly 2013; Wang et al. 2015; Wang and Loreau 2014). Loreau and Mazancourt (2008) argue that using temporal correlations fail when applied to multiple (greater than two) species or communities. Specifically, with two species, this method is bounded from -1 to 1, but for a large number of species, this method averages over pairwise correlations, bounding the measure from 0 to 1 (Loreau and Mazancourt 2008). On the other hand, the covariance matrix method cre-
ates a standardized measure between 0 and 1 regardless of community size. Therefore, the
covariance matrix method has been favored over the correlation method in recent studies
(Thibaut and Conolly 2013; Wang et al. 2015; Wang and Loreau 2014).

Many studies have focused on the relationship between synchrony and stability where
fluctuations are quantified by population variance (Abbott 2011; Allen et al. 1993; Jansen
1999; Wang et al. 2015); however, these studies do not explicitly analyze synchrony in
relation to oscillation type. Generally, only chaos is studied because of its tendency to occur
in low synchrony populations (Allen et al. 1993). Chaotic oscillations with low synchrony
have been observed under low dispersal probabilities in stochastic systems (Allen et al. 1993;
Heino et al. 1997; Ranta et al. 1998). However, other studies have observed high synchrony
chaos under intermediate levels of dispersal in deterministic systems (Udwadia and Raju
1998; Ylikarjula et al. 2000). Understanding synchrony in relation to chaos and the other
oscillation types can elucidate some of the mechanisms that affect synchrony in fluctuating
populations.

In this paper, we explore the effects of dispersal on the dynamics of a two-patch model
of a species with 1, 2, or 3 life-history stages. We investigate the relationship between
attractor type, synchrony, and dispersal while identifying mechanisms driving oscillations
using bifurcation and eigenvalue analysis. Our models are parameterized using amphibian
literature with discrete life stage development, high fecundity rates, and low larval survival
(McCaffery et al. 2014; Vonesh and De la Cruz 2002; Willson and Hopkins 2013). Dispersal
may occur in any life stage, and for simplicity, we assume constant dispersal of individuals
in one life stage while keeping dispersal of individuals in all other life stages at zero. We first
analyze stability of single-patch dynamics using eigenvalue analysis, and classify attractor
types of oscillating populations using the Lyapunov exponent. By combining bifurcation,
Lyapunov exponent, and eigenvalue plots, we identify dynamical shifts and routes to chaos
under a range of different survival parameters. Next, we investigate the effects of dispersal
on attractor type and synchrony in two-patch models of a species with 1, 2, and 3 life-history
stages. We use two measures of dispersal to quantify dispersal magnitudes and symmetries.
Additionally, we use bifurcation and eigenvalue plots of numerically solved fixed points to determine shifts in dynamics and routes to chaos. Our results find that chaotic attractors do not have consistently low synchrony but have synchrony levels similar to the attractors preceding chaos. We also identify a number of different bifurcations including saddle-node, pitchfork, transcritical, Hopf, and flip bifurcations as well as a variety of routes to chaos including period-doubling, phase-locking, and crisis. Additionally, multiple stable and unstable fixed points occur in the two-patch model along with multiple stable steady states induced by dispersal. This work offers a new perspective on synchrony-stability research by using bifurcation, eigenvalue, and Lyapunov exponent analyses, and demonstrates some of the methods that can be used to analyze metapopulation systems with multiple coexisting steady states.

3.2 Mathematical model formulation

3.2.1 The two-patch model

Given a metapopulation with 2 patches and \( n \) life stages, we define a patch-based population vector \( \mathbf{u}_n(t) \in \mathbb{R}^{2n} \) as

\[
\mathbf{u}_n(t) = \begin{bmatrix} u_{1n}(t) \\ u_{2n}(t) \end{bmatrix}, \tag{3.1}
\]

where \( u_{xn}(t) \in \mathbb{R}^n, x = 1, 2, \) is a subvector of length \( n \) and represents population distributions for \( n \) life stages in patch \( x \) at time \( t \). For each of the 3 life stage models, we have
\[ u_{x1} = \begin{bmatrix} a_x \end{bmatrix}, \text{ when } n = 1, \quad (3.2a) \]

\[ u_{x2} = \begin{bmatrix} j_x \\ a_x \end{bmatrix}, \text{ when } n = 2, \quad (3.2b) \]

\[ u_{x3} = \begin{bmatrix} j_x \\ s_x \\ a_x \end{bmatrix}, \text{ when } n = 3, \quad (3.2c) \]

where \( j_x, s_x, \) and \( a_x \) are the juvenile, subadult, and adult densities in patch \( x \).

We define a two-patch, \( n \)-stage model where each patch experiences demographic processes independent of the other patch and interactions between them occur through a dispersal mechanism. The projection matrix \( A_n \in \mathbb{R}^{2n,2n} \) is defined as

\[
A_n = \begin{bmatrix}
(1 - \sigma_{1n})B_{1n} & \sigma_{2n}B_{2n} \\
\sigma_{1n}B_{1n} & (1 - \sigma_{2n})B_{2n}
\end{bmatrix},
\]

(3.3)

where \( \sigma_{xn} \in \mathbb{R}^{n,n} \) is a diagonal matrix with entries that represent dispersal probabilities for \( n \) life stages from patch \( x = 1, 2 \), and \( B_{xn} \in \mathbb{R}^{n,n} \) is the demographic matrix for patch \( x = 1, 2 \). The diagonal submatrices in \( A_n \), \( (1 - \sigma_{xn})B_{xn} \), represent demographic processes of individuals who remain in patch \( x \) while the off-diagonal submatrices, \( \sigma_{xn}B_{xn} \), represent demographic and dispersal processes of individuals dispersing from patch \( x \).

With the notations above, the two-patch model with multiple life stages is expressed as

\[ u_n(t + 1) = A_n u_n(t). \quad (3.4) \]

Note that demographic processes (e.g., fecundity, survival) occur prior to dispersal processes.
3.2.2 Local patch demographics

Local patch dynamics without dispersal are used to understand how each patch behaves in isolation. For an $n$-stage model, local populations are projected using

$$
u_{xn}(t + 1) = B_{xn} u_{xn}(t), \quad x = 1, 2. \tag{3.5}$$

Assuming local patch dynamics are identical, patch indices are omitted and (3.5) simplifies to

$$u_n(t + 1) = B_n u_n(t). \tag{3.6}$$

For a single-stage population, new recruits mature into breeding adults ($a$) after surviving their first year. The demographic matrix is defined as

$$B_1 = \begin{bmatrix} S_a + FH_a \end{bmatrix}, \tag{3.7}$$

with corresponding population vector $u_1$ defined in (3.2a), $S_a$ is the survival probability of adults, $F$ is the fecundity rate, and $H_a$ is the density-dependent first-year survival (see section 3.2.2). The fecundity term includes adult breeding probability, clutch size, and egg survival.

For a 2-stage population, new recruits grow into juveniles ($j$) in their first year then have some probability of transitioning to breeding adults in subsequent years. The demographic matrix is defined as

$$B_2 = \begin{bmatrix} S_j (1 - P_{ja}) & FH_a \\ S_j P_{ja} & S_a \end{bmatrix}, \tag{3.8}$$

with corresponding population vector $u_2$ defined in (3.2b), $P_{ja}$ is the probability of transitioning from the juvenile to adult stage, and $S_j$ is the survival probability of juveniles.

Finally, a 3-stage population assumes an intermediate non-breeding stage (subadults; $s$) with survival probabilities similar to adults. We assume that individuals may only transition
to latter stages in single-stage increments. The demographic matrix is defined as

\[
B_3 = \begin{bmatrix}
S_j(1 - P_{js}) & 0 & FH_a \\
S_jP_{js} & S_s(1 - P_{sa}) & 0 \\
0 & S_sP_{sa} & S_a
\end{bmatrix},
\]

(3.9)

with corresponding population vector \( u_3 \) defined in (3.2c), \( P_{js} \) and \( P_{sa} \) are transition probabilities from juvenile to subadult and subadult to adult, and \( S_s \) is the survival probability of subadults.

**Nonlinear larval survival**

In each patch, we assume negative density-dependent survival in first-year individuals to represent, for example, larval competition. We use the Hassell competition model to define first-year survival based on the number of breeding adults in the same year (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012),

\[
H_a = \frac{L}{(1 + \frac{F_a}{K})^\gamma},
\]

(3.10)

where \( L \) is the maximum first-year survival, \( K \) is the carrying capacity, and \( \gamma \) is the density-dependent exponent. Other forms of density-dependent survival have been used in amphibian models including the Ricker model (Hellriegel 2000) and Gompertz model (Bendik and Dries 2018; Băncilă et al. 2016). We choose the Hassell competition model because of its realism and use in multiple amphibian studies (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012).

Competition in the Hassell equation depends on the density-dependent exponent, \( \gamma \), where \( \gamma \) is inversely proportional to patch size (Anazawa 2019). For smaller values of \( \gamma \), individuals undergo ‘contest’ competition while for large values of \( \gamma \), individuals undergo ‘scramble’ competition (Hassell 1975). In contest competition, some proportion of the individuals receive sufficient resources for survival while the rest do not survive. In scramble competition, all resources are ‘shared’ such that individuals either all die out or all survive.
For extreme cases, ideal contest competition ($\gamma = 1$) defines the Ricker model while ideal scramble competition ($\gamma \to \infty$), defines the Beverton-Holt model (Anazawa 2019).

To make this paper more accessible to the readers, we summarize some of the symbols used in this paper in Table 3.1.

Table 3.1: Symbols used in the model and analysis.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of life stages</td>
</tr>
<tr>
<td>$u_n$</td>
<td>two-patch population vector for $n$ stages</td>
</tr>
<tr>
<td>$u_{xn}$</td>
<td>$n$-stage population vector for patch $x$</td>
</tr>
<tr>
<td>$U_x$</td>
<td>population size of patch $x$</td>
</tr>
<tr>
<td>$A_n$</td>
<td>$2n \times 2n$ projection matrix</td>
</tr>
<tr>
<td>$B_{xn}$</td>
<td>$n$-stage demographic matrix for patch $x$</td>
</tr>
<tr>
<td>$\sigma_{xn}$</td>
<td>$n \times n$ diagonal matrix of dispersal probabilities for patch $x$</td>
</tr>
<tr>
<td>$J_n$</td>
<td>Jacobian matrix for $n$-stage demographic matrix</td>
</tr>
<tr>
<td>$\bar{J}_n$</td>
<td>two-patch Jacobian matrix for $n$-stages</td>
</tr>
<tr>
<td>$H_a$</td>
<td>density-dependent first-year survival</td>
</tr>
<tr>
<td>$R_H$</td>
<td>yearly recruitment rate</td>
</tr>
</tbody>
</table>

3.3 Eigenvalue analysis and dynamics for the one-patch model

We use linearization techniques evaluated at equilibrium to determine local stability of fixed points, and identify oscillation types of steady state behavior using Lyapunov exponents. Fixed points are found by setting $j(t+1) = j(t) = j^*$, $s(t+1) = s(t) = s^*$, and $a(t+1) = a(t) = a^*$ for the appropriate model and stability is determined by analyzing the dominant eigenvalue, $\lambda$, of the Jacobian matrix evaluated at the fixed point. The magnitude of the dominant eigenvalue determines whether the fixed point is stable ($|\lambda| < 1$) or unstable ($|\lambda| > 1$). For stable fixed points ($|\lambda| < 1$), asymptotic behavior of populations that start near the fixed point will converge to the fixed point. Additionally, convergence
to an equilibrium may occur monotonically when the dominant eigenvalue is real (node), or 
non-monotonically when the dominant eigenvalue contains a nonzero imaginary part (spiral).
For unstable fixed points ($|\lambda| > 1$), nearby trajectories are repelled from the fixed point 
and will converge to the nearest attractor. Oscillations occur when the dominant eigenvalue 
does not have an imaginary part and is less than $-1$, or the imaginary part of the dominant 
eigenvalue is nonzero and $|\lambda| > 1$. We classify oscillations into three attractor types (in-
vARIANT cycles, k-cycles, and chaos) based on the calculated Lyapunov exponent (see section 
3.3.3).

3.3.1 The Jacobian matrix

We define Jacobians for any time $t$ to allow for Lyapunov exponent calculations. Be-
cause the only density-dependent term is in the larval stage, all Jacobians have the following 
density-dependent larval survival term,

$$
\hat{H}(t) = \frac{R_H}{(1 + \frac{F a(t)}{K})^\gamma} - \frac{\gamma F R_H}{K \left(1 + \frac{F a(t)}{K}\right)^{\gamma+1}} a(t).
$$

(3.11)

where $R_H = FL$ represents yearly recruitment rate.

The Jacobians are then defined as

- for the 1-stage model,

$$
J_1(t) = S_a + \hat{H}(t),
$$

(3.12)

- for the 2-stage model,

$$
J_2(t) = \begin{bmatrix}
S_j (1 - P_{ja}) & \hat{H}(t) \\
S_j P_{ja} & S_a
\end{bmatrix},
$$

(3.13)

- and for the 3-stage model,

$$
J_3(t) = \begin{bmatrix}
S_j (1 - P_{js}) & 0 & \hat{H}(t) \\
S_j P_{js} & S_s (1 - P_{sa}) & 0 \\
0 & S_s P_{sa} & S_a
\end{bmatrix}.
$$

(3.14)
When evaluated at equilibrium, these Jacobians are used for local stability analysis; otherwise, long-term temporal averages over the Jacobian are used for attractor identification (see section 3.3.3).

### 3.3.2 Analytic solutions of the Jacobian at equilibrium

We analytically solve for Jacobians evaluated at the trivial and nontrivial fixed points for the one-patch model. With 1 life-history stage, the trivial steady state is \( a^* = 0 \) and its Jacobian is calculated as

\[
J^0_1 = S_a + R_H. \tag{3.15}
\]

The nontrivial equilibrium is calculated as

\[
a^* = \frac{K}{F} \left( \frac{R_H}{1 - S_a} \right)^{1/\gamma} - 1, \tag{3.16}
\]

with its Jacobian given by

\[
J^*_1 = 1 - \gamma(1 - S_a) \left( 1 - \left( \frac{1 - S_a}{R_H} \right)^{1/\gamma} \right). \tag{3.17}
\]

For 2 life-history stages, the trivial steady state is \( j^* = a^* = 0 \) with its Jacobian given by

\[
J^0_2 = \begin{bmatrix}
S_j(1 - P_{ja}) & R_H \\
S_jP_{ja} & S_a
\end{bmatrix}. \tag{3.18}
\]

The nontrivial equilibrium for the 2-stage model is calculated as

\[
\begin{align*}
\dot{j}^* &= \frac{\dot{H}}{1 - S_j(1 - P_{ja})}, \\
a^* &= \frac{K}{F} \left( \frac{R_H}{1 - S_a \psi_{ja}} \right)^{1/\gamma} - 1, \tag{3.19}
\end{align*}
\]

\[
\begin{align*}
\dot{a}^* &= \frac{K}{F} \left( \frac{R_H}{1 - S_a \psi_{ja}} \right)^{1/\gamma} - 1,
\end{align*}
\]
where $\tilde{H} = a^*\hat{H}$, and $\psi_{ja} = \frac{P_{ja}S_j}{1-S_j(1-P_{ja})}$ represents the total fraction of juveniles that eventually mature to adults (see appendix for details). The corresponding Jacobian is

$$J_2^* = \begin{bmatrix} S_j(1 - P_{ja}) & \frac{1 - S_a}{\psi_{ja}} \left(1 - \gamma \left(1 - \left(\frac{R_H\psi_{ja}}{1-S_a}\right)^{-\frac{1}{\gamma}}\right)\right) \\ S_jP_{ja} & S_a \end{bmatrix}.$$  \hspace{1cm} (3.20)

Finally, for 3 life-history stages, the trivial equilibrium is $j^* = s^* = a^* = 0$ with Jacobian

$$J_3^0 = \begin{bmatrix} S_j(1 - P_{js}) & 0 & R_H \\ S_jP_{js} & S_s(1 - P_{sa}) & 0 \\ 0 & S_sP_{sa} & S_a \end{bmatrix}.$$  \hspace{1cm} (3.21)

The nontrivial equilibrium for the 3-stage model is

$$j^* = \psi_{js}\tilde{H}$$

$$s^* = \psi_{js}\psi_{sa}\tilde{H}$$

$$a^* = \frac{K}{F} \left( \left(\frac{R_H}{1-S_a}\psi_{js}\psi_{sa}\right)^{1/\gamma} - 1 \right),$$  \hspace{1cm} (3.22)

where $\psi_{js} = \frac{P_{ja}S_j}{1-S_j(1-P_{js})}$ and $\psi_{sa} = \frac{P_{sa}S_s}{1-S_s(1-P_{sa})}$ are the total fraction of individuals that eventually mature to the next stage (see appendix for details). The corresponding Jacobian is

$$J_3^* = \begin{bmatrix} S_j(1 - P_{js}) & 0 & \frac{1 - S_a}{\psi_{js}\psi_{sa}} \left(1 - \gamma \left(1 - \left(\frac{R_H\psi_{js}\psi_{sa}}{1-S_a}\right)^{-\frac{1}{\gamma}}\right)\right) \\ S_jP_{js} & S_s(1 - P_{sa}) & 0 \\ 0 & S_sP_{sa} & S_a \end{bmatrix}.$$  \hspace{1cm} (3.23)

### 3.3.3 Attractor types

We classify oscillating populations into three attractor types: (a) invariant cycles; (b) k-cycles; and (c) chaos, using the Lyapunov exponent defined for matrix population models
Caswell (2001),

\[ \lambda_e = \lim_{T \to \infty} \frac{1}{T} \ln |J_n(T - 1) \cdots J_n(0) u_n(0)|. \]  

(3.24)

Here, \( J_n(t) \) is the \( n \)-stage Jacobian matrix at time \( t \), \( u_n(0) \) is the initial population vector, and \( |\cdot| \) is the vector magnitude. Note that \( \lambda_e \) depends on \( u_n(0) \) but results remain the same with vectors in the same basin of attraction (Caswell 2001; Gyllenberg et al. 1992; Strogatz 2015). See section 3.4.3 for systems with fractal basins of attraction.

Oscillations are classified based on the Lyapunov exponent value where \( \lambda_e < 0 \) are \( k \)-cycles, \( \lambda_e = 0 \) are invariant cycles, and \( \lambda_e > 0 \), are chaos (Caswell 2001). Examples of each attractor type are summarized in Figure 3.1. Because numerical simulations introduce calculation error, we identify invariant cycles if \( \lambda_e \) is sufficiently close to 0 (i.e., \( |\lambda_e| < 0.01 \)). Identifying unstable dynamics without closed form eigenvalue analyses is a nontrivial task and even graphical representations of different oscillations may be ambiguous. We visually inspect selected results and find that the tolerance above sufficiently distinguishes invariant cycles from chaos and \( k \)-cycles.

Intuitively, the Lyapunov exponent measures the averaged progression of two nearby trajectories in time. If the two trajectories approach an equilibrium, the difference between them approaches 0. For oscillating populations, the difference between the two trajectories approaches 0 for in-phase \( k \)-cycles (\( \lambda_e < 0 \)) or reaches some constant value for invariant cycles (\( \lambda_e = 0 \)). With chaos (\( \lambda_e > 0 \)), the trajectories never converge so the difference between them increases as time moves forward. The Jacobian matrix represents the dynamics of the deviation vector; hence, the product of all temporal Jacobian matrices describes total deviation in time.
Fig. 3.1: Phase planes for juvenile and adult populations in the one-patch, 2-stage model with calculated Lyapunov exponent for an invariant cycle (A; \( \lambda_e = 0 \)), low-point k-cycle (B; \( \lambda_e < 0 \)), high-point k-cycle (C; \( \lambda_e < 0 \)), and chaos (D; \( \lambda_e > 0 \)). The k-cycles are formed by the phase-locked invariant cycles while retaining the shape of the closed loop. Chaos is developed as a wrinkle forms in the invariant cycle, creating sensitivity to initial conditions. Simulations are created by plotting the final 9000 juvenile and adult densities for \( R_H = 167.333 \), and \( S_a = 0.366(A), 0.228(B), 0.139(C), \) and 0.099(D).

### 3.3.4 Phase planes for the one-patch model

We investigate dynamics in the one-patch model of species with 1, 2, and 3 life-history stages by varying the range of pairwise parameter sets with all other parameters fixed. Parameters are based on estimates used in amphibian literature and we try to maintain consistent total populations across the different life stage models (Table 3.2). We determine that all trivial solutions in the one-patch models (\( j^* = s^* = a^* = 0 \)) are real and unstable for all parameter ranges except for \( L = 0.01 \) and \( 0.01 \leq S_a \leq 0.42 \) in the 3-stage model. In this region, the trivial equilibrium is stable for a brief window before undergoing a transcritical bifurcation, switching stability with the nontrivial fixed point. This means that global extinction will occur for sufficiently low adult and larval survival.

For each parameter combination, we first determine stability using eigenvalue analysis of the Jacobian matrix at the nontrivial fixed points. When oscillations occur, we classify the oscillation type using the Lyapunov exponent by initializing each life stage with 20 indi-
viduals and run simulations for $T = 10000$ time steps to ensure convergence to asymptotic behavior. The Lyapunov exponent is calculated using the final 9000 time steps to identify attractor type. Note that the carrying capacity ranges from 40000 to 80000 but time series simulations typically have populations with less than 200 individuals. Although carrying capacities seem high, the ratio between fecundity and carrying capacity ($F/K$) is similar to equivalent coefficients estimated in other studies (Vonesh and De la Cruz 2002; Willson et al. 2012).

Table 3.2: Default vital rates and values used in the model. Vital rates are based on ranges used in different amphibian systems.

<table>
<thead>
<tr>
<th>Vital rate</th>
<th>Symbol</th>
<th>1-stage</th>
<th>2-stage</th>
<th>3-stage</th>
<th>Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fecundity</td>
<td>$F$</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>(125)</td>
</tr>
<tr>
<td>Maximum first-year survival</td>
<td>$L$</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>(186; 192)</td>
</tr>
<tr>
<td>Carrying capacity</td>
<td>$K$</td>
<td>80000</td>
<td>50000</td>
<td>40000</td>
<td>(192)</td>
</tr>
<tr>
<td>Density-dependent exponent</td>
<td>$\gamma$</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>(186)</td>
</tr>
<tr>
<td>Juvenile survival</td>
<td>$S_j$</td>
<td>-</td>
<td>0.4</td>
<td>0.45</td>
<td>(125; 186; 192)</td>
</tr>
<tr>
<td>Subadult survival</td>
<td>$S_s$</td>
<td>-</td>
<td>-</td>
<td>0.7</td>
<td>(125)</td>
</tr>
<tr>
<td>Adult survival</td>
<td>$S_a$</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>(125; 186; 192)</td>
</tr>
<tr>
<td>Transition from juvenile to adult</td>
<td>$P_{ja}$</td>
<td>-</td>
<td>0.5</td>
<td>-</td>
<td>(186)</td>
</tr>
<tr>
<td>Transition from juvenile to subadult</td>
<td>$P_{js}$</td>
<td>-</td>
<td>-</td>
<td>0.3</td>
<td>(125)</td>
</tr>
<tr>
<td>Transition from subadult to adult</td>
<td>$P_{sa}$</td>
<td>-</td>
<td>-</td>
<td>0.6</td>
<td>(125)</td>
</tr>
</tbody>
</table>

Citations are as follows: 125-McCaffery and Maxell (2010); 186-Vonesh and De la Cruz (2002); 192-Willson et al. (2012)

We vary the yearly recruitment rate, $R_H$, the density-dependent exponent, $\gamma$, and adult survival, $S_a$ to understand how small parameter changes affect system dynamics. Figure 3.2 show contour plots of the calculated dominant eigenvalues for each model. The bold lines denote where bifurcations of the nontrivial steady state occur ($|\lambda| = 1$). The 1-stage model experiences flip bifurcations, causing the birth of k-cycles while the 2- and 3-stage models
experience Hopf bifurcations, creating invariant cycles. Additionally, in the 3-stage model, a transcritical bifurcation occurs for low adult survival and yearly recruitment where the population experiences global extinction. This is the only region in the one-patch model where the trivial fixed point is stable.

Figure 3.3 shows where different attractor types occur under the different parameter ranges. Equilibrium regions are white while oscillating regions are colored based on attractor type and k-cycle period. The number of points in a k-cycle are numerically estimated by finding the last point in the cycle within 0.001 of the final point. We identify all 2, 4, 8, and 16 point k-cycles as well as k-cycles with periods greater than 16 (16+) and less than 16 that were not already identified (other cycles). In general, increasing the number of stages ‘simplifies’ the dynamics, i.e., chaos is most prevalent in the 1-stage model (A, D), a mix of dynamics are found in the 2-stage model (B, E), and invariant cycles are most prevalent in the 3-stage model (C, F). In the 1-stage model, k-cycles occur through flip bifurcations that undergo period-doubling routes to chaos. Among the chaos regions, periodic windows appear as low-point k-cycles. In the 2- and 3-stage models, Hopf bifurcations create invariant cycles that become phase-locked to varying degrees. Specifically, the 2-stage model has large windows of low-point cycles, high-point cycles, and chaos while the 3-stage model has fewer phase-locked regions of low-point k-cycles.
Fig. 3.2: Dominant eigenvalues ($\lambda$) for a range of adult survival ($S_a$), yearly recruitment rate ($R_H$), and density-dependent exponent ($\gamma$) for the one-patch model. The solid black lines represent bifurcations in the nontrivial fixed points. The 1-stage model has flip bifurcations, creating k-cycles whereas the 2- and 3-stage models undergo Hopf bifurcations, inducing invariant cycles. Additionally, under low adult survival and fecundity in the 3-stage model (C), the nontrivial fixed point experiences a transcritical bifurcation where it switches stability with the trivial fixed point.
Fig. 3.3: Attractor planes for a range of adult survival ($S_a$), yearly recruitment rate ($R_H$), and density-dependent exponent ($\gamma$) for the one-patch model. Attractor type is determined with the Lyapunov exponent with subclassification of k-cycles into 2, 4, 8, 16, 16+ period cycles, and all other k-cycles with less than 16 points (other cycles). Flip bifurcations create k-cycles in the 1-stage model where chaos arises under period-doubling routes with periodic windows of low-point k-cycles. For the 2- and 3-stage models, invariant cycles are created through a Hopf bifurcation with phase-locked regions generating low-point k-cycles in the 3-stage model and a mix of low-point cycles, high-point cycles, and chaos in the 2-stage model.

Close inspection of bifurcation diagrams with calculated Lyapunov exponents allow us to verify results found in the eigenvalue and attractor plots. Bifurcation plots (Figure 3.4A-C) are created by plotting patch 1 adults in the final 50 time steps vertically for each $S_a$ value, and the Lyapunov exponents (Figure 3.4D-F) are calculated using the final 9000 time steps. We analyze a range of adult survival values and find that a flip bifurcation occurs at $S_a \approx 0.58$ in the 1-stage model, creating k-cycles that lead to chaos through a period-doubling route. The 2- and 3-stage models experience Hopf bifurcations at $S_a \approx 0.2$ in the 3-stage model and $S_a \approx 0.6$ in the 2- and 3-stage models. Additionally, low-point phase-locked k-cycles form for a few small regions in the 2-stage model, marked by negative $\lambda_e$. 
Fig. 3.4: One-patch bifurcation diagrams (A, B, C) with calculated Lyapunov exponents (D, E, F) and dominant eigenvalues (G, H, I) allow us to analyze bifurcations under a range of adult survival probabilities. In the 1-stage model, k-cycles occur through a flip bifurcation, creating chaos through a period-doubling route. In the 2-stage model, Hopf bifurcations create invariant cycles with phase-locked k-cycles whereas invariant cycles persist in the 3-stage model. Bifurcations and attractor types are verified with respective Lyapunov exponents (2nd row) and dominant eigenvalues (3rd row).

3.4 Two-patch model dynamics under constant dispersal probabilities

We analyze local stability of numerically solved fixed points in the two-patch model using eigenvalue analysis, and steady state behavior using Lyapunov exponents and population synchrony. We find that multiple fixed points exist under different dispersal regions which creates some difficulty in performing broad-scale analysis of dynamics. Therefore, to maintain consistency across model outputs, we analyze asymptotic behavior of only one trajectory using initial conditions defined in the following paragraph. Note that this does not find all alternate steady states as that requires extensive analysis of localized parameters. To account for this, we analyze a select few scenarios using bifurcation and eigenvalue analysis under multiple initial conditions to understand some of the different dynamics found in our results.

To analyze the effects of dispersal on the two-patch model, we select pairwise $S_a$ values that initialize the model with a variety of dynamics without dispersal (see Table 3.3).
Attractor and synchrony planes allow us to understand general system behavior while bifurcation and stability plots of fixed points for select parameter ranges provide additional information on some of the bifurcation types and alternate steady states that exist. Each simulation is initialized with steady states from the corresponding one-patch simulation. For example, in the 1-stage model, if adult survival in patch 1 is 0.1 \( (S_{a1} = 0.1) \) and adult survival in patch 2 is 0.7 \( (S_{a2} = 0.7) \) then patch 1 is initialized with one of the points on the local k-cycle while patch 2 is initialized with approximately 50 individuals corresponding to the stable equilibrium. We identify oscillations by running our models for \( T = 10000 \) time steps and analyzing the range of the final 50 time steps. A steady state is classified as an oscillation if the range of any population is greater than 0.01; otherwise, the steady state is an equilibrium. Attractor type is determined by calculating the Lyapunov exponent (\( \lambda_e \) in equation 3.24) using the final 9000 time steps and the two-patch Jacobian defined by

\[
J_n(t) = \begin{bmatrix}
(1 - \sigma_{1n}J_{1n}(t)) & \sigma_{2n}J_{2n}(t) \\
\sigma_{1n}J_{1n}(t) & (1 - \sigma_{2n})J_{2n}(t)
\end{bmatrix},
\] (3.25)

where \( J_{xn} \) is the \( n \)-stage Jacobian for patch \( x \) with corresponding two-patch population vector \( u_{xn} \). Finally, we calculate population synchrony using patch population sizes in the final 500 time steps (see section 3.4.2).

For each pair of adult survival values, we test 100 dispersal probabilities ranging from 0 to 1 for each life stage. With 15 pairwise adult survival probabilities and 10000 pairwise dispersal probabilities for each life stage, we have a total of 150,000 simulations for the 1-stage model, 300,000 simulations for the 2-stage model, and 450,000 simulations for the 3-stage model (all attractor, synchrony, and eigenvalue simulations are in the appendix). We summarize our results with the following analysis.

### 3.4.1 Dispersal symmetry

We measure dispersal symmetry in two ways: 1) comparing the proportion of dispersing individuals in one patch to the total proportion of dispersing individuals, and 2) partitioning
Table 3.3: Adult survival used in the two-patch systems with local stability dynamics and Lyapunov exponent ($\lambda_e$).

<table>
<thead>
<tr>
<th>$S_a$</th>
<th>1-stage Dynamics ($\lambda_e$)</th>
<th>2-stage Dynamics ($\lambda_e$)</th>
<th>3-stage Dynamics ($\lambda_e$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>K-cycle (-0.369)</td>
<td>Invariant cycle (0.000)</td>
<td>Stable (-0.003)</td>
</tr>
<tr>
<td>0.3</td>
<td>Chaos (0.286)</td>
<td>Invariant cycle (0.000)</td>
<td>Invariant cycle (0.000)</td>
</tr>
<tr>
<td>0.5</td>
<td>K-cycle (-0.356)</td>
<td>Invariant cycle (0.000)</td>
<td>Invariant cycle (0.000)</td>
</tr>
<tr>
<td>0.7</td>
<td>Stable (-0.083)</td>
<td>Stable (-0.060)</td>
<td>Stable (-0.0061)</td>
</tr>
<tr>
<td>0.9</td>
<td>Stable (-0.954)</td>
<td>Stable (-0.083)</td>
<td>Stable (-0.083)</td>
</tr>
</tbody>
</table>

the dispersal probability plane into zones. We use populations averaged over the final time steps $\hat{u}_{xn} = \frac{1}{500} \sum_{\tau=1}^{500} u_{xn}(\tau)$ to define the asymmetry in per capita dispersal from each patch,

$$\alpha = \frac{N_{\sigma_1}/N_1}{N_{\sigma_1}/N_1 + N_{\sigma_2}/N_2},$$  \hspace{1cm} (3.26)

where $N_{\sigma x} = \sum_n \sigma_{xn} \hat{u}_{xn}$ is the number of individuals dispersing from patch $x = 1, 2$, and $N_1 = \sum_n \hat{u}_{1n}$ is the total population size of patch 1. Dispersal is symmetric when $\alpha \approx 0.5$, and asymmetric when $\alpha \approx 0$ (mostly dispersing from patch 2) and $\alpha \approx 1$ (mostly dispersing from patch 1).

We group the dispersal plane into 5 zones that represent different dispersal probabilities and symmetries, as shown in Figure 3.5. They are: low symmetric dispersal (zone 1), intermediate symmetric dispersal (zone 2), high symmetric dispersal (zone 3), low asymmetric dispersal (zone 4), and high asymmetric dispersal (zone 5). Partitioning dispersal regions into different zones provides insight into expected system dynamics based on dispersal symmetry and degree. Note that the number of simulations in each zone differs based on cover area: zones 1 and 3 have 210 simulations, zone 2 has 3260 simulations, and zones 4 and 5 have 3160 simulations.
Fig. 3.5: The 5 zones for the dispersal plane that represent different dispersal probabilities and symmetries. Zone 1 represents low symmetric dispersal, zone 2 represents intermediate symmetric dispersal, zone 3 represents high symmetric dispersal, zone 4 represents low asymmetric dispersal, and zone 5 represents high asymmetric dispersal.

3.4.2 Patch synchrony

We measure population synchrony between the two patches using patch population sizes in the final 500 time steps. This method sums over all entries in the temporal covariance matrix and divides by the square of the sum over the standard deviations (Loreau and Mazancourt 2008),

$$\beta = \frac{\sum_{i,k} \text{cov}(\hat{U}_i, \hat{U}_k)}{\left( \sum_{i} \sqrt{\text{cov}(\hat{U}_i, \hat{U}_i)} \right)^2},$$

(3.27)

for $i = 1, 2$ and $k = 1, 2$. The vector, $\hat{U}_i$, is defined by

$$\hat{U}_i = \begin{bmatrix} U_i(T - 499) \\ U_i(T - 498) \\ \vdots \\ U_i(T) \end{bmatrix},$$

where $U_i(t) = \sum_n u_{in}(t)$ are patch $i$ population sizes, and $\text{cov}(\hat{U}_i, \hat{U}_k)$ are the temporal covariance measures between $\hat{U}_i$ and $\hat{U}_k$. This method measures the variance of population
sizes across all patches and is standardized between 0 and 1. \( \beta \approx 1 \) indicates that the populations are strongly positively correlated (high synchrony) while \( \beta \approx 0 \) indicates that the populations are strongly negatively correlated (low synchrony). Systems with at least one stable population have perfect synchrony (only found with zero dispersal); hence even small dispersal probabilities produce cycles across all populations. Additionally, this method is robust to population magnitudes (i.e., vertical shifts in population abundances), and measures near-perfect synchrony for populations with large amplitude differences.

### 3.4.3 Fractal basins of attraction

Dispersal can create multiple steady states under different initial conditions even when all parameters remain the same, called fractal basins of attraction (Hastings 1993; Udwa-dia and Raju 1998). To demonstrate this, we investigate two parameter sets where fractal basins are produced in the 1-stage model by testing a range of initial conditions for patch 1 \((a_1(0))\) and patch 2 \((a_2(0))\) while keeping all parameters constant. The results are summarized in Figure 3.6. We find that fractal basins of attraction occur under relatively high dispersal. In addition, different initial conditions may produce stable equilibria (white), and either k-cycles (black) or chaos (red). Note that we found fractal basins of attraction only in the 1-stage model, and not in the 2- or 3-stage models. More instances of fractal basins of attraction may occur in our simulations but identifying their locations is non-trivial and beyond the scope of this project. Instead, understanding that fractal basins of attraction exist in our models shows that some of our results may change under different initial conditions.
Fig. 3.6: Dispersal may create multiple end state dynamics under different initial conditions even though all vital rate parameters are unchanged. We vary initial conditions \((a_1(0), a_2(0))\) while fixing all parameters to produce fractal basins of attraction. In A, we use \(\sigma_1 = 0.87\) and \(\sigma_2 = 0.96\), and the system either experiences chaos (red) or stable dynamics (white). In B, we use \(\sigma_1 = 0.87\) and \(\sigma_2 = 0.9\), and the system either experiences k-cycles (black) or stable dynamics (white). In both scenarios, \(S_{a1} = 0.1\) and \(S_{a2} = 0.3\).

3.4.4 Two-patch model attractor type and synchrony results

We find that dynamics are preserved under small vital rate perturbations and dispersal may induce oscillations even if both local populations are at equilibrium. For example, the homogeneous systems in the 1-stage model \((S_{a1} = S_{a2})\) contain oscillating regions in the scenarios where both patches are at equilibrium in the absence of dispersal (B, C) and appear in similar dispersal regions as the scenario with slightly lower adult survival (A) (Figure 3.7). We also find evidence of dispersal-induced oscillations for one heterogeneous scenario in the 1-stage model, \(S_{a1} = 0.7, S_{a2} = 0.9\), three scenarios in the 2-stage model \((S_{a1} = S_{a2} = 0.7; S_{a1} = 0.7, S_{a2} = 0.9; S_{a1} = S_{a2} = 0.9)\), and two scenarios in the 3-stage model \((S_{a1} = 0.1, S_{a2} = 0.7; S_{a1} = 0.1, S_{a2} = 0.9)\) (see appendix). Note that dispersal-induced oscillations have only been observed under adult dispersal, presumably because of the direct effects of adults on larval-stage survival.
Fig. 3.7: Attractor planes for homogeneous metapopulations ($S_{a1} = S_{a2}$) in the one stage model show that dispersal regions containing oscillations are similar under small vital rate perturbations. Attractor type is classified as equilibria (white), invariant cycles (blue), k-cycles (black), and chaos (red). K-cycles are further classified based on their periodicity. Dispersal-induced oscillations occur for high adult survival (B, C) in similar unstable regions as intermediate adult survival (A).

Figure 3.8A-C shows the relative frequency of per capita symmetric dispersal ($\alpha$) for each attractor type for the 1-, 2-, and 3-stage models, and Figure 3.8D shows the proportion that each attractor type appears in each zone. In the 1-stage model, chaotic results (red) occur most frequently under symmetric dispersal (zones 1, 2, and 3), k-cycles (black) occur almost uniformly across different symmetries and zones, and invariant cycles (blue) occur under high dispersal (zones 3 and 5). In the 2-stage model, oscillations mostly occur under symmetric dispersal with the majority of k-cycles appearing under highly symmetric dispersal (zone 3). In the 3-stage model, invariant cycles occur under symmetric dispersal while the few instances of chaos are found in the extremes, i.e., full dispersal from one patch to the other. equilibria (purple) generally have a more uniform distribution in all three models, showing little correlation with dispersal symmetry. Note that the number of chaos events in the 2- and 3-stage models are sparse compared to the number of invariant and k-cycles.
Fig. 3.8: The relative frequency of per capita symmetric dispersal ($\alpha$) for each attractor type for the 1-stage (A), 2-stage (B), and 3-stage (C) models, and the fraction that each dispersal type appears in each zone (D). Unstable occurrences are classified as invariant cycles (blue), k-cycles (black), chaos (red), and stable equilibria (purple). The number of occurrences for each attractor type are noted in parentheses. Chaos is most prevalent under symmetric dispersal in the 1-stage model, invariant cycles occur under asymmetric dispersal in the 1-stage model and symmetric dispersal in the 2- and 3-stage models, and k-cycles occur for all dispersal symmetries in the 1-stage model, under high symmetric dispersal in the 2-stage model, and rarely in the 3-stage model. Stable equilibria are present for all symmetries across all models.

To summarize population synchrony ($\beta$), we plot synchrony histograms for each attractor type in each zone (Figure 3.9). For example, of the 2498 1-stage simulations with low symmetric dispersal (top left panel in Figure 3.9), 20 exhibited invariant cycles with $\sim 80\%$ being perfectly synchronized (blue), 1824 exhibited k-cycles with $\sim 70\%$ being perfectly synchronized (black), and 654 exhibited chaos with synchrony being mostly between 0.8 and 1 (red). Across all three life stage models, zone 1 has the most mixed results due to low dispersal probabilities from local populations (first row). As dispersal increases, we find more consistency within each zone. For the 1-stage model, higher dispersal regions (zones 2, 3, and 5) have consistently lower synchrony than lower dispersal zones (zones 1 and 4). For the 2-stage model, invariant cycles maintain near-perfect synchrony across all zones. The few instances of chaos either have perfect or intermediate synchrony ($\beta \approx 0.5$), and
k-cycles occur with very low synchrony in zone 3 and high synchrony in all other zones. For the 3-stage model, we find consistently high synchrony for all unstable points including a few instances of chaos.

Overall, low and high symmetric dispersal (zones 1 and 3) produce the most variety in synchrony and attractor type, especially for the 1- and 2-stage models. The variety in zone 1 can be expected since they are reflective of local population dynamics without dispersal. However, high symmetric dispersal (zone 3) may significantly destabilize dynamics, producing a relatively high number of chaos events in the 1-stage model and low-synchrony k-cycles in the 2-stage model. The latter result is especially interesting since these are the only instances that we see such low synchrony.
Fig. 3.9: The relative frequencies for synchrony levels for each attractor type summarize the stability-synchrony relationships in the different dispersal zones. The number of occurrences that an attractor type appears in each zone is in parentheses and $\beta$ is the proportion of synchrony occurrences within each attractor type. In the 1-stage model, intermediate to high dispersal probabilities (zones 2, 3, and 5) produce lower synchrony ($\beta \approx 0.8$) for all attractor types whereas low dispersal probabilities (zones 1 and 4) have more results with near-perfect synchrony. In the 2-stage model, most results have near-perfect synchrony under sufficient dispersal (zones 2, 3, and 5) with the unique finding of near-perfect asynchronous k-cycles under very high symmetric dispersal (zone 3). In the 3-stage model, highly synchronous invariant cycles are consistently high across all zones.

### 3.4.5 Bifurcation analysis

The results in the previous section only capture trajectories of one steady state even though multiple steady states may exist. We use bifurcation and eigenvalue plots of select dispersal probabilities to analyze some of the alternate attractors and fixed points that appear in the two-patch model (Figures 3.10 and 3.11). For each $\sigma_{a1}$ value, we find all
nontrivial fixed points by numerically solving for $u_n^* = A_n^* u_n^*$ using trust-region root-finding techniques with 200 randomly selected start conditions (Coleman and Li 1996). See the appendix for details. We use these fixed points to numerically compute eigenvalues for local stability analysis. To identify all attractors for each $\sigma_{a1}$ value, we initialize simulations with 100 random initial conditions and run the two-patch model for 1000 time steps. This ensures that populations start within the basins of attraction for each unique attractor.

In plots A-C, we represent attractors at chosen $\sigma_{a1}$ values by plotting the population size of adults in patch 1 for the final 50 time steps (black dots). Additionally, we plot all numerically solved fixed points for each $\sigma_{a1}$ (red circles). For each fixed point, we plot the real part (black circles) and magnitude (blue dots) of the dominant eigenvalue in plots D-F. Trajectories near unstable fixed points ($|\lambda| > 1$) will converge to one of the attractors, and we identify local bifurcations and alternate steady states by closely inspecting the bifurcation and eigenvalue plots. Note that we do not attempt to identify unstable limit cycles as it requires systematic analysis of global dynamics and bifurcations. We select 3 scenarios in the two-patch, 1-stage model, 2 scenarios in the 2-stage model, and 1 scenario in the 3-stage model to show some of the different bifurcations that occur in our simulations. All parameters are fixed except for $\sigma_{a2}$ which may either be fixed or equal to $\sigma_{a1}$.

In Figure 3.10A, D, a 2-cycle begins for low $\sigma_{a1}$ that experiences a period-doubling route to chaos at $\sigma_{a1} \approx 0.25$. This persists until it undergoes a crisis at $\sigma_{a1} \approx 0.8$, creating a sudden shift to a stable equilibrium. The stable fixed point undergoes a Hopf bifurcation where it loses stability and an invariant cycle appears at $\sigma_{a1} \approx 0.9$. In plots B, E, a stable equilibrium persists throughout the $\sigma_{a1}$ range and coexists with a k-cycle that is produced through a 2-cycle saddle-node bifurcation at $\sigma_{a1} \approx 0.7$. The k-cycle experiences a period-doubling route to chaos that is suddenly destroyed at $\sigma_{a1} \approx 0.87$ via a crisis. Here, a new fixed point is created that quickly experiences a flip bifurcation, creating a stable equilibrium. The fixed point then undergoes a Hopf bifurcation, causing the birth of an invariant cycle that appears for a small window. Finally, in plots C, F, chaos induced by period-doubling routes persist for $0.15 < \sigma_{a1} < 0.9$. An unstable fixed point appears at
$\sigma_{a1} \approx 0.6$ ($|\lambda| > 5$), and a pitchfork bifurcation occurs at $\sigma_{a1} \approx 0.75$ creating two unstable fixed points. As $\sigma_{a1}$ increases, the unstable fixed points gain stability for a brief window before Hopf bifurcations cause the two stable equilibria to lose stability, creating invariant cycles that quickly become chaotic attractors. The qualitative behavior of the attractors in columns 2 and 3 are verified with phase planes and iteration maps for select $\sigma_{a1}$ in the appendix.

Figure 3.11 shows bifurcation plots and dominant eigenvalues for the 2-stage model (columns 1 and 2) and the 3-stage model (column 3). In plots A, D, we find that an invariant cycle with constant amplitude persists until $\sigma_{a1} \approx 0.9$ where the dominant eigenvalue loses its imaginary component while the real component is less than $-1$. This creates a sudden shift from invariant cycles to 2-point k-cycles with increasing amplitude as $\sigma_{a1}$ increases. Plots B, E show a small window of invariant cycles occurring for $0.3 < \sigma_{j1} < 0.4$, and another destruction of the imaginary component of the dominant eigenvalue occurring at $\sigma_{j1} \approx 0.65$, shifting the fixed point from a stable spiral to a stable node. Finally, in plots C, F, a stable equilibrium persists for all $\sigma_{s1}$ while a saddle-node bifurcation occurs at $\sigma_{s1} \approx 0.8$, creating asymptotically stable and unstable fixed points.
Fig. 3.10: Bifurcation and eigenvalue plots of all global attractors (black dots), fixed points (red circles), and dominant eigenvalues of fixed points for the two-patch, 1-stage model. As $\sigma_{a1}$ increases in plots A, D, k-cycles undergo a period-doubling route to chaos ($\sigma_{a1} \approx 0.45$) which then experiences a crisis at $\sigma_{a1} \approx 0.8$, creating a brief window of stability as it collides with an unstable orbit. At $\sigma_{a1} \approx 0.9$, the stable equilibrium undergoes a Hopf bifurcation, creating an invariant cycle. In plots B, E, a k-cycle appears at $\sigma_{a1} \approx 0.7$ and undergoes a period-doubling route to chaos. This persists until a new fixed point is created at $\sigma_{a1} \approx 0.87$. The new fixed point quickly undergoes a flip bifurcation to Hopf bifurcation, causing the alternate attractor to shift from a k-cycle to equilibrium to invariant cycle. In plots C, F, chaos persists under phase-locked k-cycles for $0.15 < \sigma_{a1} < 0.9$ with a period doubling route occurring at $\sigma_{a1} \approx 0.1$. Two unstable fixed points are created at $\sigma_{a1} \approx 0.75$ via a pitchfork bifurcation where only the chaotic attractor exists. The fixed points gain stability for a brief window before once again losing stability via Hopf bifurcations where invariant cycles are created that quickly phase-lock to chaos (see appendix). Parameters used are $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.6$ for plots A, D, $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.96$ for plots B, E, and $S_{a1} = S_{a2} = 0.3$, $\sigma_{a2} = \sigma_{a1}$ for plots C, F.
Fig. 3.11: Bifurcation and eigenvalue plots of all global attractors (black dots), fixed points (red circles), and dominant eigenvalues of fixed points for the two-patch, 2-stage model (columns 1 and 2), and 3-stage model (column 3). In plots A, D, an invariant cycle persists until the loss of the imaginary component in the dominant eigenvalue shifts the invariant cycle to a 2-point k-cycle. In plots B, E, a Hopf bifurcation marks the creation of an invariant cycle for a brief window at $0.3 < \sigma_{j1} < 0.4$ with a stable equilibrium occurring elsewhere. In plots C, F, a saddle-node bifurcation creates two fixed points at $\sigma_{s1} \approx 0.8$, one of which is stable, resulting in two stable nontrivial fixed points. Parameters used are $S_{a1} = S_{a2} = 0.5$, $\sigma_{a2} = \sigma_{a1}$ for plots A, D, $S_{a1} = 0.5$, $S_{a2} = 0.9$, $\sigma_{j2} = 0.8$ for plots B, E, and $S_{a1} = S_{a2} = 0.9$, $\sigma_{s2} = 0.9$ for plots C, F.

3.5 Conclusions

In this paper, we used stage-structured matrix models with density-dependent first-year survival to analyze the effects of dispersal in a two-patch system of a species with up to 3 life-history stages. We used eigenvalue, bifurcation, and Lyapunov exponent analysis to identify qualitative shifts in dynamics and multiple steady states. We observed a variety of bifurcation types and routes to chaos, including period-doubling, phase-locking, and crisis. Dispersal probabilities and symmetries were categorized into 5 zones to compare the relationship between synchrony and attractor type. We found that populations experiencing chaotic fluctuations are not necessarily less synchronized than other oscillation types but rather show similar synchrony as the attractors that precede chaos. Analyzing the relationship between oscillation type and synchrony has not been done before, and we found that the mechanisms that drive bifurcations and the dispersal regions they occupy are indicators of population synchrony regardless of attractor type. This is especially important for synchrony assumptions in chaotic attractors.
3.5.1 Dispersal effects on attractor types

Our result that low to moderate dispersal created equilibrium dynamics while high dispersal may induce oscillations is consistent with previous studies (Abbott 2011; Amarasekare 1998; Dey et al. 2014; Doebeli 1995; Ives et al. 2003; Wang et al. 2015; Wang and Loreau 2014; Ylikarjula et al. 2000). However, where these studies compared synchrony with population variance, we defined oscillations based on attractor type. We found that chaos was most prevalent in the 1-stage model, generally occurring under symmetric dispersal. The onset of chaos varied and were induced by period-doublings, phase-lockings, or crises (see Figure 3.10). Additionally, low to moderate dispersal in the 1-stage model generally created k-cycles via flip bifurcations whereas high dispersal induced invariant cycles via Hopf bifurcations that quickly transitioned to phase-locked k-cycles and chaos. In the 2-stage model, invariant cycles persisted under symmetric dispersal except for high symmetric adult dispersal for moderate to high adult survival. Here, two-point k-cycles were created after suddenly losing the imaginary part of the dominant eigenvalue. The 3-stage model showed persistent invariant cycles largely among symmetric dispersal regions. Cycles were created through Hopf bifurcations and were generally found under moderate adult survival.

We assumed constant dispersal probabilities though different dynamics may result under different dispersal rules (Amarasekare 1998; Kendall and Fox 1998; Ruxton 1996; Ylikarjula et al. 2000). For example, Ylikarjula et al. (2000) found a variety of results when testing both density-dependent and density-independent dispersal; however, they did not find any general differences between the two dispersal types on stability. On the other hand, Amarasekare (1998) found that strong density-dependent dispersal destabilized dynamics. The consequences of dispersal strategies are seemingly sensitive to model characteristics, including density-dependent dispersal type, dispersal mortality, demographic density-dependence type, and growth rates (Amarasekare 1998; Ripa 2000; Ylikarjula et al. 2000). Even with simple dispersal assumptions, we found complicated dynamics in our results, but these are expected to change under different dispersal behavior and rules.

One of the more interesting results we found was the creation of oscillations when both
local populations were initially stable. Previous studies have observed this phenomenon for density-dependent and density-independent dispersal (Kendall and Fox 1998; Ruxton 1996; Ylikarjula et al. 2000). Ylikarjula et al. (2000) found dispersal-induced cycles from fully-stable homogeneous populations under both positive and negative density-dependent dispersal. We found that oscillations induced by dispersal were due to the small effects of vital rate perturbations (Figure 3.7), which explains why dispersal may induce oscillations in systems where both patches were stable in the absence of dispersal.

### 3.5.2 Dispersal effects on synchrony

We found that constant dispersal had a nonlinear effect on population synchrony. Specifically, low dispersal probabilities initially synchronized populations while moderate to high dispersal may rapidly desynchronize populations. This was found for high asymmetric dispersal in the 1-stage model and high symmetric dispersal in the 2-stage model. Asymmetric dispersal has been shown to decrease synchrony in especially one-directional dispersal patterns (Doebeli 1995). Dispersal type and the number of patches in a metapopulation seemingly play a large role on synchrony (Kendall and Fox 1998; Ylikarjula et al. 2000). For example, Ylikarjula et al. (2000) found that synchrony depended on whether dispersal was density-dependent or density-independent. To our knowledge, the nearly perfect asynchronous k-cycles under high symmetric dispersal created in our two-patch model has not been reported in prior studies. We found that this sudden shift was due to the loss of the imaginary part in the dominant eigenvalue.

We assumed that population fluctuations were driven by density-dependent demographic processes. However, fluctuations, and consequently synchrony, could be influenced by environmental processes (Ranta et al. 1997). In fact, Ripa (2000) argues that the Moran effect is ever-present and should not be discounted when analyzing the synchrony-dispersal relationship. The effects of dispersal on population synchrony then depend on the degree of correlated stochasticity and fluctuations induced by intrinsic dynamics (Ripa 2000). Because of the nature of cycles induced by demographic processes, it is much easier to maintain synchrony among populations with intrinsically-induced cycles than a system where popu-
lation fluctuations are driven by uncorrelated stochasticity (Ripa 2000). Therefore, we may expect different results under stochastically-driven fluctuations than fluctuations driven by demographic processes.

3.5.3 Implications for amphibian populations

Amphibian movement behavior is largely unknown due to the difficulty in reliably tracking individuals (Bull 2005; Ross et al. 1999; Semlitsch 2010). We took a theoretical approach to these types of systems by analyzing a range of dispersal probabilities in multi-stage demographic models that have been used in previous amphibian literature (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012). Although we did not explicitly define landscape geography, distances between patches can play an important role in metapopulation dynamics (Holland and Hastings 2008). Patches further from each other can create more complex dynamics while decreasing synchrony (Björnstad et al. 1999; Kaneko 1985; Kendall and Fox 1998; Ripa 2000). The negative relationship between synchrony and distance agrees with populations driven by environmental change as weather patterns are less correlated at greater distances (Ranta et al. 1999). Similarly, oscillations are more prevalent as distance increases since less dispersal is expected across greater distances. Although outside the scope of this study, it would be interesting to explore the effects of dispersal on different network structures and dispersal strategies.

Chaotic oscillations are typically considered to enhance viability due to naturally noisy dynamics and low synchrony; however, this may only be true for chaos induced by environmental variation with low dispersal rates (Allen et al. 1993; Heino et al. 1997; Ylikarjula et al. 2000). We found that chaos was not necessarily less synchronous than other attractor types; rather, populations exhibiting chaotic dynamics had similar synchrony as the attractor types preceding chaos. Specifically, chaos was created through one of three mechanisms (period-doubling, phase-locking, and crisis), and synchrony did not experience any sudden shifts throughout these routes to chaos. This means that chaos did not have any desynchronizing effects on population synchrony in our model. That being said, it is not certain that all routes to chaos preserve synchrony, and it would be interesting to explore other routes
to chaos to determine whether or not chaos has any desynchronizing effects in deterministic models.

The notion that chaos encourages population viability due to its desynchronizing effects does not hold for the deterministic model presented here. Instead, we found that synchrony in chaos is similar to synchrony in other attractor types, and that dispersal symmetry and magnitudes largely drive changes in synchrony across populations. That being said, our results do not cover all bifurcation types and routes to chaos, and there may be instances where the desynchronizing effects of chaos take hold in different models. Thus, more research is needed to understand how attractor types and dispersal relate to synchrony in metapopulation dynamics.
CHAPTER 4
SENSITIVITY ANALYSIS OF HABITAT CONTRIBUTIONS IN A
DENSITY-DEPENDENT COLUMBIA SPOTTED FROG
METAPOPULATION MODEL

Abstract
In metapopulation systems, it is important to analyze both local and global population
dynamics to understand population viability and extinction risk. In classical metapopula-
tion theory, local habitats, or patches, are classified as sources and sinks based on local
demography and the degree to which they interact with proximal habitats. The contribu-
tion metric relates demographic and dispersal processes to source-sink theory by quantifying
patch self-recruitment and emigration rates. Understanding the productivity of patches
allows conservation managers to determine how to properly manage connected habitats.
Additionally, it is important to understand how changes in demography affect patch con-
tributions and population sizes. In this paper, we explore the relationships between patch
contributions, patch population sizes, and total population sizes to understand how in-
dividual habitats influence metapopulation dynamics. We develop elasticity analysis for
habitat contributions using matrix calculus to understand patch contribution and popula-
tion size responses to vital rate perturbations. We apply our analysis to a high-elevation
Columbia spotted frog population under a range of density-dependent dispersal rates and
distance-dependent dispersal distributions. We also consider two dispersal pathways based
on landscape geography; a full system where individuals have complete access to all patches
and a partial system where dispersal is governed by waterway networks. We find that
negative density-dependent dispersal produces similar contributions among patches regardless
of dispersal pathways whereas centrally-located patches have lower contributions but
higher population sizes under positive density-dependent dispersal. When applying elasticity analysis to patch contributions and patch population sizes, we find that the direct effects of vital rate perturbations decrease patch contributions under negative density-dependent dispersal and increase patch contributions under constant and positive density-dependent dispersal. On the other hand, patch sizes increase in the perturbed patch regardless of density-dependent dispersal type. Additionally, total population size elasticities are similar across all simulations regardless of dispersal type and behavior. We use these results to identify potential management strategies that maintain patch type heterogeneity by restoring or degrading select patches. These results show the value in comparing elasticities of different variables to more fully understand system responses to environmental and anthropogenic disturbances.
4.1 Introduction

Many species inhabit different regions or habitats of a landscape where populations in each habitat have unique demographic and dispersal processes. Interactions between populations occur through dispersal processes and the aggregate interactions across all habitats influence both local and global dynamics. In classical metapopulation theory, individual habitats are classified as sources and sinks based on population demographics and dispersal behavior (Holt 1984; Pulliam 1988). Sources have the ability to sustain themselves while supporting other habitats whereas sinks rely on immigration from nearby patches to maintain viable populations. Identifying sources and sinks in a connected network has important implications for conservation management (Crowder et al. 2000; Semlitsch 2000; Strasser et al. 2012). For example, preserving sink habitats may actually do more harm than good while conserving source habitats in low quality systems may increase total population size and decrease extinction risk (Crowder et al. 2000; Semlitsch 2000). Accurate identification of sources and sinks is critical for ensuring appropriate habitat management action.

There has been increasing interest in connecting metapopulation demography with source-sink theory to understand population dynamics in networked systems (Diffendorfer 1998; Krkošek and Lewis 2010; Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). These are achieved using mathematical models generally built on the birth, immigration, death, emigration model (Diffendorfer 1998; Krkošek and Lewis 2010; Pulliam 1988). In recent literature, one particular model has been favored that uses per capita measurements of fecundity, survival, and dispersal (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). This model classifies patches as sources and sinks based on local demographic processes and connections with proximal patches. Specifically, patches that produce recruits for sufficient patch retention while also contributing to the metapopulation through dispersal are sources whereas patches with low recruitment and emigration are sinks (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). Calculation of the contribution metric is relatively straightforward and only uses information from the projection matrix without
any additional analysis methods (e.g., eigenvalue analysis). Additionally, the contribution metric can readily extend to pathways based on per capita movement along migration paths (Sample et al. 2019; Wiederholt et al. 2018).

An important question in population ecology asks how changes in vital rate parameters affect different variables of interest, i.e., sensitivity analysis. Sensitivity analysis measures changes in output variables given absolute changes in model parameters. A potentially more useful measure when comparing parameters with different scales is elasticity analysis which measures proportional parameter changes (Benton and Grant 1999). Sensitivity and elasticity analyses can be used to test management scenarios without the costly effort of field work and can identify error tolerances in parameter estimates. It can also be used to identify parameters that are most important for promoting population growth and viability, providing vital information for conservation efforts (Crowder et al. 2000; De Kroon et al. 2000). In many cases, population growth rates are commonly assessed under perturbed parameters and have been used in a variety of ecological applications (Caswell 2001; De Kroon et al. 2000; Grant and Benton 2000; Hunter and Caswell 2005). Sensitivities to other dependent variables such as population distributions and fitness also have important implications for management action, population ecology, and evolutionary biology (Benton and Grant 1999; Caswell 2009; 2012; Crowder et al. 2000; Flockhart et al. 2015; Grant and Benton 2003; Oberhauser et al. 2017).

Many studies use numerical differentiation to calculate sensitivities and elasticities (Benton and Grant 1999; Crowder et al. 2000; Flockhart et al. 2015; Grant and Benton 2003; Oberhauser et al. 2017). Although these calculations are straightforward, they come with high computational cost and potentially unreliable numerical derivatives in especially highly variable environments (Benton and Grant 1996; 1999). Alternatively, sensitivity analysis can be computed using matrix calculus to quickly analyze large parameter ranges and sensitivities of multiple parameters at once (Caswell 2009; 2012; Hunter and Caswell 2005). We develop a general formula for sensitivity and elasticity analysis of the contribution metric to changes in low-level parameters in density-dependent matrix metapopulation models.
Our models are developed with the vec-permutation approach which classifies individuals by their demographic processes and dispersal behavior (Caswell 2012; Hunter and Caswell 2005). Our analysis builds on sensitivity analysis of equilibria and other dependent variables in density-dependent population models (Caswell 2009; 2012).

We focus on a high-elevation Columbia spotted frog (*Rana luteiventris*) population located in the Selway-Bitterroot Wilderness, Montana. This study site has seven breeding ponds classified into three waterbody types based on their hydroperiod (permanent, semi-permanent, ephemeral). Previous studies have shown that this population is susceptible to harsh climate conditions though habitat heterogeneity may increase population persistence under unpredictable climates (McCaffery et al. 2014; 2012).

Previous research on this population have not considered interpatch interactions largely due to the difficulty in reliably tracking individuals (Bull 2005; Ross et al. 1999; Semlitsch 2010). Little is known about amphibian dispersal in general; however, juveniles (pre-breeding amphibians) tend to exhibit much more active movement and generally have higher dispersal probabilities than adults (Funk et al. 2005a; Gamble et al. 2007; McCaffery et al. 2014; Semlitsch 2010; Smith and Green 2005). Reasons for dispersing from a breeding pond to a new pond could be due to habitat quality, predator presence, and pond demography (Boualit et al. 2019; Buxton and Sperry 2017; Cayuela et al. 2018; Gamble et al. 2007; Tournier et al. 2017). Additionally, density effects on dispersal may vary. Where some studies find that amphibians tend to remain at populated breeding sites (Buxton and Sperry 2017; Cayuela et al. 2018; Gamble et al. 2007), others have shown that amphibians vacate densely populated sites to avoid breeding and resource competition (Hellriegel 2000). Reasons for dispersing to a new pond for breeding are also unknown but may be attributed to environmental and social factors (Cayuela et al. 2020). If breeding ponds are sufficiently close to each other, individuals may use perceptual or olfactory cues to locate favorable breeding habitats (Buxton and Sperry 2017). However, many studies show that dispersal strategies are described by the linear distances between ponds, indicating that amphibians exhibit random movement when searching for a suitable breeding pond (Boualit et al. 2019;

In this paper, we develop sensitivity and elasticity analysis of the contribution metric for general metapopulation matrix models, and apply our methods to a Columbia spotted frog population. To explore different dispersal behaviors in this system, we use a distance-based dispersal strategy with a range of density-dependent dispersal rates for two dispersal strategies based on landscape geography. We first analyze pond contributions, pond sizes, and total population sizes at equilibrium to understand metapopulation dynamics with parameter estimates informed by spotted frog literature. We assume constant demographic rates except for larval survival which is regulated based on egg density and pond hydroperiod (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012). We apply elasticity analysis to understand the effects of pond-specific survival probabilities on pond contributions and populations. We find that density-dependent dispersal type (positive versus negative density-dependence) plays a large role in both the perturbed and unperturbed systems. Without perturbations, pond contributions have much more varied results under positive density-dependent dispersal than negative density-dependent dispersal and the arrangement of contributions differ under different landscape networks. Elasticity analysis shows that the direct effects of pond perturbations on pond contributions are smallest under negative density-dependent dispersal but change signs and are much larger under constant and negative density-dependent dispersal. On the other hand, pond sizes all experience positive shifts with the largest changes occurring under negative density-dependent dispersal. Additionally, we find that total population elasticities are robust to dispersal type and landscape network, indicating that the total population may experience similar changes regardless of dispersal strategies. We use these results to determine best management practices for encouraging pond heterogeneity across the metapopulation.

We organize this paper as follows. In section 2, we define the matrix metapopulation model using vec-permutation decomposition methods. Then we introduce the patch contribution metric to quantify habitat importance based on local demography and dispersal
rates, and develop sensitivity and elasticity analysis on population sizes and patch contributions. In section 3, we apply these methods to the Columbia spotted frog population in the Selway-Bitterroot Wilderness. We assume that larval survival is regulated by the number of eggs hatched, and dispersal is defined with density-dependent emigration rates and distance-dependent dispersal distributions. We also use two dispersal strategies based on the landscape geography; a full system that allows individuals to travel to any pond at any time, and a partial system that restricts movement based on the waterway network. We then define the formulas used to calculate sensitivities to population vectors and vital rate parameters in the Columbia spotted frog population. We analyze patch contributions and equilibrium populations in the unperturbed model to understand the original system then compare those results to their elasticities. These results are used to make suggestions for potential management action. Finally, we conclude with a discussion of our results and provide context to existing literature and ecology applications.

4.2 Methods

4.2.1 The matrix model

We begin by constructing a discrete-time matrix model composed of two matrix components; a demographic matrix that describes within-patch life histories and a dispersal matrix that describes between-patch interactions. The demographic matrix is defined by a patch-based population vector while the dispersal matrix is defined by a stage-based population vector. We denote vector and matrix variables with bolded letters and scalar variables with unbolded letters. For a metapopulation with \( k \) patches and \( n \) life stages, the patch-based population vector is defined as \( \mathbf{u}(t) = [\mathbf{u}_1(t), \mathbf{u}_2(t), \ldots, \mathbf{u}_k(t)]^T \), where each subvector, \( \mathbf{u}_x(t) \in \mathbb{R}^{n,1} \), represents the population distributions for all life stages in patch \( x \) at time \( t \). The stage-based population vector is defined as \( \mathbf{\hat{u}}(t) = [\mathbf{\hat{u}}_1(t), \mathbf{\hat{u}}_2(t), \ldots, \mathbf{\hat{u}}_n(t)]^T \), where each subvector, \( \mathbf{\hat{u}}_y(t) \in \mathbb{R}^{k,1} \), represents the population distribution of life stage \( y \) across all patches at time \( t \). The demographic and dispersal components are then related by the
vec-permutation matrix

\[
V = \sum_{x=1}^{k} \sum_{y=1}^{n} E_{xy} \otimes E_{xy}^T,
\]

(4.1)

where \( E_{xy} \in \mathbb{R}^{k,n} \) has a one in the \((x, y)\) position and zeros elsewhere, and \( \otimes \) is the Kronecker product (Henderson and Searle 1981; Hunter and Caswell 2005). The vec-permutation matrix relates the two population vectors with \( \hat{u} = Vu \) and \( u = V^T \hat{u} \).

We define block matrices for the demographic processes, \( B \), and dispersal processes, \( M \),

\[
B = \begin{bmatrix}
B_1 & 0 & \ldots & 0 \\
0 & B_2 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & B_k
\end{bmatrix},
\]

\[
M = \begin{bmatrix}
M_1 & 0 & \ldots & 0 \\
0 & M_2 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & M_n
\end{bmatrix},
\]

(4.2)

where each diagonal submatrix represents demographic processes in patch \( x \) (\( B_x \)), and dispersal processes in life stage \( y \) (\( M_y \)). In other words, each demographic submatrix, \( B_x \in \mathbb{R}^{n,n} \), represents growth, survival, and fecundity for patch \( x \), and each dispersal submatrix, \( M_y \in \mathbb{R}^{k,k} \), represents the distribution of individuals in life stage \( y \) to all other patches.

The demographic and dispersal matrices are combined using the vec-permutation matrix. The formulation of the governing projection matrix depends on whether the population vector is organized by patch or stage and the order in which demography and dispersal occur. We assume a patch-based model with demographic processes occurring prior to dispersal,

\[
A = V^T M V B.
\]

(4.3)

The projection matrix \( A \) can be defined as a block-matrix with \( n \times n \) blocks. Each diagonal submatrix, \( A_{xx} \in \mathbb{R}^{n,n} \), represents the demographic processes of individuals that remain in patch \( x \), and every other submatrix, \( A_{px} \in \mathbb{R}^{n,n} \), represents the demographic processes of individuals that disperse from patch \( x \) to patch \( p \). This matrix combines both demographic
and dispersal processes and projects the population by one time step using

\[ u(t + 1) = A u(t). \]  

(4.4)

Note that the projection matrix can be arranged by stage or patch with demography or dispersal processes occurring first through cyclic permutations. Changing the order of these processes does not affect eigenvalue spectra; however, population results will change based on census timing (Caswell 2001; Hunter and Caswell 2005).

4.2.2 The contribution metric

The contribution of individuals in a life history stage at a patch is the sum of all demographic and emigration processes, and is calculated by summing over all rows in each column of the projection matrix,

\[ C(t) = \mathbf{1}^T_k A(t), \]  

(4.5)

where \( C(t) = [C_1(t), C_2(t), \cdots, C_k(t)] \) for subvectors \( C_x(t) \) that represent contributions of all life stages in patch \( x \).

4.2.3 Habitat contribution metric

Given the population vector in patch \( x \) (\( u_x(t) \)), the total number of individuals in patch \( x \) at time \( t + 1 \) can be expressed as the sum of the number of individuals that remain in the patch and the number of individuals dispersing into the patch,

\[ U_x(t + 1) = \mathbf{1}^T_n A_{xx}(t) u_x(t) + \sum_{p \neq x} \mathbf{1}^T_n A_{xp}(t) u_p(t), \]  

(4.6)
where $\mathbf{1}_n \in \mathbb{R}^{n,1}$ is a column vector of ones. Assuming that all life stages have equal weights, we define the per capita growth rate of patch $x$ (Runge et al. 2006),

$$
\frac{U_x(t + 1)}{U_x(t)} = \frac{\mathbf{1}_n^T \mathbf{A}_{xx}(t) \mathbf{u}_x(t)}{U_x(t)} + \sum_{p \neq x} \frac{\mathbf{1}_n^T \mathbf{A}_{xp}(t) \mathbf{u}_p(t)}{U_x(t)}
$$

$$
\lambda_x(t) = R_x(t) + I_x(t),
$$

where $\lambda_x(t) = \frac{U_x(t+1)}{U_x(t)}$ is the growth rate in patch $x$, $R_x(t) = \frac{\mathbf{1}_n^T \mathbf{A}_{xx}(t) \mathbf{u}_x(t)}{U_x(t)}$ is the per capita self-recruitment rate, representing the ability for a patch to retain itself without immigration, and $I_x(t) = \frac{\sum_{p \neq x} \mathbf{1}_n^T \mathbf{A}_{xp}(t) \mathbf{u}_p(t)}{U_x(t)}$ is the per capita immigration rate. This is well-defined for all nonzero patch sizes.

The degree to which a patch can maintain its population while contributing to the global population is defined as

$$
C_x(t) = \frac{\mathbf{1}_n^T \mathbf{A}_{xx}(t) \mathbf{u}_x(t)}{U_x(t)} + \frac{\sum_{p \neq x} \mathbf{1}_n^T \mathbf{A}_{xp}(t) \mathbf{u}_p(t)}{U_x(t)}
$$

$$
= R_x(t) + E_x(t),
$$

(4.7)

where $E_x(t) = \frac{\sum_{p \neq x} \mathbf{1}_n^T \mathbf{A}_{xp}(t) \mathbf{u}_p(t)}{U_x(t)}$ is the per capita emigration rate normalized by the population size in patch $x$ (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). An alternative form for equation (4.7) can be written by substituting $R_x(t) = \lambda_x(t) - I_x(t)$,

$$
C_x(t) = \lambda_x(t) - I_x(t) + E_x(t).
$$

(4.8)

Equations (4.7) and (4.8) are the per capita habitat contribution for patch $x$ and measures the relative contribution of a patch to the metapopulation (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). At equilibrium, the contribution metric is closely connected to the per capita growth rate and stable state distribution (Runge et al. 2006) Patches with a contribution greater than 1 ($C_x > 1$) are classified as sources while patches with a contribution less than 1 ($C_x < 1$) are classified as sinks. At equilibrium ($\mathbf{u}(t + 1) = \mathbf{u}(t)$), $\lambda_x = 1$ and $C_x$ can be determined by the dispersal rates, $I_x$ and $E_x$. Specifically, if $E_x > I_x$, 

$\lambda_x = 1$ and $C_x$ can be determined by the dispersal rates, $I_x$ and $E_x$. Specifically, if $E_x > I_x$, 

$\lambda_x = 1$ and $C_x$ can be determined by the dispersal rates, $I_x$ and $E_x$. Specifically, if $E_x > I_x$, 

$\lambda_x = 1$ and $C_x$ can be determined by the dispersal rates, $I_x$ and $E_x$. Specifically, if $E_x > I_x$,
the patch is classified as a source, and if $E_x < I_x$, the patch is classified as a sink.

### 4.2.4 Sensitivity of population vectors

We follow the procedure used by (Caswell 2009; 2012) to perform sensitivities of patch equilibria to parameters in $A$. For a projection matrix dependent on density and some parameter $\theta$, we can write $A = A(u, \theta)$. The governing equation at equilibrium is

$$u^* = A^*(u^*, \theta)u^*, \quad (4.9)$$

where $A^*(u^*, \theta)$ is the projection matrix at equilibrium and $u^*$ is the equilibrium population vector. Taking the derivative of $u^*$ with respect to $\theta$, we get

$$\frac{d u^*}{d \theta} = \frac{d A^*}{d \theta} u^* + A^* \frac{d u^*}{d \theta}. \quad (4.10)$$

We apply the vec operator (vec) to the first term on the right-hand side in equation (4.10). The vec operator transforms matrices into column vectors by stacking each column and is used to take derivatives of a matrix with respect to a vector (Caswell 2009). It is known (Magnus and Neudecker 1985; Neudecker 1969) that applying the vec operator to the product of a matrix with a vector gives

$$\text{vec} \left( \frac{d A^*}{d \theta} u^* \right) = \left( (u^*)^T \otimes I_{kn} \right) \frac{d \text{vec} A^*}{d \theta}. \quad (4.11)$$

Note that the vec operator on a vector results in the original vector, vec$u^* = u^*$. Since $A^*$ is density-dependent, we use the chain rule to solve for $\frac{d \text{vec} A^*}{d \theta}$,

$$\frac{d \text{vec} A^*}{d \theta} = \frac{\partial \text{vec} A^*}{\partial \theta} + \frac{\partial \text{vec} A^*}{\partial (u^*)^T} \frac{d u^*}{d \theta}. \quad (4.12)$$

where $\frac{\partial \text{vec} A^*}{\partial (u^*)^T} = \begin{bmatrix} \frac{\partial \text{vec} A^*}{\partial u^*_{11}} & \frac{\partial \text{vec} A^*}{\partial u^*_{12}} & \ldots & \frac{\partial \text{vec} A^*}{\partial u^*_{kn}} \end{bmatrix}$ combines the vector form of the derivative of every entry in $A^*$ with respect to every entry in $u^*$. 
Finally, substituting equations (4.11) and (4.12) into equation (4.10) and solving for \( \frac{d{u^*}}{d\theta} \), we get

\[
\frac{d{u^*}}{d\theta} = \left( (u^*)^T \otimes I_{kn} \right) \left( \frac{\partial \text{vec} A^*}{\partial \theta} + \frac{\partial \text{vec} A^*}{\partial (u^*)^T} \frac{d{u^*}}{d\theta} \right) + A^* \frac{d{u^*}}{d\theta}
\]

\[
\frac{d{u^*}}{d\theta} = \left( I_{kn} - ((u^*)^T \otimes I_{kn}) \frac{\partial \text{vec} A^*}{\partial (u^*)^T} - A^* \right)^{-1} \left( (u^*)^T \otimes I_{kn} \right) \frac{\partial \text{vec} A^*}{\partial \theta}.
\]

(4.13)

Note that \( (I_{kn} - ((u^*)^T \otimes I_{kn}) \frac{\partial \text{vec} A^*}{\partial (u^*)^T} - A^*) \) is invertible as long as the linearization of \( A^* \) does not contain any eigenvalues equal to 1, i.e., +1 bifurcations (transcritical, saddle node, pitchfork) are not well defined (Caswell 2009).

Solving for \( \frac{d{u^*}}{d\theta} \) has some useful applications. Sensitivities of the total population are solved by summing over all entries,

\[
\frac{dU^*}{d\theta} = \sum_{x=1}^{k} \sum_{y=1}^{n} \frac{d{u^*}_{xy}}{d\theta},
\]

(4.14)

and patch sensitivities by summing over all life stages in patch \( x \),

\[
\frac{dU^*_x}{d\theta} = \sum_{y=1}^{n} \frac{d{u^*}_{xy}}{d\theta}.
\]

(4.15)

### 4.2.5 Sensitivity of patch contributions

Using the results above, we now solve for sensitivity of patch contributions to small parameter perturbations. Given the equilibrium patch contribution, \( C^*_x = R^*_x + E^*_x \), the derivative with respect to \( \theta \) is

\[
\frac{dC^*_x}{d\theta} = \frac{1}{(U^*_x)^2} \left[ I_n^T \frac{dA^*_{xx}}{d\theta} u^*_x + I_n^T A^*_{xx} \frac{d{u^*}_x}{d\theta} + \left( \sum_{p \neq x} I_n^T \frac{dA^*_{px}}{d\theta} u^*_x + I_n^T A^*_{px} \frac{d{u^*}_x}{d\theta} \right) \right] U^*_x
\]

\[
\frac{1}{(U^*_x)^2} \left[ I_n^T A^*_{xx} u^*_x + \sum_{p \neq x} I_n^T A^*_{px} u^*_x \right] \frac{dU^*_x}{d\theta}.
\]

(4.16)
Note that the final term is $C^*_x$, so we can simplify the equation to

$$\frac{dC^*_x}{d\theta} = \frac{1}{U^*_x} \left[ \sum_{p=1}^k \tilde{1}^T_n \frac{dA^*_p x}{d\theta} u^*_x + \sum_{p=1}^k \tilde{1}^T_n A^*_p x \frac{du^*_x}{d\theta} - C^*_x \frac{dU^*_x}{d\theta} \right]. \quad (4.17)$$

Finally, we use the chain rule on $dA^*_p x / d\theta$ to get

$$\frac{dC^*_x}{d\theta} = \frac{1}{U^*_x} \left[ \sum_{p=1}^k (u^*_x)^T \otimes \tilde{1}^T_n \left( \frac{\partial vec A^*_p x}{\partial \theta} + \frac{\partial vec A^*_p x}{\partial u^*_x} \frac{du^*_x}{d\theta} \right) + \sum_{p=1}^k \tilde{1}^T_n A^*_p x \frac{du^*_x}{d\theta} - C^*_x \frac{dU^*_x}{d\theta} \right]. \quad (4.18)$$

This is the equation for sensitivity analysis of the contribution metric.

### 4.2.6 Elasticity

It can be useful to compare proportional parameter change rather than absolute change, especially when vital rates are measured on different scales. This is done with elasticity analysis by scaling sensitivity by the ratio between the perturbed parameter and the output (Caswell 2001; Hunter and Caswell 2005). Specifically, the elasticity of a population is $\frac{\theta}{u^*_x} \frac{du^*_x}{d\theta}$ and the elasticity of patch contribution is $\frac{\theta}{C^*_x} \frac{dC^*_x}{d\theta}$.

### 4.3 Application to a Columbia spotted frog population

#### 4.3.1 Study area

We apply our analysis to a Columbia spotted frog population located in the Selway-Bitterroot Wilderness, Montana. For this system, we define patches as breeding ponds located in the area (Figure 4.2) that are classified based on their waterbody hydroperiod (permanent, semi-permanent, ephemeral). Permanent ponds are characterized by waterbodies deeper than 1 meter, never dry out, and able to support an abundance of larvae, semi-permanent ponds may dry out in particularly warm summers but also have the highest recruitment rates, and ephemeral ponds are characterized by shallow water no deeper than 1 meter and can dry out as early as August, preventing all larvae from developing into
Table 4.1: Symbols used in the model and analysis.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of life stages</td>
</tr>
<tr>
<td>$k$</td>
<td>number of patches</td>
</tr>
<tr>
<td>$u$</td>
<td>patch-based global population vector</td>
</tr>
<tr>
<td>$u_x$</td>
<td>n-stage population vector for patch $x$</td>
</tr>
<tr>
<td>$U_x$</td>
<td>population size of patch $x$</td>
</tr>
<tr>
<td>$U$</td>
<td>total population size</td>
</tr>
<tr>
<td>$A$</td>
<td>$nk \times nk$ projection matrix</td>
</tr>
<tr>
<td>$A_{op}$</td>
<td>$n \times n$ projection submatrix for individuals dispersing from patch $p$ to patch $o$</td>
</tr>
<tr>
<td>$B$</td>
<td>$n \times k$ demographic matrix</td>
</tr>
<tr>
<td>$B_x$</td>
<td>n-stage demographic matrix for patch $x$</td>
</tr>
<tr>
<td>$M$</td>
<td>$n \times k$ dispersal matrix</td>
</tr>
<tr>
<td>$M_y$</td>
<td>$k$-patch dispersal matrix for life stage $y$</td>
</tr>
<tr>
<td>$\sigma_y$</td>
<td>$k \times k$ diagonal matrix of dispersal probabilities for life stage $y$</td>
</tr>
<tr>
<td>$D$</td>
<td>$k \times k$ matrix of distribution probabilities of dispersing individuals</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>pathway weight matrix</td>
</tr>
<tr>
<td>$V$</td>
<td>vec-permutation matrix</td>
</tr>
<tr>
<td>$I_k$</td>
<td>$k \times k$ identity matrix</td>
</tr>
<tr>
<td>$\vec{1}_n$</td>
<td>$n \times 1$ vector of ones</td>
</tr>
<tr>
<td>$H_x$</td>
<td>density-dependent first-year survival for patch $x$</td>
</tr>
<tr>
<td>$L_x$</td>
<td>maximum larval survival in patch $x$</td>
</tr>
<tr>
<td>$K_x$</td>
<td>larval carrying capacity for patch $x$</td>
</tr>
<tr>
<td>$\gamma_x$</td>
<td>density-dependent exponent for larval survival in patch $x$</td>
</tr>
<tr>
<td>$\lambda_x$</td>
<td>growth rate for patch $x$</td>
</tr>
<tr>
<td>$C_x$</td>
<td>per capita patch contribution for patch $x$</td>
</tr>
<tr>
<td>$R_x$</td>
<td>per capita self-recruitment rate for patch $x$</td>
</tr>
<tr>
<td>$I_x$</td>
<td>per capita immigration rate for patch $x$</td>
</tr>
<tr>
<td>$E_x$</td>
<td>per capita emigration rate for patch $x$</td>
</tr>
<tr>
<td>vec$(\cdot)$</td>
<td>vec operator</td>
</tr>
</tbody>
</table>
metamorphs (Loman 2002; McCaffery et al. 2014). The seven breeding ponds are classified into three permanent ponds \((P_1, P_2, P_3)\), two semi-permanent ponds \((S_1, S_2)\), and two ephemeral ponds \((E_1, E_2)\) (Figure 4.2).

We assume that the Columbia spotted frog develops in three distinct life stages; juvenile \((j)\), subadult \((s)\), and adult \((a)\) (McCaffery et al. 2014; McCaffery and Maxell 2010; McCaffery et al. 2012). New recruits who survive their first year develop into juveniles. Subadults and adults are similar in size with similar vital rates; however, only adults are capable of breeding. We define demographic processes for all life stages where first-year individuals experience density-dependent growth based on their natal pond. For notation, we define ponds \(x = \{P_1, P_2, P_3, S_1, S_2, E_1, E_2\}\) and life stages \(y = \{j, s, a\}\).

### 4.3.2 Within pond demographics

Demographic processes for pond \(x\) are defined by the submatrix

\[
B_x = \begin{bmatrix}
(1 - P_{js})S_j & 0 & FH_x S_m \\
0 & (1 - P_{sa})S_s & 0 \\
0 & P_{sa}S_s & S_a
\end{bmatrix},
\]

(4.19)

with juvenile, subadult, and adult survival \((S_j, S_s, \text{and } S_a)\), probability of transitioning from juvenile to subadult \((P_{js})\) and subadult to adult \((P_{sa})\), fecundity \((F)\), pond-specific egg to metamorph survival \((H_x)\), and metamorph to yearling survival \((S_m)\).

We assume all vital rates are constant except for egg to metamorph survival, \(H_x\), which is based on the Hassell competition model (Halley et al. 1996; Vonesh and De la Cruz 2002; Willson and Hopkins 2013; Willson et al. 2012),

\[
H_x(t) = \frac{L_x}{\left(1 + \frac{F_{a_x}(t)}{K_x}\right)^{\gamma_x}},
\]

(4.20)

where \(L_x\) is the maximum larval survival, \(K_x\) is the larval carrying capacity, and \(\gamma_x\) is the density-dependent exponent for pond \(x\). This model relies on the number of eggs laid as a
function of adults in pond \( x \) \( (F_{a_x}) \).

Competition in the Hassell equation depends on the density-dependent exponent, \( \gamma_x \), where \( \gamma_x \) is inversely proportional to waterbody size which affects resource allocation (Anazawa 2019). For smaller values of \( \gamma_x \), individuals undergo ‘contest’ competition while for large values of \( \gamma_x \), individuals undergo ‘scramble’ competition (Hassell 1975). In contest competition, some proportion of individuals receive sufficient resources for survival while the rest do not survive. In scramble competition, all resources are ‘shared’ such that individuals either all die out or all survive. For extreme cases, ideal contest competition \( (\gamma_x = 1) \) defines the Ricker model while ideal scramble competition \( (\gamma_x \rightarrow \infty) \) defines the Beverton-Holt model (Anazawa 2019).

### 4.3.3 Dispersal strategies

We divide dispersal processes into two components: outgoing dispersal probabilities (probability of an individual leaving a pond) and dispersal distributions (allocation of dispersing individuals to all other ponds). A dispersal probability matrix, \( \sigma_y \), defines the probability for individuals in life stage \( y \) to disperse from their current pond, and a proportional movement matrix, \( D \), defines the distribution of dispersing individuals to all other ponds. These are defined as

\[
\sigma_y = \begin{bmatrix}
\sigma_{1y} & 0 & \ldots & 0 \\
0 & \sigma_{2y} & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & \sigma_{ky}
\end{bmatrix}, \quad 
D = \begin{bmatrix}
0 & D_{12} & \ldots & D_{1k} \\
D_{21} & 0 & \ldots & D_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
D_{k1} & D_{k2} & \ldots & 0
\end{bmatrix}, \quad (4.21)
\]

where \( \sigma_{xy} \) is the probability that an individual in life stage \( y \) disperses from pond \( x \), and \( D_{px} \) represents the proportion of dispersing individuals that disperse from pond \( x \) to pond \( p \). Note that without dispersal mortality, each column in \( D \) sums to unity.

The dispersal submatrix for individuals in life stage \( y \) is defined as

\[
M_y = (I_k - \sigma_y) + D \sigma_y, \quad (4.22)
\]
where $I_k \in \mathbb{R}^{k,k}$ is an identity matrix. The term, $(I_k - \sigma_y)$, represents the proportion of individuals remaining in the source pond, and $D \sigma_y$ represents the distribution of dispersing individuals. Note that each column in $M_y$ sums to unity.

We assume dispersal in only the juvenile stage and zero dispersal in subadults and adults, i.e., $\sigma_s$ and $\sigma_a$ are zero matrices. We use the density-dependent dispersal model defined in Ylikarjula et al. (2000) where dispersal is dependent on the size of the population in the source pond. Given the number of total individuals in pond $x$, $U_x(t) = \sum_{y=1}^{n} u_{xy}(t)$, the proportion of juveniles dispersing from pond $x$ is

$$\sigma_{xj}(t) = \frac{\sigma_{\text{max}}}{1 + e^{\beta(\kappa_x - U_x(t))}},$$

(4.23)

where $\sigma_{\text{max}}$ is the maximum juvenile dispersal probability, the steepness of the curve depends on the shape parameter, $\beta$, and the inflection point is determined by $\kappa_x$. Negative $\beta$ represents negative density-dependent dispersal (fewer individuals disperse for larger populations) and positive $\beta$ represents positive density-dependent dispersal (more individuals disperse for larger populations). $\beta = 0$ represents constant dispersal such that $\sigma_{xj}(t) = \sigma_{\text{max}}/2$. See Figure 4.1A for an example of different $\beta$ values under a range of densities with an inflection point at $\kappa = 50$.

Distance-based dispersal is defined by the negative exponential function based on linear distances between breeding ponds (Breden 1987; Halley et al. 1996; Zamberletti et al. 2018). This assumes that dispersing individuals have a higher probability of dispersing to ponds closer to their current breeding pond than ponds further away. This is defined as

$$D_{px} = \begin{cases} 
0 & p = x \\
\frac{\alpha_{px}}{d_x} e^{-\frac{d_{px}}{r}} & p \neq x
\end{cases}, \quad \bar{d}_x = \sum_{i \neq x} \alpha_{ix} e^{-\frac{d_{ix}}{r}},$$

(4.24)

where $r$ is the mean dispersal distance, $\alpha_{px}$ is a weight for individuals moving from pond $x$ to pond $p$, $d_{px}$ is the distance from pond $x$ to pond $p$, and $\bar{d}_x$ is the normalizing constant to establish unity in all column sums. The mean dispersal distance, $r$, determines the effects
Fig. 4.1: Examples of how dispersal probabilities change as $\beta$ and $r$ change. Dispersal from a patch (A) is negative density-dependent (dash line), positive density-dependent (solid line), or constant (dash-dot line) when $\beta$ is negative, positive, or zero. As $r$ increases (B), the probability of dispersing to all ponds becomes more uniform (dash-dot line) whereas low $r$ values distribute individuals to nearby ponds (dash line). We set $\sigma_{\text{max}} = 0.5$ in A and used $e^{-d/r}$ in B where $d$ is the distance between patches.

that distances between ponds have on the dispersal behavior. For small $r$ values, dispersing individuals are more likely to disperse to nearby ponds than ponds further away. As $r$ increases, the dispersal distribution becomes more uniform such that dispersing individuals are just as likely to disperse to nearby ponds as they are to ponds further away. See Figure 4.1B for an example of high and low mean dispersal distances.

4.3.4 Fully vs partially connected systems

Amphibians are known to travel along waterways and the decision to disperse to a new pond may depend on landscape features. We define two dispersal schemes based on metapopulation geography; a fully connected system where individuals have access to all other ponds, and a partially connected system where individuals move along the waterway network (Figure 4.2). Weights are defined in $\alpha_{op}$ where $\alpha_{op} = 1$ indicates a directional connected path between ponds $p$ and $o$ while $\alpha_{op} = 0$ indicates that the path is disconnected. Figure 4.2B shows the pathways for the partially connected system where individuals move along the arrows determined by the stream network. The pathway weight matrix for the
partially connected system is

\[ \mathbf{\alpha} = \begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1
\end{bmatrix}. \] (4.25)

Note that the pathway weight matrix for the fully connected system is a matrix of all ones.

Table 4.2: Distances (m) between each pond in the study site. Measurements were taken by using the Google Maps linear distance measuring tool between center points of each pond.

<table>
<thead>
<tr>
<th></th>
<th>(P_1)</th>
<th>(P_2)</th>
<th>(P_3)</th>
<th>(S_1)</th>
<th>(S_2)</th>
<th>(E_1)</th>
<th>(E_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(P_1)</td>
<td>0</td>
<td>233.9</td>
<td>1160.4</td>
<td>844.7</td>
<td>330.0</td>
<td>293.1</td>
<td>597.6</td>
</tr>
<tr>
<td>(P_2)</td>
<td>233.9</td>
<td>0</td>
<td>929.5</td>
<td>637.1</td>
<td>101.6</td>
<td>154.2</td>
<td>441.4</td>
</tr>
<tr>
<td>(P_3)</td>
<td>1160.4</td>
<td>929.5</td>
<td>0</td>
<td>626.6</td>
<td>845.9</td>
<td>953.7</td>
<td>716.2</td>
</tr>
<tr>
<td>(S_1)</td>
<td>844.7</td>
<td>637.1</td>
<td>626.6</td>
<td>0</td>
<td>537.1</td>
<td>556.0</td>
<td>778.3</td>
</tr>
<tr>
<td>(S_2)</td>
<td>330.0</td>
<td>101.6</td>
<td>845.9</td>
<td>537.1</td>
<td>0</td>
<td>138.3</td>
<td>437.4</td>
</tr>
<tr>
<td>(E_1)</td>
<td>293.1</td>
<td>154.2</td>
<td>953.7</td>
<td>556.0</td>
<td>138.3</td>
<td>0</td>
<td>571.6</td>
</tr>
<tr>
<td>(E_2)</td>
<td>597.6</td>
<td>441.4</td>
<td>716.2</td>
<td>778.3</td>
<td>437.4</td>
<td>571.6</td>
<td>0</td>
</tr>
</tbody>
</table>

4.3.5 Sensitivity analysis

The sensitivity analysis defined above requires derivatives of the projection matrix with respect to \(u\) and \(\theta\). We can decompose \(\frac{\partial \text{vec} \mathbf{A}^*}{\partial (u^*)^T}\) to

\[
\frac{\partial \text{vec} \mathbf{A}^*}{\partial (u^*)^T} = \left( (\mathbf{V}\mathbf{B}^*)^T \otimes \mathbf{V}^T \right) \frac{\partial \text{vec} \mathbf{M}^*}{\partial (u^*)^T} + \left( \mathbf{I}_{kn} \otimes (\mathbf{V}^T\mathbf{M}^*\mathbf{V}) \right) \frac{\partial \text{vec} \mathbf{B}^*}{\partial (u^*)^T},
\] (4.26)

where \(\mathbf{M}^*\) and \(\mathbf{B}^*\) are the dispersal and demographic matrices at equilibrium.

Similarly, the derivative \(\frac{\partial \text{vec} \mathbf{A}^*}{\partial \theta}\) can be decomposed to

\[
\frac{\partial \text{vec} \mathbf{A}^*}{\partial \theta} = \left( (\mathbf{V}\mathbf{B}^*)^T \otimes \mathbf{V}^T \right) \frac{\partial \text{vec} \mathbf{M}^*}{\partial \theta} + \left( \mathbf{I}_{kn} \otimes (\mathbf{V}^T\mathbf{M}^*\mathbf{V}) \right) \frac{\partial \text{vec} \mathbf{B}^*}{\partial \theta},
\] (4.27)
Fig. 4.2: Individuals may disperse to ponds based on waterway networks. Our system consists of seven ponds with a river network connecting some of the ponds to each other (A). Individuals that choose to disperse may only be able to disperse to the nearest pond connected by a waterway (B).

We use these decompositions to more readily solve for our derivatives.

**Sensitivity to population vector**

The only density-dependent terms in $M^*$ are in the juvenile stage submatrix, $M^*_j$; hence, $M^*_s$ and $M^*_a$ are zero matrices while the derivative of $M^*_j$ with respect to individuals in pond $x$ and stage $y$ is

$$\frac{\partial M^*_j}{\partial u_{xy}} = (D - I_k) \frac{\partial \sigma_j}{\partial u_{xy}}, \quad (4.28)$$

where diagonal entries in $\frac{\partial \sigma_j}{\partial u_{xy}}$ are defined as,

$$\frac{\partial \sigma_{\bar{x}j}}{\partial u_{x\bar{y}}} = \begin{cases} 
\sigma_{\text{max}} \beta e^{\beta (\bar{x} - U^*_\bar{x})} 
\left(1 + e^{\beta (\bar{x} - U^*_\bar{x})}\right)^2, & \bar{x} = x \\
0, & \bar{x} \neq x 
\end{cases},$$

for $\bar{x} \in \{P_1, P_2, \ldots, E_2\}$. 
The demographic matrix $B^*$ is dependent only on the adult population in each pond,

$$
\frac{\partial B^*_x}{\partial u_{xy}} = \begin{bmatrix}
0 & 0 & F \frac{\partial H_x}{\partial u_{xy}} S_m \\
0 & 0 & 0 \\
0 & 0 & 0 
\end{bmatrix}, \quad (4.29)
$$

where,

$$
\frac{\partial H_x}{\partial u_{xy}} = \begin{cases}
\frac{-L_x \gamma_x F}{K_x \left(1 + \frac{F a_x}{K_x}\right)^{\gamma_x + 1}}, & \bar{x} = x, \ y = a \\
0, & \bar{x} \neq x, \ y = j, s
\end{cases}
$$

Sensitivity to larval survival parameters

Breeding ponds are largely impacted by environmental and anthropogenic changes which have direct implications for larval survival. We find sensitivities of patch contribution and population sizes to the larval survival parameters $L_x, K_x,$ and $\gamma_x$. Note that in the projection matrix, the dispersal submatrix is not dependent on vital rates so $\partial M^*_x / \partial \theta$ is a zero matrix while the demographic submatrix $B^*_x$ is

$$
\frac{\partial B^*_x}{\partial \theta} = \begin{bmatrix}
0 & 0 & F \frac{\partial H_x}{\partial \theta} S_m \\
0 & 0 & 0 \\
0 & 0 & 0 
\end{bmatrix}. \quad (4.30)
$$

The specific derivatives are defined by the following equations,

- For $L_x$,

$$
\frac{\partial H_x}{\partial L_x} = \frac{1}{\left(1 + \frac{F a_x}{K_x}\right)^{\gamma_x}}. \quad (4.31)
$$

- For $K_x$,

$$
\frac{\partial H_x}{\partial K_x} = \frac{L_x \gamma_x F a_x}{K_x^2 \left(1 + \frac{F a_x}{K_x}\right)^{\gamma_x + 1}}. \quad (4.32)
$$
Table 4.3: Default vital rates and values used in the model. Vital rates are based on ranges defined in McCaffery and Maxell (2010).

<table>
<thead>
<tr>
<th>Vital rate</th>
<th>Symbol</th>
<th>Perm</th>
<th>Semi-perm</th>
<th>Eph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fecundity</td>
<td>$F$</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Metamorph-to-yearling survival</td>
<td>$S_m$</td>
<td>0.45</td>
<td>0.45</td>
<td>0.45</td>
</tr>
<tr>
<td>Juvenile survival</td>
<td>$S_j$</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>Subadult survival</td>
<td>$S_s$</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>Adult survival</td>
<td>$S_a$</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>Transition from juvenile to subadult</td>
<td>$P_{js}$</td>
<td>0.3</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Transition from subadult to adult</td>
<td>$P_{sa}$</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>Maximum larval survival</td>
<td>$L_x$</td>
<td>0.099</td>
<td>0.3</td>
<td>0.052</td>
</tr>
<tr>
<td>Larval carrying capacity</td>
<td>$K_x$</td>
<td>425</td>
<td>371</td>
<td>998</td>
</tr>
<tr>
<td>Maximum larval survival</td>
<td>$\gamma_x$</td>
<td>0.615</td>
<td>1.11</td>
<td>0.746</td>
</tr>
<tr>
<td>Dispersal carrying capacity</td>
<td>$\kappa_x$</td>
<td>50</td>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>Maximum juvenile dispersal</td>
<td>$\sigma_{\text{max}}$</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

- For $\gamma_x$,

$$
\frac{\partial H_x}{\partial \gamma_x} = - \frac{L_x \ln \left( 1 + \frac{F a_{x}}{K_x} \right)}{\left( 1 + \frac{F a_{x}}{K_x} \right)^{\gamma_x}}. 
$$

(4.33)

Note that for $\bar{x} \neq x$, the derivatives equal 0.

4.4 Results

4.4.1 Fitting density-dependent larval survival to data

We use egg count and egg to metamorph survival data that were collected by McCafferey (2010) between the years 2001 and 2009 although survival data for the year 2005 is missing due to forest fires in the region. We exclude years that did not produce any recruits as they are typically due to climate effects (e.g., dry summers and early winters). Specifically, we exclude years 2008 and 2009 from $P_1$, 2002, 2006, 2008, and 2009 from $P_2$, 2002, 2008, and 2009 from $P_3$, 2009 from $S_2$, 2003, 2006, and 2007 from $E_1$, and 2006, 2007, 2008, and 2009 from $E_2$.

We fit the Hassell competition model by minimizing the root mean squared error with all pond-specific egg to metamorph survival data. We use interior-point optimization with
Fig. 4.3: We fit the Hassell competition model to egg count data for each pond type to define pond-specific larval survival. Permanent ponds (A) have high carrying capacities where larval survival is less affected by pond densities. Semi-permanent ponds (B) have the highest maximum survival but cannot sustain a large number of recruits. Ephemeral ponds (C) have consistently lower survival than permanent and semi-permanent ponds.

function tolerance $1e^{-6}$. We initialize the minimizer with a range of parameters using Latin hypercube sampling to account for multiple local minima. Specifically, we sample from 1000 evenly-spaced values for the following ranges in the larval survival parameters: $0.01 \leq L_x \leq 0.3$, $10 \leq K_x \leq 1000$, and $0.01 \leq \gamma_x \leq 100$. We find that two local minima occur for each pond type (see appendix). Fitted parameters for global minima are $L_P = 0.099$, $K_P = 425$, $\gamma_P = 0.615$ in permanent ponds, $L_S = 0.300$, $K_S = 371$, $\gamma_S = 1.11$ in semi-permanent ponds, and $L_E = 0.052$, $K_E = 998$, $\gamma_E = 0.746$ in ephemeral ponds. Figure 4.3 shows pond-specific survival data (ex, circle, star) with the fitted Hassell competition model as a function of eggs laid (dashed line). We expect that larval survival is highest in semi-permanent ponds and lowest in ephemeral ponds (McCafferey 2010). The fitted parameters for permanent and semi-permanent ponds are consistent with our intuition for larval survival ($L_x$), carrying capacity ($K_x$), and waterbody size ($\gamma_x$); however, carrying capacity and waterbody size in ephemeral ponds are not as expected. This is most likely due to only having 8 data points to fit the model.

4.4.2 Dispersal effects on pond dynamics

We use numerical simulations to assess habitat quality, equilibrium populations, and their sensitivities to larval survival parameters. First, we find asymptotic populations without dispersal by running simulations for 2000 time steps with $M = I_{kn}$ using the parameters defined in Table 4.3. Each pond is initialized with 20 individuals, i.e., $\mathbf{u}_x(0) = [20, 20, 20]$. 
End-state populations for each local pond are $\tilde{u}^*_P_1 = \tilde{u}^*_P_2 = \tilde{u}^*_P_3 = [99, 19, 20]$, $\tilde{u}^*_S_1 = \tilde{u}^*_S_2 = [48, 9, 9]$, and $\tilde{u}^*_E_1 = \tilde{u}^*_E_2 = [29, 5, 6]$. We also calculate local pond abundances, $\tilde{U}^*_P_1 = \tilde{U}^*_P_2 = \tilde{U}^*_P_3 = 138$, $\tilde{U}^*_S_1 = \tilde{U}^*_S_2 = 66$, and $\tilde{U}^*_E_1 = \tilde{U}^*_E_2 = 40$, and total population size $\tilde{U}^* = 244$.

Using end-states for the local pond abundances without dispersal ($\tilde{u}^*_x$) as initial conditions, we run simulations with the parameters defined in Table 4.3 for a range of $r$ and $\beta$ values. We test 200 $r$ values between 50 and 1000 to reflect mean dispersal distances relevant to the distances between ponds (Table 4.2), and 100 $\beta$ values between $-0.2$ and 0.2. We find that larger magnitudes of $\beta$ do not provide qualitatively different results. Each simulation is run for 2000 time steps, and we use the final solution to define the metapopulation vector ($u^*$), pond population vector ($u^*_x$), pond size ($U^*_x$), and pond contribution ($C^*_x$). Previous work on this population have assumed equal weights in each life stage for growth rate and viability analysis, so we define our pond contributions without any special considerations for individual life stages (McCaffery et al. 2014; McCaffery and Maxell 2010; McCaffery et al. 2012).

Figure 4.4 shows how dispersal affects pond contributions under a range of density-dependent dispersal values ($\beta$). Each line represents pond contributions averaged across all $r$ simulations while the shaded regions represent standard deviations. We categorize dispersal based on the strength and direction of density-dependence: strong negative density-dependent dispersal ($\beta < -0.1$), weak negative density-dependent dispersal ($-0.1 < \beta < 0$), weak positive density-dependent dispersal ($0 < \beta < 0.1$), and strong positive density-dependent dispersal ($\beta > 0.1$). We find tighter deviations under strong negative density-dependent dispersal ($\beta < -0.1$) and larger deviations under positive density-dependent dispersal ($\beta > 0$). Under weak negative density-dependent dispersal ($-0.1 < \beta < 0$), pond contributions and populations experience drastic shifts, especially ephemeral pond 2 ($E_2$) in the partial system. Under positive density-dependent dispersal, permanent ponds have mostly higher contributions while all other ponds have mostly lower contributions except for $S_1$ and $E_2$ in the full system, and $P_2$ and $E_2$ in the partial system. In general, ponds with
larger populations (permanent ponds) benefit from positive density-dependent dispersal while ponds with smaller populations (ephemeral ponds) benefit from negative density-dependent dispersal, and vice-versa. This makes sense since negative density-dependent dispersal allows larger ponds to retain individuals while positive density-dependent dispersal distributes populations more uniformly.

Figure 4.5 shows differences in pond contributions between the full and partial systems averaged over all $r$ values for strong negative density-dependent dispersal ($\beta < -0.1$; open circle), weak negative density-dependent dispersal ($-0.1 < \beta < 0$; ex), weak positive density-dependent dispersal ($0 < \beta < 0.1$; star) and strong positive density-dependent dispersal simulations ($\beta > 0.1$; closed circle) with standard deviation bars (black lines). We find that there are little differences in pond contributions between the full and partial systems for strong negative density-dependent dispersal ($\beta < -0.1$). For weak density-dependent dispersal ($-0.1 < \beta < 0.1$) and strong positive density-dependent dispersal ($\beta > 0.1$), $P_1$, $S_2$, $E_1$, and $E_2$ generally have higher contributions while $P_2$ and $S_1$ have lower contributions in the partial system than the full system. $P_3$ experiences similar contributions between the partial and full systems. This is explained by the dispersal patterns in the partial system. Specifically, $S_1$ receives all of the dispersing individuals from $P_3$, significantly decreasing pond contribution. $P_1$, $S_2$, $E_1$, and $E_2$ all share dispersing individuals from $P_2$ so the number of incoming individuals into each of these ponds is decreased, increasing pond contributions.

### 4.4.3 Population size and contribution elasticity analyses

We now investigate the effects of vital rate perturbations to pond populations, total populations, and pond contributions by calculating elasticities to larval survival parameters in each pond ($L_x, K_x, \gamma_x$). We use representative $r$ and $\beta$ values to show sensitivities under low and high mean dispersal distances ($r = 50$, $r = 1000$), negative density-dependent dispersal $\beta = -0.2$, positive density-dependent dispersal ($\beta = 0.2$), and constant dispersal ($\beta = 0$). First, we verify our sensitivity equations by plotting calculated sensitivities with simulated perturbations. To numerically solve for sensitivities, we increase the parameter
Fig. 4.4: Pond populations (A, B) and contributions (C, D) for a range of density-dependent dispersal ($\beta$) averaged over all $r$ values (solid lines) with standard deviations (shaded area). Under strong negative density-dependent dispersal ($\beta < -0.1$), contributions are all approximately 1 while populations reflect local pond demographics. For weak negative density-dependent dispersal ($-0.1 < \beta < 0$), contributions and populations experience drastic changes with large (full) or small (partial) standard deviations. Under positive density-dependent dispersal ($\beta > 0$), contributions and populations vary based on pond demography and geographic locations, where centrally-located ponds have higher populations and lower contributions depending on feedbacks between connected ponds. In particular, $P_2$ has the highest population in both the full and partial systems but is a source in the full systems and sink in the partial system because of high immigration coming from ponds $P_1$, $S_2$, $E_1$, and $E_2$. 
values defined in Table 4.3 one parameter at a time by a fixed amount (0.001) and subtract
the results obtained in the previous section from the perturbed solution. For example, given
a small increase in pond $P_1$ larval survival, $L'_P = L_P + \Delta L_P$, with perturbed population
vector $u'$, the total change between the original population and perturbed population is
$\Delta u = u' - u$ with estimated sensitivity, $\Delta u/\Delta \theta$. Figure 4.6 shows the estimated sensitivities
on the $x$-axis with calculated sensitivity analysis on the $y$-axis for the local pond size ($u_x$)
and pond contribution ($C_x$) in the full (top row) and partial (bottom row) systems. We
use pseudo-logarithms to plot both axes, i.e., for some deviation $\Delta u_{xy}$, $\text{plog}_{10}(\Delta u_{xy}) = \text{sign}(\Delta u_{xy}) \log_{10}(1 + |\Delta u_{xy}|)$, where $\text{sign}(\cdot)$ is the sign of the value, and $|\cdot|$ is the absolute
value (John and Draper 1980). This measure scales all values to a logarithm scale while
preserving zeros and negatives. The one-to-one correlations in Figure 4.6 verify that our
sensitivity metrics for $u_x$ and $C_x$ work.

Figure 4.7 shows elasticities of total populations ($\frac{\partial U}{\partial \theta} \frac{dU}{d\theta}$) averaged over the different $r$
and $\beta$ values in the full (A) and partial (B) systems. The error bars represent standard
deviations across all simulations. We find that elasticities are similar across all simulations
regardless of density-dependence type, mean dispersal distance, and full and partial systems.
Fig. 4.6: We validate our sensitivity analysis (\(\partial\)) by comparing results with perturbed simulations (\(\Delta\)) to perturbations in larval survival parameters. Plots A and C compare pond size sensitivities while plots B and D compare contribution sensitivities. The linear relationship between the simulated perturbations (\(x\)-axis) with the calculated sensitivities (\(y\)-axis) show that our sensitivity analysis accurately represents simulations. All results are scaled by the pseudo-logarithm to account for large-scale differences.
$L_x$ and $K_x$ increase total populations while $\gamma_x$ decreases total populations with the degree of change dependent on pond type. Perturbations in high fecundity ponds ($P_1$, $P_2$, $P_3$) have the largest effects on total population sizes regardless of parameter shifts. However, the lower fecundity ponds ($S_1$, $S_2$, $E_1$, $E_2$) have equally smaller effects on total population sizes under larval survival and carrying capacity changes but semi-permanent ponds have slightly higher effects under changes in the density-dependent exponent. These results are consistent for both the full and partial systems.

The consequences of habitat alterations have both direct and indirect effects on all habitats. Figure 4.8 shows elasticities of pond contributions (A) and pond population sizes (B) for direct effects (colored symbols) and indirect effects (black dots) averaged across both $r$ values and the full versus partial systems. For example, increasing larval survival ($L_x$) under constant density-dependent dispersal ($\beta = 0$) increases both pond contributions and pond population sizes in the altered pond (direct effect); however, all other ponds experience lower contributions but higher population sizes (indirect effects). In general, parameter perturbations have smaller effects on pond contributions under negative density-dependent dispersal but contributions are higher under constant and positive density-dependent dispersal (Figure 4.8A). The opposite effect occurs for pond size elasticities (Figure 4.8B); direct effects on population sizes are largest under negative density-dependent dispersal while the indirect effects on population size increases for constant and positive density-dependent dispersal.

A qualitative shift in elasticities of pond contribution occurs when $\beta$ changes from negative density-dependent dispersal to constant dispersal. Specifically, perturbations in $L_x$ and $K_x$ produce lower contributions under negative density-dependent dispersal and higher contributions for constant dispersal and positive density-dependent dispersal; vice-versa for $\gamma_x$. This shift supports our intuition of the contribution metric. That is, increasing survival in a pond subsequently increases population abundances. Under negative density-dependent dispersal, larger populations have lower emigration rates, decreasing pond contribution. Alternatively, positive density-dependent dispersal, and even constant dispersal, have larger
emigration rates for larger populations, increasing pond contributions. Therefore, we expect change dependent on density-dependent dispersal type for pond contribution elasticities. This sign change does not occur for pond population size elasticities.

4.4.4 Implications of elasticity analysis

Without perturbations, negative density-dependent dispersal produces larger end-state populations and smaller variances whereas positive density-dependent dispersal produces smaller populations with larger variances (Figure 4.4). These results generally remain the same between the full and partial systems. On the other hand, pond contributions are largely affected by density-dependent dispersal and landscape network. Specifically, under positive density-dependent dispersal, $P_2$ experiences lower contributions in the partial system than in the full system, ephemeral ponds generally experience higher contributions while semi-permanent ponds experience either higher or lower contributions in the partial system. On the contrary, the degree to which total populations change under small parameter shifts depends on local pond survival regardless of pond size discrepancies across the different dispersal behavior. Specifically, we can expect to see higher total population increases when changes are made to the permanent ponds.

When managing for local populations, we find that the degree and direction of the effects of management on population size and contribution depend on density-dependent
Fig. 4.8: Elasticities of pond contributions (A) and pond population sizes (B) to larval survival parameters averaged over $r = 50,1000$ and both the full and partial systems. Changes in pond-specific larval survival have both direct (colored symbols) and indirect (black dots) effects on each pond. Under negative density-dependent dispersal ($\beta = -0.2$), pond contributions show very little change under larval survival perturbations while pond sizes experience much larger direct changes. For constant dispersal ($\beta = 0$) and positive density-dependent dispersal ($\beta = 0.2$), all ponds experience noticeable direct and indirect changes under larval survival perturbations. Additionally, pond contributions experience a sign change as density-dependent dispersal shifts from negative to constant to positive. This sign change does not occur for pond size.

dispersal type and pond type (Figure 4.8). Under negative density-dependent dispersal, management action causes small negative shifts in pond contributions but consistently increases pond sizes in the managed pond while showing little negative changes in all other ponds. This shows that the impacts of management action are mostly positive with little drawbacks in pond size and contribution. Under constant and positive density-dependent dispersal, care should be taken when taking management action. Specifically, when managing for a pond, the managed pond experiences higher contribution and size. Additionally, all other ponds experience higher population sizes but lower contributions. This discrepancy is important when the number of source ponds are scarce with near-sink contributions ($C_x \approx 1$).

This result brings up an important point. If dispersal behavior is known in a network, management action will depend on the conservation objective. For example, if we choose to manage pond $E_2$, it may shift from a source to sink under negative density-dependent dispersal or sink to source under positive density-dependent dispersal in the full system, potentially inducing population rescue. On the other hand, if the goal is to increase total population size, a reliable option is to restore one of the permanent ponds, producing large increases to the total population size. In particular, restoring pond $P_2$ may decrease
its contribution but only by a small amount under negative density-dependent dispersal (Figure 4.8A). However, under positive density-dependent dispersal, $P_2$ will experience a higher pond contribution, pond size, and total population size. Therefore, preserving high fecundity ponds generally benefit metapopulation viability whereas preserving low quality ponds has less of an impact on the metapopulation but could induce local rescue.

### 4.4.5 Implications for conservation management

Previous work on this population has shown that maintaining heterogeneity in pond hydroperiods produces more consistent recruitment and reduces extinction risk in variable and extreme climates (McCaffery et al. 2014). Therefore, one strategy for managing this metapopulation is to ensure at least one source exists in each pond type. We achieve this by altering waterbody size, typically done by filling or draining waterbodies. Specifically, we identify ponds with the largest contributions in each pond type and impose a strategy that maintains sources or induces rescue in sinks.

Table 4.4 shows contributions for all ponds in each scenario where bold values are the largest contribution in each pond type. Based on these bolded values, if all three are greater than 1 (indicating sources are present in each pond type), then no management action is necessary (NA). If two of these values are greater than 1 (sources occur in two pond types and only sinks occur in the third pond type), then restoring the sink will increase its contribution. Finally, if two of these values are less than 1 (sources only occur in one pond type), we degrade a pond that is not one of the three bolded ponds to increase contributions in the two sinks. We use elasticities of contributions to waterbody size $\left( \frac{\gamma_x}{C_x} \right)$ to determine largest total contribution increases in the sink habitats.

We find that management action is not necessary under negative density-dependent dispersal, and this is true for both the full and partial systems. Under high mean dispersal distances ($r = 1000$) in the full system, we find that habitat degradation will produce the largest contribution increases for ponds $S_1$ and $E_2$, potentially inducing rescue in one or both of these ponds. Under low mean dispersal distances ($r = 50$), we find that restoring $S_1$ will increase its contribution while subsequently decreasing contributions in all other
Table 4.4: Management strategies to maintain pond type heterogeneity in the metapopulation. Strategies may restore (up arrow), degrade (down arrow), or do nothing to (NA) a pond to increase contributions in other ponds. Bolded numbers are largest contributions in each pond type.

<table>
<thead>
<tr>
<th>β</th>
<th>r</th>
<th>$C_{P_1}$</th>
<th>$C_{P_2}$</th>
<th>$C_{P_3}$</th>
<th>$C_{S_1}$</th>
<th>$C_{S_2}$</th>
<th>$C_{E_1}$</th>
<th>$C_{E_2}$</th>
<th>Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Full</td>
<td>-0.2</td>
<td>50</td>
<td>1.00</td>
<td>0.999</td>
<td>1.00</td>
<td>1.00</td>
<td>0.999</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1000</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>50</td>
<td>1.27</td>
<td>0.965</td>
<td>1.28</td>
<td>1.15</td>
<td>0.760</td>
<td>0.907</td>
<td>1.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>1.08</td>
<td>1.05</td>
<td>1.16</td>
<td>0.986</td>
<td>0.900</td>
<td>0.898</td>
<td>0.932</td>
</tr>
<tr>
<td>Partial</td>
<td>-0.2</td>
<td>50</td>
<td>1.00</td>
<td>0.998</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1000</td>
<td>1.00</td>
<td>0.998</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>50</td>
<td>1.28</td>
<td>0.916</td>
<td>1.21</td>
<td>0.979</td>
<td>0.824</td>
<td>0.990</td>
<td>1.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>1.22</td>
<td>0.882</td>
<td>1.21</td>
<td>0.910</td>
<td>1.01</td>
<td>0.961</td>
<td>1.12</td>
</tr>
</tbody>
</table>

In some cases, the indirect effects of pond restoration may decrease contributions in the other source ponds such that they become sinks, but the source ponds in these instances are sufficiently high ($C_x \geq 1.05$), making this less likely from happening.

Care should be taken when degrading habitats. Although contributions increase in proximal ponds, total population size will decrease (Figure 4.7). Therefore, this strategy may only be feasible for thriving populations. If a metapopulation is at risk of extinction, it may be more beneficial to restore a pond that produces the largest increase in total population size rather than degrade a pond to increase contributions in other habitats.

4.5 Discussion

In this paper, we developed sensitivity and elasticity analysis for habitat contributions in density-dependent matrix metapopulation models. These analysis methods can be used to determine how small changes in parameters affect patch importance in a network. We applied our analysis to a high-elevation Columbia spotted frog population in Montana and found that the strength and direction of sensitivities relied largely on density-dependent dispersal type but not so much on dispersal distances and metapopulation geography. Specifically, pond contributions experienced smaller changes to perturbations
under negative density-dependent dispersal while larger changes occurred under constant and positive density-dependent dispersal. On the other hand, parameter perturbations had the opposite effect on pond sizes such that the largest changes occurred under negative density-dependent dispersal while constant and positive density-dependent dispersal experienced smaller changes. This emphasizes the importance of analyzing multiple outputs when determining options for management action.

Ranking habitats in spatially structured populations allow managers to readily compare patches given their within-patch demography and relationship with proximal populations (Runge et al. 2006; Sample et al. 2019; Wiederholt et al. 2018). We expanded on this theory by developing sensitivity analysis for habitat contributions. To our knowledge, only one paper analyzed the effects of parameter perturbations on the contribution metric. Erickson et al. (2018) used a linear model to describe common loon (Gavia immer) migration and found that increasing survival probabilities produced proportional increases in habitat contributions. This makes sense in a density-independent model since sensitivity of habitat contributions only depends on the perturbed life stage and is always positive. In this paper, we found that habitat contributions may increase or decrease depending on dispersal strategy in a density-dependent model. In future studies, it would be interesting to understand the conditions necessary for positive and negative contributions under different dispersal scenarios and migratory routes.

Amphibians and other semi-aquatic organisms such as snakes, turtles, and insects rely on emergent vegetation and waterbody hydroperiod for early stage development (Bull 2005; McCaffery et al. 2014; Semlitsch 2010). Ponds and other waterbodies are affected by environmental factors such as annual runoff (Gould et al. 2019; Ray et al. 2016), snow water equivalency (McCaffery and Maxell 2010; McCaffery et al. 2012), and drought (Hossack et al. 2013; Patla and Keinath 2005a) which in turn affect breeding success and first-year recruitment. Where permanent and larger waterbodies are more robust to environmental change, ephemeral and smaller waterbodies are greatly affected by annual precipitation and runoff. We found that the largest changes to total population size occurred in permanent
ponds when proportional changes occurred across all ponds. However, if ephemeral ponds experience larger changes in survival probabilities due to extreme climates, we may see even larger changes to the total population size. Our results only consider small parameter perturbations to individual ponds but under correlated weather patterns across the whole population, we should see drastic changes in the total population as well as disproportional changes in local populations.

We restricted our analysis to steady-state equilibria for simplicity and as a starting point for more complicated systems. Previous studies have used sensitivity analysis on fluctuating populations and transient behavior which have direct implications for population extinction and parameter uncertainty (Benton and Grant 1999; Caswell 2007; 2009; Grant 1997; Grant and Benton 2000). Additionally, many species exhibit migratory behavior and sensitivity analysis of the contribution metric will allow us to understand the impacts of management action on different migratory sites (Erickson et al. 2018; Sample et al. 2019; Wiederholt et al. 2018).

Care should be taken when interpreting sensitivities of single-parameter perturbations. Even though the biological significance is implicitly defined in our analysis, there has not been direct empirical evidence confirming the validity of single-parameter sensitivity analysis (Benton and Grant 1999). Additionally, parameter estimates in biology are typically very uncertain, and perturbations in only one parameter may not accurately assess uncertainty. Global sensitivity analysis resolves this issue by simultaneously changing multiple parameters to identify uncertainties across a large parameter space (Marino et al. 2008). This analysis is especially useful for stochastic and nonlinear models due to variability in output variables (Renardy et al. 2019). Cursory analysis of global sensitivities for survival and dispersal parameters for possible parameter ranges showed similar confidence in survival parameters on total population size although ephemeral pond carrying capacity had smaller effects on total population size than the other survival rates (see appendix). This means that even though we are uncertain of our fitted parameters due to lack of data, shifts in ephemeral pond carrying capacities would show little change to our total population size.
results. That being said, there would still be significant differences in individual pond sizes and contributions regardless of the parameter being refitted. Therefore, the conclusions we made in this analysis may change if we use different parameter values and for more reliable analysis, we would need more data on frog survival and abundance.

We assumed nonzero patch sizes for our patch contribution analysis. In many metapopulation systems, patches may be unoccupied for multiple years and rely on immigration for recolonization (Semlitsch 2010; Zylstra et al. 2019). In such cases, it is possible to measure habitat contributions by calculating per capita contributions of all pathways connecting to the empty habitat (Erickson et al. 2018). These methods are especially useful for migratory species where habitats are unoccupied in certain seasons and stochastic systems where local habitats may experience extinction for some period of time. Extending our model to include unoccupied sites would be a useful tool for analyzing transient and migratory populations.

As climate becomes more severe and unpredictable, it is important to understand how management actions can encourage viability in populations at risk of extinction. We showed that multiple factors should be considered when making these decisions. Specifically, dispersal behavior plays a large role in patch importance and managing one patch increases its population size but contributions and sizes of proximal patches may decrease depending on density-dependence dispersal type and dispersal pathways. When total population sizes are especially low, it may be most beneficial to focus management on high survival patches. This shows the complexity of conservation management and emphasizes the importance of understanding not only demographic processes within patches but also movement behavior between patches.
CHAPTER 5

CONCLUSION

This dissertation broadly covered theoretical ecology research, connecting mathematical theory to applications relevant for conservation management and population dynamics. In the first chapter, we reviewed background information for the mathematical theory used in this paper and their applications to ecology. The second chapter was a meta-analysis of time series data that used scaling theory and fast-slow dynamical systems theory to estimate and compare relative rates of density and phenotypic change. Chapter 3 analyzed metapopulation stability and synchrony in two patch models with 1, 2, and 3 life history stages. Chapter 4 analyzed metapopulation viability in a seven patch system using source-sink theory in demographic metapopulations with sensitivity analysis. The metapopulation analyses were applied to general amphibian literature in the two patch model while the seven patch model focused on a high elevation Columbia spotted frog population located in the Selway-Bitterroot Wilderness, Montana. These models investigated the role of dispersal in metapopulation dynamics to understand the processes driving amphibian dynamics and habitats most vital to metapopulation viability.

In the second chapter, we performed a meta-analysis of eco-evolutionary data to estimate and compare density rates of change with phenotypic rates of change using fast-slow dynamical systems theory and scaling theory. Specifically, we analyzed a collection of time series data to determine whether the total population density and mean phenotype changed on similar time scales. Additionally, we compared density and phenotypic rates of change for traits with plastic and evolutionary modes of adaptation. Our results showed that phenotypic rates of change can be equal to or faster than changes in population densities for both modes of adaptation. This suggests that both plasticity and evolution have the potential to drive phenotypic change that is just as fast, if not faster, than changes in population densities.
In chapter 3, we determined regions of attractor types among a range of constant dispersal probabilities in two patch models for 1, 2, and 3 life stage systems. We assumed constant dispersal between the two patches for 1 life stage at a time and tested a full range of dispersal probabilities from each patch, i.e., individuals in either patch may exhibit a range of dispersal probabilities. Using discrete-time matrix population models based on amphibian literature, we classified oscillations into three types: invariant cycles, k-cycles, and chaos. Additionally, we measured the level of synchrony between populations using covariance matrix analysis to understand the degree to which populations fluctuated in the same direction. We categorized dispersal rates into 5 zones that represented different dispersal symmetries and intensities. We found that low levels of dispersal may initially stabilize and synchronize dynamics while moderate to high levels of dispersal may create oscillations and desynchronize dynamics. These results were consistent across the three life stage models though unique results appeared in the different life stages. We found that synchrony levels in chaos were not consistently lower than other attractor types, but rather reflected synchrony of attractor types in nearby dispersal ranges. This showed that chaos does not necessarily have lower synchrony in intrinsically-driven oscillations, which contradicts the perception that chaos creates lower synchrony. Many studies have investigated the relationship between stability and synchrony where stability is defined by population variation, whereas we showed how the different attractor types affect synchrony using bifurcation and eigenvalue analysis.

In chapter 4, we applied metapopulation viability theory to a seven patch Columbia spotted frog population located in western Montana. Specifically, we used the habitat contribution metric which classified local habitats as sources and sinks based on per capita recruitment and dispersal rates. We extended the utility of habitat contribution by developing sensitivity analysis of habitat contribution for perturbations in model parameters. We performed sensitivity analysis on patch contribution and equilibrium populations to larval survival probabilities under different density-dependent dispersal strategies and dispersal scenarios that reflect landscape geography. We compared patch contributions and their
elasticities to equilibrium populations and their elasticities to understand the relationship between dispersal rates, patch contributions, and populations. We found that patch contributions and total population sizes were dependent on density-dependent dispersal type while contributions differed between the two landscape-driven dispersal strategies. Elasticity analysis showed that total metapopulation elasticities did not change regardless of dispersal scenario; however, local population elasticities had the most notable effects on low survival patches under negative density-dependent dispersal. As density-dependent dispersal shifted from negative to positive, local population elasticities became more uniform across all populations. Patch contribution elasticities were similar while also experiencing a sign-flip as density-dependence shifted from negative to positive. For example, increasing larval survival in a patch decreased contribution of that patch under negative density-dependent dispersal, whereas the contribution experienced an increase under constant and positive density-dependent dispersal. Based on this analysis, we provided suggestions for management options to encourage heterogeneous pond-types across the population.
REFERENCES


———. 1999. Elasticity as an important tool in evolutionary and population ecology. TREE 14:467–471.


Funk, W., A. Greene, P. Corn, and F. Allendorf. 2005b. High dispersal in a frog species suggests that it is vulnerable to habitat fragmentation. Biology Letters 1:13–16.


Maxell, B. A. 2000. Management of montana’s amphibians: a review of factors that may present a risk to population viability and accounts on the identification, distribution,
taxonomy, habitat use, natural history, and the status and conservation of individual species. Tech. rep., University of Montana, Missoula, MT.


APPENDIX A

Supplementary Information for ‘Evolution and plastic phenotypic change can be just as fast as changes in population densities’

A.1 The link between feedbacks and rates of change

Here, we show how the rates of density and phenotypic change affect the strengths of density, density-phenotype, and phenotype feedbacks. Density feedbacks are the effects species’ densities have on the dynamics of population densities, phenotype feedbacks are the effects species’ traits have on the dynamics of species traits, and density-phenotype feedbacks are reciprocal between species’ population densities and traits.

Mathematically, the presence/absence of these feedbacks is determined by how the rates of change depend on the variables, i.e., the coupling between the variables. To illustrate this, let $x$ and $z$ be the density and population-level phenotype of a species with dynamics $\frac{dx}{dt} = \epsilon_x f(x, z)$ and $\frac{dz}{dt} = \epsilon_z g(x, z)$, respectively, where $\epsilon_x$ and $\epsilon_z$ are the maximum rates of phenotypic change and $-1 \leq f, g \leq 1$; see the section “Distinguishing between fast and slow rates of change” in the main text for more details. A density feedback is present if the density dynamics depends on the density of the species, i.e., $f(x, z)$ depends on the value of $x$. Said another way, a density feedback is present if there is a coupling between the density dynamics and the value of the species density. Mathematically, a feedback is present if $\partial f(x, z)/\partial x \neq 0$ for some range of $x$. For example, the exponential growth equation $\frac{dx}{dt} = zx$ has a density feedback whereas the growth rate $\frac{dx}{dt} = z$ does not have a density feedback. Similarly, a phenotype feedback is present if the phenotype dynamics depend on the phenotype of the species, i.e., the phenotype dynamics are coupled to the value of the phenotype, which mathematically means $\partial g(x, z)/\partial z \neq 0$ for some range of $z$. The density-phenotype feedback is present if the density and phenotype dynamics depend on the value of the other variable, i.e., $\partial f(x, z)/\partial z \neq 0$ and $\partial g(x, z)/\partial x \neq 0$.
total, the different feedbacks are present/absent depending on whether the dynamics of the variables are coupled, which mathematically can be captured by non-zero derivatives of the equations describing the dynamics.

The strengths of those feedbacks depend on both the strength of the coupling between the variables (i.e., the magnitudes of the partial derivatives in the previous paragraph) and the speeds of the dynamics (i.e., the magnitudes of the rates of change). To see this, without loss of generality we Taylor expand the differential equations about $t = 0$ to get

\[
\frac{dx}{dt}(t) = \left. \frac{dx}{dt}(t) \right|_{t=0} + \left. \frac{d^2 x}{dt^2}(t) \right|_{t=0} t + \mathcal{O}(t^2)
\]  
(A.1)

\[
\frac{dz}{dt}(t) = \left. \frac{dz}{dt}(t) \right|_{t=0} + \left. \frac{d^2 z}{dt^2}(t) \right|_{t=0} t + \mathcal{O}(t^2)
\]  
(A.2)

We focus on the terms that are linear in $t$ because the higher order terms are smaller for $t \ll 1$. Equation (S.2) shows that the density dynamics depend on (i) how the density dynamics are coupled to the species’ density ($\frac{\partial}{\partial x} \frac{dx}{dt}$) and phenotype ($\frac{\partial}{\partial z} \frac{dz}{dt}$) and (ii) the rates of change of the density ($\frac{dx}{dt}$) and phenotype ($\frac{dz}{dt}$). Combining those factors defines the density feedback and one component of the density-phenotype feedback. Similarly, equation (S.5) shows that the phenotype dynamics depend on (i) how the phenotype dynamics are coupled to the species’ density ($\frac{\partial}{\partial x} \frac{dz}{dt}$) and phenotype ($\frac{\partial}{\partial z} \frac{dz}{dt}$) and (ii) the rates of change of the density ($\frac{dx}{dt}$) and phenotype ($\frac{dz}{dt}$). Combining those factors defines the phenotype
feedback and the other component of the density-phenotype feedback.

Equations (S.3) and (S.6) show that the strengths of the effect of the density feedback, the density-phenotype feedback, and the phenotype feedback are proportional to $\epsilon_x^2$, $\epsilon_x \epsilon_y$, and $\epsilon_z^2$, respectively. This means that if the coupling between variables is relatively equal, then the strengths of the different feedbacks can often be predicted by the maximum rates of change of the variables ($\epsilon_x, \epsilon_z$). For example, if the phenotype dynamics are much faster than the density dynamics (i.e., $\epsilon_x \ll \epsilon_z$) and the coupling between all variables is comparable (i.e., $\frac{\partial}{\partial x} \frac{dx}{dt}$ and $\frac{\partial}{\partial z} \frac{dx}{dt}$ are similar in magnitude), then the density-phenotype feedback term in equation (S.3) will be much larger than the density feedback term. Thus, while density and density-phenotype feedbacks both affect the population density dynamics, the density-phenotype feedback has a much larger effect because the phenotype dynamics are faster.

We note four things about and related to the above. First, one variable being faster than the other (e.g., $\epsilon_x > \epsilon_z$) does not necessarily imply that the feedbacks involving the faster variable are always larger than the feedbacks involving the slower variable. This is because the strengths of the feedbacks are determined jointly by (i) the maximum rate of change of involved variables (e.g., $\epsilon_x$), (ii) the realized rate of change at that point in time (e.g., $f(x, z)$) and (iii) the coupling between the variables (e.g., $\partial g/\partial x$). For example, the density-phenotype feedback in equation (S.6) may be weaker than the density feedback when the density dynamics are faster than the phenotypic dynamics ($\epsilon_x > \epsilon_z$) if coupling between the phenotype and density is weak ($\partial g/\partial x$, e.g., changes in density have little effect on selection) or if the species density is currently changing slowly ($f \approx 0$ because the system is near the $x$-nullcline). That being said, if the coupling between all variables is similar in magnitude, then in many regions of parameter space the maximum rates of change will determine which feedbacks are stronger or weaker. This means that identifying faster versus slower variables using maximum rates of change can help identify specific feedbacks that are more likely to have stronger effects than other feedbacks, but that does not imply one feedback has stronger effects than another feedback at all points in time.

Second, the above is directly related to theory in previous studies (Cortez 2018; Cortez
that have measured the strengths of the effects of different feedbacks on equilibrium stability. In those studies, feedbacks are defined by submatrices of the Jacobian and their determinants. The above equations also depend on the Jacobian (although not evaluated at equilibrium). To see this, we rewrite equations (S.2)-(S.5) as

\[
\begin{pmatrix}
\frac{dx}{dt}(t) \\
\frac{dz}{dt}(t)
\end{pmatrix} = \begin{pmatrix}
\frac{dx}{dt}(0) \\
\frac{dz}{dt}(0)
\end{pmatrix} + \begin{pmatrix}
\frac{\partial}{\partial x} & \frac{\partial}{\partial z} \\
\frac{\partial}{\partial x} & \frac{\partial}{\partial z}
\end{pmatrix} \begin{pmatrix}
x(t) \\
y(t)
\end{pmatrix} + t + O(t^2).
\]

(A.7)

where the 2x2 matrix is the Jacobian of the model.

Third, comparing the maximum rates of density and phenotypic change allows one to make predictions about whether density, density-phenotype, and phenotype feedbacks are more likely to have stronger effects on the system dynamics. However, this does not mean that those feedbacks have the strongest effects on the system dynamics. This is because the densities or phenotypes of other species or other environmental variables may have stronger effects on the dynamics of the system. Nonetheless, even when other variables are affecting the dynamics of the system, comparing maximum rates of change still provides insight into whether the density, density-phenotype, or phenotype feedbacks of a focal species are more likely to have stronger effects on the system dynamics.

To see this, let \( x \) and \( z \) denote the density and population-level phenotype for a focal species and let \( y_i(t) \) \((1 \leq i \leq n)\) denote all other variables in the system. Each \( y_i(t) \) could be the density or trait of another interacting species or some other environmental variable (e.g., temperature). We model this system using the system of differential equations,

\[
\begin{align*}
\frac{dx}{dt} &= \epsilon_x f(x, z, y_1, ..., y_n) \\
\frac{dz}{dt} &= \epsilon_z g(x, z, y_1, ..., y_n) \\
\frac{dy_i}{dt} &= \epsilon_i h_i(x, z, y_1, ..., y_n, t), \quad 1 \leq i \leq n.
\end{align*}
\]

(A.8)
The dynamics of $y_i$ can depend on $x$ and $z$ (e.g., when $y_i$ is the density of an interacting species) or be independent of those variables (e.g., if $y_i$ is temperature then it may only be a function of time, $h_i(t)$). Taylor expanding the $dx/dt$ and $dz/dt$ equations as before yields

$$
\frac{dx}{dt}(t) = \epsilon_x f|_{t=0} + \epsilon_x \epsilon_z \frac{\partial f}{\partial x} h(t)|_{t=0} + \epsilon_x \epsilon_z \frac{\partial f}{\partial z} g|_{t=0} + \Omega(t^2)
$$

(A.9)

$$
\frac{dz}{dt}(t) = \epsilon_z g|_{t=0} + \epsilon_x \epsilon_z \frac{\partial g}{\partial x} f|_{t=0} + \epsilon_x \epsilon_z \frac{\partial g}{\partial z} g|_{t=0} + \Omega(t^2)
$$

(A.10)

These equations are same as equations (S.3) and (S.6) except that they account for how the other variables affects the dynamics of the focal species’ density and trait.

Importantly, the relative strengths of the density, density-phenotype and phenotype feedbacks involving the focal species’ density and trait are unaffected by the dynamics of the additional variables. For example, if the phenotype dynamics of the focal species are much faster than its density dynamics ($\epsilon_z \gg \epsilon_x$) and there is sufficient coupling between the variables, then the effect of the focal species’ density-phenotype feedback on the density dynamics of the focal species are stronger than the effects of the focal species’ density feedbacks. However, without information about the rates of change of the other variables ($y_i$) and how the dynamics of the focal species are coupled to them, it is not possible to determine whether the density, density-phenotype, and phenotype feedbacks involving just the focal species are weaker, stronger or comparable to the effects of the feedbacks involving the other variables.

Finally, while the above focuses on feedbacks between two variables, everything extends to feedbacks between any combination of species’ densities and population-level traits by adding additional equations and using linear algebra. The mathematics simply involves
applying the linear algebra illustrated in equation (S.7) to equations analogous to equations (S.9) and (S.10) for the appropriate set of variables. For example, the feedbacks involving just \( x, z \) and \( y_1 \) would be derived by Taylor expanding the \( dx/dt, dz/dt, \) and \( dy_1/dt \) equations and computing a 3x3 submatrix of the Jacobian. As another example, the feedbacks between all variables in model (S.8) are defined by equation (S.7) where \( (x(t), z(t)) \) is replaced by \( (x(t), z(t), y_1(t), ..., y_n(t)) \) (and similarly for the derivatives) and the matrix is the full Jacobian for the model.

A.2 Details about empirical data sets

The data sets for all analyzed studies are included in the supplementary files uploaded to the Dryad Digital Repository (Grosklos and Cortez 2020). The data from Edeline et al. (2008), Haafke et al. (2016), Hiltunen et al. (2014), Kasada et al. (2014), and Sanchez and Gore (2013) were shared with us by the authors of those studies. The data for all other studies was extracted from published PDFs. One study (Le Galliard et al. 2005) analyzed in the DeLong et al. (2016) study was excluded from our analysis because it was unclear how to compute the total density of the population.

For most studies the population-level trait was the mean value of a quantitative phenotype (e.g., beak size or wing length). For eight of the studies (6 with heritable traits, 2 with an unknown mode of adaptation), the original data consisted of densities or abundances of two different phenotypes (e.g., susceptible and resistant bacterial strains, or red and yellow color morphs). For these systems, we used the frequency of one phenotype as the population-level trait. Specifically, if \( a_1, ..., a_n \) and \( b_1, ..., b_n \) are the densities of each phenotype at time points \( t_1, ..., t_n \), then the total density at each point is \( x_i = a_i + b_i \) and the population-level trait is \( z_i = a_i / x_i \). Using the proportion of one phenotype as the population-level trait is appropriate for these systems because in any system where individuals can only take on one of two phenotypes, there is a one-to-one mapping between the mean phenotype and the proportion of one phenotype. In particular, if phenotype \( i \) has trait value \( v_i \) density \( n_i \), then the average trait value can be uniquely written in terms of the proportion of phenotype 1 \( (p_1) \) using \( (v_1n_1 + v_2n_2) / (n_1 + n_2) = v_1p_1 + v_2(1 - p_1) \). We
note that because the clones in the Fenchel and Jonsson (1988) study have different plastic responses to predation, each individual’s trait is a reaction norm.

The estimated scaled maximum rates of density and phenotypic change for each published time series are listed in Table A.1. Multiple lines for a single study mean that multiple time series from that study were analyzed. The scaled maximum rates listed in Table 1 and plotted in Figure 3A are the averages of those in Table A.1. Figure 3B in the main text plots the ratio of the average maximum phenotypic rate and the average maximum density rate, i.e., \( \langle \epsilon_{ph} \rangle / \langle \epsilon_{den} \rangle \). The ratio for each individual time series is listed in Table A.1 and plotted in Figure A.1.

We classify each study as either a field-based study (FS) or a constant condition laboratory experiment (LE); see column Exp in Table A.1. Field-based studies involved data collected in a natural setting. Temporal dynamics in these populations were driven by a combination of species interactions and natural environmental forcing (e.g., seasonal forcing). We note that the Campbell and Echternacht (2003) study was unique in that it seeded uninhabited islands with a small number of individuals and observed the population in subsequent years. Laboratory studies involved communities held under constant environmental conditions. For all laboratory studies, the communities were either (i) seeded with populations whose densities and phenotypic distributions were not at equilibrium values or (ii) seeded initially with monomorphic populations and then additional genotypes were introduced at a later point in time. In all cases, the temporal dynamics in these studies are driven by factors internal to the system (e.g., predatory interactions) and not by external environmental forcing.

Because the sampling interval could potentially affect our results, Table A.1 also lists the total number of samples for each time series (SS), total time elapsed (Total T), average sampling interval size (Avg SI), and the maximum time between consecutive measurements (Max SI). For some studies, these values differed between the density and trait time series, but because these differences were nominal, we only reported the values for the density time series in Table A.1. The average number of samples collected per generation for every
study is plotted in Figure A.2. There are 17 studies with at least one sample per generation on average. We expect that undersampling is not an issue for these studies. There are 13 studies where the average number of samples is less than one per generation (studies coded by c, j, k, l, m, q, r, B, C, E, F, H, and η), with 4 of those studies (coded by c, j, q, and h) having less than 1 sample per 10 generations on average. As noted in the main text, 6 of the studies (coded j, k, l, m, H, and η) are unlikely to be affected by the sparse sampling because the evolving species is a prey in a predator-prey system and the sampling interval was a small fraction of the predator generation time (which limits the period of the cycles). Undersampling could be affecting the results of the remaining 7 studies (coded by c, q, r, B, C, E, and F). This is less likely for the Caron et al. (1985), DeLong et al. (2014), Gonzalez et al. (1993), and Lürling et al. (2005) studies (coded by B, C, E, and F), which have one sample per 1.4-4 generations on average and maximum gaps between consecutive samples that range between 3.9 and 5.1 generations. Additional evidence of this for DeLong et al. (2014) (coded by C) is provided by the tight fit between the data and mechanistic model in that study; see Figure 1 of that study. Undersampling has a greater likelihood of affecting the results for the other studies (coded by c, q, and r) which have one sample per 4.6-84 generations on average and maximum gaps between consecutive samples that range between 5 and 220 generations.

A.3 Model simulations

The simulations in Figures 1 and 2 are from the Abrams and Matsuda (1997) model, modified so that the prey density and prey trait equations are multiplied by dimensionless time scale parameters $\epsilon_x$ and $\epsilon_z$. This model is,

\[
\begin{align*}
\frac{dx}{dt} &= \epsilon_x \left( rx + qxz - kx^2 - \frac{axyz}{1 + hxz} \right) \\
\frac{dy}{dt} &= \frac{caxyz}{1 + hxz} - my \\
\frac{dz}{dt} &= \epsilon_z V \left( q - \frac{ay}{1 + hxz} \right) e^{-s/(z-s)},
\end{align*}
\]

where $x$ is the prey density, $y$ is the predator density, and $z$ is the population-level prey
trait (defense). The parameters are: prey attack rate, $a$, prey to predator density conversion rate, $c$, predator handling time, $h$, predator per capita death rate, $m$, trait-depended prey maximum exponential growth rate, $r + qz$, and prey intraspecific competition coefficient, $k$. In the trait equation, $V$ is the standing genetic variation and the function $\exp(-s/(z-s))$ bounds the trait above the small value $s$.

For the simulation in Figure 1, the parameters are $r = 2.5$, $q = 0.8$, $k = 1$, $a = 1$, $h = 1$, $c = 1$, $m = 0.5$, $V = 1$, $s = 0.001$, $\epsilon_x = 1$, and $\epsilon_z = 0.01$ and the initial conditions are $(x, y, z) = (5, 5, 0.5)$. For all panels in Figure 2, the parameters are $r = 0.25$, $q = 0.08$, $k = 0.1$, $a = 0.1$, $h = 1$, $c = 10$, $m = 0.5$, $V = 0.175$, and $s = 0.001$ and the initial conditions are $(x, y, z) = (5, 1, 1.5)$. These values were chosen such that $\epsilon_{ph} \approx \epsilon_{den}$ when estimated from simulations where $\epsilon_x = \epsilon_z = 1$. In panels A-D, the step size for all simulations is 1 time unit, $\epsilon_x = 0.1$ for all simulations, and $\epsilon_z$ is set to 2, 1, 0.5, 0.1, and 0.05. Panels E-H use the same parameters as panels A-D, respectively, with step sizes of 1 (black symbols), 4, 10, 20, 40, 100, and 200 (lightest gray symbols) time units.

A.4 Comparison of results from methods using alternative scalings and summary statistics

We compared the rates of density and phenotypic change estimated using the method in this study, the method in DeLong et al. (2016), and methods that use alternative scalings. The rows of Figure A.3 involve different scalings of the rates and the columns of Figure A.3 differ in terms of comparing (left) maximum rates and (right) averaged rates. Figure A.3A shows the estimated maximum rates of change scaled by the range of the data (and the species generation time, $G$). Figure A.3B shows the estimated maximum rates of change where the derivatives are per capita rates, computed using $|(x_{i+1} - x_i)/(t_{i+1} - t_i)| \cdot (G/x_i)$ and $|(z_{i+1} - z_i)/(t_{i+1} - t_i)| \cdot (G/z_i)$. Figure A.3C shows the estimated maximum rates of change of the unscaled rates, computed using $|(x_{i+1} - x_i)/(t_{i+1} - t_i)|$ and $|(z_{i+1} - z_i)/(t_{i+1} - t_i)|$. Figure A.3D shows the estimated maximum rates of change scaled by the standard deviation of the time series; this is computed using $|(x_{i+1} - x_i)/(t_{i+1} - t_i)| \cdot (G/std(x_i))$ and $|(z_{i+1} - z_i)/(t_{i+1} - t_i)| \cdot (G/std(z_i))$ where $std(x_i)$ and $std(z_i)$ are the stand deviations.
of the density and phenotype time series. Panels E-H show average rates of change scaled in the same way as panels A-D. Figure A.3A uses the method from this study and Figure A.3F uses the method from DeLong et al. (2016).

When computing per capita rates of change (as in DeLong et al. (2016)), some data points in the time series were omitted because the trait or population density took on a value of 0, which results in a divide by zero error when computing per capita rates. We omitted 1 data point in Becks et al. (2010), 20 data points in Kasada et al. (2014), and 14 data points in Sanchez and Gore (2013).

Comparing Figure A.3A,E and Figure A.3B,F shows that the key factor driving the different results between our study and DeLong et al. (2016) is the choice of scaling (i.e., scaling by range versus per capita rates). In general, the increased scatter of the data points in Figure A.3B,F is due to small divisor issues. Specifically, if the population density \( x \) is small in magnitude, then small amounts of variation in \( x \) cause large changes in the value of \( (1/x)(dx/dt) \); the same applies to when the trait value \( z \) is small in magnitude. These small divisor issues result in scaled maximum rates of change that are much larger than expected. This in turn results in some studies having maximum or average phenotypic rates that are more than 10 times faster or more than 10 times slower than the corresponding rates of change in density. For example, the placement of data set r (Schrag and Mittler 1996) is caused, in part, by the per capita rate of phenotypic change being massive because the mean phenotype is close to zero. These small divisor issues and the omitted data points in the previous paragraph highlight the perils of using per capita rates when comparing rates of density and phenotypic change.

Comparing Figure A.3A,E and Figure A.3D,H shows that our results are qualitatively unchanged if the maximum or average rates of change are scaled by the sample standard deviation. Specifically, scaling by sample standard deviation produces maximum phenotypic rates of change that are between 0.4 and 2.5 times faster than maximum density rates of change.
Fig. A.1: Ratios of maximum phenotypic rates and maximum density rates for all time series analyzed in this study. Replicates from the same study are connected by solid vertical gray lines. Studies are colored according to the mode of adaptation: genetic (a–r; red), plastic (A–H; blue), and unknown (α – ψ; brown). Symbols correspond to studies listed in Tables 1 and S1. The horizontal lines denote when phenotypic rates are two times faster (upper dashed line), equal to (solid line), and two times slower (lower dashed line) than density rates of change.
Fig. A.2: The average number of samples collected per generation for all time series analyzed in this study. Replicates from the same study are connected by solid vertical gray lines. Symbols correspond to studies listed in Tables 1 and S1 and colored according to the mode of adaptation: genetic (a−r; red), plastic (A−H; blue), and unknown (α − ψ; brown). The dashed line represent studies where the average number of samples collected were once per generation. Studies that lie below the dashed line indicate that less than one sample was collected per generation.
Fig. A.3: Comparison of the methods for estimating rates of density and phenotypic change used in the current study and in DeLong et al. (2016). Rates of density and phenotypic change estimated (left column) using maximum derivatives or (right column) average derivatives that are either (4th row) scaled by the standard deviation, (3rd row) unscaled, (2nd row) scaled by the magnitude of the values (i.e., per capita rates), or (1st row) scaled by the range of the data. Panel A shows the results from the method used in the current study. Panel F shows the results from the method used in DeLong et al. (2016). In all panels, the black line denotes the 1:1 line and the upper and lower dashed lines denote where phenotypic rates of change are two times faster or two times slower, respectively, than density rates of change. Symbols correspond to studies listed in Table 1.
Table A.1: Estimated scaled maximum rates of density and phenotypic change for each time series.

<table>
<thead>
<tr>
<th>Code</th>
<th>Study</th>
<th>Figure</th>
<th>Trait</th>
<th>Data Type</th>
<th>$\epsilon_{\text{den}}$</th>
<th>$\epsilon_{\text{ph}}$</th>
<th>$\epsilon_{\text{ph}}/\epsilon_{\text{den}}$</th>
<th>SS</th>
<th>Total T</th>
<th>Avg SI</th>
<th>Max SI</th>
<th>Gen</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Becks et al. (2010)</td>
<td>4a</td>
<td>Clump size</td>
<td>E</td>
<td>0.3</td>
<td>0.72</td>
<td>2.4</td>
<td>22</td>
<td>21</td>
<td>1</td>
<td>1.3</td>
<td>0.78</td>
<td>days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4b</td>
<td></td>
<td></td>
<td>0.25</td>
<td>0.46</td>
<td>1.8</td>
<td>26</td>
<td>26</td>
<td>1</td>
<td>2.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4c</td>
<td></td>
<td></td>
<td>0.55</td>
<td>0.58</td>
<td>1</td>
<td>42</td>
<td>43</td>
<td>1</td>
<td>2.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4d</td>
<td></td>
<td></td>
<td>0.41</td>
<td>0.54</td>
<td>1.3</td>
<td>32</td>
<td>31</td>
<td>1</td>
<td>1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4e</td>
<td></td>
<td></td>
<td>0.34</td>
<td>0.72</td>
<td>2.1</td>
<td>44</td>
<td>59</td>
<td>1.4</td>
<td>2.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4f</td>
<td></td>
<td></td>
<td>0.36</td>
<td>0.75</td>
<td>2.1</td>
<td>42</td>
<td>59</td>
<td>1.4</td>
<td>2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>Becks et al. (2012)</td>
<td>1a</td>
<td>Clump size</td>
<td>E</td>
<td>0.39</td>
<td>0.4</td>
<td>1</td>
<td>79</td>
<td>101</td>
<td>1.3</td>
<td>3.6</td>
<td>0.78</td>
<td>days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1b</td>
<td></td>
<td></td>
<td>8.6</td>
<td>18</td>
<td>2.1</td>
<td>79</td>
<td>114</td>
<td>1.5</td>
<td>3.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1c</td>
<td></td>
<td></td>
<td>0.32</td>
<td>0.42</td>
<td>1.3</td>
<td>46</td>
<td>47</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1d</td>
<td></td>
<td></td>
<td>0.93</td>
<td>0.75</td>
<td>0.9</td>
<td>74</td>
<td>73</td>
<td>1</td>
<td>1.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>Bohannon and Lenski (1998)</td>
<td>2a</td>
<td>Proportion susceptible</td>
<td>E</td>
<td>0.011</td>
<td>0.012</td>
<td>1.1</td>
<td>19</td>
<td>385</td>
<td>21</td>
<td>55</td>
<td>0.25</td>
<td>hrs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2b</td>
<td></td>
<td></td>
<td>0.0086</td>
<td>0.013</td>
<td>1.5</td>
<td>18</td>
<td>386</td>
<td>23</td>
<td>55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>Coltman et al. (2003)</td>
<td>2b, 2c</td>
<td>Horn length (adult male)</td>
<td>E</td>
<td>1.8</td>
<td>2.4</td>
<td>1.3</td>
<td>31</td>
<td>30</td>
<td>1</td>
<td>1.1</td>
<td>6</td>
<td>yrs</td>
</tr>
<tr>
<td>e</td>
<td>Fussman et al. (2003)</td>
<td>2a</td>
<td>Proportion mictic (female)</td>
<td>E</td>
<td>1.4</td>
<td>1.3</td>
<td>0.92</td>
<td>25</td>
<td>25</td>
<td>1</td>
<td>1.5</td>
<td>2.2</td>
<td>days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2b</td>
<td></td>
<td></td>
<td>0.71</td>
<td>1.5</td>
<td>2.2</td>
<td>25</td>
<td>26</td>
<td>1</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2c</td>
<td></td>
<td></td>
<td>1</td>
<td>1.1</td>
<td>1</td>
<td>26</td>
<td>26</td>
<td>1</td>
<td>1.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>Grant and Grant (2002)</td>
<td>5a, 1b</td>
<td>G. fortis Beak length</td>
<td>E</td>
<td>2.9</td>
<td>2.1</td>
<td>0.7</td>
<td>30</td>
<td>29</td>
<td>1</td>
<td>1.1</td>
<td>4.5</td>
<td>yrs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5a, 1e</td>
<td>G. fortis Beak shape</td>
<td></td>
<td>2.9</td>
<td>1.9</td>
<td>0.64</td>
<td>30</td>
<td>29</td>
<td>1</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>Grant and Grant (2002)</td>
<td>5a, 1c</td>
<td>G. scandens Beak length</td>
<td>E</td>
<td>2.9</td>
<td>1.9</td>
<td>0.93</td>
<td>30</td>
<td>29</td>
<td>1</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>h</td>
<td>Grant and Grant (2002)</td>
<td>5a, 1f</td>
<td>G. scandens Beak shape</td>
<td></td>
<td>2</td>
<td>1.1</td>
<td>0.55</td>
<td>30</td>
<td>29</td>
<td>1</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>i</td>
<td>Haafke et al. (2016)</td>
<td>3E(top)</td>
<td></td>
<td>O</td>
<td>0.065</td>
<td>0.062</td>
<td>0.94</td>
<td>80</td>
<td>102</td>
<td>1.3</td>
<td>22</td>
<td>0.11</td>
<td>wks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3E(bot)</td>
<td></td>
<td></td>
<td>0.053</td>
<td>0.074</td>
<td>1.4</td>
<td>80</td>
<td>102</td>
<td>1.3</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STA</td>
<td></td>
<td></td>
<td>0.053</td>
<td>0.092</td>
<td>1.7</td>
<td>80</td>
<td>102</td>
<td>1.3</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STB</td>
<td></td>
<td></td>
<td>0.046</td>
<td>0.077</td>
<td>1.7</td>
<td>80</td>
<td>102</td>
<td>1.3</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>Hiltunen and Becks (2014)</td>
<td>2b, 2e</td>
<td>Proportion defended</td>
<td>E</td>
<td>0.54</td>
<td>0.17</td>
<td>0.32</td>
<td>15</td>
<td>28</td>
<td>2</td>
<td>2.1</td>
<td>1</td>
<td>days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2c, 2f</td>
<td></td>
<td></td>
<td>0.48</td>
<td>0.22</td>
<td>0.45</td>
<td>15</td>
<td>28</td>
<td>2</td>
<td>2.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2d, 2g</td>
<td></td>
<td></td>
<td>0.27</td>
<td>0.3</td>
<td>1.1</td>
<td>15</td>
<td>28</td>
<td>2</td>
<td>2.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>l</td>
<td>Hiltunen et al. (2014)</td>
<td>2.3a, 2.3b</td>
<td>Clump size</td>
<td>O</td>
<td>0.19</td>
<td>0.17</td>
<td>0.88</td>
<td>177</td>
<td>176</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.3e, 2.3f</td>
<td></td>
<td></td>
<td>0.21</td>
<td>0.35</td>
<td>1.7</td>
<td>223</td>
<td>222</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.3g, 2.3h</td>
<td></td>
<td></td>
<td>0.17</td>
<td>0.39</td>
<td>2.2</td>
<td>157</td>
<td>157</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4a, 2.4b</td>
<td></td>
<td></td>
<td>0.13</td>
<td>0.42</td>
<td>3.2</td>
<td>185</td>
<td>184</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4c, 2.4d</td>
<td></td>
<td></td>
<td>0.16</td>
<td>0.46</td>
<td>2.8</td>
<td>190</td>
<td>189</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4e, 2.4f</td>
<td></td>
<td></td>
<td>0.21</td>
<td>0.5</td>
<td>2.4</td>
<td>186</td>
<td>186</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4g, 2.4h</td>
<td></td>
<td></td>
<td>0.21</td>
<td>0.39</td>
<td>1.9</td>
<td>66</td>
<td>65</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.5a, 2.5b</td>
<td></td>
<td></td>
<td>0.18</td>
<td>0.29</td>
<td>1.7</td>
<td>190</td>
<td>189</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.5c, 2.5f</td>
<td></td>
<td></td>
<td>0.11</td>
<td>0.38</td>
<td>3.5</td>
<td>24</td>
<td>23</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.5e, 2.5d</td>
<td></td>
<td></td>
<td>0.095</td>
<td>0.43</td>
<td>4.5</td>
<td>187</td>
<td>186</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.5g, 2.5h</td>
<td></td>
<td></td>
<td>0.15</td>
<td>0.38</td>
<td>2.6</td>
<td>66</td>
<td>65</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Columns are the study code (a–r for genetic traits, A–H for plastic traits, and α–ψ for unknown) whether the data was extracted from PDF (E) or obtained from authors of the study (O), experiment type (Exp), calculated maximum rates of density ($\epsilon_{\text{den}}$) and phenotypic ($\epsilon_{\text{ph}}$) change and their ratio ($\epsilon_{\text{ph}}/\epsilon_{\text{den}}$), sample size (SS), total time elapsed (Tot T), average sampling interval size (Avg SI), maximum sampling interval size (Max SI), generation time (Gen), and time unit (Unit). Multiple rows for the same study correspond to different replicates.

*The density data is from Figure 1a of Ezard et al. (2009) and the phenotypic data is from Figure 1c of Ozgul et al. (2009).*
Table S1 (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Study</th>
<th>Figure</th>
<th>Trait</th>
<th>Data</th>
<th>Type</th>
<th>ϵ_{den}</th>
<th>ϵ_{ph}</th>
<th>ϵ_{ph}/ϵ_{den}</th>
<th>SS</th>
<th>Total T</th>
<th>Avg SI</th>
<th>Max SI</th>
<th>Gen Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>Kasada et al. (2014)</td>
<td>2A,B</td>
<td>Proportion defended (costly tradeoff)</td>
<td>O</td>
<td>LE</td>
<td>0.17</td>
<td>0.38</td>
<td>2.2</td>
<td>43</td>
<td>58</td>
<td>1.4</td>
<td>4</td>
<td>1 days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2C,D</td>
<td></td>
<td></td>
<td></td>
<td>0.48</td>
<td>0.43</td>
<td>0.9</td>
<td>54</td>
<td>69</td>
<td>1.3</td>
<td>4.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2E,F</td>
<td></td>
<td></td>
<td></td>
<td>0.64</td>
<td>0.47</td>
<td>0.72</td>
<td>33</td>
<td>40</td>
<td>1.3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td></td>
<td>3A,B</td>
<td>Proportion defended (cheap tradeoff)</td>
<td></td>
<td></td>
<td>1.1</td>
<td>0.39</td>
<td>0.36</td>
<td>57</td>
<td>59</td>
<td>1.1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3C,D</td>
<td></td>
<td></td>
<td></td>
<td>0.72</td>
<td>0.74</td>
<td>1</td>
<td>54</td>
<td>63</td>
<td>1.2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3E,F</td>
<td></td>
<td></td>
<td></td>
<td>0.68</td>
<td>0.59</td>
<td>0.86</td>
<td>59</td>
<td>72</td>
<td>1.2</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3G,H</td>
<td></td>
<td></td>
<td></td>
<td>0.78</td>
<td>0.58</td>
<td>0.74</td>
<td>56</td>
<td>69</td>
<td>1.3</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>o</td>
<td>Milot et al. (2011)</td>
<td></td>
<td>Age at first reproduction</td>
<td>E</td>
<td>FS</td>
<td>0.58</td>
<td>0.77</td>
<td>1.3</td>
<td>8</td>
<td>149</td>
<td>21</td>
<td>31</td>
<td>27 yrs</td>
</tr>
<tr>
<td>p</td>
<td>Robinson et al. (2006)</td>
<td>2</td>
<td>Proportion normal horned</td>
<td>E</td>
<td>FS</td>
<td>2.7</td>
<td>6.1</td>
<td>2.2</td>
<td>18</td>
<td>18</td>
<td>1.1</td>
<td>2</td>
<td>6.5 yrs</td>
</tr>
<tr>
<td>q</td>
<td>Sanchez and Gore (2013)</td>
<td>2b</td>
<td>Proportion cooperator</td>
<td>O</td>
<td>LE</td>
<td>0.041</td>
<td>0.1</td>
<td>2.4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0.1 days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2c</td>
<td></td>
<td></td>
<td></td>
<td>0.037</td>
<td>0.063</td>
<td>1.7</td>
<td>7</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>Schrag and Mittler (1996)</td>
<td>5a</td>
<td>Proportion resistant</td>
<td>E</td>
<td>LE</td>
<td>0.29</td>
<td>0.21</td>
<td>0.74</td>
<td>7</td>
<td>8</td>
<td>1.3</td>
<td>2.3</td>
<td>0.283 hrs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5b</td>
<td></td>
<td></td>
<td></td>
<td>0.26</td>
<td>0.12</td>
<td>0.46</td>
<td>7</td>
<td>8</td>
<td>1.3</td>
<td>2.3</td>
<td></td>
</tr>
</tbody>
</table>

**Plastic**

| A    | Campbell and Echternacht (2003) | 3, Table 2 | Snout-vent length | E | FS | 0.61 | 0.76 | 1.2 | 4 | 3 | 0.99 | 0.99 | 1 yrs |
|      |       | 3, Table 2 | | | | 0.42 | 0.81 | 2 | 4 | 3 | 0.99 | 0.99 |
| B    | Caron et al. (1985) | 6 | Cell volume | E | LE | 0.22 | 0.21 | 0.93 | 9 | 4 | 0.51 | 1.1 | 0.25 days |
| C    | DeLong et al. (2014) | 1b, 1d | Cell volume | E | LE | 0.11 | 0.14 | 1.2 | 9 | 8 | 0.98 | 1.2 | 0.25 days |
| D    | Fenchel and Jonsson (1988) | 2a | Cell volume | E | LE | 1.9 | 3.1 | 1.6 | 15 | 115 | 8.2 | 40 | 12 hrs |
|      |       | 2b | | | | 1.5 | 1.9 | 1.3 | 15 | 73 | 5.2 | 16 |
| E    | Gonzalez et al. (1993) | 1a, 1b | Cell volume | E | LE | 0.28 | 0.2 | 0.74 | 8 | 164 | 23 | 51 | 10 hrs |
|      |       | 1a, 1b | | | | 0.24 | 0.24 | 1 | 8 | 165 | 24 | 52 |
| F    | Lürling et al. (2005) | 1A, 2 | Clump size | E | LE | 0.55 | 0.27 | 0.9 | 36 | 50 | 1.4 | 3.2 | 1 days |
| G    | Ozgul et al. (2010) | 1b, 1c | Weight | E | FS | 1.7 | 3.1 | 1.9 | 31 | 32 | 1.1 | 1.9 | 2 yrs |
| H    | Suzuki et al. (2017) | 3a, 3c | Cell volume | E | LE | 0.023 | 0.017 | 0.72 | 20 | 32 | 1.7 | 3 | 0.05 days |

**Unknown**

| α    | Brown and Brown (2013) | 1a, 1c | Wing length (nest) | E | FS | 0.69 | 0.47 | 0.68 | 30 | 29 | 1 | 1.2 | 1.5 yrs |
|      |       | 1a, 1c | Wing length (roadkill) | FS | 0.69 | 0.71 | 1 | 25 | 29 | 1.2 | 4 |
| β    | Coltman et al. (2003) | 2a, 2c | Weight (adult male) | E | FS | 1.8 | 2.8 | 1.5 | 31 | 30 | 1 | 1.1 | 6 yrs |
| γ    | Edeline et al. (2008) | 1c, 1d | Body length | O | FS | 2.2 | 3.7 | 1.7 | 51 | 50 | 1 | 1.2 | 4 yrs |
|      |       | 1c, 1d | | | | 2.4 | 3.7 | 1.6 | 51 | 50 | 1 | 1 |
|      |       | 1c, 1d | | | | 2.4 | 3.5 | 1.5 | 51 | 50 | 1 | 1 |
| δ    | Ezard et al. (2009) | 1a, Ozgul 1c | Weight | E | FS | 4.3 | 5.6 | 1.3 | 16 | 15 | 1 | 1.1 | 6.5 yrs |

Note: ϵ denotes the coefficient of variation (CV), den denotes the denominator of the CV, φ denotes the mean of the trait, and SI denotes the selection intensity.
<table>
<thead>
<tr>
<th>Code</th>
<th>Study</th>
<th>Figure</th>
<th>Trait</th>
<th>Data</th>
<th>Type</th>
<th>$\epsilon_{\text{den}}$</th>
<th>$\epsilon_{\text{ph}}$</th>
<th>$\epsilon_{\text{ph}}/\epsilon_{\text{den}}$</th>
<th>SS Total</th>
<th>Avg</th>
<th>Max</th>
<th>Gen</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\eta$</td>
<td>Fischer et al. (2014)</td>
<td>D2</td>
<td>Proportion strain 1</td>
<td>E</td>
<td>LE</td>
<td>0.27</td>
<td>0.33</td>
<td>1.2</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td>0.78</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.18</td>
<td>0.28</td>
<td>1.6</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.23</td>
<td>0.25</td>
<td>1.1</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.24</td>
<td>0.28</td>
<td>1.2</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.21</td>
<td>0.32</td>
<td>1.5</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.21</td>
<td>0.2</td>
<td>0.95</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.18</td>
<td>0.28</td>
<td>1.5</td>
<td>16</td>
<td>34</td>
<td>2.3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.15</td>
<td>0.2</td>
<td>1.4</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.22</td>
<td>0.17</td>
<td>0.79</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.22</td>
<td>0.31</td>
<td>1.4</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D2</td>
<td></td>
<td>E</td>
<td>LE</td>
<td>0.25</td>
<td>0.18</td>
<td>0.71</td>
<td>16</td>
<td>35</td>
<td>2.3</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td>$\theta$</td>
<td>Nakazawa et al. (2007)</td>
<td>2b, 3a</td>
<td>Body length</td>
<td>E</td>
<td>FS</td>
<td>0.72</td>
<td>0.69</td>
<td>0.95</td>
<td>28</td>
<td>27</td>
<td>1</td>
<td>1.2</td>
<td>1.25</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Schoener et al. (2002)</td>
<td>2b, 3b</td>
<td>Wet weight</td>
<td>E</td>
<td>FS</td>
<td>0.72</td>
<td>0.79</td>
<td>1.1</td>
<td>28</td>
<td>27</td>
<td>1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>Schoener et al. (2002)</td>
<td>5A, 8</td>
<td>Hindlimb length</td>
<td>E</td>
<td>FS</td>
<td>0.62</td>
<td>0.66</td>
<td>1.1</td>
<td>3</td>
<td>2</td>
<td>0.99</td>
<td>0.99</td>
<td>1</td>
</tr>
<tr>
<td>$\xi$</td>
<td>Schoener et al. (2002)</td>
<td>5A, 8</td>
<td>Lamellae number</td>
<td>E</td>
<td>FS</td>
<td>0.62</td>
<td>0.94</td>
<td>0.94</td>
<td>3</td>
<td>2</td>
<td>0.99</td>
<td>0.99</td>
<td></td>
</tr>
<tr>
<td>$\rho$</td>
<td>Sinervo et al. (2000)</td>
<td>1d, 1c</td>
<td>Clutch size</td>
<td>E</td>
<td>FS</td>
<td>0.62</td>
<td>0.72</td>
<td>1.2</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>1.1</td>
<td>1</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Sinervo et al. (2000)</td>
<td>1d, 1e</td>
<td>Egg mass</td>
<td>E</td>
<td>FS</td>
<td>0.62</td>
<td>0.87</td>
<td>1.4</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>1.1</td>
<td></td>
</tr>
<tr>
<td>$\tau$</td>
<td>Swain et al. (2007)</td>
<td>1d, 1f</td>
<td>Proportion orange-throated (female)</td>
<td>E</td>
<td>FS</td>
<td>0.6</td>
<td>0.99</td>
<td>1.7</td>
<td>11</td>
<td>10</td>
<td>0.99</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$\phi$</td>
<td>Swain et al. (2007)</td>
<td>1d, 1f</td>
<td>Body length at age 6 years</td>
<td>E</td>
<td>FS</td>
<td>0.62</td>
<td>0.93</td>
<td>1.5</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>1.1</td>
<td></td>
</tr>
<tr>
<td>$\psi$</td>
<td>Tessier et al. (1992)</td>
<td>1a, 5b</td>
<td>Body size</td>
<td>E</td>
<td>FS</td>
<td>1.5</td>
<td>1.4</td>
<td>0.99</td>
<td>40</td>
<td>33</td>
<td>0.85</td>
<td>1.9</td>
<td>4.5</td>
</tr>
<tr>
<td></td>
<td>Tessier et al. (1992)</td>
<td>1b, 5b</td>
<td>Body size</td>
<td>E</td>
<td>FS</td>
<td>2.7</td>
<td>2.7</td>
<td>0.97</td>
<td>17</td>
<td>81</td>
<td>5.1</td>
<td>15</td>
<td>30</td>
</tr>
</tbody>
</table>

Note: $\epsilon_{\text{den}}$, $\epsilon_{\text{ph}}$, $\epsilon_{\text{ph}}/\epsilon_{\text{den}}$, SS Total, Avg, Max, Gen, Unit are the data types and values associated with each code and study.
APPENDIX B

Supplementary Information for ‘Chaos does not drive lower synchrony for intrinsically-induced population fluctuations’

B.1 Proof of vec-permutation matrix conversion

Here, we prove the vec-permutation decomposition. First, we can define a general projection matrix, $A$, with $n$ life stages and $k$ patches in terms of demographic and dispersal processes separately,

$$A_{px} = \begin{cases} 
B_x(I_n - \sigma), & p = x \\
B_x\sigma D_{px}, & p \neq x 
\end{cases} \tag{B.1}$$

Here, $I_n$ represents the $n \times n$ identity matrix, $B_x$ is the $n \times n$ demographic submatrix, $\sigma$ is the $n \times n$ dispersal rate matrix, and $D_{px}$ is the proportion of dispersing individuals that move from patch $x$ to patch $p$. This function states that every submatrix along the diagonal, $A_{px}$ for $p = x$, undergoes some demographic process, $B_x$ for the proportion of individuals that remain in the patch, $I_n - \sigma$. The off-diagonal entries, $A_{px}$ for $p \neq x$, represents the proportion of individuals in patch $x$ that disperse to patch $p$ who then undergo the demographic process in patch $x$ defined by $B_x$.

This function can be rewritten as one formula using the Kronecker product, $\otimes$,

$$A = B(I_k \otimes (I_n - \sigma)) + BD \otimes \sigma$$
$$= B(I_k \otimes (I_n - \sigma) + D \otimes \sigma,$$

where $I_k$ is the $k \times k$ identity matrix, and $(I_k \otimes (I_n - \sigma) + D \otimes \sigma$) is a matrix composed of diagonal submatrices with elements that represent the proportion of individuals that disperse from one patch to another.

Now, given two matrices of the same size, $Q$ and $R$, with some vec-permutation, $V$,
the Kronecker product of the two matrices can be reversed using the identity, \( R \otimes Q = V^T (Q \otimes R) V \) (Henderson and Searle 1981). With this identity, we can define \( A \) as,

\[
A = B [I_k \otimes (I_n - \sigma) + D \otimes \sigma] \\
= B [V^T ((I_n - \sigma) \otimes I_k) V + V^T ((\sigma) \otimes D)V] \\
= B V^T [(I_n - \sigma) \otimes I_k + (\sigma) \otimes D] V \\
= B V^T M V,
\]

where \( M = (I_n - \sigma) \otimes I_k + \sigma \otimes D \) is a block diagonal matrix organized by life stage such that each submatrix, \( M_{px} \), is a \( k \times k \) matrix that represents the proportion of individuals that disperse from patch \( x \) to patch \( p \) (with diagonal entries representing the proportion of individuals that remain in the patch).

### B.2 Total maturation probability

To show the eventual probability of maturation from one life-history stage to the next, consider a species with 2 life-history stages (juveniles, adults) with juvenile survival \( (S_j) \) and juvenile to adult transition probability \( (P_{ja}) \). Given the number of juveniles at any point in time, the probability that a juvenile transitions to an adult in \( t \) time steps is \( S_j P_{ja} (S_j(1 - P_{ja}))^{t-1} \). Then the fraction of all individuals that mature to adults after \( T \) time steps is \( \sum_{t=1}^{T} S_j P_{ja} (S_j(1 - P_{ja}))^{t-1} \). Taking the limit as \( T \) goes to infinity gives the total fraction of all individuals that eventually mature into adults, \( \lim_{T \to \infty} \sum_{t=1}^{T} S_j P_{ja} (S_j(1 - P_{ja}))^{t-1} = \frac{S_j P_{ja}}{1 - S_j(1 - P_{ja})} = \psi_{ja} \). The same formulation can be used for juvenile to subadult transitions \( (\psi_{js}) \), and subadult to adult transitions \( (\psi_{sa}) \).

### B.3 Bifurcation analysis in the one patch model

We use bifurcation plots with Lyapunov exponent and eigenvalue plots to determine bifurcation types and routes to chaos in the one patch model. Bifurcation plots are created by plotting adult populations in the final 50 time steps vertically for each \( S_a \) value, and the Lyapunov exponents \( (\lambda_e) \) are calculated using the final 9000 time steps. Eigenvalues are
Fig. B.1: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 1-stage, 1 patch model. We find that k-cycles appear through flip bifurcations at $S_a \approx 0.6$. From there, we find two different routes to chaos. For $R_H = 96.77$, chaos occurs in one section ($0.2 \leq S_a \leq 0.4$) through period doublings in the k-cycles. Period doublings also induce chaos for $R_H = 247.98$ and $R_H = 399.19$ on the left-most and right-most chaotic bounds, and for chaos found in the intermediate section of $R_H = 399.19$. However, for $S_a \approx 0.2$ and $S_a \approx 0.35$, a window of stable 3-cycles appear through a phase-locking route.

calculated numerically using analytically solved fixed points (see main text).
Fig. B.2: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 1-stage, 1 patch model. Similar to bifurcations in $R_H$, $k$-cycles form through flip bifurcations and transition to chaos through period doublings. We also see small windows of stable $k$-cycles within chaos regions.
Fig. B.3: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 2-stage, 1 patch model. Hopf bifurcations create invariant cycles for $S_a \approx 0.6$ which experience phase-locking that create k-cycles in different regions. The phase-locked k-cycles may then either return to invariant cycles (e.g., $0.3 < S_a < 0.35$ for $R_H = 96.77$) or is replaced with chaos (e.g., $0.05 < S_a < 0.1$ for $R_H = 247.98$). In these plots, the invariant cycles develop wrinkles that mix trajectories, creating sensitivities to initial conditions.
Fig. B.4: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 2-stage, 1 patch model. Similar to the $R_H$ bifurcations, invariant cycles appear under Hopf bifurcations which are then phase-locked to k-cycles and in some cases chaos.
Fig. B.5: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select yearly recruitment ($R_H$) under all possible adult survival probabilities ($S_a$) in the 3-stage, 1 patch model. We find that invariant cycles occur through Hopf bifurcations with windows of phase-locked k-cycles.
Fig. B.6: Bifurcation diagrams (top row), Lyapunov exponents (middle row), and dominant eigenvalue (bottom row) for select $\gamma$ under all possible adult survival probabilities ($S_a$) in the 3-stage, 1 patch model. We find that invariant cycles occur through Hopf bifurcations.
B.4 Calculating multiple end states

To identify multiple fixed points for eigenvalue analysis, and end state behavior for bifurcation analysis, we numerically solve for fixed points in the projection matrix using trust-region root-finding techniques under a range of starting conditions (Coleman and Li 1996). Specifically, we solve for $A_n u_n = u_n$ with 200 random initial populations with each life stage ranging between 1 and 200 individuals. We then calculate eigenvalues of the Jacobian matrix with the numerically estimated fixed points. Then, we run simulations of the model initialized with 100 random initial conditions ranging between 1 and 200 individuals to ensure that simulations cover all basins of attractions. Bifurcation plots then use adult populations in patch 1 to represent end state behavior.
B.5 Two-patch model results

We plot attractor types, synchrony results, and dominant eigenvalues for all two patch simulations used in the main text. Each simulation was initialized with end states from the corresponding 1-patch simulation. Models were run for $T = 10000$ time steps and dynamics were identified by analyzing the range of the final 50 time steps. Simulations were classified as equilibria if the range of all populations were less than 0.01. Otherwise, they were classified as oscillating and attractor type was determined by calculating the Lyapunov exponent ($\lambda_e$) using the final 9000 time steps and the two patch Jacobian defined by,

$$\bar{J}_n(t) = \begin{bmatrix} (1 - \sigma_1)J_{n1}(t) & \sigma_2 J_{n2}(t) \\ \sigma_1 J_{n1}(t) & (1 - \sigma_2)J_{n2}(t) \end{bmatrix},$$

(B.2)

where $J_{nx}$ is the $n-$stage Jacobian for patch $x$ with corresponding 2-patch population vector $\bar{u}_n$. Stability types were classified as k-cycles for $\lambda_e < 0$, invariant cycles for $\lambda_e = 0$, and chaos (or strange attractors) for $\lambda_e > 0$. Additionally, we calculated total patch population synchrony using the final 500 time steps, and dominant eigenvalues. Note that the dominant eigenvalues may not reflect the simulations in the oscillation planes due to multiple fixed points and instead are used to show some of the bifurcations that occur in these models.

Each subplot represents results for 100 dispersal probabilities ranging from 0 to 1 for each life stage. This was done for all selected pairwise adult survival probabilities. With 15 pairwise adult survival probabilities and 10000 pairwise dispersal probabilities for each life stage provided a total of 150,000 simulations for the 1-stage model, 300,000 simulations for the 2-stage model, and 450,000 simulations for the 3-stage model.
Fig. B.7: Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 1-stage model.
Fig. B.8: Eigenvalue planes for adult dispersal in the two patch, 1-stage model. Solid black lines represent where bifurcations occur ($|\lambda| = 1$).
Fig. B.9: Attractor (left) and synchrony planes (right) for juvenile dispersal in the two patch, 2-stage model.
Fig. B.10: Eigenvalue planes for juvenile dispersal in the two patch, 2-stage model. The black regions in the top-right corners of the last two rows are a mix of eigenvalues of unstable steady states created under high dispersal probabilities.
Fig. B.11: Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 2-stage model.
Fig. B.12: Eigenvalue planes for adult dispersal in the two patch, 2-stage model.
Fig. B.13: Attractor (left) and synchrony planes (right) for juvenile dispersal in the two patch, 3-stage model.
Fig. B.14: Eigenvalue planes for juvenile dispersal in the two patch, 3-stage model. The small black region in the final plot is due to a saddle-node bifurcation that created a stable and unstable equilibrium.
Fig. B.15: Attractor (left) and synchrony planes (right) for subadult dispersal in the two patch, 3-stage model.
Fig. B.16: Eigenvalue planes for subadult dispersal in the two patch, 3-stage model. The small black region in the final plot is due to a saddle-node bifurcation that created a stable and unstable equilibrium.
Fig. B.17: Attractor (left) and synchrony planes (right) for adult dispersal in the two patch, 3-stage model.
Fig. B.18: Eigenvalue planes for adult dispersal in the two patch, 3-stage model.
Fig. B.19: Phase planes for the two patch, 1-stage model with parameters $S_{a1} = 0.1$, $S_{a2} = 0.3$, $\sigma_{a2} = 0.96$, and patch 1 dispersal $\sigma_{a1} = 0.75$ (A), $\sigma_{a1} = 0.85$ (B), and $\sigma_{a1} = 0.95$ (C). We find that two stable fixed points exist in each of these cases. First, a 4-point k-cycle occurs in (A), then as $\sigma_{a1}$ increases, the chaos exists via a period-doubling route from a flip bifurcation. Note that at least 2 unstable fixed points exist among these phase planes (verified with numerical simulations initialized by multiple fixed points).

B.6 Multiple end state phase planes

We found that dispersal may create multiple nontrivial fixed points. To demonstrate this, we plot phase planes for select scenarios in the two patch, 1-stage model that exemplify how multiple basins of attraction creates multiple nontrivial steady states.
Fig. B.20: Phase planes for the two patch, 1-stage model with parameters $S_{a1} = 0.3$, $S_{a2} = 0.3$, $\sigma_{a2} = \sigma_{a1} = 0.75$ (A), $\sigma_{a2} = \sigma_{a1} = 0.85$ (B), $\sigma_{a2} = \sigma_{a1} = 0.95$ (C). What starts as chaos formed from k-cycles (A) turns into two unique fixed points (B) then finally three unique chaotic cycles identified by red dots, circles, or stars (C). Each cycle in (C) is found by closely examining end-state behavior of simulations under initial conditions that are slightly perturbed from the different fixed points. Among these cycles exist two unstable equilibria. Note that because the two patches are symmetric with symmetric dispersal, the two stable equilibria in (B) and two of the chaotic cycles in (C; star, dot) have symmetric fixed points.
B.7 2-cycle saddle-node bifurcation

Figure B.21 are iteration maps that show the creation of a stable and unstable (saddle) 2-cycle produced from a 2-cycle saddle-node bifurcation that appears in Figure 3.10B in the main text. Plots were generated by fixing the adult stage population in patch 2 to the stable equilibrium while changing the adult population in patch 1 \((a_1)\). \(f(a_1, a_2)\) and \(f^2(a_1, a_2)\) are the first- and second-iterate maps of adult populations in patch 1. Specifically, \(a_1(t+1) = (1 - \sigma_{21})(S_{a_1} + FH_{a_1}(t))a_1(t) + \sigma_{12}(S_{a_2} + FH_{a_2}(t))a_2(t) = f(a_1(t), a_2(t))\) and \(a_2(t+2) = f(f(a_1(t), a_2(t))) = f^2(a_1(t), a_2(t))\). Fixed points are located at the intersections of the graph to the one-to-one line. The upper intersections in all plots represent the stable equilibrium that persists throughout the dispersal range in Figure 3.10B. The intersections in the second-iterate maps (plots D-F) show the locations of the 2-cycle orbits created at a 2-cycle saddle-node bifurcation between \(\sigma_{a_1} = 0.6\) and \(\sigma_{a_1} = 0.7\). Specifically, the 2-cycle saddle-node bifurcation produces a period-2 stable k-cycle and its associated period-2 saddle (unstable) k-cycle.
Fig. B.21: First- and second-iterate maps of the two-patch, 1 life-history stage model for $S_{a_1} = 0.1$, $S_{a_2} = 0.3$, $\sigma_{a_2} = 0.96$ (Figure 3.10B in the main text). The upper intersections in all plots represent the persistent stable equilibrium for this scenario. In plot D, $f^2(a_1, a_2)$ lies near the one-to-one line but does not intersect. As $\sigma_{a_1}$ increases, the graph crosses the one-to-one line, creating a stable and saddle 2-point k-cycle via 2-cycle saddle-node bifurcation.
B.8 Matlab code

The following are Matlab scripts to generate simulations, analysis, and figures for the two patch model. All scripts were run in Matlab R2021a.

- `make_demographic_matrix.m`: Creates demographic matrix for the one and two patch model. For two patches, this creates a diagonal matrix with submatrices that represent demography in each patch.

- `calc_eigs.m`: Calculate dominant eigenvalues solved Jacobians for fixed points in the one patch model.

- `create_Jacobian.m`: Creates demographic Jacobians for the one and two patch models (does not require fixed point densities). The output is a diagonal matrix that is multiplied by the dispersal matrix to acquire the projection matrix Jacobian.

- `calculate_Lyapunov_exp.m`: Calculates Lyapunov exponent for the one patch model.

- `calculate_Lyapunov_exp_2patch.m`: Calculates the Lyapunov exponent for the two patch model. Also outputs indices where different attractor types occur.

- `calculate_synchrony.m`: Calculates total population synchrony.

- `make_vec_matrix.m`: Creates vec-permutation matrix for relating stage-based matrices with patch-based matrices.

- `analyze_twoParms.m`: Loops through demographic parameter ranges in the one patch model for $R_H$, $S_{a}$, and $\gamma$, and calculates Lyapunov exponents for each simulation. Used in `onpatch_end_state.m`.

- `onpatch_end_state.m`: Runs simulations of the one patch model. Saves attractor type, Lyapunov exponent, end state populations, and dominant eigenvalues. Save files are used for manuscript figures.

- `onpatch_phase_planes.m`: Creates phase plane plots for the one patch, 2-stage model to show different attractor types.
- **Analyze_one_patch_save_data.m**: Creates figures for the main one patch results in manuscript. Includes all attractor planes, eigenvalue planes, select bifurcation plots, select Lyapunov exponent plots, and select eigenvalue plots.

- **make_distance_based_dispersal.m**: Creates dispersal submatrix, $D$, for the dispersal matrix $M$.

- **Zone_plane.m**: Creates figure that identifies dispersal zone locations.

- **analyze_parms_2patch.m**: Loops through dispersal probabilities in the two patch model and numerically solve fixed points, Lyapunov exponents, and synchrony. Used in twopatch_end_state.m.

- **twopatch_end_state.m**: Runs simulations of the two patch model. Saves attractor type, Lyapunov exponent, end state populations, dominant eigenvalues, and synchrony. Save files are used for manuscript figures.

- **Analyze_two_patch_save_data.m**: Creates figures for the main two patch results in the manuscript. This includes all attractor planes, synchrony planes, specific attractor planes that show dispersal-induced oscillations, and all histograms that summarize all results.

- **chaos_fractal_basins.m**: Creates fractal basin plots that show the existence of multiple steady states in the two patch model.

- **two_patch_bifurcations.m**: Creates bifurcation plots for select models to show different fixed points, steady states, and eigenvalues.

- **two_patch_iteration_maps.m**: Creates iterate maps used in the appendix that show the creation of a stable and unstable 2-cycle that forms through a 2-cycle saddle-node.

- **two_patch_phase_planes.m**: Creates phase plane figures for select scenarios to show multiple steady states.
• twopatch_lambda_plane.m: Creates eigenvalues plane figures for all scenarios in the two patch model. Only one fixed point is solved that may or may not reflect the steady state shown in the attractor planes.
function [A_diag] = make_demographic_matrix(parms_in, a_t, num_stages)
% Creates a diagonal matrix with submatrices that represent density
% dependent demographic processes for a single-stage organism

num_patches = size(parms_in,2);

% Make diagonal matrix with transition matrices for each pond
for ii = 1:num_patches

  % Parameters
  F = parms_in(1,ii); % Fecundity
  Sn = parms_in(2,ii); % Mean survival from metamorph to age 1
  Sj = parms_in(3,ii); % juvenile survival
  Ss = parms_in(4,ii); % subadult survival
  Sa = parms_in(5,ii); % female adult survival
  Pjs = parms_in(6,ii); % juvenile to subadult transition
  Psf = parms_in(7,ii); % subadult to female adult transition
  K = parms_in(8,ii); % carrying cap
  gamma = parms_in(9,ii); % density-dependent exponent

  % density dependent larval survival
  e_to_m_surv = Sn.*(1./(1+a_t(ii).*F./K).^gamma);

  if num_stages == 1
    % 1 stage demographic matrix
    B_x = Sa + F*e_to_m_surv;
  elseif num_stages == 2
    % 2 stage demographic matrix for pond x
    B_x = [Sjs*(1-Psf) F*e_to_m_surv; Sj*Psf Sa];
  else
    % 3 stage demographic matrix for pond x
    B_x = [Sjs*(1-Pjs) 0 F*e_to_m_surv; Sj*Pjs Ss*(1-Psf) 0; 0 Ss*Psf Sa];
  end

% Diagonal Matrix
\[ A_{\text{diag}}((ii - 1) \ast \text{num\_stages} + 1: ii \ast \text{num\_stages}, (ii - 1) \ast \text{num\_stages} + 1: ii \ast \text{num\_stages}) = B \cdot x \]

end

dep
function [lambda_out,lambda_0] = calc_eigs(num_stages, all_parms)

%% Calculate eigenvalues of trivial and nontrivial fixed points using
%% Jacobian matrix for up to 3 life stages in the one patch model

% Parameters
F = all_parms(1);          % Fecundity
L = all_parms(2);          % first-year survival
Sj = all_parms(3);         % juvenile survival
Ss = all_parms(4);         % subadult survival
Sa = all_parms(5);         % adult survival
Pjs = all_parms(6);        % juvenile to subadult transition
Psa = all_parms(7);        % subadult to female adult transition
K = all_parms(8);          % carrying capacity
gamma = all_parms(9);      % density-dependent exponent

if num_stages == 1

  % trivial equib
  J_0 = Sa+F*L;
  % nontrivial equib
  J = 1 - gamma*(1-Sa)*(1-((1-Sa)/(F*L))^(1/gamma));

elseif num_stages == 2

  % trivial equib
  J_0 = [Sj*(1-Psa) 0 0; Sj*Psa Sa];
  % nontrivial equib
  psi_ja = Psa*Sj/(1-Sj*(1-Psa));
  J = [Sj*(1-Psa) (1-Sa)/psi_ja*(1-gamma*(1-((1-Sa)/(F*L*psi_ja)))^(1/gamma));
      Sj*Psa Sa]

else

  % trivial equib
  J_0 = [Sj*(1-Pjs) 0 0; Sj*Pjs Ss*(1-Psa) 0; 0 Ss*Psa Sa];
  % nontrivial equib
  psi_js = Pjs*Sj/(1-Sj*(1-Pjs));
  psi_sa = Psa*Ss*(1-Ss*(1-Psa));
  J = [Sj*(1-Pjs) 0 (1-Sa)/psi_js*psi_sa*b@x-(1-gamma*(1-((1-Sa)/(F*L*psi_js*psi_sa)))^(1/gamma)));
      Sj*Pjs Ss*(1-Psa) 0; 0 Ss*Psa Sa];
end

% trivial eigenvalue
E_0 = eig(J_0);
[~, I_0] = max(abs(E_0));
lambda_0 = E_0(I_0);

% nontrivial eigenvalue
E = eig(J);
[~, I] = max(abs(E));
lambda_out = E(I);
end
function J_diag = create_Jacobian(tot_patches, num_stages, all_parms, N)
% Create Jacobian for 1, 2, or 3 stages and multiple patches
% Density-dependence exists in larval survival only

% Define local patch Jacobian matrix
for ii = 1:tot_patches
    % Parameters
    F = all_parms(1, ii); % Fecundity
    L = all_parms(2, ii); % first-year survival
    Sj = all_parms(3, ii); % juvenile survival
    Ss = all_parms(4, ii); % subadult survival
    Sa = all_parms(5, ii); % adult survival
    Pjs = all_parms(6, ii); % juvenile to subadult transition
    Psa = all_parms(7, ii); % subadult to female adult transition
    K = all_parms(8, ii); % carrying capacity
    gamma = all_parms(9, ii); % density-dependent exponent

    Hx = L./(1+N(num_stages*(ii-1)+num_stages).*F./K).^gamma;

    dfdN = F.*Hx-gamma.*F.^2./(K.*(1+F.*N(num_stages*(ii-1)+num_stages).*K)...*Hx.*N(num_stages*(ii-1)+num_stages));

    if num_stages == 1
        J = Sa + dfdN;
    elseif num_stages == 2
        J = [Sj*(1-Psa) dfdN; Sj*Psa Sa];
    else
        J = [Sj*(1-Pjs) 0 dfdN; Sj*Pjs Ss*(1-Psa) 0; 0 Ss*Psa Sa];
    end

    J_diag((ii-1)*num_stages+1:ii*num_stages,(ii-1)*num_stages+1:ii*num_stages) = J;
end
end
function lyap_exp_out = calculate_Lyapunov_exp(N_in, num_stages, all_parms)

end_N = N_in(:,1000:end);
N = end_N;
X = end_N(:,1);

% Loop through final time steps
for tt = 1:size(end_N,2)
    J_diag = create_Jacobian(1, num_stages, all_parms, N(:,tt));
    % % project population by one time step
    A = J_diag;
    X(:,tt+1) = A*X(:,tt);
    lyap_exp(tt+1) = log(norm(X(:,tt+1))/tt);
end

lyap.exp(isnan(lyap.exp))=[];
lyap.exp(isinf(lyap.exp))=[];

lyap.exp.out = lyap.exp(end);

end
function [thresh, chaos_thresh, inv_cycle_thresh] = ...
    calculate_Lyapunov_exp_2patch(N_in, num_stages, all_parms, sj1, sj2, sigma_inds)

% Calculate Lyapunov exponent for 2patch model

% Make diagonal block dispersal matrix (organized by life stage)
D = ones(2);
D = D - diag(diag(D));
sigma_new = zeros(num_stages); % disp rate matrix
M = kron(eye(size(sigma_new)),sigma_new,eye(size(D))) + kron(sigma_new,D);

V = make_vec_matrix(2,num_stages);

thresh = -1;
chaos_thresh = -1;
inv_cycle_thresh = -1;

% Fixed dispersal
sigma_j_vector = [sj1 sj2]';

% Change dispersal to specified life stage
M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) +...
    sigma_j_vector' .* D;

end N = N_in(:,1000:end);
N = end N;
X = end N(:,1);
for tt = 1:size(end N,2)
    J_diag = create_Jacobian(2, num_stages, all_parms, N(:,tt));
    % % project population by one time step
    A = V'*M*V*J_diag;
    X(:,tt+1) = A*X(:,tt);
    lyap_exp(tt+1) = log(norm(X(:,tt+1)))/tt;
end
lyap_exp(isnan(lyap_exp)) = [];
lyap_exp(isinf(lyap_exp)) = [];
if lyap_exp(end) > 0.001
    chaos_thresh = 1;
elseif lyap_exp(end) < -0.001
    thresh = 1;
else
inv_cycle_thresh(end) = 1;
end

end
function N_sync = calculate_synchrony(N_in, num_stages)

% Calculate total population synchrony
% final densities in each patch
N_in(N_in<10^{-20}) = 0; % round small rounding errors to 0
% Total population synchrony
N_cov_temp = cov([sum(N_in(1:num_stages,:),1);sum(N_in(num_stages+1:end,:),1)]');
N_sync = sum(N_cov_temp(:))/(sum(diag((N_cov_temp).^0.5)))^2;
end
function V = make_vec_matrix(tot_ponds, num_stages)

% Create vec-permutation matrix for relating stage-based with patch-based % models
V = zeros(tot_ponds*num_stages);
I = eye(tot_ponds*num_stages);
k = 1;
for i=1:num_stages
    for j=i:num_stages:tot_ponds*num_stages
        V(k,:) = I(j,:);
        k = k+1;
    end
end
end
function [lyap_exp, max_lambda, end_N] = analyze_two_parms(all_parms, parm_1, parm_2, range_1, range_2, num_stages)

% % Loop through parameter spaces and calculate Lyapunov exponent for
% % attractor type identification

T = 10000; % time steps
IC = [20 20 20]; % initial conditions
for p1 = 1:length(range_1)
    all_parms(parm_1, num_stages) = range_1(p1);
    for p2 = 1:length(range_2)
        all_parms(parm_2, num_stages) = range_2(p2);

        % % Calculate eigenvalues
        [max_lambda(p1, p2), lambda_0] = calc_eigs(num_stages, all_parms(:, num_stages));

        N = IC(1:num_stages)';
        for tt = 1:T
            % num egg masses in previous time step
            a_t = N(num_stages:end, tt);
            % % density dependent larval survival
            B = make_demographic_matrix(all_parms(:, num_stages), a_t, num_stages);
            N(:, tt+1) = B*N(:, tt);
        end
    end
end_N(p1, p2, :) = N(:, end-100:end);

% If cycles occur, calculate lyapunov exp
if abs(max_lambda(1))>1 && abs(lambda_0)>1
    lyap_exp(p1, p2)...
        = calculate_Lyapunov_exp(N, num_stages, all_parms(:, num_stages));
else
    lyap_exp(p1, p2) = NaN;
end
function onepatch_end_state

% Runs simulations for the one-patch model, calculates unstable attractor
% type, and saves files.
% NOTE: end-state .mat files use a lot of memory
%
% clear all; close all

% Parameters
F = [500 500 500]; % Fecundity
L = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent

% Analysis

% Define parm ranges
all_parms = [F; L; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
min_parms = [1, .01, .01, .01, .01, .01, .01, 1000, 1]; % min values
max_parms = [500, 1, .99, .99, .99, .99, .99, 150000, 100]; % max values

% Define parm indices
num_tests = 100;

% Plot each phase diagram for two parameters at a time
for p1 = 1:length(test_parms)
  for p2 = 1:length(test_parms)
    if p2 > p1
      ind1 = test_parms(p1);
      ind2 = test_parms(p2);
      range1 = linspace(min_parms(ind1),max_parms(ind1),num_tests);
      range2 = linspace(min_parms(ind2),max_parms(ind2),num_tests);
[lyap_exp,max_lambda,end_N] = analyze_two_parms(allParms,inds_1,...
    inds_2,range_1,range_2,stages);

% % save Lyapunov exponent
% lyap_data = ['one_patch_save_data/Lyap_1patch_1',num2str(stages),'stage_1',
    '.parm_labels{pl}','.',parm_labels{p2}','.csv'];
% csvwrite(lyap_data,lyap_exp)
% % save N
% filename = ['one_patch_save_data/','num2str(stages)',',stage_1',
    'parm_labels{pl}',',',parm_labels{p2}',',csv'];
% save(filename,'end_N')
% save lambda
% lambda_data = ['one_patch_save_data/Lambda_1patch_1',num2str(stages),'
    stage_1',',parm_labels{pl}',',',parm_labels{p2}',',csv'];
% csvwrite(lambda_data,max_lambda)

end

end
end
end
function onepatch_phase_planes

% % Plot phase planes with different attractor types and Lyapunov exponents

% clear all; close all
% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first—year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density—dependent exponent
mean_parms_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
all parms = mean_parms_all(:, 2);

% % Analysis

subfig_labels = 'ABCDE';
um_stages = 2;
parm_1 = 2;
parm_2 = 5;

% Initializations
T = 10000;
IC = [20 20 20];
allparms(parm_1) = 167.333/500;
range_2 = [0.366364 0.227778 0.138687 0.0990909];
for p2 = 1:length(range_2)
    figure
    allparms(parm_2) = range_2(p2);

    N = IC(1:num_stages)';
    for tt = 1:T
        % num egg masses in previous time step
        a_t = N(num_stages:num_stages:end, tt);
        % % density dependent larval survival
        B = make_demographic_matrix(all parms, a_t, num_stages);
\[ N(:, t_t+1) = B*N(:, t_t); \]

\textbf{end}

\% If cycles occur, find lyapunov exp
lyap_exp = calculate_Lyapunov_exp\(N, num\_stages, all\_parms\);

plot\(N(1, end-8999:end), N(2, end-8999:end), 'k.', 'markersize', 12\)

hold on
annotation(gcf, 'textbox', ...
[0.18928571428571 0.835666668298699 0.357142857142858 0.0785714269393967],...
'String', ['\lambda e = ' num2str(round(100*lyap\_exp)/100)],...
'LineStyle', 'none',...
'FontSize', 15);

% Create textbox
annotation(gcf, 'textbox', ...
[0.839383838383838 0.842995169082125 0.0495050505050505 0.0772946859903378],...
'String', num2str(subfig\_labels(p2)),...
'LineStyle', 'none',...
'FontWeight', 'bold',...
'FontSize', 15,...
'FitBoxToText', 'off');

\textbf{set} (gcf, 'Position', [500 476 495 414])
xlabel('j')
ylabel('a')
xlim([0 300])
ylim([0 60])
\textbf{set} (gca, 'fontsize', 17)

\% \textbf{print} (gcf, ['1 patch\_figs/Lyap\_sample_', num2str(p2), '.eps'], '-depsc', '-r600')

\textbf{end}
function Analyze_one_patch_save_data

% Create figures for attractor planes, eigenvalue planes, and bifurcation
% plots, eigenvalue plots, and Lyapunov plots for select scenarios in the
% one patch model

% All files used are located in save_data
% Figures are saved in 1patch_figs

% Define parm ranges
min_parms = [1, 1, 0.01, 0.01, 0.01, 0.01, 1000, 1]; %min values
max_parms = [1000, 500, 0.99, 0.99, 0.99, 0.99, 0.99, 150000, 100]; %max values

test_parms = [2 9 5]; % parm indices to test
parm_labels_all = {'\text{RH}', '\gamma', 'S_a'};
parm_labels = {'\text{RH}', '\gamma', 'S_a'};

num_tests = 100;

%% Plot chaos phase planes for each scenario
subfig_labels = 'ABCDE';
for stages = 1:3
    figure
    for pl = 1:length(test_parms)-1
        subplot(2,1,pl);
        plot(-1,-1,'b.','markersize',10)
        hold on
        plot(-1,-1,'.','color', [.6 .6 .6], 'markersize',10)
        plot(-1,-1,'.', 'color', [.4 .4 .4], 'markersize',10)
        plot(-1,-1,'.', 'color', [.2 .2 .2], 'markersize',10)
        plot(-1,-1,'.', 'color', [0 0 0], 'markersize',10)
        plot(-1,-1,'.', 'color', [0.8500 0.3250 0.0980], 'markersize',10)
        plot(-1,-1,'.', 'color', [0.9290 0.6940 0.1250], 'markersize',10)
        plot(-1,-1,'r.', 'markersize',10)
        kcyclexthresh = -1*ones(num_tests, num_tests);
        chaos_thresh = -1*ones(num_tests, num_tests);
        inv_cycle_thresh = -1*ones(num_tests, num_tests);
        two_cycle = -1*ones(num_tests, num_tests);
four_cycle = -1*ones(num_tests,num_tests);
eight_cycle = -1*ones(num_tests,num_tests);
sixteen_cycle = -1*ones(num_tests,num_tests);
sixteen_plus_cycle = -1*ones(num_tests,num_tests);
odd_cycle = -1*ones(num_tests,num_tests);

% % Load save file
lyap_file = ['one_patch_save_data/Lyap_1patch_' , num2str(stages),... 
    'stage_' , parm_labels{1},'_' , parm_labels{3},'.csv'];
lyap.data = csvread(lyap_file);
N.data = ['one_patch_save_data/\',num2str(stages),'
    stage_' , parm_labels{1},'_' , parm_labels{3},'.csv'];
load,N = load(N.data);
end_N = load,N/end_N;
lambda_file = ['one_patch_save_data/Lambda_1patch_' , num2str(stages),... 
    'stage_' , parm_labels{1},'_' , parm_labels{3},'.csv'];
lambda.data = csvread(lambda_file);

% Identify attractor type
k_cycle_thresh(lyap.data<-0.001) = 1;
chaos_thresh(lyap.data>0.001) = 1;
inv_cycle_thresh(abs(lyap.data)<=0.001) = 1;

% calculate k number
for parm1 = 1:num_tests
    for parm2 = 1:num_tests
        if k_cycle_thresh(parm1,parm2)==1
            N = squeeze(end_N(parm1,parm2,:,:));
        end
        % Check for cycles
        if abs(lambda.data(parm1,parm2))>1
            % if any(max(N')-min(N')>0.01)
            % calculate k-point
            if stages == 1
                final_N = N(end);
                k_temp = find(abs(N - final_N)<0.001);
            else
                final_N = sum(N(:,end),1);
                k_temp = find(abs(sum(N,1) - final_N)<0.001);
            end
        end
    end
end
end

% K-cycles may have high k-points - Identify as 16+
if length(k_temp)==1
    k_cycle_period = 17; % for near-chaos events
else
    k_cycle_period = k_temp(end)-k_temp(end-1);
end
if k_cycle_period ==2
    two_cycle(parm1, parm2) = 1;
elseif k_cycle_period ==4
    four_cycle(parm1, parm2) = 1;
elseif k_cycle_period ==8
    eight_cycle(parm1, parm2) = 1;
elseif k_cycle_period == 16
    sixteen_cycle(parm1, parm2) = 1;
elseif k_cycle_period > 16
    sixteen_plus_cycle(parm1, parm2) = 1;
else
    odd_cycle(parm1, parm2) = 1;
end
end
end
end
end

inds_1 = test_parms(p1);
inds_2 = test_parms(3);
range_1 = linspace(min_parms(inds_1), max_parms(inds_1), num_tests);
range_2 = linspace(min_parms(inds_2), max_parms(inds_2), num_tests);
if range_1(1) == 0
    range_1 = 0.001;
end
plot(range_2, range_1.*inv_cycle_thresh,'b.'),
hold on
plot(range_2, range_1.*two_cycle,'.','color', [.6 .6 .6])
plot(range_2, range_1.*four_cycle,'.','color', [.4 .4 .4])
plot(range_2, range_1.*eight_cycle,'.','color', [.2 .2 .2])
plot(range_2, range_1.*sixteen_cycle,'.','color', [0 0 0])
plot(range_2, range_1.*sixteen_plus_cycle,'.','color', [0.8500 0.3250 0.0980])
```matlab
plot(range_2, range_1.*odd_cycle,'o','color',[0.9290 0.6940 0.1250]);
plot(range_2, range_1.*chaos_thresh,'r.'),
xlim([range_2(1) range_2(end)]),
ylim([range_1(1) range_1(end)]),
if pl == 2
    xlabel({parm_labels_all{3};num2str(stages),'-stage'});
end
ylabel(parm_labels_all{pl}),
title(num2str(subfig_labels(stages+3)));
end
subplot(2,1,1)
if stages == 3
    legend('Invariant Cycles', '2 Cycles', '4 Cycles', '8 Cycles', '16 Cycles', '
16+ Cycles', 'Other Cycles','Chaos');
end

%% Save chaos plots
set(gcf,'Position',[680 399 308 579]),
title(num2str(subfig_labels(stages)));

%% Plot lambda planes for each scenario
subfig_labels = 'ABCDEF';
for stages = 1:3
    figure
    for pl = 1:length(test_parms)-1
        subplot(2,1,pl);
        % % Load save file
        lambda_file = ['one_patch_save_data/Lambda_onepatch_' num2str(stages),... 'stage.', parm_labels{pl}, '_' parm_labels{3},'.csv'];
        lambda_data = csvread(lambda_file);
        ind_1 = test_parms(pl);
        ind_2 = test_parms(3);
        range_1 = linspace(min_parms(ind_1),max_parms(ind_1),num_tests);
        range_2 = linspace(min_parms(ind_2),max_parms(ind_2),num_tests);
```
 contour (range_2, range_1, real(lambda_data), 'showtext', 'on')
 hold on
 [~ , c] = contour (range_2, range_1, abs(lambda_data), [1 1], 'k-');
 c.LineWidth = 3;

 xlim([range_2(1) range_2(end)])
 ylim([range_1(1) range_1(end)])
 if p1 == 2
 xlabel({parm_labels_all{3};[num2str(stages),'-stage']})
 end
 ylabel(parm_labels_all{p1})
 title(num2str(subfig_labels(stages + 3)))
 end
 subplot(2, 1, 1)
 if stages == 3
 legend('real(lambda)', '|lambda| = 1')
 end

%%% Save chaos plots
 set(gcf, 'Position', [680 299 308 579])
 title(num2str(subfig_labels(stages)))

%%% Plot bifurcations for select scenarios
 colors = [0.4940 0.1840 0.5560; 0.4660 0.6740 0.1880; 0.8500 0.3250 0.0980];
 for stages = 1:3
  for p1 = 1:length(test_parms)-1
   % % Load save file
   N_data = ['one_patch_save_data/', num2str(stages), 'stage', parm_labels{p1}, '_'
     , parm_labels{3}, '.mat'];
   load_N = load(N_data);
   end_N = load_N.end_N;

   inds_1 = test_parms(p1);
   inds_2 = test_parms(3);
   range_1 = linspace(min_parms(inds_1), max_parms(inds_1), num_tests);
   range_2 = linspace(min_parms(inds_2), max_parms(inds_2), num_tests);
sp_ind = 0;
figure1 = figure;
for bifs = [20,50,80]
    sp_ind = sp_ind+1;
    figure
    % Bifurcation plots
    subplot(1,3,sp_ind);
    if stages == 1
        plot(0,0,'.','color', colors(3,:))
        hold on
    elseif stages == 2
        plot(0,0,'.','color', colors(1,:))
        hold on
        plot(0,0,'.','color', colors(3,:))
    else
        plot(0,0,'.','color', colors(1,:))
        hold on
        plot(0,0,'.','color', colors(2,:))
        plot(0,0,'.','color', colors(3,:))
    end
    if stages == 1
        plot(range_2,squeeze(end_N(bifs,:,1,:)),'.','color', colors(3,:),'
     markersize',3)
    elseif stages == 2
        plot(range_2,squeeze(end_N(bifs,:,1,:)),'.','color', colors(1,:),' markersize',3)
        hold on
        plot(range_2,squeeze(end_N(bifs,:,2,:)),'.','color', colors(3,:),' markersize',3)
    else
        plot(range_2,squeeze(end_N(bifs,:,1,:)),'.','color', colors(1,:),' markersize',3)
        hold on
        plot(range_2,squeeze(end_N(bifs,:,2,:)),'.','color', colors(2,:),' markersize',3)
        plot(range_2,squeeze(end_N(bifs,:,3,:)),'.','color', colors(3,:),' markersize',3)
    end
    title({param_labels_all{pl},' = ',num2str(range_1(bifs))})
if sp_ind ==1
ylabel('End-behavior')

end

xlabel('S_a')

end

set(gcf, 'Position',[86.6000 639.4000 791.2000 148.8000])

% Create textbox
annotation(figure1,'textbox',...
    [0.130434782608696 0.655913978494624 0.0293225480283115 0.1844383870967745],...
    'String','A',...
    'LineStyle','none',...
    'FitBoxToText','off');

annotation(figure1,'textbox',...
    [0.411021233569262 0.653225806451615 0.0293225480283116 0.1844383870967745],...
    'String','B',...
    'LineStyle','none',...
    'FitBoxToText','off');

annotation(figure1,'textbox',...
    [0.692618806875632 0.645161290322584 0.0293225480283115 0.1844383870967745],...
    'String','C',...
    'LineStyle','none',...
    'FitBoxToText','off');

if stages == 1
    legend({'Adult'},{ 'Fontsize',8})
elseif stages == 2
    legend({'Juvenile','Adult'}, 'Fontsize',8)
else
    legend({'Juvenile','Subadult','Adult'}, 'Fontsize',8)
end

% % Save figure

% print(gcf,['1patch_figs/1patch_bif',num2str(stages)],...
%     parm_labels{pl}, 'stage.eps', '-depsc', '-r600')

end

end

close all


%% Plot Lyapunov exp for select scenarios

for stages = 1:3
for pl = 1:length(test_parms)-1

% % Load save file
lyap_file = ['one_patch_save_data/Lyap', ... 'patch', num2str(stages), ... 'stage_', parm_labels{pl}, '_', parm_labels{3}, '.csv'];
lyap_data = csvread(lyap_file);

inds_2 = test_parms(3);
range_2 = linspace(min_parms(inds_2), max_parms(inds_2), num_tests);
sp_ind = 0;
figure1 = figure;
for bifs = [20, 50, 80]
    sp_ind = sp_ind+1;
    subplot(1,3,sp_ind);
    plot(range_2, lyap_data(bifs,:), 'k--')
    hold on
    plot([range_2(1) range_2(end)], [0 0], 'k--')
    if sp_ind ==1
        ylabel('\lambda_e')
    end
    xlabel(parm_labels_all{3})
end

set(gcf, 'Position', [86.6000 639.4000 791.2000 148.8000])
% Create textbox
annotation(figure1, 'textbox', ...)
    [0.130434782608696 0.755913978494624 0.0293225480283115 ... 0.18443870967745], ...'
    'String', 'D', ...
    'LineStyle', 'none', ...
    'FitBoxToText', 'off');
annotation(figure1, 'textbox', ...)
    [0.411021233569262 0.753225806451615 0.0293225480283116 ... 0.18443870967745], ...'
    'String', 'E', ...
    'LineStyle', 'none', ...
    'FitBoxToText', 'off');
annotation(figure1, 'textbox', ...)
    [0.692618806875632 0.745161290322584 0.0293225480283115 ... 0.18443870967745], ...
'String', 'F', ...
'LineStyle', 'none', ...
'FitBoxToText', 'off');

%% Save figure
% print(gcf,['1patch_figs/1patch_Lyap',num2str(stages),...]
% parm_labels{p1},'stage.eps','-depsc','-r600')
end
end
close all

%% Plot lambda exp for select scenarios
for stages = 1:3
    for pl = 1:length(test_parms)-1
        % % Load save file
        lambda_file = ['one_patch_save_data/Lambda1patch',num2str(stages),...]
                      'stage',parm_labels{pl},',', parm_labels{3},'.csv'];
        lambda_data = csvread(lambda_file);
        
       inds_2 = test_parms(3);
        range_2 = linspace(min_parms(inds_2),max_parms(inds_2),num_tests);
        sp_ind = 0;
        figure1 = figure;
        for bifs = [20,50,80]
            sp_ind = sp_ind+1;
            subplot(1,3,sp_ind);
            plot(range_2,real(lambda_data(bifs,:)),'k-')
            hold on
            plot(range_2,abs(lambda_data(bifs,:)),'b-')
            plot([range_2(1) range_2(end)], [-1 -1],'k--')
            plot([range_2(1) range_2(end)], [1 1],'k--')
            if sp_ind ==1
                ylabel('\lambda')
            end
            xlabel(parm_labels_all{3})
            ylim([min(real(lambda_data(bifs,:))) max(abs(lambda_data(bifs,:)))]
        end
        legend('real(\lambda)', '|\lambda|')
set(gcf, 'Position',[86.6000 639.4000 791.2000 148.8000])
% Create textbox
annotation(figure1,'textbox',...[0.130434782608696 0.755913978494624 0.0293225480283115 0.184483870967745],...'
String','G',...
LineStyle','none',...
'FitBoxToText','off');
annotation(figure1,'textbox',...[0.411021233569262 0.753225806451615 0.0293225480283116 0.184483870967745],...'
String','H',...
LineStyle','none',...
'FitBoxToText','off');
annotation(figure1,'textbox',...[0.692618806875632 0.745161290322584 0.0293225480283115 0.184483870967745],...'
String','I',...
LineStyle','none',...
'FitBoxToText','off');

% % Save figure

print(gcf,['1patch_figs/1patch_lambda_',num2str(stages),...'
parm_labels{pl},'stage.eps'],['-depsc','-r600'])

end
end

close all

end
function [D_out] = make_distance_based_dispersal(tot_patches)

% Create distance based dispersal for two patch model
D_out = ones(tot_patches);
D_out = D_out - diag(diag(D_out));

end
% Create illustration of dispersal zones

figure
plot([0 0.2], [0.2,0], 'k-')
hold on
plot([0.2 1], [0.8,0], 'k-')
plot([0 0.8], [0.2,1], 'k-')
plot([.8 1], [1,0.8], 'k-')
plot([0 0.4], [1,0.6], 'k-')
plot([0.6 1], [0.4,0], 'k-')
set(gcf, 'Position', [651 577 480 401])
xlabel('\sigma_1')
ylabel('\sigma_2')

% Create textboxes
annotation(gcf, 'textbox',...[0.122842096927463 0.14629044988161 0.138798701298701 0.0656666666666669],...'String','Zone 1',...'LineStyle','none',...'FitBoxToText','off');
annotation(gcf, 'textbox',...[0.38055907507127 0.741883714811893 0.127201457079507 0.0656666666666671],...'String','Zone 5',...'LineStyle','none',...'FitBoxToText','off');
annotation(gcf, 'textbox',...[0.531335128286347 0.24464614575112 0.127201457079506 0.065666666666667],...'String','Zone 4',...'LineStyle','none',...'FitBoxToText','off');
annotation(gcf, 'textbox',...[0.724239784605638 0.415916863983164 0.127201457079506 0.065666666666667],...'String','Zone 5',...'LineStyle','none',...'FitBoxToText','off');
annotation(gcf, 'textbox',...[0.79075863161229 0.860668245198634 0.127201457079506 0.0656666666666671],...'String','Zone 3',...'LineStyle','none',...'FitBoxToText','off');
annotation(gcf,'textbox',... 
    [0.452276666727906 0.50581425940542 0.128654577130187 0.0656666666666669],...
    'String','Zone 2',...
    'LineStyle','none',...
    'FitBoxToText','off');
annotation(gcf,'textbox',... 
    [0.203175483053531 0.573375427519075 0.124984162179285 0.0656666666666671],...
    'String','Zone 4',...
    'LineStyle','none',...
    'FitBoxToText','off');
set(gca,'Fontsize',12)
print(gcf,'patch.figs/Zone_plane','-depsc','-r600')
function [N_sync, lyap_exp, lambda_0, k_cycle_period] = ...
    analyze_Parms_2Patch(sj1, sj2, IC, stage, tot_ponds, T, sigma_inds, D, all_Parms, V, M)

% % Loop through parameter spaces and calculate Lyapunov exponent for
% attractor type identification

IC_new = repmat(IC(1:stage), tot_ponds, 1);
k_cycle_period = -isones(length(sj1), length(sj1));

for p1 = 1:length(sj1)
    for p2 = 1:length(sj2)
        sigma_j_vector = [sj1(p1) sj2(p1)]';
        % % Change dispersal to specified life stage
        M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) + ...
            sigma_j_vector' .* D;
        N = IC_new;

        % % Numerically solve for fixed points
        if stage == 1
            fun = @(a_x) one_stage(all_parms, sigma_j_vector, a_x);
        elseif stage == 2
            fun = @(a_x) two_stage(all_parms, 1 - diag(M), a_x);
        else
            fun = @(a_x) three_stage(all_parms, 1 - diag(M), a_x);
        end
        IC = IC_new;
        [aa, ~, exitflag] = fsolve(fun, IC);

        if exitflag == 0 && all(aa > -0.001)
            J_diag = create_Jacobian(2, stage, all_parms, aa);
            A_J = V' * M * V * J_diag;
            % calculate eigenvalue
            E_0 = eig(A_J);
            [~, I_0] = max(abs(E_0));
            lambda_0(p1, p2) = E_0(I_0);
        else
            disp('Solution not found')
        end
    end
end

for tt = 1:T
    % num egg masses in previous time step
\[ a_t = N(\text{stage:stage:end}, tt); \]

\% \% density dependent larval survival
\[ B = \text{make_demographic_matrix(all_parms}, a_t, \text{stage}); \]

\% \% project population by one time step
\[ A = V' * M * V * B; \]
\[ N(:, tt+1) = A * N(:, tt); \]

end

\% final densities in each pond
\[ \text{end}_N = N(:, \text{end}-499: \text{end}); \]
\[ \text{max}_N = \text{max}(\text{end}_N'); \]
\[ \text{min}_N = \text{min}(\text{end}_N'); \]

\[ \text{all}_N(p1, p2, ::) = \text{end}_N; \]

\% If cycles occur, find lyapunov exp
\if any(\text{max}_N - \text{min}_N > .01)
    \[ [\text{lyap}_\text{exp}(p1, p2), k_{\text{cycle_period}}(p1, p2)] \]
    \[ = \text{calculate_Lyapunov_exp_2patch}(N, \text{stage}, \text{all_parms}, s_j1(p1), s_j2(p2), \text{sigma inds}); \]
    \[ N_{\text{sync}}(p1, p2) = \text{calculate_synchrony}(N(:, \text{end}-499: \text{end}), \text{stage}); \]
\else
    lyap\_exp(p1, p2) = -1;
    \[ N_{\text{sync}}(p1, p2) = 1; \]
\end{lyap\_exp(p1, p2) = -1;
\[ N_{\text{sync}}(p1, p2) = 1; \]
end
end
end

function \[ F \_\text{out} = \text{one\_stage}(\text{parms_in}, s_j\_\text{vec}, a_x) \]

\% Fixed point solutions for the two patch, 1-stage model
\% Parameters
\[ F = \text{parms_in}(1,1); \quad \% \text{Fecundity} \]
\[ S_n = \text{parms_in}(2,1); \quad \% \text{Mean survival from metamorph to age 1} \]
\[ S_j = \text{parms_in}(3,1); \quad \% \text{Juvenile survival} \]
\[ S_s = \text{parms_in}(4,1); \quad \% \text{Subadult survival} \]
\[ S_a = \text{parms_in}(5,1); \quad \% \text{Female adult survival} \]
\[ P_{js} = \text{parms_in}(6,1); \quad \% \text{Juvenile to subadult transition} \]
\[ P_{sf} = \text{parms_in}(7,1); \quad \% \text{Subadult to female adult transition} \]
Kx = parms_in(8,1); % carrying cap
gamma_x = parms_in(9,1); % density-dependent exponent

sj1 = sj_vec(1);
sj2 = sj_vec(2);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sa1 + F*Sn.*((1./(1+a_x(1).*F./Kx)).^gamma_x)*)((1-sj1)*a_x(1) + ...
            (Sa2 + F*Sn.*((1./(1+a_x(2).*F./Kx)).^gamma_x)))*(sj2*a_x(2) - a_x(1);
F_out(2) = (Sa1 + F*Sn.*((1./(1+a_x(1).*F./Kx)).^gamma_x)*)((1-sj2)*a_x(2) - a_x(2);
        end

function F_out = two_stage(parms_in, sj_vec, a_x)

% Fixed point solutions for the two patch, 2-stage model

% Parameters
F = params_in(1,1); % Fecundity
Sn = params_in(2,1); % Mean survival from metamorph to age 1
Sj = params_in(3,1); % juvenile survival
Ss = params_in(4,1); % subadult survival
Sa = params_in(5,1); % female adult survival
Pjs = params_in(6,1); % juvenile to subadult transition
Psf = params_in(7,1); % subadult to female adult transition
Kx = params_in(8,1); % carrying cap
gamma_x = params_in(9,1); % density-dependent exponent

sj_vec = sj_vec([1,3,2,4]);
Sa1 = params_in(5,1);
Sa2 = params_in(5,2);

F_out(1) = (Sj*(1-Psf)*a_x(1)*(1-sj_vec(1)) + F*Sn.*((1.+a_x(2).*F./Kx).^gamma_x)*a_x(2)*s_j(vec(1))) + ...
            (Sj*(1-Psf)*a_x(3)*sj_vec(3) + F*Sn.*((1.+a_x(4).*F./Kx).^gamma_x)*a_x(4)*s_j(vec(3))) - a_x(1);
F_out(2) = (Sj*Psf*a_x(1)*(1-sj_vec(2)) + Sa1*a_x(2))]*(1-sj_vec(2))+...
            (Sj*Psf*a_x(3)*sj_vec(4) + Sa2*a_x(4)*s_j(vec(4))) - a_x(2);
F_out(3) = (Sj*(1-Psf)*a_x(1)*sj_vec(1) + F*Sn.*((1.+a_x(2).*F./Kx).^gamma_x)*a_x(2)*s_j(vec(1)))+...
\begin{align*}
(S_j \cdot (1 - P_{fs}) \cdot a_{x}(3) \cdot (1 - s_{jvec}(3)) + F \cdot S_{n} \cdot (1 ./ (1 + a_{x}(4) \cdot F ./ K_{x}) \cdot \gamma_{x}) \cdot a_{x}(3)) - a_{x}(3) = 0;
F_{out}(1) = (S_j \cdot P_{js} \cdot a_{x}(1) \cdot s_{jvec}(1) + F \cdot S_{n} \cdot (1 ./ (1 + a_{x}(3) \cdot F ./ K_{x}) \cdot \gamma_{x}) \cdot a_{x}(3) - a_{x}(4) = 0;
F_{out}(4) = (S_j \cdot Psfs \cdot a_{x}(4) \cdot (1 - s_{jvec}(4)) \cdot (1 - s_{jvec}(4)) + Sa_{2} \cdot a_{x}(4) \cdot (1 - s_{jvec}(4)) - a_{x}(4) = 0;
end

function F_{out} = three_{stage}(parms_{in}, s_{jvec}, a_{x})
\end{align*}
end
function twopatch_end_state
%
% Runs simulations for the two-patch model, calculates unstable attractor
% type and synchrony, and saves files. NOTE: end STATE .mat files use a lot
% of memory
%
% clear all; close all
%

%%%% Analysis

% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent
mean parms_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
IC_def = [20 20 20]'; % initial conditions
num_tests = 100;
T = 10000;
sj1 = linspace(0,1,num_tests);
sj2 = linspace(0,1,num_tests);

% % Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

for stage = 1:3
    mean parms = mean parms_all(:,stage)';
% Create vec-permutation matrix
V = make_vec_matrix(tot_ponds, stage);
% Make diagonal block dispersal matrix (organized by life stage)
sigma_new = zeros(stage); % disp rate matrix
M = kron(eye(size(sigma_new)) - sigma_new, eye(size(D))) + kron(sigma_new, D);

for sigma_y = 1:stage
    sigma_inds = (sigma_y - 1)*tot_ponds + 1:sigma_y*tot_ponds;
    for s1 = 1:length(Sa_var)
        for s2 = 1:length(Sa_var)
            % simulate upper half of parameters
            if s2 >= s1
                % change adult survival and gamma to test parm
                all parms = repmat(mean_parms', 1, tot_ponds);
                all_parms(5,:) = [Sa_var(s1) Sa_var(s2)];

            end
            % get stable state with no dispersal
            IC_0 = repmat(IC_def(1:stage), tot_ponds, 1);
            N = IC_0;
            for tt = 1:T
                % Fixed dispersal
                sigma_j_vector = [0 0]';

                % % Change dispersal to specified life stage
                M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) +...  
                    sigma_j_vector'*D;

                % num egg masses in previous time step
                a_t = N(stage:stage:end, tt);
                % % density dependent larval survival
                B = make_demographic_matrix(all_parms, a_t, stage);

                % % project population by one time step
                A = V'*MsVsB;
                N(:, tt+1) = A*N(:, tt);
            end
        end
    end
end

IC = N(:, end);
[N_sync, lyap_exp, lambda_0, k_cycles] =...
analyze_parms_2patch(sj1,sj2,IC,stage,tot_ponds,T,sigma inds,D,all_parms,V,M);

%% % save Lyapunov exponent
% lyap_data = ['two_patch_save_data/Lyap',',num2str(stage)',',stage',',sigma',
num2str(sigma_y)',',_scn',',num2str(s1)',',-',',num2str(s2)',',.csv'];
% csvwrite(lyap_data,lyap_exp)
%% % save lambda
% lambda_data = ['two_patch_save_data/Lambda',',num2str(stage)',',stage',',sigma',
num2str(sigma_y)',',_scn',',num2str(s1)',',-',',num2str(s2)',',.csv'];
% csvwrite(lambda_data,lambda_0)
%% % save synchrony
% Nsync_data = ['two_patch_save_data/Sync',',num2str(stage)',',stage',',sigma',
num2str(sigma_y)',',_scn',',num2str(s1)',',-',',num2str(s2)',',.csv'];
% csvwrite(Nsync_data,N_sync)
%% % save k-cycles
% kcycle_data = ['two_patch_save_data/K_cycle',',num2str(stage)',',stage',',sigma',
num2str(sigma_y)',',_scn',',num2str(s1)',',-',',num2str(s2)',',.csv'];
% csvwrite(kcycle_data,k_cycles)
%% % save N
% filename = ['two_patch_save_data/',',num2str(stage)',',stage',',sigma',',num2str(
 sigma_y)',',_scn',',num2str(s1)',',-',',num2str(s2)',',.mat'];
% save(filename,'all_N')

end
end
end
end
end
function Analyze_two_patch_save_data

% Produce plots of attractor planes, synchrony planes, and histograms of saved
% end-behavior solutions and Lyapunov exponents
% Data is taken from files in 'two_patch_save_data'

% Initializations based on simulation initializations
sj1 = linspace(0,1,100);
sj2 = linspace(0,1,100);

[all_zones,zone inds] = define_zones;

%%%% Analysis

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

%%%% Plot chaos phase planes for all scenarios
stage_label = ['aaa';'jaa';'jsa'];
for stage = 1:3
    for Sa1 = 1:5
        for Sa2 = 1:5
            if Sa2 >= Sa1
                for sigma_y = 1:stage
                    figure
                    % % Load save file
                    lyap_file = ['two_patch_save_data/Lyap_','num2str(stage)','.','stage_','.','sigma_','.','num2str(Sa1)','.','scn_','.','num2str(Sa2)','.','csv'
                    ];
                    lyap_data = csvread(lyap_file);
                    lambda_file = ['two_patch_save_data/Lambda_','num2str(stage)','.','stage_','.','sigma_','.','num2str(Sa1)','.','scn_','.','num2str(Sa2)','.','csv'
                    ];
                    lambda_data = csvread(lambda_file);
                    k_cycle_file = ['two_patch_save_data/K_cycle_','num2str(stage)','.','stage_','.','sigma_','.','num2str(Sa1)','.','scn_','.','num2str(Sa2)','.','csv'
                    ];
                    k_cycle_data = csvread(k_cycle_file);

                    % % plot chaos planes

end
end
end
end
end
plot(1,1,'b.')
hold on
plot(1,1,'c','color', [.6 .6 .6])
plot(1,1,'c','color', [.4 .4 .4])
plot(1,1,'c','color', [.2 .2 .2])
plot(1,1,'c','color', [0 0 0])
plot(1,1,'c','color', [.9 .45 .13])
plot(1,1,'c','color', [.93 .69 .13])
plot(1,1,'r.')</p>

odd_cycles = plot_chaos_plane(lambda_data,sj1,sj2,sigma_y,stage,
   lyap_data,k_cycle_data);

xticks([0 0.25 0.5 0.75 1])
yticks([0 0.25 0.5 0.75 1])
set(gcf, 'Position', [500, 500, 313, 284])

% % create legends and axis labels for selected plots
if Sa1 == 1
   ylabel(['S_{a2} = ',num2str(Sa_var(Sa2))]; ['\sigma_{',stage,'label(' stage ,sigma_y'),'2}'])
   set(gcf, 'Position', [500, 500, 377, 290])
end
if Sa2 == 5
   xlabel(['\sigma_{',stage,'label(' stage ,sigma_y'),'1}']; ['S_{a1} = ',
         num2str(Sa_var(Sa1))])
if Sa1 == 1
   set(gcf, 'Position', [500, 500, 377, 319.2])
else
   set(gcf, 'Position', [500, 500, 331, 312])
end
end
dbcl(0,15)
if Sa1 == 5
legend('Invariant Cycles', '2 Cycles', '4 Cycles', '8 Cycles', ...
   '16 Cycles', '16+ Cycles', 'Other Cycles', 'Chaos', 'fontsize',10)
end

% % Save chaos plots
   print(gcf,['2pctcgs/blap_Scn','num2str(Sa1),-',num2str(Sa2),...
   '2patch',stage,'stg',sigma_y,':num2str(sigma_y),'.eps'],['-depsc2',
   ','-r600])
close all

end
end
end
end
end
end

%% Plot sync planes for all scenarios
stage_label = ['aaa';'jaa';'jsa'];
for stage = 1:3
for Sa1 = 1:5
for Sa2 = 1:5
    if Sa2 >= Sa1
        for sigma_y = 1:stage
            figure
            % % Load save file
            sync_file = ['two_patch_save_data/Sync',num2str(stage),',stage',',','sigma',',num2str(sigma_y),',',',',num2str(Sa1),',',',',num2str(Sa2),'.csv'];
            sync_data = csvread(sync_file);
            % % plot synchrony planes
            create_sync_planes(sync_data, sj1, sj2, stage, sigma_y)
            xticks([0 0.25 0.5 0.75 1])
            yticks([0 0.25 0.5 0.75 1])
            % % create legends and axis labels for selected plots
            set(gcf, 'Position', [500, 500, 313, 284])
            if Sa1 == 1
                ylabel(['\textit{S}_{a2} = ',num2str(Sa_var(Sa2));'\textit{\sigma} = ',num2str(sigma_y);',\textit{\sigma}_{jn} = ',num2str(Sa1);',\textit{\sigma}_{ja} = ',num2str(Sa2);']
            set(gcf, 'Position', [500, 500, 377, 290])
        end
        if Sa2 == 5
            xlabel(['\textit{\sigma}_{jn} = ',stage_label(sigma_y),',\textit{\sigma}_{ja} = ',num2str(Sa1);']
            if Sa1 == 1
                set(gcf, 'Position', [500, 500, 377, 319.2])
            else
            end
        end
    end
end
end
```matlab
set(gcf,'Position',[500,500,331,312])
end
end
set(gca,'fontsize',15)

%%% Save sync plots
print(gcf,['2patch_figs/Sync_Scn',num2str(Sa1),'-','num2str(Sa2) .... '
'_.',num2str(stage),'_stg_sigma_y_','num2str(sigma_y)','.eps'],'-depsc','-r600')

close all
end
end
end
end

%%% Plot chaos phase planes for dispersal–induced instability scenarios
stage_label = ['aaa';'jaa';'jsa'];
title_label = 'ABC';
inds = 0;
for stage = 1
    for Sa1 = 3:5
        for Sa2 = 3:5
            if Sa2 == Sa1
                for sigma_y = stage
                    inds = inds+1;
                    figure
                    % % Load save file
                    lyap_file = ['two_patch_save_data/Lyap_','num2str(stage)','_stage_','...
                    'sigma','num2str(sigma_y)',['.scn',num2str(Sa1),'-','num2str(Sa2)','.csv']
                    ];
                    lyap_data = csvread(lyap_file);
                    lambda_file = ['two_patch_save_data/Lambda_','num2str(stage)','_stage_','...
                    'sigma','num2str(sigma_y)',['.scn',num2str(Sa1),'-','num2str(Sa2)','.csv']
                    ];
                    lambda_data = csvread(lambda_file);

                    % % plot chaos planes
                    plot(-1,-1,'b')
                end
            end
        end
    end
end
```
hold on
plot([-1,-1,'.','color', [.6 .6 .6]])
plot([-1,-1,'.','color', [.4 .4 .4]])
plot([-1,-1,'.','color', [.2 .2 .2]])
plot([-1,-1,'.','color', [0 0 0]])
plot([-1,-1,'.','color', [.9 0.45 0.13]])
plot([-1,-1,'.','color', [.93 .69 .13]])
plot([-1,-1,'r.'])
odd_cycles = plot_chaos_plane(lambda_data,sj1, sj2, sigma_y, stage, lyap_data);
if Sa1 == 5
legend('Invariant Cycles', '2 Cycles', '4 Cycles', '8 Cycles', ...
     '16 Cycles', '16+ Cycles', 'Other Cycles', 'Chaos')
end
ylabel(['\sigma{', stage_label(stage,sigma_y), '}']);
xlabel(['\sigma{', stage_label(stage,sigma_y), '}','S_a = ',num2str( Sa_var(Sa1))])
set(gcf, 'Position', [500 578 212 212])
title(num2str(title_label(inds)))

% % Save chaos plots
% print(gcf,['2 patch_figs/Lyap2_Scn',num2str(Sa1),'-',num2str(Sa2),...
%      '_',num2str(stage),'_stg_sigm_y', num2str(sigma_y),'.eps']); -depsc
%
% close all
end
end
end
end

% % Create histograms for symmetric dispersal across all simulations
titles = 'ABC';
for stage = 1:3
    sigma1 = repmat(sj1,1,100);
    sigma2 = reshape(repmat(sj2,100,1),1,10000);

    % Initialize all data vars
    k_cycle_sync_total = [];
    chaos_sync_total = [];

    ...
inv_cycle_sync_total = []; k_cycle_N_total = []; chaos_N_total = []; inv_cycle_N_total = []; stable_N_total = []; k_cycle_disp_total = []; chaos_disp_total = []; inv_cycle_disp_total = []; stable_disp_total = []; s_to_u_sync_total = []; s_to_u_disp_total = []; s_to_u_N_total = []; k_cycle_sync_all = []; chaos_sync_all = []; inv_cycle_sync_all = []; k_cycle_sync_same_stage = []; chaos_sync_same_stage = []; inv_cycle_sync_same_stage = []; k_cycle_sync_diff_stage = []; chaos_sync_diff_stage = []; inv_cycle_sync_diff_stage = []; k_cycle_sync_same_patch = []; chaos_sync_same_patch = []; inv_cycle_sync_same_patch = [];

for sigma_y = 1:stage
    for Sa1 = 1:size(Sa_var,2)
        for Sa2 = 1:size(Sa_var,2)
            if Sa2 >= Sa1
                lyap_file = ['two_patch_save_data/Lyap', num2str(stage), 'stage', 'sigma', num2str(sigma_y), '_scn', num2str(Sa1), '=&', num2str(Sa2), '.csv'];
                lyap_data = csvread(lyap_file);
                lambda_file = ['two_patch_save_data/Lambda', num2str(stage), 'stage', 'sigma', num2str(sigma_y), '_scn', num2str(Sa1), '=&', num2str(Sa2), '.csv'];
                lambda_data = csvread(lambda_file);
[k_cycle_N_sync_total, k_cycle_N_sync_all, k_cycle_N_sync_stage, 
  k_cycle_N_sync_diff_stage, k_cycle_N_sync_same_patch, ...  
chaos_N_sync_total, chaos_N_sync_all, chaos_N_sync_stage, 
  chaos_N_sync_diff_stage, chaos_N_sync_same_patch, ...  
inv_cycle_N_sync_total, inv_cycle_N_sync_all, inv_cycle_N_sync_stage, 
  inv_cycle_N_sync_diff_stage, inv_cycle_N_sync_same_patch, 
s_to_u_sync, ...]

k_cycle_disp, chaos_disp, inv_cycle_disp, stable_disp, s_to_u_disp, 
k_cycle_N, chaos_N, inv_cycle_N, stable_N, s_to_u_N] = ...
all_results(lambda_data, sigma1, sigma2, sigma_y, stage, lyap_data);

% Accumulate all results
k_cycle_sync_total = [k_cycle_sync_total; k_cycle_N_sync_total];
chaos_sync_total = [chaos_sync_total; chaos_N_sync_total];
inv_cycle_sync_total = [inv_cycle_sync_total; inv_cycle_N_sync_total];
k_cycle_sync_all = [k_cycle_sync_all; k_cycle_N_sync_all];
chaos_sync_all = [chaos_sync_all; chaos_N_sync_all];
inv_cycle_sync_all = [inv_cycle_sync_all; inv_cycle_N_sync_all];
k_cycle_sync_same_stage = [k_cycle_sync_same_stage; 
k_cycle_N_sync_same_stage];
chaos_sync_same_stage = [chaos_sync_same_stage; chaos_N_sync_same_stage];
inv_cycle_sync_same_stage = [inv_cycle_sync_same_stage; 
  inv_cycle_N_sync_same_stage];
k_cycle_sync_diff_stage = [k_cycle_sync_diff_stage; 
k_cycle_N_sync_diff_stage];
chaos_sync_diff_stage = [chaos_sync_diff_stage; 
  chaos_N_sync_diff_stage];
inv_cycle_sync_diff_stage = [inv_cycle_sync_diff_stage; 
  inv_cycle_N_sync_diff_stage];
k_cycle_sync_same_patch = [k_cycle_sync_same_patch; 
k_cycle_N_sync_same_patch];
chaos_sync_same_patch = [chaos_sync_same_patch; 
  chaos_N_sync_same_patch];
inv_cycle_sync_same_patch = [inv_cycle_sync_same_patch; 
  inv_cycle_N_sync_same_patch];
s_to_u_sync_total = [s_to_u_sync_total; s_to_u_sync];

k_cycle_disp_total = [k_cycle_disp_total; k_cycle_disp];
chaos_disp_total = [chaos_disp_total; chaos_disp];
inv_cycle_disp_total = [inv_cycle_disp_total; inv_cycle_disp];
stable_disp_total = [stable_disp_total;stable_disp];
s_to_u_disp_total = [s_to_u_disp_total;s_to_u_disp];

k_cycle_N_total = [k_cycle_N_total;k_cycle_N];
chaos_N_total = [chaos_N_total;chaos_N];
inv_cycle_N_total = [inv_cycle_N_total;inv_cycle_N];
stable_N_total = [stable_N_total;stable_N];
s_to_u_N_total = [s_to_u_N_total;s_to_u_N];

end
end
end
end

all_sync = [k_cycle_sync_total;chaos_sync_total;inv_cycle_sync_total];
all_disp = [k_cycle_disp_total;chaos Disp_total;inv_cycle_disp_total];
all_N = [k_cycle_N_total;chaos Disp_total;inv_cycle_N_total];

%% Plot histogram of proportion dispersing individuals
figure
dispersal_prop_hist(k_cycle_N_total,chaos_N_total,inv_cycle_N_total,stable_N_total,...
    k_cycle Disp_total,chaos Disp_total,inv_cycle Disp_total,stable Disp_total,stage)
title(num2str(titles(stage)))
set(gcf, 'Position', [680 722 442 256])
set(gca, 'Fontsize',12)
legend({'Invariant cycles ','num2str(length(inv_cycle Disp total)),'}),...'
    ['K-cycles ',num2str(length(k_cycle Disp total)),'],['Chaos (',...
    num2str(length(chaos Disp total)),'],['Stable (',num2str(length(stable N total
    )),']),'Fontsize',8,'Location','Northwest')
% print(gcf,['2 patch_figs/prop_hists_',num2str(stage),'.stg_eps'],$depse','r600')

end

%% Plot synchrony histograms for each zone
for stage = 1:3
    figure
    % % cycle through each zone individually
    for zi = 1:length(zone_inds)-1
        sigma1 = sj1(all_zones(1,zone inds(zi)+1:zone inds(zi+1)));
sigma2 = sj2(all_zones(2,zone inds(zi)+1:zone inds(zi+1)));
k_cycle_sync_total = []; chaos_sync_total = []; inv_cycle_sync_total = []; k_cycle_N_total = []; chaos_N_total = []; inv_cycle_N_total = [];
stable_N_total = []; k_cycle_disp_total = []; chaos_disp_total = []; inv_cycle_disp_total = [];
stable_disp_total = []; s_to_u_sync_total = []; s_to_u_disp_total = []; s_to_u_N_total = []; k_cycle_sync_all = []; chaos_sync_all = []; inv_cycle_sync_all = [];
k_cycle_sync_same_stage = []; chaos_sync_same_stage = []; inv_cycle_sync_same_stage = []; k_cycle_sync_diff_stage = [];
chaos_sync_diff_stage = []; inv_cycle_sync_diff_stage = []; k_cycle_sync_same_patch = []; chaos_sync_same_patch = [];
inv_cycle_sync_same_patch = [];

for sigma_y = 1:stage
    for Sa1 = 1:size(Sa_var,2)
        for Sa2 = 1:size(Sa_var,2)
            if Sa2 >= Sa1
                % Load save file
                load_file = ['two_patch_save_data/','num2str(stage),'stage','sigma',
                             num2str(sigma_y),'_scn',num2str(Sa1),'-_','-',num2str(Sa2),'.mat'];
                load_N = load(load_file);
                N = load_N.all_N;
                lyap_file = ['two_patch_save_data/Lyap/','num2str(stage)',stage',
                             'sigma',num2str(sigma_y),'_scn',num2str(Sa1),'-_','-',num2str(Sa2),'.csv'];
            end
        end
    end
end
lyap_data = csvread(lyap_file);

[k_cycle_N_sync_total, k_cycle_N_sync_all, k_cycle_N_sync_stage,
   k_cycle_N_sync_diff_stage, k_cycle_N_sync_same_patch, ...]
chaos_N_sync_total, chaos_N_sync_all, chaos_N_sync_stage,
   chaos_N_sync_diff_stage, chaos_N_sync_same_patch, ...
inv_cycle_N_sync_total, inv_cycle_N_sync_all, inv_cycle_N_sync_stage,
   inv_cycle_N_sync_diff_stage, inv_cycle_N_sync_same_patch,
s_to_u_sync, ...
k_cycle_disp, chaos_disp, inv_cycle_disp, stable_disp, s_to_u Disp,
   k_cycle_N, chaos_N, inv_cycle_N, stable_N, s_to_u_N] = ...;
all_results(N, sigm1, sigm2, sigm_y, stage, lyap_data);

k_cycle_sync_total = [k_cycle_sync_total; k_cycle_N_sync_total];
chaos_sync_total = [chaos_sync_total; chaos_N_sync_total];
inv_cycle_sync_total = [inv_cycle_sync_total; inv_cycle_N_sync_total];
k_cycle_sync_all = [inv_cycle_sync_total; inv_cycle_N_sync_total];
chaos_sync_all = [chaos_sync_all; chaos_N_sync_all];
inv_cycle_sync_all = [inv_cycle_sync_all; inv_cycle_N_sync_all];
k_cycle_sync_same_stage = [k_cycle_sync_same_stage;
   k_cycle_N_sync_same_stage];
chaos_sync_same_stage = [chaos_sync_same_stage; chaos_N_sync_same_stage];
inv_cycle_sync_same_stage = [inv_cycle_sync_same_stage;
   inv_cycle_N_sync_same_stage];
k_cycle_sync_diff_stage = [k_cycle_sync_diff_stage;
   k_cycle_N_sync_diff_stage];
chaos_sync_diff_stage = [chaos_sync_diff_stage;
   chaos_N_sync_diff_stage];
inv_cycle_sync_diff_stage = [inv_cycle_sync_diff_stage;
   inv_cycle_N_sync_diff_stage];
k_cycle_sync_same_patch = [k_cycle_sync_same_patch;
   k_cycle_N_sync_same_patch];
chaos_sync_same_patch = [chaos_sync_same_patch;
   chaos_N_sync_same_patch];
inv_cycle_sync_same_patch = [inv_cycle_sync_same_patch;
   inv_cycle_N_sync_same_patch];
s_to_u_sync_total = [s_to_u_sync_total; s_to_u_sync];

k_cycleDisp_total = [k_cycleDisp_total; k_cycleDisp];
chaosDisp_total = [chaosDisp_total; chaosDisp];
inv_cycle_disp_total = [inv_cycle_disp_total; inv_cycle_disp];
stable_disp_total = [stable Disp_total; stable Disp];
s_to_u_disp_total = [s_to_u Disp_total; s_to_u Disp];

k_cycle_N_total = [k_cycle_N_total; k_cycle_N];
chaos_N_total = [chaos_N_total; chaos_N];
inv_cycle_N_total = [inv_cycle_N_total; inv_cycle_N];
stable_N_total = [stable N_total; stable N];
s_to_u_N_total = [s_to_u N_total; s_to_u N];

e nd

d end

d end

d end

all_sync = [k_cycle sync_total; chaos sync_total; inv_cycle sync_total];
all Disp = [k_cycle Disp_total; chaos Disp_total; inv_cycle Disp_total];
all_N = [k_cycle N_total; chaos N_total; inv_cycle N_total];

% % Plot sync histograms for all synchrony combinations
subplot (5,1,zi)
total sync_hists (k_cycle sync_total, chaos sync_total, inv_cycle sync_total, stage)
 if stage == 1
 y label ('Relative frequency')
 end
 if zi == 5
 x label ({'\beta \textbf{; num2str (stage), '} - stage'})
 end
t  itle ([{'Zone ', num2str (zi)}])
 set (gca, 'Fontsize', 12)
y label ('Relative frequency', 'Fontsize', 10)
legend ({{'Invariant cycles (' num2str (length (inv_cycle sync_total)), ')'}},...
 {'K-cycles (' num2str (length (k_cycle sync_total)), ') '},
 {'Chaos (' num2str (length (chaos sync_total)), ') '}}, 'Fontsize', 8, 'Location', 'Northwest')

% % summary statistics
 total_results = (zone inds (zi + 1) - zone inds (zi)) * 15 * stage;
 num unstable (stage, zi) = length (all sync);
 num unstable types (stage, zi, :) = [length (inv_cycle sync_total), length...
 k_cycle sync_total],
 length (chaos sync_total);
percent_unstable(stage, zi) = length(all_sync)/total_results;
percent_unstable_types(stage, zi,:) = [length(inv_cycle_sync_total), length(k_cycle_sync_total),...
    length(chaos_sync_total)]/total_results;
mean_sync(stage, zi) = mean(all_sync);
sd_sync(stage, zi) = std(all_sync,1);
end

set(gcf, 'Position', [454 48 365 9 4 5]);
% print(gcf,['2patch_figs/sync_freq_zones',num2str(stage),'.stg.eps','.--depsc','.--r600','])
end

%% Create bar plot
figure
ba = bar([squeeze(percent_unstable_types(1,:,:));zeros(1,3);squeeze(
    percent_unstable_types(2,:,:))...;
    zeros(1,3);squeeze(percent_unstable_types(3,:,:))],'stacked', 'FaceColor','flat');
ba(1).CData = [0 0 1];
ba(2).CData = [0 0 0];
ba(3).CData = [1 0 0];
xticks(0:17)
xticklabels({'','1','2','3','4','5','6','1','2','3','4','5','6','1','2','3','4','5'})
xlabel({'Zones Zones Zones';...
    '1=stage 2=stage 3=stage'})
ylabel('Fraction Unstable')
title('D')
set(gca, 'Position', [680 722 442 256])
set(gca, 'FontSize',12)
legend({'Invariant cycles','K-cycles','Chaos'},'FontSize',8)
print(gcf,'2patch_figs/unstable_zones_hist.eps','.--depsc','.--r600')
end

%% Corollary functions
function [all_zones,zone_inds] = define_zones
% % Define indices within each zone
zl_inds = [];

z2_inds = []; 
z3_inds = []; 
z4_inds = []; 
z5_inds = []; 
for pp = 20:-1:1
    z1_inds = [z1_inds, [1:pp:(1-pp)*ones(1,pp)]]; 
z3_inds = [z3_inds, [101-pp:100:(pp+80)*ones(1,pp)]]; 
end 
for pp = 1:20
    z2_inds = [z2_inds, [pp+1:79+pp:22-pp:100-pp]]; 
end 
z2_inds = [z2_inds, [21:100;1:80]]; 
for pp = 1:40
    z4_inds = [z4_inds, [21+pp:100-pp:pp*ones(1,80-2*pp)]]; 
z4_inds = [z4_inds, [pp*ones(1,81-2*pp);21+pp:101-pp]]; 
z5_inds = [z5_inds, [pp+1:80-pp:(101-pp)*ones(1,80-2*pp)]]; 
z5_inds = [z5_inds, [(101-pp)*ones(1,81-2*pp);pp:80-pp]]; 
end 
all_zones = [z1_inds,z2_inds,z3_inds,z4 inds,z5 inds]; 
zone_inds = [0,length(z1_inds),length(z1 inds)+length(z2_inds),length(z1 inds)+length(z2 inds)+length(z3 inds),... 
             length(z1 inds)+length(z2 inds)+length(z3 inds)+length(z4 inds),... 
             length(z1 inds)+length(z2 inds)+length(z3 inds)+length(z4 inds)+length(z5 inds)]; 
end 

function unique_cycles = plot_chaos_plane(lambda_in, sj1, sj2, sigma_y, num_stages, 
                              lyap_data, k_cycle_data) 

    stage_label = ['aaa';'jaa';'jsa']; 
k_cycle_thresh = -l*ones(length(sj1), length(sj2)); 
chaos_thresh = -l*ones(length(sj1), length(sj2)); 
inv_cycle_thresh = -l*ones(length(sj1), length(sj2)); 
two_cycle = -l*ones(length(sj1), length(sj2)); 
four_cycle = -l*ones(length(sj1), length(sj2)); 
eight_cycle = -l*ones(length(sj1), length(sj2)); 
sixteen_cycle = -l*ones(length(sj1), length(sj2)); 
odd_cycle = -l*ones(length(sj1), length(sj2)); 
sixteen_plus_cycle = -l*ones(length(sj1), length(sj2));
unique_cycles = -1*ones(length(sj1), length(sj2));

for p1 = 1:length(sj1)
    for p2 = 1:length(sj1)
        % If cycles occur, determine chaos
        if abs(lambda_in(p1,p2))>1
            if lyap_data(p1,p2)<-0.001
                k_cycle_thresh(p1,p2)=1;
            
            if k_cycle_data(p1,p2) ==2
                two_cycle(p1,p2) = 1;
            elseif k_cycle_data(p1,p2) ==4
                four_cycle(p1,p2) = 1;
            elseif k_cycle_data(p1,p2) ==8
                eight_cycle(p1,p2) = 1;
            elseif k_cycle_data(p1,p2) ==16
                sixteen_cycle(p1,p2) = 1;
            elseif k_cycle_data(p1,p2) >16
                odd_cycle(p1,p2) = 1;
            else
                if k_cycle_data(p1,p2) > 0
                    sixteen_plus_cycle(p1,p2) = 1;
                end
            end
        end
    end
end

if lyap_data(p1,p2)>0.001
    chaos_thresh(p1,p2)=1;
else
    inv_cycle_thresh(p1,p2)=1;
end

end

sj1(1) = 0.001;
sj2(1) = 0.001;
plot(sj1, sj2.*inv_cycle_thresh, 'b. ')
hold on
plot(sj1, sj2.*two_cycle, '.', 'color', [.6 .6 .6])
plot(sj1, sj2.*four_cycle, '.', 'color', [.4 .4 .4])
plot(sj1, sj2.*eight_cycle, '.', 'color', [.2 .2 .2])
plot(sj1, sj2.*sixteen_cycle,','color',[0 0 0])
plot(sj1, sj2.*sixteen_plus_cycle,','color',[.9 .45 .13])
plot(sj1, sj2.*odd_cycle,','color',[.93 .69 .13])
plot(sj1, sj2.*chaos_thresh,'r')
xlim([sj1(1) sj1(end)])
ylim([sj2(1) sj2(end)])
xlabel(['\sigma{',stage_label(num_stages,sigma_y),'1}'])
ylabel(['\sigma{',stage_label(num_stages,sigma_y),'2}'])
end

function create_sync_planes(sync_data, sj1, sj2, num_stages, sigma_y)
% Plot synchrony planes for dispersal ranges

stage_label = ['aaa';'jaa';'jsa'];
N_sync = sync_data;
% % Plot synchrony
N_sync(N_sync>0.9999999999) = 1;
N_sync(1,1) = 0; % force initial point to zero to get consistent contour lines
contourf(sj1, sj2,N_sync, linspace(0,1,21),'ShowText','on')
colormap(spring)
xlim([sj1(1) sj1(end)])
ylim([sj2(1) sj2(end)])
xlabel(['\sigma{',stage_label(num_stages,sigma_y),'1}'])
ylabel(['\sigma{',stage_label(num_stages,sigma_y),'2}'])
end

function [k_cycle_N_sync_total, k_cycle_N_sync_all, k_cycle_N_sync_stage,
  k_cycle_N_sync_diff_stage, k_cycle_N_sync_same_patch,...
  chaos_N_sync_total, chaos_N_sync_all, chaos_N_sync_stage, chaos_N_sync_diff_stage,
  chaos_N_sync_same_patch,...
  inv_cycle_N_sync_total, inv_cycle_N_sync_all, inv_cycle_N_sync_stage,
  inv_cycle_N_sync_diff_stage, inv_cycle_N_sync_same_patch, s_to_u_N_sync_total,
  ...]
  k_cycle_disp, chaos_disp, inv_cycle_disp, stable_disp, s_to_u_disp, k_cycle_N, chaos_N,
  inv_cycle_N, stable_N, s_to_u_N] = ...
  all_results(N_in,sj1, sj2, sigma_y, num_stages, lyap_data)
% Collect results for synchrony and attractor histogram and bar plots

% % Initializations
if num\_stages == 2
    diff\_stage\_ind = [1 4; 2 3];
    same\_patch\_ind = [1 2; 3 4];
else if num\_stages == 3
    diff\_stage\_ind = [1 5; 1 6; 2 4; 2 6; 3 4; 3 5];
    same\_patch\_ind = [1 2; 1 3; 2 3; 4 5; 4 6; 5 6];
end

k\_cycle\_N\_sync\_total = [];
k\_cycle\_N\_sync\_all = [];
k\_cycle\_N\_sync\_stage = [];
chaos\_N\_sync\_total = [];
chaos\_N\_sync\_all = [];
chaos\_N\_sync\_stage = [];
inv\_cycle\_N\_sync\_total = [];
inv\_cycle\_N\_sync\_all = [];
inv\_cycle\_N\_sync\_stage = [];
s\_to\_u\_N\_sync\_total = [];

end\_N0 = squeeze(N\_in(1,1,:,:));
max\_N0 = max(end\_N0,[],2);
min\_N0 = min(end\_N0,[],2);

stable\_thresh = -l\_ones(length(sj1), length(sj2));
k\_cycle\_thresh = -l\_ones(length(sj1), length(sj2));
chaos\_thresh = -l\_ones(length(sj1), length(sj2));
inv\_cycle\_thresh = -l\_ones(length(sj1), length(sj2));
stable\_to\_unstable\_thresh = -l\_ones(length(sj1), length(sj2));

for s1 = 1:length(sj1)
    p1 = round(sj1(s1)*100+1);
    p1(p1>50)=p1-1; % make up for rounding error
    p2 = round(sj2(s1)*100+1);
    p2(p2>50)=p2-1; % make up for rounding error
    end\_N = squeeze(N\_in(p1,p2,:,:));
    max\_N = max(end\_N,[],2);
    min\_N = min(end\_N,[],2);

    % calculate mean number of dispersing individuals
    N\_dispersing\_1(s1) = mean(sj1(s1)*end\_N(sigma\_y,:));
    N\_dispersing\_2(s1) = mean(sj2(s1)*end\_N(sigma\_y+num\_stages,:));
% calculate mean number of individuals in each patch
mean_1(s1) = mean(sum(end_N(1:num_stages,:,:),1));
mean_2(s1) = mean(sum(end_N(num_stages+1:end,:,:),1));

% If cycles occur, determine attractor type and calculate synchrony
if any(max_N - min_N > .01)
    if lyap_data(p1,p2)<-0.001
        k_cycle_thres(s1)=1;
    end
    % Calculate synchrony for total patch populations,
    % same-stage populations, and between-stage populations
    cov_total = cov([sum(end_N(1:num_stages,:,:),1);sum(end_N(num_stages+1:end,:,:),1)]');
k_cycle_N_sync_total = [k_cycle_N_sync_total;sum(cov_total(:))/(sum(diag((cov_total).^5)))^2];

cov_all = cov(end_N');
k_cycle_N_sync_all = [k_cycle_N_sync_all;sum(cov_all(:))/(sum(diag((cov_all).^5)))^2];

    for ss = 1:num_stages
        cov_same_stage = cov([end_N(ss,:);end_N(num_stages+ss,:)]');
k_cycle_N_sync_stage = [k_cycle_N_sync_stage;...
            sum(cov_same_stage(:))/(sum(diag((cov_same_stage).^5)))^2];
    end

    if num_stages >1
        for yy = 1:size(diff_stage_ind,1)
            cov_diff_stage = cov([end_N(diff_stage_ind(yy,:),:);end_N(diff_stage_ind(yy,2,:),:)']);
k_cycle_N_sync_diff_stage = [k_cycle_N_sync_diff_stage;...
                sum(cov_diff_stage(:))/(sum(diag((cov_diff_stage).^5)))^2];
        end

        for xx = 1:size(same_patch_ind,1)
            cov_same_patch = cov([end_N(same_patch_ind(xx,:),:);end_N(same_patch_ind(xx,2,:),:)']);
k_cycle_N_sync_same_patch = [k_cycle_N_sync_same_patch;...
                sum(cov_same_patch(:))/(sum(diag((cov_same_patch).^5)))^2];
        end
    end
end
else if lyap_data(p1,p2)>0.001
    chaos_thresh(s1)=1;

    % Calculate synchrony for total patch populations,
    % same-stage populations, and between-stage populations
    cov_total = cov([sum(end_N(1:num_stages,:)';1);sum(end_N(num_stages+1:end,:)',1)]');
    chaos_N_sync_total = [chaos_N_sync_total;sum(cov_total(:))/(sum(diag(cov_total).^5)))^2];
    cov_all = cov(end_N');
    chaos_N_sync_all = [chaos_N_sync_all;sum(cov_all(:))/(sum(diag(cov_all).^5)))^2];

    for ss = 1:num_stages
        cov_same_stage = cov([end_N(ss,:);end_N(num_stages+ss,:)]');
        chaos_N_sync_stage = [chaos_N_sync_stage;...
            sum(cov_same_stage(:))/(sum(diag(cov_same_stage).^5)))^2];
    end
    if num_stages>1
        for yy = 1:size(diff_stage_ind,1)
            cov_diff_stage = cov([end_N(diff_stage_ind(yy,1,:);end_N(diff_stage_ind(yy,2,:))']');
            chaos_N_sync_diff_stage = [chaos_N_sync_diff_stage;...
                sum(cov_diff_stage(:))/(sum(diag(cov_diff_stage).^5)))^2];
        end
        for xx = 1:size(same_patch_ind,1)
            cov_same_patch = cov([end_N(same_patch_ind(xx,1,:);end_N(same_patch_ind(xx,2,:))']');
            chaos_N_sync_same_patch = [chaos_N_sync_same_patch;...
                sum(cov_same_patch(:))/(sum(diag(cov_same_patch).^5)))^2];
        end
    end
end

else
    inv_cycle_thresh(s1)=1;

    % Calculate synchrony for total patch populations,
    % same-stage populations, and between-stage populations
\[
\text{cov\_total} = \text{cov}([\text{sum(end\_N(1:num\_stages,:)},1);\text{sum(end\_N(num\_stages+1:end,:)},1)])']; \\
\text{inv\_cycle\_N\_sync\_total} = [\text{inv\_cycle\_N\_sync\_total};\text{sum(cov\_total(:))}/(\text{sum(diag((cov\_total).^0.5))}).^2]; \\
\text{cov\_all} = \text{cov}(\text{end\_N}'); \\
\text{inv\_cycle\_N\_sync\_all} = [\text{inv\_cycle\_N\_sync\_all};\text{sum(cov\_all(:))}/(\text{sum(diag((cov\_all).^0.5))}).^2]; \\
\text{for ss} = 1:\text{num\_stages} \\
\text{cov\_same\_stage} = \text{cov}([\text{end\_N(ss,:)};\text{end\_N(num\_stages+ss,:)}']); \\
\text{inv\_cycle\_N\_sync\_stage} = [\text{inv\_cycle\_N\_sync\_stage};... \text{sum(cov\_same\_stage(:))}/(\text{sum(diag((cov\_same\_stage).^0.5))}).^2]; \\
\text{end} \\
\text{if num\_stages} > 1 \\
\text{for yy} = 1:\text{size(\text{diff\_stage\_ind},1)} \\
\text{cov\_diff\_stage} = \text{cov}([\text{end\_N(diff\_stage\_ind(yy,1,:)};\text{end\_N(diff\_stage\_ind(yy,2,:),:)}']); \\
\text{inv\_cycle\_N\_sync\_diff\_stage} = [\text{inv\_cycle\_N\_sync\_diff\_stage};... \text{sum(cov\_diff\_stage(:))}/(\text{sum(diag((cov\_diff\_stage).^0.5))}).^2]; \\
\text{end} \\
\text{for xx} = 1:\text{size(\text{same\_patch\_ind},1)} \\
\text{cov\_same\_patch} = \text{cov}([\text{end\_N(same\_patch\_ind(xx,1,:)};\text{end\_N(same\_patch\_ind(xx,2,:),:)}']); \\
\text{inv\_cycle\_N\_sync\_same\_patch} = [\text{inv\_cycle\_N\_sync\_same\_patch};... \text{sum(cov\_same\_patch(:))}/(\text{sum(diag((cov\_same\_patch).^0.5))}).^2]; \\
\text{end} \\
\text{end} \\
\text{end} \\
\text{if all(max\_N0 - min\_N0 < 0.01) } \\
\text{stable\_to\_unstable\_thresh(s1) = 1;} \\
\text{cov\_total} = \text{cov}([\text{sum(end\_N(1:num\_stages,:)},1);\text{sum(end\_N(num\_stages+1:end,:)},1)])']; \\
\text{s\_to\_u\_N\_sync\_total} = [\text{s\_to\_u\_N\_sync\_total};\text{sum(cov\_total(:))}/(\text{sum(diag((cov\_total).^0.5))}).^2]; \\
\text{end}
if all(max_N0 - min_N0 < 0.01)
    stable_to_unstable_thresh(s1) = 1;
else
    stable_thresh(s1) = 1;
end

k_cycle_inds = find(k_cycle_thresh == 1);
chaos_inds = find(chaos_thresh == 1);
inv_cycle_inds = find(inv_cycle_thresh == 1);
stable_inds = find(stablethresh == 1);
stable_to_unstable_inds = find(stable_to_unstable_thresh == 1);

k_cycle_disp = [N_dispersing_1(k_cycle_inds)', N_dispersing_2(k_cycle_inds)'];
chaos_disp = [N_dispersing_1(chaos_inds)', N_dispersing_2(chaos_inds)'];
inv_cycle_disp = [N_dispersing_1(inv_cycle_inds)', N_dispersing_2(inv_cycle_inds)'];
stable_disp = [N_dispersing_1(stable_inds)', N_dispersing_2(stable_inds)'];
s_to_u_disp = [N_dispersing_1(stable_to_unstable_inds)', N_dispersing_2(stable_to_unstable_inds)'];

k_cycle_N = [mean_1(k_cycle_inds)', mean_2(k_cycle_inds)'];
chaos_N = [mean_1(chaos_inds)', mean_2(chaos_inds)'];
inv_cycle_N = [mean_1(inv_cycle_inds)', mean_2(inv_cycle_inds)'];
stable_N = [mean_1(stable_inds)', mean_2(stable_inds)'];
s_to_u_N = [mean_1(stable_to_unstable_inds)', mean_2(stable_to_unstable inds)'];

end

function dispersal_prop_hist(k_cycle_N_total, chaos_N_total, inv_cycle_N_total, stable_N_total, ...
    k_cycle_disp_total, chaos_disp_total, inv_cycle_disp_total, stable_disp_total, stage)

    edges = linspace(0,1,50);
k_cycle_proportion = k_cycle_disp_total./k_cycle_N_total;
chaos_proportion = chaos_disp_total./chaos_N_total;
in v_cycle.proportion = inv_cycle_disp_total./inv_cycle_N_total;
stable_proportion = stable_disp_total./stable_N_total;
histogram(inv_cycle_proportion(:,1)./(inv_cycle_proportion(:,1)+inv_cycle_proportion(:,2)), edges, 'FaceColor','b', 'Normalization','probability')
hold on
histogram(k_cycle_proportion(:,1)./(k_cycle_proportion(:,1)+k_cycle_proportion(:,2)),
    edges, 'FaceColor','k', 'Normalization','probability')
histogram(chaos_proportion(:,1)./(chaos_proportion(:,1)+chaos_proportion(:,2)), edges,
    'FaceColor','r', 'Normalization','probability')
histogram(stable_proportion(:,1)./(stable_proportion(:,1)+stable_proportion(:,2)),
    edges, 'FaceColor',[0.4940 0.1840 0.5560], 'Normalization','probability')
xlabel({'
\alpha'; [num2str(stage),'=' stage]})
ylabel('Relative frequency')
if stage == 3
    ylim([0 0.2])
end
end

function total_sync_hists(k_cycle_sync_total, chaos_sync_total, inv_cycle_sync_total, 
    stage)

    % Create histograms of all results

    edges = linspace(0,1,50);
    histogram(inv_cycle_sync_total, edges, 'FaceColor','b', 'Normalization','probability')
    hold on
    histogram(k_cycle_sync_total, edges, 'FaceColor','k', 'Normalization','probability')
    histogram(chaos_sync_total, edges, 'FaceColor','r', 'Normalization','probability')
    xlabel('\beta')
ylabel('Relative frequency')
end
function chaos_fractal BASINS

%% runs simulations for scenarios sensitive to initial conditions. Plots
%% and saves stability type for a range of initial conditions.

%% clear all; close all

%% Analysis

% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent
mean_prams_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
T = 10000;
num_tests = 100;
IC = [20 20 20]'; % initial conditions

% Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

%% Specify scenario
stage = 1;
all_prams = repmat(mean_prams_all(:,1),1,tot_ponds);
all_prams(5,1) = Sa_var(1);
all_prams(5,2) = Sa_var(2);
s1 = 0.87;
s2_all = [0.9 0.96];
sigma_y = 1;

% Initializations
% % Create vec_permutation matrix
V = make_vec_matrix(tot_ponds, stage);

% Make diagonal block dispersal matrix (organized by life stage)
sigma_new = zeros(stage); % disp rate matrix
M = kron(eye(size(sigma_new))==sigma_new,eye(size(D))) + kron(sigma_new,D);

% Indices that correspond to life stage in sigma
sigma_inds = (sigma_y==1)*tot_ponds+1:sigma_y*tot_ponds;

% Plot fractal basin for selected scenario
for s2 = s2_all
    figure
    k_cycle_thresh = -l*ones(num_tests, num_tests);
    chaos_thresh = -l*ones(num_tests, num_tests);
    inv_cycle_thresh = -l*ones(num_tests, num_tests);
    stage_label = ['aaa';'jaa';'jsa'];

    % Loop through different initial conditions
    IC_new = repmat(IC(1:stage),tot_ponds,1);
    % Fixed dispersal
    sigma_j_vector = [s1 s2]';

    % Change dispersal to specified life stage
    M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) +...
    sigma_j_vector'.*D;
    for IC1 = 1:num_tests
        for IC2 = 1:num_tests
            N = IC_new;
            N(sigma_y) = IC1;
            N(sigma_y+stage) = IC2;
            for tt = 1:T
                % num egg masses in previous time step
                a_t = N(stage:stage:end,tt);
% density dependent larval survival
B = make_demographic_matrix(all_parms, a_t, stage);

% project population by one time step
A = V'*M*V*B;
N(:, t+1) = A*N(:, t);
end

% final densities in each pond
end_N = N(:, end-99:end);
max_N = max(end_N');
min_N = min(end_N');

% Determine cycles with simulation ranges
if any(max_N - min_N > .01)
    [k_cycle_thresh(IC1, IC2), chaos_thresh(IC1, IC2), inv_cycle_thresh(IC1, IC2)]...
    = calculate_Lyapunov_exp2patch(N, stage, all_parms, s1, s2, sigma_inds);
end
end
end

plot(1:num_tests, (1:num_tests).*k_cycle_thresh,'k')
hold on
plot(1:num_tests, (1:num_tests).*chaos_thresh,'r')
plot(1:num_tests, (1:num_tests).*inv_cycle_thresh,'b')
xlim([1 num_tests])
ylim([1 num_tests])
xlabel([stage_label(stage,sigma_y),'_1(0)'])
ylabel([stage_label(stage,sigma_y),'_2(0)'])
set(gcf, 'Position',[680 742 280 236]);
if s2 == s2_all(1)
title('B')
else
    title('A')
end
function two_patch_bifurcations_main
% Bifurcations with multiple initial conditions
% Plots 1) bifurcation plots of fixed points with end state simulations and
% eigenvalues of fixed points

%% Analysis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent
meanParms_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
num_tests = 100;
T = 10000;
sj1 = linspace(0,1,num_tests);

% % Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

% Define all scenarios
stage_all = [1 1 1 2 2 3];
Sa1_all = [1 1 2 3 3 5];
Sa2_all = [2 2 2 3 5 5];
sigma_y_all = [1 1 1 2 1 2];
sigma_ind_all = [60 96 -1 -1 80 90];
stage_labels = ['aaa';'jaa';'jaa'];
bif_titles = 'ABCABC';
eig_titles = 'DEFDEF';

for scenario = 1:length(stage_all)
    figure
    if scenario == 3 || scenario == 6
        plot([-1,-1,'k.'])
        hold on
        plot([-1,-1,'ro','markersize',3])
    end
    stage = stage_all(scenario);
    Sa1 = Sa1_all(scenario);
    Sa2 = Sa2_all(scenario);
    sigma_y = sigma_y_all(scenario);
    sj_ind = sigma_ind_all(scenario);
    mean_parms = mean_parms_all(:,stage);
    % Create vec-permutation matrix
    V = make_vec_matrix(tot_ponds, stage);
    sigma_inds = (sigma_y-1)*tot_ponds+1:sigma_y*tot_ponds;
    if sj_ind < 0
        sj2=sj1;
    else
        sj2=sj1(sj_ind)*ones(1,length(sj1));
    end
    % Make diagonal block dispersal matrix (organized by life stage)
    sigma_new = zeros(stage); % disp rate matrix
    M = kron(eye(size(sigma_new))-sigma_new,eye(size(D))) + kron(sigma_new,D);

    % change adult survival and gamma to test parm
    all parms = repmat(mean_parms',1,tot_ponds);
    all_parms(5,:) = [Sa_var(Sa1) Sa_var(Sa2)];

    for pl = 1:length(sj1)
        % Fixed dispersal
        sigma_j_vector = [sj1(pl) sj2(pl)]';
        % Change dispersal to specified life stage
        M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) + ...
                                   sigma_j_vector'*D;
if stage == 1
    fun = @(a,x) one_stage(all_parms, sigma_j_vector, a, x);
elseif stage == 2
    fun = @(a,x) two_stage(all_parms, diag(M), a, x);
else
    fun = @(a,x) three_stage(all_parms, diag(M), a, x);
end

% Find trivial state eigenvalue
a_all = zeros(2*stage, 1);
J_diag = create_Jacobian(2, stage, all_parms, a_all);
A_J = V'*M*V'*J_diag;

% calculate eigenvalue
E_0 = eig(A_J);
[~, I_0] = max(abs(E_0));
lambda_keep = E_0(I_0);

for ee = 1:200
    IC = randsample(100, tot_ponds*stage, 1);

    % % Uncomment if need to find more basins of attraction
    %
    % N = IC;
    %
    % for tt = 1:T
    %
    %     % num egg masses in previous time step
    %     a_t = N(stage:stage:end, tt);
    %
    %     % density dependent larval survival
    %     B = make_demographic_matrix(all_parms, a_t, stage);
    %
    %
    %     % project population by one time step
    %     A = V'*M*V*B;
    %     N(:, tt+1) = A*N(:, tt);
    %
    % end
    %
    % final densities in each pond
    end_N = N(:, end-49:end);
    % plot(sj1(p1),end_N(stage,:), 'k. ')
    hold on

    % Solve for roots
    [aa,~, exitflag] = fsolve(fun, IC);
    if exitflag ~= 0 && all(aa>0.001)
        J_diag = create_Jacobian(2, stage, all_parms, aa);
        A_J = V'*M*V'*J_diag;
% calculate eigenvalue
E_0 = eig(A_J);
[~, I_0] = max(abs(E_0));
lambda_0 = E_0(I_0);

% Find unique fixed points
if any(abs(a_all-aa)>0.0001)
    lambda_keep = [lambda_keep, lambda_0];
    a_all = [a_all, aa];
end

% perturb IC to avoid simulating unstable equibs
N = abs(aa)+0.0001;
for tt = 1:T
    % num egg masses in previous time step
    a_t = N(stage:stage:end,tt);
    % density dependent larval survival
    B = make_demographic_matrix(all_parms, a_t, stage);

    % project population by one time step
    A = V'*M*V*B;
    N(:,tt+1) = A*N(:,tt);
end

% final densities in each pond
end_N = N(:,end-49:end);
plot(sj1(pl),end_N(stage,:), 'k.' )
hold on
end

plot(sj1(pl),a_all(stage,:), 'ro', 'markersize',3)
xlabel(['\sigma{',sigma_labels(stage, sigma_y),'}'])
ylabel('End-behavior')

set(gcf, 'Position', [680 639 447 239]);
set(gca, 'Fontsize',16)
end

if scenario ==3 || scenario == 6
legend('Simulation','Fixed Point', 'fontsize',12, 'location', 'northwest')
end

title(bif_titles(scenario))
xlim([0 1])
% save bifurcation results
% print(gcf,[''2patch_figs/twopatch_bifurcation_''num2str(stage),'-'stg_Sa,'...''
% num2str(Sal),num2str(Sa2),'_sigma','num2str(sigma_y),'_s_j','num2str(s_j_ind)',
% multi_IC.eps'],'-depsc','-r600')
close all
end

% % Calculate eigenvalues of fixed points
for scenario = 1:length(stage_all)
    figure
    plot([-1,-1,'ko'])
    hold on
    plot([-1, -1,'b.'])
    stage = stage_all(scenario);
    Sal = Sal_all(scenario);
    Sa2 = Sa2_all(scenario);
    sigma_y = sigma_y_all(scenario);
    s_j_ind = sigma_ind_all(scenario);
    mean_parms = mean_parms_all(:,stage)';
    % Create vec-permutation matrix
    V = make_vec_matrix(tot_ponds, stage);
    sigma_inds = (sigma_y-1)*tot_ponds+1:sigma_y*tot_ponds;
    if s_j_ind <0
        s_j2=s_j1;
    else
        s_j2=s_j1(s_j_ind)*ones(1,length(s_j1));
    end
    % Make diagonal block dispersal matrix (organized by life stage)
    sigma_new = zeros(stage); % disp rate matrix
    M = kron(eye(size(sigma_new))-sigma_new,eye(size(D))) + kron(sigma_new,D);
    % change adult survival and gamma to test parm
    all_parms = repmat(mean_parms',1,tot_ponds);
    all_parms(5,:) = [Sa_var(Sal) Sa_var(Sa2)];
    for p1=1:length(s_j1)
        % Fixed dispersal
        sigma_j_vector = [s_j1(p1) s_j2(p1)'];
    end
%% Change dispersal to specified life stage
M(sigma_i, sigma_j) = eye(size(D)) - diag(sigma_j_vector) +
   sigma_j_vector'.*D;

if stage == 1
    fun = @(a,x) one_stage(all_parms,sigma_j_vector,a,x);
elseif stage == 2
    fun = @(a,x) two_stage(all_parms,1-diag(M),a,x);
else
    fun = @(a,x) three_stage(all_parms,1-diag(M),a,x);
end

% Find trivial state eigenvalue
a_all = zeros(2*stage,1);
[J_diag,~] = create_Jacobian(2, stage, all_parms, a_all);
A_J = V*MsVs*J_diag;

% calculate eigenvalue
E_0 = eig(A_J);
 [~, I_0] = max(abs(E_0));
lambda_keep = E_0(I_0);

for ee = 1:200
    IC = randsample(100,tot_ponds*stage);
    [aa,~,exitflag] = fsolve(fun, IC);

    if exitflag~=0&& all(aa>0.001)
        [J_diag,~] = create_Jacobian(2, stage, all_parms, aa);
        A_J = V*MsVs*J_diag;

        % calculate eigenvalue
        E_0 = eig(A_J);
        [~, I_0] = max(abs(E_0));
        lambda_0 = E_0(I_0);
        if any(abs(a_all-aa)>0.0001)
            lambda_keep = [lambda_keep, lambda_0];
            a_all = [a_all, aa];
        end
    end
end

plot(sj1(p1),real(lambda_keep),'ko','markersize',3)
hold on
plot(sj1(p1),abs(lambda_keep),'b.'
plot([0 1],[1 1],'k-')
plot([0 1],[−1 −1],'k−')
end
xlabel(['\sigma_{',stage_labels(stage,sigma_y),'}'])
ylabel('\lambda')
xlim([0 1])
if stage == 1
    ylim([−5 5])
elseif stage == 2
    ylim([−2 2])
else
    ylim([0.8 1.2])
end
set(gcf,'Position',[680 639 417 239]);
set(gca,'FontSize',16)
if scenario == 3 || scenario == 6
    legend('real(\lambda)','\sigma',12,'location','northwest')
end
title(eig_titles(scenario))

% save bifurcation results
% print(gcf,['2patch_figs/twopatch_lambda','num2str(stage),'--stage_Sa','...
%   num2str(Sa1),num2str(Sa2),'_sigma',num2str(sigma_y),'_sj',num2str(sj_ind),'multi_IC.eps','--depsc','--r600'])
close all
end
end

function F_out = one_stage(parms_in,sj_vec,a_x)
% Fixed point solutions for the two patch, 1-stage model
% Parameters
F = parms_in(1,1); % Fecundity
Sn = parms_in(2,1); % Mean survival from metamorph to age 1
Sj = parms_in(3,1); % juvenile survival
Ss = parms_in(4,1); % subadult survival
Sa = parms_in(5,1); % female adult survival
Pjs = parms_in(6,1); % juvenile to subadult transition
Psf = parms_in(7,1); % subadult to female adult transition
Kx = parms_in(8,1); % carrying cap
gamma_x = parms_in(9,1); % density-dependent exponent
sj1 = sj_vec(1);
sj2 = sj_vec(2);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sa1 + FsSn.*(1./(1+a_x(1).*F./Kx).ˆgamma_x))'*(1-sj1)*a_x(1) + ...
(Sa2 + FsSn.*(1./(1+a_x(2).*F./Kx).ˆgamma_x))'*sj2*a_x(2) - a_x(1);
F_out(2) = (Sa1 + FsSn.*(1./(1+a_x(1).*F./Kx).ˆgamma_x))'*sj1*a_x(1) + ...
(Sa2 + FsSn.*(1./(1+a_x(2).*F./Kx).ˆgamma_x))'*(1-sj2)*a_x(2) - a_x(2);
end

function F_out = two_stage(parms_in, sj_vec, a_x)

% Fixed point solutions for the two patch, 2-stage model
% Parameters
F = parms_in(1,1); % Fecundity
Sn = parms_in(2,1); % Mean survival from metamorph to age 1
Sj = parms_in(3,1); % juvenile survival
Ss = parms_in(4,1); % subadult survival
Sa = parms_in(5,1); % female adult survival
Pjs = parms_in(6,1); % juvenile to subadult transition
Psf = parms_in(7,1); % subadult to female adult transition
Kx = parms_in(8,1); % carrying cap
gamma_x = parms_in(9,1); % density-dependent exponent

sj_vec = sj_vec([1,3,2,4]);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sj*(1-Psf)*a_x(1)*(1-sj_vec(1)) + FsSn.*(1./(1+a_x(1).*F./Kx).ˆgamma_x)*
             a_x(2)*(1-sj_vec(1)) + ...
     (Sj*(1-Psf)*a_x(3)*sj_vec(3) + FsSn.*(1./(1+a_x(4).*F./Kx).ˆgamma_x)*
             a_x(4)*sj_vec(3)) - a_x(1);
F_out(2) = (Sj*Psf*a_x(1)*(1-sj_vec(2)) + Sal*a_x(2)*(1-sj_vec(2)) + ...
     (Sj*Psf*a_x(3)*sj_vec(4) + Sa2*a_x(4)*sj_vec(4)) - a_x(2);
F_out(3) = (Sj*(1-Psf)*a_x(1)*sj_vec(1) + FsSn.*(1./(1+a_x(2).*F./Kx).ˆgamma_x)*
             a_x(2)*sj_vec(1) + ...
     (Sj*(1-Psf)*a_x(3)*(1-sj_vec(3)) + FsSn.*(1./(1+a_x(4).*F./Kx).ˆgamma_x)*
             a_x(4)*(1-sj_vec(3))) - a_x(3);
F_out(4) = (Sj*Psf*a_x(1)*sj_vec(2) + Sal*a_x(2)*sj_vec(2) + ...
\[
(S_j \cdot Psf \cdot a \cdot x (3) \cdot (1 - s_j \cdot vec (4)) + Sa2 \cdot a \cdot x (4) \cdot (1 - s_j \cdot vec (4))) = a \cdot x (4);
\]
end

function \( F_{\text{out}} = \text{three\_stage}(\text{parms\_in}, s_j \cdot vec, a \cdot x) \)

% Fixed point solutions for the two patch, 3-stage model

% Parameters
\[ F = \text{parms\_in}(1,1); \] % Fecundity
\[ Sn = \text{parms\_in}(2,1); \] % Mean survival from metamorph to age 1
\[ Sj = \text{parms\_in}(3,1); \] % juvenile survival
\[ Ss = \text{parms\_in}(4,1); \] % subadult survival
\[ Sa = \text{parms\_in}(5,1); \] % female adult survival
\[ Pjs = \text{parms\_in}(6,1); \] % juvenile to subadult transition
\[ Psf = \text{parms\_in}(7,1); \] % subadult to female adult transition
\[ Kx = \text{parms\_in}(8,1); \] % carrying cap
\[ \gamma_x = \text{parms\_in}(9,1); \] % density-dependent exponent

\[ s_j \cdot vec = s_j \cdot vec([1, 3, 5, 2, 4, 6]); \]
\[ Sa1 = \text{parms\_in}(5,1); \]
\[ Sa2 = \text{parms\_in}(5,2); \]

\[ F_{\text{out}}(1) = (S_j \cdot (1 - P) \cdot a \cdot x (1) \cdot (1 - s_j \cdot vec (1)) + F \cdot Sn \cdot (1/(1 + a \cdot x (3) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) = a \cdot x (1); \]
\[ F_{\text{out}}(2) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (1); \]
\[ F_{\text{out}}(3) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (2); \]
\[ F_{\text{out}}(4) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (3); \]
\[ F_{\text{out}}(5) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (4); \]
\[ F_{\text{out}}(6) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (5); \]
\[ F_{\text{out}}(7) = (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) +... \]
\[ (S_j \cdot Pjs \cdot a \cdot x (4) \cdot s_j \cdot vec (4) + F \cdot Sn \cdot (1/(1 + a \cdot x (6) \cdot F / Kx) \cdot gamma_x) \cdot a \cdot x (6) \cdot s_j \cdot vec (4)) - a \cdot x (6); \]
end
function two_patch_iteration_map

% Phase planes of select scenarios with multiple ICs

%% Analysis

rng('default')

% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent
mean parms all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
num_tests = 100;
T = 10000;
sj2_OG = linspace(0,1,num_tests);
% test specific dispersal probabilities
sj_alt = [.6 .7 .8];

%% Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

% Define all scenarios
stage_all = [1];
Sa1_all = [1 2];
Sa2_all = [2 2];
sigma_y_all = [1 1];
sigma_ind_all = [96 -1];

sub_titles = 'ABCDEF';
for scenario = 1:length(stage_all)
    stage = stage_all(scenario);
    Sa1 = Sa1_all(scenario);
    Sa2 = Sa2_all(scenario);
    sigma_y = sigma_y_all(scenario);
    sj_ind = sigma_ind_all(scenario);
    meanparms = meanparms_all(:,stage)';

    % Create vec-permutation matrix
    V = make_vec_matrix(tot_ponds, stage);
    sigma seks = (sigma_y = 1)*tot_ponds+1:sigma_y*tot_ponds;

    if sj_ind < 0
        sj2 = sj_alt;
    else
        sj2 = sj2_OG(sj_ind)*ones(1,length(sj_alt));
    end

    % Make diagonal block dispersal matrix (organized by life stage)
    sigma_new = zeros(stage); % disp rate matrix
    M = kron(eye(size(sigma_new)) - sigma_new, eye(size(D))) + kron(sigma_new,D);

    % change adult survival and gamma to test parm
    allparms = repmat(meanparms',1,tot_ponds);
    allparms(5,:) = [Sa_var(Sa1) Sa_var(Sa2)];

    for p1 = 3:length(sj_alt)
        figure
        subplot(2,1,1)
        plot([0 200],[0 200],'k-')
        hold on
        title(['\sigma_{a1} = ', num2str(sj_alt(p1))])
        subplot(2,1,2)
        plot([0 200],[0 200],'k-')
        hold on

        % Fixed dispersal
        sigma_j_vector = [sj_alt(p1) sj2(p1)]';
        % % Change dispersal to specified life stage
        M(sigma_seks, sigma_seks) = eye(size(D)) - diag(sigma_j_vector) + ...
                           sigma_j_vector.'*D;

        % Find one steady state to use for adults in patch 2
\[ N = \begin{bmatrix} 100 & 100 \end{bmatrix}; \]

\begin{verbatim}
for tt = 1:T
    % num egg masses in previous time step
    a_t = N(stage:stage:end, tt);
    % density dependent larval survival
    B = make_demographic_matrix(allParms, a_t, stage);
    % project population by one time step
    A = V' * M * V * B;
    N(:, tt + 1) = A * N(:, tt);
end

% Loop through adult abundances in patch 1
for a_1 = 0:0.5:200
    a_t = [a_1; N(2, end)];
    % density dependent larval survival
    B = make_demographic_matrix(allParms, a_t, stage);
    % project population by one time step
    A = V' * M * V * B;
    f1 = A * a_t;
    subplot(2, 1, 1)
    plot(a_1, f1(1), 'k.' )
    B = make_demographic_matrix(allParms, f1, stage);
    A = V' * M * V * B;
    f2 = A * f1;
    xlabel('a_1')
    ylabel('f(a_1,a_2)')
    ylim([0 200])
    % Second iteration
    subplot(2, 1, 2)
    plot(a_1, f2(1), 'k.' )
    xlabel('a_1')
    ylabel('f^2(a_1,a_2)')
end
end
end
end
end
end
\end{verbatim}
annotation(gcf,'textbox',...  
[0.29129197080292 0.812901639344262 0.0924306569343066 0.114754098360658],...
'String',sub_titles(p1),...
'LineStyle','none',...
'FontSize',12,...  
'FitBoxToText','off');  % save results
annotation(gcf,'textbox',...  
[0.29129197080292 0.335901639344262 0.0924306569343066 0.114754098360658],...
'String',sub_titles(p1+3),...
'LineStyle','none',...
'FontSize',12,...  
'FitBoxToText','off');  % save results
print(gcf,['2patch_figs/twopatch_iterate_map',num2str(stage),'=stg_Sa',...  
num2str(Sa1),num2str(Sa2),'_sigma',num2str(sigma_y),'_sj',num2str(sj_ind),num2str(...  
p1),'_multi_IC.eps',']','-depsc','-r600')

% close all
end

end

end
function two_patch_phase_planes

% Phase planes of select scenarios with multiple ICs

%% Analysis

rng('default')

% Parameters
F = [500 500 500];  % Fecundity
Sn = [0.2 0.2 0.2];  % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45];   % juvenile survival [.1 .7]
Ss = [0 0 0.7];      % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6];  % adult survival [.2 .8]
Pjs = [0 0 0.3];      % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6];   % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000];  % carrying capacity
gamma_x = [30 30 30];  % density-dependent exponent
mean_parms_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
num_tests = 100;
T = 10000;
sj2_OG = linspace(0,1,num_tests);
% test specific dispersal probabilities
sj_alt = [.75 .85 .95];

% % Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

% Define all scenarios
stage_all = [1 1];
Sa1_all = [1 2];
Sa2_all = [2 2];
sigma_y_all = [1 1];
sigma_ind_all = [96 -1];

sub_titles = 'ABC';
\textbf{for} scenario = 1:length(stage_all) \\
stage = stage_all(scenario); \\
Sa1 = Sa1_all(scenario); \\
Sa2 = Sa2_all(scenario); \\
sigma_y = sigma_y_all(scenario); \\
sj_ind = sigma_ind_all(scenario); \\
mean_parms = mean_parms_all(:,stage)'; \\
% Create vec-permutation matrix \\
V = make_vec_matrix(tot_ponds, stage); \\
sigma_inds = (sigma_y - 1)*tot_ponds + 1: sigma_y * tot_ponds; \\
if sj_ind <0 \\
  sj2 = sj_alt; \\
else \\
  sj2 = sj2_OG(sj_ind) * ones(1, length(sj_alt)); \\
end \\
% Make diagonal block dispersal matrix (organized by life stage) \\
sigma_new = zeros(stage); % dispersal rate matrix \\
M = kron(eye(size(sigma_new)) - sigma_new, eye(size(D))) + kron(sigma_new, D); \\
% change adult survival and gamma to test parm \\
all_parms = repmat(mean_parms', 1, tot_ponds); \\
all_parms(5,:) = [Sa_var(Sa1) Sa_var(Sa2)]; \\
\textbf{for} p1 = 1:length(sj_alt) \\
  figure \\
  % Fixed dispersal \\
  sigma_j_vector = [sj_alt(p1) sj2(p1)]'; \\
  % % Change dispersal to specified life stage \\
  M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) + ... \\
  diag(sigma_j_vector)' * D; \\
  if stage == 1 \\
    fun = @(a)x one_stage(all_parms, sigma_j_vector, a_x); \\
  elseif stage == 2 \\
    fun = @(a)x two_stage(all_parms, 1-diag(M), a_x); \\
  else \\
    fun = @(a)x three_stage(all_parms, 1-diag(M), a_x); \\
  end \\
% Find trivial state eigenvalue
a_all = zeros(2*stage,1);
J_diag = create_Jacobian(2, stage, all_parms, a_all);
A_J = V'*M*V*J_diag;

% calculate eigenvalue
E_0 = eig(A_J);
[~, I_0] = max(abs(E_0));
lambda_keep = E_0(I_0);
if scenario == 2 && p1 == 3
  % Manually pick 3 IC for different basins of attraction
  a_all = [59 20; 20 59; 0 45];
syms = ['.r'; '.rx'; '.ro'; '.rs'];
  for ee = 1:length(a_all)
    N = a_all(ee,:);
    for tt = 1:T
      % num egg masses in previous time step
      a_t = N(stage:stage:end,tt);
      % density dependent larval survival
      B = make_demographic_matrix(all_parms, a_t, stage);

      % % project population by one time step
      A = V'*M*V*B;
      N(:,tt+1) = A*N(:,tt);
    end
    end_N = N(:,end-99:end);
    plot(end_N(stage,:), end_N(2*stage,:), syms(ee,:))
    hold on
  end
else
  for ee = 1:500
    IC = randsample(100, tot_ponds*stage, 1);
    [aa,~,exitflag] = fsolve(fun, IC);

    if exitflag==0 && all(aa>0.001)
      J_diag = create_Jacobian(2, stage, all_parms, aa);
      A_J = V'*M*V*J_diag;
      % calculate eigenvalue
      E_0 = eig(A_J);
      [~, I_0] = max(abs(E_0));
      lambda_0 = E_0(I_0);
      if all(abs(a_all-AAa) > 0.01)
lambda_keep = [lambda_keep, lambda_0];
a_all = [a_all, aa];

% Run simulation with random initial condition
N = IC;
for tt = 1:T
    % num egg masses in previous time step
    a_t = N(stage:stage:end,tt);
    % % density dependent larval survival
    B = make_demographic_matrix(all_parms, a_t, stage);

    % % project population by one time step
    A = V'*M*V*B;
    N(:, tt+1) = A*N(:, tt);
end

[lyap_exp, ~] = ...
calculate_Lyapunov_exp_2patch(N, stage, all_parms, sigma_j_vector(1),...
 sigma_j_vector(2), sigma inds);

% final densities in each pond
end_N = N(:, end-99:end);
if max(end_N(1,:))-min(end_N(1,:))<0.001
    plot(end_N(stage,:),end_N(2*stage,:), 'kx')
else
    if lyap_exp<-0.01
        plot(end_N(stage,:),end_N(2*stage,:), 'k.' )
    elseif abs(lyap_exp)<0.01
        plot(end_N(stage,:),end_N(2*stage,:), 'b.' )
    else
        plot(end_N(stage,:),end_N(2*stage,:), 'r.' )
    end
end
hold on
end
end
end
xlabel('a_1')
ylabel('a_2')
set(gcf, 'Position', [197 534 274 244]);
set(gca,'Fontsize',16)
annotation(gcf,'textbox',... 
[0.32129197080292 0.795901639344262 0.0924306569343066 0.114754098360658],...
'String',sub_titles(p1),...
'LineStyle','none',...
'FontSize',12,...
'FitBoxToText','off'); % save results

% print(gcf,['2patch_figs/twopatch_phase_plane',num2str(stage),'=stg_Sa',... 
num2str(Sa1),num2str(Sa2),'_sigma',num2str(sigma_y),'sj_','num2str(sj_ind),... 
num2str(p1),'multi_IC.eps'],'=depsc','=r600')

end

close all
end

end

function F_out = one_stage(parms_in,sj_vec,a_x)
% Fixed point solutions for the two patch, 1-stage model
% Parameters
F = parms_in(1,1); % Fecundity
Sn = parms_in(2,1); % Mean survival from metamorph to age 1
Sj = parms_in(3,1); % juvenile survival
Ss = parms_in(4,1); % subadult survival
Sa = parms_in(5,1); % female adult survival
Pjs = parms_in(6,1); % juvenile to subadult transition
Psf = parms_in(7,1); % subadult to female adult transition
Kx = parms_in(8,1); % carrying cap
gamma_x = parms_in(9,1); % density-dependent exponent

sj1 = sj_vec(1);
sj2 = sj_vec(2);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sa1 + F*Sn.*(1./(1+a_x(1).*F./Kx).^gamma_x))*(1-sj1)*a_x(1) +... 
(Sa2 + F*Sn.*(1./(1+a_x(2).*F./Kx).^gamma_x))*sj2*a_x(2) - a_x(1); 
F_out(2) = (Sa1 + F*Sn.*(1./(1+a_x(1).*F./Kx).^gamma_x))*sj1*a_x(1) +... 
(Sa2 + F*Sn.*(1./(1+a_x(2).*F./Kx).^gamma_x))*(1-sj2)*a_x(2) - a_x(2);
end
function F_out = two_stage(parms_in, sj_vec, a_x)
%
% Fixed point solutions for the two patch, 2-stage model
%
% Parameters
F = parms_in(1,1);  % Fecundity
Sn = parms_in(2,1);  % Mean survival from metamorph to age 1
Sj = parms_in(3,1);  % juvenile survival
Ss = parms_in(4,1);  % subadult survival
Sa = parms_in(5,1);  % female adult survival
Pjs = parms_in(6,1); % juvenile to subadult transition
Psf = parms_in(7,1); % subadult to female adult transition
Kx = parms_in(8,1);  % carrying cap
gamma_x = parms_in(9,1);  % density-dependent exponent

sj_vec = sj_vec([1,3,2,4]);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sj*Psf*a_x(1)*(1-sj_vec(1)) + F*Sn.*(1./(1+a_x(2).*F./Kx)).^gamma_x)*a_x(2)*(1-sj_vec(1)) + ...
            (Sj*Psf*a_x(3)*s_j_vec(3) + F*Sn.*(1./(1+a_x(4).*F./Kx)).^gamma_x)*a_x(4)*s_j_vec(3)) - a_x(1);
F_out(2) = (Sj*Psf*a_x(1)*(1-sj_vec(2)) + Sa1*a_x(2))*s_j_vec(2) + ...
            (Sj*Psf*a_x(3)*s_j_vec(4) + Sa2*a_x(4)*s_j_vec(4)) = a_x(2);
F_out(3) = (Sj*Psf*a_x(1)*(1-sj_vec(1)) + F*Sn.*(1./(1+a_x(2).*F./Kx)).^gamma_x)*a_x(2)*s_j_vec(1) + ...
            (Sj*Psf*a_x(3)*(1-sj_vec(3)) + F*Sn.*(1./(1+a_x(4).*F./Kx)).^gamma_x)*a_x(4)*(1-sj_vec(3))) - a_x(3);
F_out(4) = (Sj*Psf*a_x(1)*s_j_vec(2) + Sa1*a_x(2)*s_j_vec(2)) + ...
            (Sj*Psf*a_x(3)*(1-sj_vec(4)) + Sa2*a_x(4)*(1-sj_vec(4))) = a_x(4);

end

function F_out = three_stage(parms_in, sj_vec, a_x)
%
% Fixed point solutions for the two patch, 3-stage model
%
% Parameters
F = parms_in(1,1);  % Fecundity
Sn = parms_in(2,1);  % Mean survival from metamorph to age 1
Sj = parms_in(3,1);  % juvenile survival
Ss = parms_in(4,1);  % subadult survival
Sa = parms_in(5,1);  % female adult survival
Pjs = parms_in(6,1);  % juvenile to subadult transition
Psf = parms_in(7,1);  % subadult to female adult transition
Kx = parms_in(8,1);  % carrying cap
gamma_x = parms_in(9,1);  % density-dependent exponent

sj_vec = sj_vec([1,3,5,2,4,6]);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sj*(1-Pjs)*a_x(1)*(1-sj_vec(1)) + F*Sn.*S_j.*a_x(3).*gamma_x.*a_x(6).*s_j_vec(1)) + ...
        (Sj*(1-Pjs)*a_x(4)*s_j_vec(4) + F*Sn.*S_j.*a_x(6).*gamma_x.*a_x(6).*s_j_vec(4)) - a_x(1);
F_out(2) = (Sj*Pjs*a_x(1)*(1-sj_vec(2)) + Ss*(1-Psf)*a_x(2)*(1-sj_vec(2))) + ...
        (Sj*Pss*a_x(4)*s_j_vec(5) + Ss*(1-Psf)*a_x(5)*s_j_vec(5)) - a_x(2);
F_out(3) = (Ss*Psfs*a_x(2)*(1-sj_vec(3)) + Sa1*a_x(3)*(1-sj_vec(3))) + ...
        (Ss*Psfs*a_x(5)*s_j_vec(6) + Sa2*a_x(6)*s_j_vec(6)) - a_x(3);
F_out(4) = (Sj*(1-Pjs)*a_x(1)*s_j_vec(1) + F*Sn.*S_j.*a_x(3).*gamma_x.*a_x(6).*s_j_vec(1)) + ...
        (Sj*(1-Pjs)*a_x(4)*(1-sj_vec(4)) + F*Sn.*S_j.*a_x(6).*gamma_x.*a_x(6).*s_j_vec(4)) - a_x(4);
F_out(5) = (Sj*Pjs*a_x(1)*s_j_vec(2) + Ss*(1-Psf)*a_x(2)*s_j_vec(2)) + ...
        (Sj*Pss*a_x(4)*(1-sj_vec(5)) + Ss*(1-Psf)*a_x(5)*(1-sj_vec(5))) - a_x(5);
F_out(6) = (Ss*Psfs*a_x(2)*s_j_vec(3) + Sa1*a_x(3)*s_j_vec(3)) + ...
        (Ss*Psfs*a_x(5)*(1-sj_vec(6)) + Sa2*a_x(6)*(1-sj_vec(6))) - a_x(6);

end
function twopatch_lambda_plan

%%% Create eigenvalue planes with numerically solved fixed points
% clear all; close all

%%% Analysis

% Parameters
F = [500 500 500]; % Fecundity
Sn = [0.2 0.2 0.2]; % first-year survival [.001 .5] (rough estimate)
Sj = [0 0.4 0.45]; % juvenile survival [.1 .7]
Ss = [0 0 0.7]; % subadult survival [.1 .95]
Sa = [0.6 0.6 0.6]; % adult survival [.2 .8]
Pjs = [0 0 0.3]; % juvenile to subadult transition [.05 .4]
Psa = [0 0.5 0.6]; % subadult to female adult transition [.1 1]
Kx = [80000 50000 40000]; % carrying capacity
gamma_x = [30 30 30]; % density-dependent exponent
mean_parms_all = [F; Sn; Sj; Ss; Sa; Pjs; Psa; Kx; gamma_x];
tot_ponds = 2;

% Define scenarios
Sa_var = [0.1 0.3 0.5 0.7 0.9];

% Initializations
IC_def = [20 20 20]'; % initial conditions
num_tests = 100;
sj1 = linspace(0,1,num_tests);
sj2 = linspace(0,1,num_tests);

% % Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds);

stage_label = ['aaa';'jaa';'jsa'];

for stage = 1:3
    meanparms = mean_parms_all(:,stage)';
    % Create vec-permutation matrix
    V = make_vec_matrix(tot_ponds, stage);

    for sigma_y = 1:stage
sigma_inds = (sigma_y-1)*tot_ponds+1:sigma_y*tot_ponds;

for s1 = 1:5
 for s2 = 1:5
  % simulate upper half of parameters
  if s2>=s1
   % change adult survival and gamma to test parm
   all_parms = repmat(mean_parms',1,tot_ponds);
   all_parms(5,:) = [Sa_var(s1) Sa_var(s2)];
  end

  IC_new = repmat(IC_def(1:stage),tot_ponds,1);
  for p1 = 1:num_tests
   for p2 = 1:num_tests
    % Fixed dispersal
    sigma_j_vector = [sj1(p1) sj2(p2)]';
    % Make diagonal block dispersal matrix (organized by life stage)
    sigma_new = zeros(stage); % disp rate matrix
    M = kron(eye(size(sigma_new))-sigma_new,eye(size(D))) + kron(sigma_new,D);
    % % Change dispersal to specified life stage
    M(sigma_inds, sigma_inds) = eye(size(D)) - diag(sigma_j_vector) + ...
    sigma_j_vector'.*D;
    if stage == 1
     fun = @(a.x) one_stage(all_parms,sigma_j_vector,a.x);
    elseif stage == 2
     fun = @(a.x) two_stage(all_parms,1-diag(M),a.x);
    else
     fun = @(a.x) three_stage(all_parms,1-diag(M),a.x);
    end
    a_all = fsolve(fun, IC_new);
    if size(a_all,2)>1
     disp('more than 1 sol')
    end
   end
   [J_diag,~] = create_Jacobian(2, stage, all_parms, a_all);
   A_J = V'*M*V*J_diag;
% calculate eigenvalue
E_0 = eig(A_J);
[-, 1,0] = max(abs(E_0));
lambda_0(p1,p2) = E_0(1,0);
end
end

figure
contour(sj1,sj2,real(lambda_0)', 'showtext', 'on')
hold on
[-,c]=contour(sj2,sj1,abs(lambda_0)',[1 1]', 'k-');
c.LineWidth = 3;

xticks([0 0.25 0.5 0.75 1])
yticks([0 0.25 0.5 0.75 1])
xlabel(['\sigma\{',stage_label(stage,sigma_y),'}'])
ylabel(['\sigma\{',stage_label(stage,sigma_y),'}'])

%%% axis labels for selected plots
set(gcf, 'Position', [500, 500, 313, 284])
if s1 == 1
   ylabel(['S\{a2} = ',num2str(Sa_var(s2))];['\sigma\{',stage_label(stage,sigma_y),'}'])
   set(gcf, 'Position', [500, 500, 377, 290])
end
if s2 == 5
   xlabel(['\sigma\{',stage_label(stage,sigma_y),'}']);['S\{a1} = ',num2str(Sa_var(s1))])
if s1 == 1
   set(gcf, 'Position', [500, 500, 377, 319.2])
else
   set(gcf, 'Position', [500, 500, 331, 312])
end
end

set(gca,'fontsize',15)

%%% Save plots

% print(gcf,['2 patch.figs/LambdaScn',num2str(s1),'-',num2str(s2),...
% '_',num2str(stage),'.stg',sigma_y','.eps'],'-depsc','-r600')
close all
end
end
%% Corollary functions

function F_out = one stage(parms_in, sj_vec, a_x)
% Fixed point solutions for the two patch, 1-stage model
% Parameters
F = parms_in(1,1);  % Fecundity
Sn = parms_in(2,1);  % Mean survival from metamorph to age 1
Sj = parms_in(3,1);  % Juvenile survival
Ss = parms_in(4,1);  % Subadult survival
Sa = parms_in(5,1);  % Female adult survival
Pjs = parms_in(6,1); % Juvenile to subadult transition
Psf = parms_in(7,1); % Subadult to female adult transition
Kx = parms_in(8,1);  % Carrying capacity
gamma_x = parms_in(9,1); % Density-dependent exponent

sj1 = sj_vec(1);
sj2 = sj_vec(2);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sa1 + F*Sn.*(1/(1+a_x(1).*F./Kx).^gamma_x))*(1-sj1)*a_x(1) +... 
(Sa2 + F*Sn.*(1/(1+a_x(2).*F./Kx).^gamma_x))*sj1*a_x(2) - a_x(1); 
F_out(2) = (Sa1 + F*Sn.*(1/(1+a_x(1).*F./Kx).^gamma_x))|sj1*a_x(1) +... 
(Sa2 + F*Sn.*(1/(1+a_x(2).*F./Kx).^gamma_x))*(1-sj2)*a_x(2) - a_x(2); 

end

function F_out = two stage(parms_in, sj_vec, a_x)
% Fixed point solutions for the two patch, 2-stage model
% Parameters
F = parms_in(1,1);  % Fecundity
Sn = parms_in(2,1);  % Mean survival from metamorph to age 1
Sj = parms_in(3,1);  % Juvenile survival
Ss = parms_in(4,1);  % Subadult survival
% female adult survival
Sa = parms_in(5,1);
% juvenile to subadult transition
Pjs = parms_in(6,1);
% subadult to female adult transition
Psf = parms_in(7,1);
% carrying cap
Kx = parms_in(8,1);
% density-dependent exponent
gamma_x = parms_in(9,1);

sj_vec = sj_vec([1,3,2,4]);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

F_out(1) = (Sj*(1-Psf)*a_x(1)*(1-sj_vec(1)) + F*Sn.*(1./(1+a_x(2).*F./Kx).^gamma_x)*a_x(2)*sj_vec(1)) +...
                 (Sj*(1-Psf)*a_x(2)*(1-sj_vec(2)) + F*Sn.*(1./(1+a_x(3).*F./Kx).^gamma_x)*a_x(3)*sj_vec(2)) - a_x(1);
F_out(2) = (Sj*Psf*a_x(1)*(1-sj_vec(1)) + Sa1*a_x(2)*(1-sj_vec(2)) +...
                 (Sj*Psf*a_x(2)*(1-sj_vec(2)) + Sa1*a_x(3)*sj_vec(3)) +...
                 (Sj*Psf*a_x(3)*(1-sj_vec(3)) + Sa1*a_x(4)*(1-sj_vec(4))) - a_x(2);
F_out(3) = (Sj*(1-Psf)*a_x(1)*(1-sj_vec(1)) + F*Sn.*(1./(1+a_x(2).*F./Kx).^gamma_x)*a_x(2)*sj_vec(3)) +...
                 (Sj*(1-Psf)*a_x(2)*(1-sj_vec(2)) + Sa1*a_x(3)*sj_vec(4)) +...
                 (Sj*(1-Psf)*a_x(3)*(1-sj_vec(4)) + Sa1*a_x(4)*(1-sj_vec(4))) - a_x(3);
F_out(4) = (Sj*Psf*a_x(1)*(1-sj_vec(1)) + Sa2*a_x(2)*(1-sj_vec(2)) +...
                 (Sj*Psf*a_x(2)*(1-sj_vec(2)) + Sa2*a_x(3)*sj_vec(3)) +...
                 (Sj*Psf*a_x(3)*(1-sj_vec(3)) + Sa2*a_x(4)*(1-sj_vec(4))) - a_x(4);

end

function F_out = three_stage(parms_in, sj_vec , a_x)
% Fixed point solutions for the two patch, 3-stage model
% Parameters
F = parms_in(1,1);          % Fecundity
Sn = parms_in(2,1);         % Mean survival from metamorph to age 1
Sj = parms_in(3,1);         % juvenile survival
Ss = parms_in(4,1);         % subadult survival
Sa = parms_in(5,1);         % female adult survival
Pjs = parms_in(6,1);        % juvenile to subadult transition
Psf = parms_in(7,1);        % subadult to female adult transition
Kx = parms_in(8,1);         % carrying cap
gamma_x = parms_in(9,1);   % density-dependent exponent

sj_vec = sj_vec([1,3,5,2,4,6]);
Sa1 = parms_in(5,1);
Sa2 = parms_in(5,2);

\[ F_{out}(1) = (S_j*(1-P_j)*a_x(1)*(1-s_jvec(1)) + F*S_n.*((1/(1+a_x(3).*F./Kx)).^gamma_x)*a_x(3)*s_jvec(1)) + ... \]
\[ (S_j*(1-P_j)*a_x(4)*s_jvec(4) + F*S_n.*((1/(1+a_x(6).*F./Kx)).^gamma_x)*a_x(6)*s_jvec(4)) - a_x(1); \]

\[ F_{out}(2) = (S_j*P_j*P_j*a_x(1)*(1-s_jvec(2)) + S_s*(1-P_s)*a_x(2)*(1-s_jvec(2)) + ... \]
\[ (S_j*P_j*a_x(4)*s_jvec(5) + S_s*(1-P_s)*a_x(5)*s_jvec(5)) - a_x(2); \]

\[ F_{out}(3) = (S_s*P_s*a_x(2)*(1-s_jvec(3)) + S_a1*a_x(3)*(1-s_jvec(3)) + ... \]
\[ (S_s*P_s*a_x(5)*s_jvec(6) + S_a2*a_x(6)*s_jvec(6)) - a_x(3); \]

\[ F_{out}(4) = (S_j*(1-P_j)*a_x(1)*s_jvec(1) + F*S_n.*((1/(1+a_x(3).*F./Kx)).^gamma_x)*a_x(3)*s_jvec(1)) + ... \]
\[ (S_j*(1-P_j)*a_x(4)*(1-s_jvec(4)) + F*S_n.*((1/(1+a_x(6).*F./Kx)).^gamma_x)*a_x(6)*(1-s_jvec(4))) - a_x(4); \]

\[ F_{out}(5) = (S_j*P_j*a_x(1)*s_jvec(2) + S_s*(1-P_s)*a_x(2)*s_jvec(2)) + ... \]
\[ (S_j*P_j*a_x(4)*(1-s_jvec(5)) + S_s*(1-P_s)*a_x(5)*(1-s_jvec(5))) - a_x(5); \]

\[ F_{out}(6) = (S_s*P_s*a_x(2)*s_jvec(3) + S_a1*a_x(3)*s_jvec(3)) + ... \]
\[ (S_s*P_s*a_x(5)*(1-s_jvec(6)) + S_a2*a_x(6)*(1-s_jvec(6))) - a_x(6); \]

end
Appendix C

Supplementary Information for ‘Sensitivity analysis of habitat contributions in a density-dependent Columbia spotted frog metapopulation model’

C.1 Larval survival model fitting

We optimized the root mean squared error for parameter sets using Latin hypercube sampling. Specifically, we initialized the optimizer with 1000 different values for $S_x \in \{0.01, 0.3\}$, $K_x \in \{10, 1000\}$, and $\gamma_x \in \{0.01, 100\}$, and calculated the minimum root mean squared error between egg count data and the Hassell competition model. We found that two local minima occurred for each pond type, and investigation of the smaller mean squared error results found that permanent and semi-permanent ponds have U-shaped results whereas ephemeral ponds monotonically decrease for the parameter ranges defined. Note that only 8 data points were fit to the Hassell competition model for ephemeral ponds, and more data is needed for more reliable results.
Fig. C.1: Optimized root mean squared error estimates for a range of parameters sampled from the Latin hypercube sampling method shows that two local minima occur for each pond type (subfigures A, E, I). Plotting all lower-valued root mean squared errors for each parameter shows that absolute minima occur for permanent and semi-permanent ponds; however, ephemeral ponds never reach a local minimum. This is most likely due to the small sample size in ephemeral ponds.
Fig. C.2: Total population elasticities to pond-specific parameters for the six $r - \beta$ combinations in the full (A) and partial (B) systems. Each bar represents total population elasticity to the pond-specific parameter specified by color.

C.2 Local sensitivity analysis

We plot results for all total population elasticities and pond contribution and pond size elasticities. Summaries for each plot can be found in the main text.
Fig. C.3: Elasticities of pond contribution to model parameters averaged over $r = 50, 1000$ and the full and partial systems. Each bar represents specific pond elasticity to the parameter defined on the $x$-axis.
Fig. C.4: Elasticities of pond size to model parameters averaged over $r = 50, 1000$ and the full and partial systems. Each bar represents specific pond elasticity to the parameter defined on the $x$-axis.
C.3 Global sensitivity analysis

We use the Partial Rank Correlation Coefficient (PRCC) to determine linear relationships between output variables for a range of input parameters (Marino et al. 2008). Given a vector of feasible values for a parameter $\theta$, and vector of outputs $z$, the PRCC between the two vectors is defined as,

\[
\text{PRCC}(\theta, z) = \frac{\text{cov}(\theta, z)}{\text{sd}(\theta)\text{sd}(z)},
\]

(C.1)

where cov$(\theta, z)$ is the covariance between $\theta$ and $z$, and sd$(\theta)$ and sd$(z)$ are the standard deviations of $\theta$ and $z$. PRCC is bounded between $-1$ and 1 with negative values representing negative correlations between $\theta$ and $z$, positive values representing positive correlations between $\theta$ and $z$, and near-zero values representing no correlation or a nonlinear relationship between $\theta$ and $z$.

We calculate the PRCC for total population size ($U$), pond size ($U_x$), and pond contribution $C_x$ for the following ranges in larval survival and dispersal parameters: $L_x = \{0.01, 0.3\}$, $K_x = \{10, 1000\}$, $\gamma_x = \{0.01, 100\}$, $r = \{50, 1000\}$, and $\beta = \{-0.2, 0.2\}$. Figure C.5 shows global sensitivities of total population size to each parameter, and Figure C.6 shows global sensitivities of each pond contribution and size to each parameter. In general, we find that there were no significant differences in sensitivities among the different parameters. In other words, the magnitude of change in population sizes and contributions were similar regardless of changes in the vital rate parameter. This is true for all but carrying capacity in the ephemeral ponds ($K_E$) that show noticeably lower sensitivities. These values also represent confidence in our vital rate estimates which show the degree to which we can expect changes under different parameter regimes. This means that changes in each parameter may have drastic effects on contribution metrics, lowering our confidence in our results. This is unavoidable with the information we currently have, and may only be remedied with more data.
Fig. C.5: Global sensitivity of the total population to larval survival parameters, $r$, and $\beta$. We find that the magnitude of most sensitivities are less than 0.2, indicating that confidence in these estimates are similar.
Fig. C.6: Global sensitivity of pond contribution and size to larval survival parameters, $r$, and $\beta$. We find that the magnitude of some sensitivities are greater than 0.5, indicating that confidence in these parameter estimates are low. This emphasizes the importance of ensuring accurate parameter estimates.
C.4 Matlab code

The following are Matlab scripts to generate simulations, analysis, and figures for the two patch model. All scripts were run in Matlab R2021a.

- `define_Parms.m`: Demographic parameter values used for analysis.
- `fit_dens_dep_surv.m`: Fits Hassell competition model to egg to metamorph survival data. Generates root mean squared error plots (appendix), and fitted data plots (main text).
- `make_den_dep_demographic_matrix.m`: Creates a diagonal matrix with submatrices that represent density dependent demographic processes for each pond. Combined with the dispersal matrix produces the projection matrix.
- `make_distance_based_dispersal.m`: Creates relative dispersal distance matrix using pond distances found in McCaffery et al. 2014 (fig. 1).
- `model_projection.m`: Runs simulations of the projection matrix. Calculates per capita contribution metrics for ponds and individuals.
- `calculate_dCx.m`: Calculates contribution metrics using equations presented in the main text.
- `dispersal_effects_on_dynamics.m`: Calculates per capita contribution metrics and end state population sizes for the full and fragmented systems. Generates plots for pond contributions under different $\beta$ and $r$ ranges, and difference between partial and full system contributions.
- `verify_sensitivity_equations.m`: Compares contributions calculated by simulated perturbations and sensitivity analysis.
- `sensitivity_analysis.m`: Calculates local sensitivities of population sizes and contributions to $L_x$, $K_x$, and $\gamma_x$. Generates all sensitivity plots used in the manuscript and proposed management strategies.
• PRCC_model.m: Performs Pearson rank correlation coefficient analysis across a range of survival and dispersal parameters and their effects on pond contributions and populations.
function [parms_out, S_means, num_ponds] = define_parms

% Parameters mean
Pb = 0.5; % breeding probability
Sm = 0.45; % survival from metamorph to age 1
Sj = 0.45; % juvenile survival
Ss = 0.7; % subadult survival
Sf = 0.6; % female adult survival
Pjs = 0.3; % juvenile to subadult transition
Pjf = 0; % juvenile to female adult transition
Psf = 0.6; % subadult to female adult transition
Cs = 1000; % clutch size

% All mean parameters and their variances
parms_out = [Pb Sm Sj Ss Sf Pjs Pjf Psf Cs];

% Egg to metamorph survival (mean and variance) for each pond
% [P1 P2 P3 S1 S2 E1 E2]
% OG
S_perm = [0.015 0.015 0.015]; % egg to metamorph surv for permanent ponds max:.1
S_semi = [0.035 0.035]; % egg to metamorph surv for semi-permanent ponds max:.2
S_eph = [0.012 0.012]; % egg to metamorph surv for ephemeral ponds max:.15

S_means = [S_perm S_semi S_eph];

% number of ponds in each category (permanent, semi-permanent, ephemeral)
num_perm = length(S_perm); num_semi = length(S_semi); num_eph = length(S_eph);
num_ponds = [num_perm, num_semi, num_eph];

end
function fit_dens_dep_surv
% fit Hassell Competition model to data using least squared sums.
% Uses data extrapolated from figures in McCaffrey 2010 (thesis)

% Get all egg to metamorph survival data
e_to_m_all = csvread('egg_to_metamorph_all.csv');

% Get all number of egg mass data
eggs_Perm = csvread('num_eggs Permanent_pond.csv');
eggs_Semi_Perm = csvread('num_eggs semi Permanent_pond.csv');
eggs_Eph = csvread('num_eggs ephemeral_pond.csv');

eggs_all = reshape([eggs_Perm(:,2); eggs_Semi_Perm(:,2); eggs_Eph(:,2)],10,7);

% Remove years 2000 and 2005 from egg mass data
eggs_all(6,:) = [];
eggs_all(1,:) = [];

% Omit 0 production years
omit_inds = [1 7; 1 8; 2 2; 2 5; 2 7; 2 8; 3 2; 3 7; 3 8; 5 8; 6 3; 6 5; ... 6 6; 7 5; 7 6; 7 7; 7 8];

for qq = 1:length(omit_inds)
e_to_m_all(omit_inds(qq,2),omit_inds(qq,1)) = 0;
eggs_all(omit_inds(qq,2),omit_inds(qq,1)) = 0;
end

% Egg to metamorph survival given the number of eggs produced in each year
N = 0:50;  % number of egg masses
Cs = 400;  % clutch size
N_h = N*Cs;  % number of hatchlings

% Fit Hassell model to data
% Initial guesses, lower and upper bounds
x1 = [.01 10 .01];
x2 = [.3 1000 100];

rng default  % For reproducibility
num_tests = 1000;
x0_all = [linspace(x1(1),x2(1),num_tests);linspace(x1(2),x2(2),num_tests); ... linspace(x1(3),x2(3),num_tests)];
% Shuffle points for Latin hypercube sampling

for pp = 1:3
    x0_shuffle(pp,:) = x0_all(pp,(randperm(length(x0_all(pp,:)))));
end

for kk = 1:length(x0_shuffle)
    x0 = x0_shuffle(:,kk);

    % Fit permanent ponds
    e_to_m_vec_perm = e_to_m_all(1:end-1,1:3);
    e_to_m_vec_perm = e_to_m_vec_perm(:,);
    e_to_m_vec_perm(e_to_m_vec_perm==0)=[];
    N_h_vec_perm = Csegg_all(1:end-1,1:3);
    N_h_vec_perm = N_h_vec_perm(:,);
    N_h_vec_perm(N_h_vec_perm==0)=[];
    fun = @(x)sum((x(1).*(1./(1+N_h_vec_perm./x(2))).^x(3))-e_to_m_vec_perm).^2);
    x_fit_perm(:,kk) = fmincon(fun,x0,[],[],[],[],x1,x2);

    % Calculate mean squared error
    MSE(kk,1) = sqrt(sum((x_fit_perm(1,kk).*((1/(1+N_h_vec_perm./x_fit_perm(2,kk)))).^2)));

    % Fit semi-permanent ponds
    e_to_m_vec_semi = e_to_m_all(:,4:5);
    e_to_m_vec_semi = e_to_m_vec_semi(:,);
    e_to_m_vec_semi(e_to_m_vec_semi==0)=[];
    N_h_vec_semi = Csegg_all(:,4:5);
    N_h_vec_semi = N_h_vec_semi(:,);
    N_h_vec_semi(N_h_vec_semi==0)=[];
    fun = @(x)sum((x(1).*(1./(1+N_h_vec_semi./x(2))).^x(3))-e_to_m_vec_semi).^2);
    x_fit_semi(:,kk) = fmincon(fun,x0,[],[],[],[],x1,x2);

    % Calculate mean squared error
    MSE(kk,2) = sqrt(sum((x_fit_semi(1,kk).*((1/(1+N_h_vec_semi./x_fit_semi(2,kk)))).^2)));

    % Fit ephemeral ponds
    e_to_m_vec_eph = e_to_m_all(:,6:7);
    e_to_m_vec_eph = e_to_m_vec_eph(:,);
    e_to_m_vec_eph(e_to_m_vec_eph==0)=[];
\[ N_{h\text{vec}eph} = \text{Cs*eggs}\_all(:,6:7); \]
\[ N_{h\text{vec}eph} = N_{h\text{vec}eph}(:,); \]
\[ N_{h\text{vec}eph}(N_{h\text{vec}eph}==0) = []; \]
\[ \text{fun} = @(x) \sum\left( x(1) \cdot (1/(1+N_{h\text{vec}eph}\_x(2)) - e_{to\_m\_vec\_eph})^2 \right); \]
\[ x_{\text{fit}\_eph}(:,kk) = \text{fmincon}(\text{fun},x0,[],[],[],[],x1,x2); \]
\[ % \% \text{Calculate mean squared error} \]
\[ \text{MSE}(kk,3) = \text{sqrt}\left( \sum\left( x_{\text{fit}\_eph}(1,kk) \cdot (1/(1+N_{h\text{vec}eph}\_x_{\text{fit}\_eph}(2,kk)) - e_{to\_m\_vec\_eph})^2 \right) \right); \]
\]
\[ \% \% \text{Find min values} \]
\[ \text{for } oo = 1:3 \]
\[ [\text{MSE}_{\text{min}}(oo),\text{MSE}\_\text{ind}(oo)] = \text{min}(\text{MSE}(:,oo)); \]
\[ \end \]
\[ \text{optimal}_\text{perm} = x_{\text{fit}\_perm}(:,\text{MSE}\_\text{ind}(1)); \]
\[ \text{optimal}_\text{semi} = x_{\text{fit}\_semi}(:,\text{MSE}\_\text{ind}(2)); \]
\[ \text{optimal}_\text{eph} = x_{\text{fit}\_eph}(:,\text{MSE}\_\text{ind}(3)); \]
\[ \text{table}([\text{optimal}_\text{perm};\text{MSE}\_\text{min}(1)], [\text{optimal}_\text{semi};\text{MSE}\_\text{min}(2)], ...) \]
\[ [\text{optimal}_\text{eph};\text{MSE}\_\text{min}(3)]) \]
\]
\[ % \% \text{Plot MSE for each parameter} \]
\[ \text{marker}\_\text{colors} = [\text{'k'};\text{'b'};\text{'r'}]; \]
\[ \text{markers} = [\text{'x'};\text{'o'};\text{'*'}]; \]
\[ \text{pond}\_\text{names} = \{\text{'Permanent'},\text{'Semi-permanent'},\text{'Ephemeral'}\}; \]
\[ \text{var}\_\text{names} = \{\text{'S'}_\text{P'},\text{'K'}_\text{P'},\text{'\gamma'}_\text{P'},\text{'S'}_\text{S'},\text{'K'}_\text{S'},\text{'\gamma'}_\text{S'},\text{'S'}_\text{E'},\text{'K'}_\text{E'},\text{'\gamma'}_\text{E'}\}; \]
\[ \text{hist}\_\text{titles} = \text{AEI}; \]
\[ \text{MSE}\_\text{titles} = \{\text{'BCD'},\text{'FGH'},\text{'JKL'}\}; \]
\[ \text{for } \text{pond} = 1:3 \]
\[ \text{figure} \]
\[ \text{histogram}(\text{MSE}(:,\text{pond}),20,\text{''facecolor''},\text{marker}\_\text{colors}(\text{pond})) \]
\[ \text{ylabel('Frequency')} \]
\[ \text{xlabel('Root mean squared error')} \]
\[ \text{set}\left(\text{gcf},\text{'Position'},[488 694 268 163]\right) \]
\[ \text{legend}(\text{pond}\_\text{names}(\text{pond}),\text{''location''},\text{''northwest''}) \]
\[ \text{title}(\text{hist}\_\text{titles}(\text{pond})) \]
\[ % \% \text{print}(\text{gcf},[\text{'7 patch\_figs/MSE\_hist\_''},\text{num2str}(\text{pond})\text{''} = \text{depsc}'', '-r600')] \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{figure} \]
for parms = 1:3
    subplot(1,3,parms)
    plot(x_fit_perm(parms,:),MSE(:,1),[marker_colors(1),markers(parms)])
    if parms == 1
        ylabel('Root mean squared error')
    end
    if parms == 3
        xlim([0.5 0.75])
    end
    ylim([MSE_min(1) -0.0001 MSE_min(1) +0.0002])
    xlabel(var_names(1,parms))
    title(MSE_titles(1,parms))
end
set(gcf,'Position',[488 641 914 215])

figure
for parms = 1:3
    subplot(1,3,parms)
    plot(x_fit_semi(parms,:),MSE(:,2),[marker_colors(2),markers(parms)])
    if parms == 1
        ylabel('Root mean squared error')
    end
    ylimit([MSE_min(2) -0.0001 MSE_min(2) +0.001])
    xlabel(var_names(2,parms))
    title(MSE_titles(2,parms))
end
set(gcf,'Position',[488 641 914 215])

figure
for parms = 1:3
    subplot(1,3,parms)
    plot(x_fit_eph(parms,:),MSE(:,3),[marker_colors(3),markers(parms)])
    if parms == 1
        ylabel('Root mean squared error')
    end
    if parms == 3
        xlim([0 1])
    end
ylim([MSE_min(3) - 0.0001 MSE_min(3) + 0.001])
xlabel(var_names(3, parms))
title(MSE_titles(3, parms))
end
set(gcf, 'Position', [488 641 914 215])

% Saved fitted vals
optimal_perm = [0.09857 425.23 0.6147];
optimal_semi = [0.29997 371.45 1.11];
optimal_eph = [0.05231 998.29 0.74556];

% Plot number of eggs to egg to metamorph survival
marker_sign = ['kx'; 'ko'; 'ks'; 'bx'; 'bo'; 'rx'; 'rs'];
figure
for ii = 1:3
    patch_eggs = eggs_all(:, ii);
    patch_eggs(patch_eggs == 0) = [];
    patch_surv = e_to_m_all(:, ii);
    patch_surv(patch_surv == 0) = [];
    plot(patch_eggs * Cs, patch_surv, marker_sign(ii,:), 'markersize', 10, 'linewidth', 1.5)
hold on
end
N_perm = optimal_perm(1) .* (1/(1+N_h./optimal_perm(2)).*optimal_perm(3));
plot(N_h, N_perm, 'k--', 'linewidth', 1.5)
ylabel('H_P')
xlabel('Eggs')
xlim([0 20000])
set(gca, 'fontsize', 12)
set(gcf, 'Position', [680 459 393 219])
legend({'P_1', 'P_2', 'P_3', 'Fitted Model'}, 'FontSize', 9)

% % print(gca,'7patch_figs/fit_perm','-depsc','-r600')

figure
for ii = 4:5
    patch_eggs = eggs_all(:, ii);
    patch_eggs(patch_eggs == 0) = [];
    patch_surv = e_to_m_all(:, ii);
    patch_surv(patch_surv == 0) = [];
end
plot(patch_eggs*Cs, patch_surv, marker_sign(ii,:), 'markersize', 10, 'linewidth', 1.5)
hold on
end
N_semi = optimal_semi(1).*((1/(1+N_h./optimal_semi(2))).^optimal_semi(3));
plot(N_h(1:15),N_semi(1:15), 'b--', 'linewidth', 1.5)
ylabel('H_S')
xlabel('Eggs')
xlim([0 5500])
set(gca,'fontsize', 12)
set(gcf, 'Position', [680 459 393 219])
legend({'S_1','S_2','Fitted Model'},'FontSize', 9)
% % print(gcf,'7patch_figs/fit_semi','-depsc','-r600')

figure
for ii = 6:7
    patch_eggs = eggs_all(:,ii);
    patch_eggs(patch_eggs == 0) = [];
    patch_surv = e_to_m_all(:,ii);
    patch_surv(patch_surv == 0) = [];
    plot(patch_eggs*Cs, patch_surv, marker_sign(ii,:), 'markersize', 10, 'linewidth', 1.5)
    hold on
end
N_eph = optimal_eph(1).*((1/(1+N_h./optimal_eph(2))).^optimal_eph(3));
plot(N_h(1:20),N_eph(1:20), 'r--', 'linewidth', 1.5)
ylabel('H_E')
xlabel('Eggs')
xlim([0 6000])
set(gca,'fontsize', 12)
set(gcf, 'Position', [680 459 393 219])
legend({'E_1','E_2','Fitted Model'},'FontSize', 9)
% % print(gcf,'7patch_figs/fit_eph','-depsc','-r600')
end
function [A_diag] = make_den_dep_demographic_matrix(parms_in, DD_vars, E_t)
% Creates a diagonal matrix with submatrices that represent density
% dependent demographic processes for each pond

% Parameters
Pb = parms_in(1); % breeding probability
Sm = parms_in(2); % Mean survival from metamorph to age 1
Sj = parms_in(3); % juvenile survival
Ss = parms_in(4); % subadult survival
Sf = parms_in(5); % female adult survival
Pjs = parms_in(6); % juvenile to subadult transition
Pjf = parms_in(7); % juvenile to female adult transition
Psf = parms_in(8); % subadult to female adult transition
Cs = parms_in(9); % clutch size

Lx = DD_vars(1,:);
Kx = DD_vars(2,:);
gamma_x = DD_vars(3,:);

num_stages = 3;
um_patches = size(DD_vars,2);

% Make diagonal matrix with transition matrices for each pond
for ii = 1:num_patches
    % density dependent larval survival
e_to_m_surv = Lx(ii).*((1./(1+E_t(ii).*Cs./Kx(ii))).^gamma_x(ii));

    % 3 stage demographic matrix for pond x
B_x = [Sj*(1-Pjs-Pjf) 0 Pb*Cs*e_to_m_surv*Sm; Sj*Pjs Ss*(1-Psf) 0; Sj*Pjf Ss*Psf Sf];

    % Diagonal Matrix
A_diag((ii-1)*num_stages+1:ii*num_stages,(ii-1)*num_stages+1:ii*num_stages) = B_x
end
end
function [D_out] = make_distance_based_dispersal(tot_ponds, alpha, r)  

% Distribution of dispersing frogs based on pond distance

% Ponds in u_loc are ordered as (P1,P2,P3,S1,S2,E1,E2)
u_loc = csvread('pond_coords.csv'); % pond locations for McCaffery et al. 2014 (fig. 1)
pond_dist = pdist2(u_loc', u_loc', 'euclidean');
neg_exponential_func = alpha.*exp(-pond_dist/r);
neg_exponential_func = neg_exponential_func - diag(diag(neg_exponential_func)); % create zeros along diagonal
d_i = repmat(sum(neg_exponential_func), tot_ponds, 1);
% probability distribution function for frog dispersal distribution
D_out = neg_exponential_func./d_i;
end
function [A,B,M,sigma_j_vector,end_u,end_u_x,Cx,patch_Cx,patch_Ox,patch_Ex,patch_Ix,end_u_all,unstable_inds]= ...
model_projection(tot_ponds,num_stages,IC,V,sigma_max,beta,DD_vars,meanParms,
kappa_x,T,M,D)

% Runs simulations of the projection matrix, calculates pond contribution
% metrics

u = IC;
for tt = 1:T
    % total pop in each pond
    U = sum(reshape(u(:,tt),num_stages,tot_ponds));

    % dd dispersal Ylikarjula
    sigma_j_vector = sigma_max(1,1)/(1 + exp(beta.*(kappa_x-U)));

    % replace juvenile dispersal with density-dependent dispersal
    M(1:tot_ponds,1:tot_ponds) = eye(size(D)) - diag(sigma_j_vector) + ...
        D*diag(sigma_j_vector);

    % num egg masses in previous time step
    E_t = u(num_stages:num_stages:end,tt)*meanParms(1);

    % % density dependent larval survival
    B = make_den_dep_demographic_matrix(meanParms,DD_vars,E_t);

    % % project population by one time step
    A = V'*MsVsB;
    u(:,tt+1) = A*u(:,tt);

    % Per capita contribution metric (Sample et al. 2019)
    Cx_all(:,tt+1) = sum(A)';

    % Per cap patch contribution
    all_inds = 1:21;
    for ii = 1:tот_ponds
        mat_inds = (ii-1)*3+1:ii*3;
        temp_inds = all_inds;
        temp_inds(mat_inds) = [];
        ux_all(ii,tt+1) = sum(u(mat_inds,tt+1));
        patch_Cx_all(ii,tt+1) = sum(u(mat_inds,tt+1).*...
% Cycles

Cx_all(mat inds, tt+1) = ux_all(ii, tt+1);

patch_Ox_all(ii, tt+1) = sum(A(mat inds, mat inds)*u(mat inds, tt+1))/ux_all(ii, tt+1);

patch_Ex_all(ii, tt+1) = sum(A(temp inds, mat inds)*u(mat inds, tt+1))/ux_all(ii, tt+1);

patch_Ix_all(ii, tt+1) = sum(A(mat inds, temp inds)*u(temp inds, tt+1))/ux_all(ii, tt+1);

end

end

end_u_all = u(:, end-49:end);
end_u = squeeze(mean(u(:, end-249:end), 2));
end_u_x = sum(reshape(end_u, num_stages, tot_ponds));

if any((max(u(:, end-49:end), [], 2) - min(u(:, end-49:end), [], 2)) > 0.01)
    unstable_inds = 1;
else
    unstable_inds = -1;
end

% Per capita contribution metric

Cx = squeeze(mean(Cx_all(:, end-449:end), 2)); % Individual contributions

% Per cap patch contribution

patch_Cx = sum(patch_Cx_all(:, end-49:end).*ux_all(:, end-49:end), 2)/sum(ux_all(:, end-49:end), 2);

% patch_Cx = squeeze(patch_Cx_all(:, end));

if all(abs(patch_Cx - squeeze(mean(patch_Cx_all(:, end-49:end), 2))) > 0.01)
    disp('Different patch_Cr')
end

%% Other contribution metrics

patch_Ox = sum(patch_Ox_all(:, end-49:end).*ux_all(:, end-49:end), 2)/sum(ux_all(:, end-49:end), 2);

patch_Ex = sum(patch_Ex_all(:, end-49:end).*ux_all(:, end-49:end), 2)/sum(ux_all(:, end-49:end), 2);
\[\text{patch}_\text{Ix} = \text{sum}(\text{patch}_\text{Ix}, \text{all}(\text{end} - 49 : \text{end}) \times \text{ux}, \text{all}(\text{end} - 49 : \text{end}), 2) / \text{sum}(\text{ux}, \text{all}(\text{end} - 49 : \text{end}), 2);\]

\[
\text{if all(abs(1 - patch}_\text{Ix}) + \text{patch}_\text{Ex} - \text{patch}_\text{Cx} > 0.001)
\]
\[
\text{disp('Patch Contribution Error (OG)');}
\]

\[
\text{end}
\]

\[
\text{end}
\]
function dCx_dalpha = calculate_dCx(tot_ponds, num_stages, dA_dtheta, dA_du_all,...
    end_u, end_u_x, du_dtheta, dux_dtheta, end_A, C_x)

    % Calculate patch contribution
    for xx = 1:tot_ponds
        xx inds = num_stages*(xx-1)+1:num_stages*xx;
        sum_dA_kx = 0;
        sum_A_kx = 0;
        for oo = 1:tot_ponds
            oo inds = num_stages*(oo-1)+1:num_stages*oo;
            partial_A = dA_dtheta(oo inds, xx inds);
            dA_dux_all = dA_du_all(oo inds, xx inds, xx inds);
            dA_dux_all = reshape(dA_dux_all, num_stages^2, num_stages);
            sum_dA_kx = sum_dA_kx + kron(end_u(xx inds)', ones(1, num_stages)) * ...
                           (partial_A(:)+dA_dux_all*du_dtheta(xx inds));
            sum_A_kx = sum_A_kx + end_A(oo inds, xx inds);
        end
        dCx_dalpha(xx) = 1./end_u_x(xx).* (sum_dA_kx + ...
                                          ones(1, num_stages)*sum_A_kx*du_dtheta(xx inds) -...
                                          C_x(xx)*dux_dtheta(xx));
    end
end
function dispersal_effects_on_dynamics

% % Main analysis for contribution metrics. Calculates contributions and
% populations for all simulations

% Define parameters
[mean_parms] = define_parms;
num_stages = 3;
tot_ponds = 7;

% pond-specific larval survival parameters
Sx = [0.09857 0.09857 0.09857 0.29997 0.29997 0.05231 0.05231];
Kx = [425.23 425.23 425.23 371.45 371.45 998.29 998.29];
gamma_x = [0.6147 0.6147 0.6147 1.11 1.11 0.74556 0.74556];
DD_vars = [Sx; Kx; gamma_x];
kappa_x = [50 50 50 40 40 30 30];

% % Constant life stage dispersal
sigma_j = 0.25; % juvenile dispersal rate
sigma_s = 0; % subadult dispersal rate
sigma_f = 0; % adult dispersal rate

sigma = diag([sigma_j, sigma_s, sigma_f]); % disp rate matrix
sigma_max = 0.5*diag(ones(num_stages,1));

% % Weights for a fully-connected system
alpha_full = ones(tot_ponds,tot_ponds);
% % Weights for a fragmented system
alpha_fragment = [1 1 0 0 0 0 0; 1 1 0 1 1 1 1; 0 0 1 1 0 0 0; 0 1 1 1 0 0 0;...
                  0 1 0 0 1 1 0; 0 1 0 0 1 1 0; 0 1 0 0 0 0 1];

% Create vec-permutation matrix
V = zeros(tot_ponds*num_stages);
I = eye(tot_ponds*num_stages);
k = 1;
for i=1:num_stages
    for j=i:num_stages:tot_ponds*num_stages
        V(k,:) = I(j,:);
        k = k+1;
    end
end
% % Initial conditions
IC_all = [20 20 20]';
T = 2000;
ra_all = linspace(1, 1000, 200);
Beta_all = linspace(-0.2, 0.2, 100);
unstable_inds = -ones(200, 100);

% % local_dynamics
% Make diagonal block dispersal matrix
u_loc = repmat(IC_all, tot_ponds, 1);
M = diag(ones(tot_ponds*num_stages, 1));
[end, end, end, end, u_loc_end, u_x_loc_end, ..., unstable_inds(r, bb)] = model_projection(tot_ponds, num_stages, uloc, V,...
    zeros(num_stages), 1, DD_vars, mean_parms, kappa_x, T, M, zeros(tot_ponds));

%% Run model for partial and full systems
for sims = 1:2
    if sims == 1
        alpha = alpha_full;
        sim_titles = 'Full';
    else
        alpha = alpha_fragment;
        sim_titles = 'Partial';
    end
    for rr = 1:length(ra_all)
        for bb = 1:length(Beta_all)
            % % Run model (with dispersal)
            % % Create distance–based dispersal matrix
            D = make_distance_based_dispersal(tot_ponds, alpha, ra_all(rr));
            % Make diagonal block dispersal matrix
            M = kron(eye(size(sigma))=sigma, eye(size(D))) + kron(sigma, D);
            [end_A, end_B, end_M, end_sigma_j, end_u(:, rr, bb), end_u_x(:, rr, bb), C(:, rr, bb), C_x
            (, rr, bb),...
            patch_Ox(:, rr, bb), patch_Ex(:, rr, bb), patch_Ix(:, rr, bb), end_u_all(:, rr, bb), unstable_inds(r, bb)] = ...
            model_projection(tot_ponds, num_stages, u_loc_end, V, sigma_max, Beta_all(bb),...
            DD_vars, mean_parms, kappa_x, 2000, M, D);
        end
    end
%% Define variables
end_U = squeeze(sum(end_u,1));
U_diff = end_U - sum(u_loc,end,1);
C_x_all(:,1,:.,sims)=C_x;

%% Save workspace
% save(["7patch_",sim_tits,'_file_2.mat'])

%% Plot beta vs C_x
figure
patch_symbols2 = ['k- ','k-- ','b- ','b-- ','r- ','r-- '];
for qq = 1:tot_ponds
    plot(Beta_all, squeeze(mean(C_x(qq,:,:),2))',patch_symbols2(qq,:)','linewidth',2)
    hold on
end
for qq = 1:tot_ponds
    sd1 = squeeze(mean(C_x(qq,:,:),2))'+squeeze(std(C_x(qq,:,:),[],2))';
    sd2 = squeeze(mean(C_x(qq,:,:),2))'-squeeze(std(C_x(qq,:,:),[],2))';
    m2 = [Beta_all, fliplr(Beta_all)];
    ranges = [sd1,fliplr(sd2)];
    h=fill(m2,ranges,'k','LineStyle','none');
    set(h,'facealpha',.2)
end
ylabel({sim_tits;'C_x*s'})
xlabel('eta')
xlim([-0.2 0.2])
set(gca,'Position',[629 570 569 298]);
set(gca,'fontsize',14);
legend({'$P_1$','$P_2$','$P_3$','$S_1$','$S_2$','$E_1$','$E_2$'},...
    'Interpreter','latex','location','northwest','Interpreter','latex','fontsize',9);
if strcmp(sim_tits,'Full')
    annotation(gcf,'textbox',...
        [0.310383838383838 0.842995169082125 0.0495050505050505 0.0772946859903378],...
        'String','C',...
        'LineStyle','none',...
        'FontWeight','bold',...
        'FontSize',15,....
        'FitBoxToText','off');
```matlab
else
    annotation(gcf,'textbox',...
        [0.310383838383838 0.842995169082125 0.0495050505050505 0.0772946859903378],...
        'String','D',...
        'LineStyle','none',...
        'FontWeight','bold',...
        'FontSize',15,...
        'FitBoxToText','off');
end
%
%

% % print(gcf,['7 patch_figs/Patch_beta_vs_Cx_','sim_tits'],'depsc','r600')
%

% % Plot beta vs U_x
figure
patch_symbols2 = ['k- ';'k- ';'h- ';'b- ';'r- ';'r- '];
for qq = 1:tot_ponds
    plot(Beta_all, squeeze(mean(end_u_x(qq,:,:),2))',patch_symbols2(qq,:),',linewidth' ,2)
hold on
end
for qq = 1:tot_ponds
    sd1 = squeeze(mean(end_u_x(qq,:,:),2))'+squeeze(std(end_u_x(qq,:,:),[],2))';
sd2 = squeeze(mean(end_u_x(qq,:,:),2))'-squeeze(std(end_u_x(qq,:,:),[],2))';
m2 = [Beta_all, fliplr(Beta_all)];
ranges = [sd1,fliplr(sd2)];
h=fill(m2,ranges,'k','LineStyle','none');
set(h,'facealpha',.2)
end
ylabel({'sim_tits','U_x*s'})
xlabel('
beta')
xlim([-0.2 0.2])
set(gcf, 'Position',[629 570 569 298]);
set(gca, 'fontsize',14);
legend({'$P_1$','$P_2$','$P_3$','$S_1$','$S_2$','$S_1$','$S_2$','$E_1$','$E_2$'},9)
if strcmp(sim_tits,'Full')
    annotation(gcf,'textbox',...
        [0.310383838383838 0.842995169082125 0.0495050505050505 0.0772946859903378],...
        'String','A',...
        'LineStyle','none','Interpreter','latex');
end
```

else
annotation(gcf,'textbox',...
[0.310383838383838 0.842995169082125 0.0495050505050505 0.0772946859903378],...
'String','B',....
'LineStyle','none',....
'FontWeight','bold',....
'FontSize',15,...
'FitBoxToText','off');
end

end

% % Load workspace (to avoid long run times)
% load('7patch_Full_file.mat')
% load('7patch_Partial_file.mat')

% % Plot mean C_x difference between full and partial systems
figure
plot([-1,1],['ko', 'markersize', 5, 'linewidth', 1])
hold on
plot([-1,1],['kx', 'markersize', 20, 'linewidth', 1])
plot([-1,1],['ks', 'markersize', 20, 'linewidth', 1])
plot([-1,1],['k.', 'markersize', 20, 'linewidth', 1])
plot([0 8], [0 0], ['k--'])
for ww = 1:tot_ponds
C_x_neg = C_x_all(ww,:,:1:25,2)-C_x_all(ww,:,:1:25,1);
C_x_neg = C_x_neg(:);
C_x_lowneg = C_x_all(ww,:,:26:50,2)-C_x_all(ww,:,:26:50,1);
C_x_lowneg = C_x_lowneg(:);
C_x_lowpos = C_x_all(ww,:,:51:75,2)-C_x_all(ww,:,:51:75,1);
C_x_lowpos = C_x_lowpos(:);
C_x_pos = C_x_all(ww,:,:76:end,2)-C_x_all(ww,:,:76:end,1);
C_x_pos = C_x_pos(:);
mean_change_neg = squeeze(mean(C_x_neg));
mean_change_lowneg = squeeze(mean(C_x_lowneg));
mean_change_lowpos = squeeze(mean(C_x_lowpos));
mean_change_pos = squeeze(mean(C_x_pos));
sd_change_neg = squeeze(std(C_x_neg));
sd_change_lowneg = squeeze(std(C_x_lowneg));
sd_change_lowpos = squeeze(std(C_x_lowpos));
sd_change_pos = squeeze(std(C_x_pos));

eba = errorbar(ww-.3, mean_change_neg, sd_change_neg, sd_change_neg);
eba.Color = [0 0 0];
eba.LineStyle = 'none';

eba = errorbar(ww-.1, mean_change_lowneg, sd_change_lowneg, sd_change_lowneg);
eba.Color = [0 0 0];
eba.LineStyle = 'none';

eba = errorbar(ww+.1, mean_change_lowpos, sd_change_lowpos, sd_change_lowpos);
eba.Color = [0 0 0];
eba.LineStyle = 'none';

eba = errorbar(ww+.3, mean_change_pos, sd_change_pos, sd_change_pos);
oba.Color = [0 0 0];
oba.LineStyle = 'none';

plot(ww-.3, mean_change_neg, 'ko', 'markersize', 7, 'linewidth', 1)
plot(ww-.1, mean_change_lowneg, 'kx', 'markersize', 7, 'linewidth', 1)
plot(ww+.1, mean_change_lowpos, 'ks', 'markersize', 7, 'linewidth', 1)
plot(ww+.3, mean_change_pos, 'k.', 'markersize', 23, 'linewidth', 1)

end

xticks(1:7)
xticklabels({'P.1', 'P.2', 'P.3', 'S.1', 'S.2', 'E.1', 'E.2'})

xlim([0 8])
set(gcf, 'Position', [911 536 500 197]);

ylabel('Partial C_x's = Full C_x's')

legend1 = legend('\beta < -0.1', '-0.1 < \beta < 0', '0 < \beta < 0.1', '\beta > 0.1');

set(legend1, 'Position', [0.3262 0.57428 0.23123 0.28036]);

set(gca, 'fontsize', 11)

% print(gcf, '7patch_figs/mean_Cx_diff', '-depsc', '-r600')
function verify_sensitivity_equations
% Validate sensitivity analysis results with simulations

% Define parameters
[mean parms] = define parms;
num stages = 3;
tot ponds = 7;

% Larval survival parameters
Sx = [0.09857 0.09857 0.09857 0.29997 0.29997 0.05231 0.05231];
Kx = [425.23 425.23 425.23 371.45 371.45 998.29 998.29];
gamma_x = [0.6147 0.6147 0.6147 1.11 1.11 0.74556 0.74556];
DD vars = [Sx; Kx; gamma_x];
kappa_x = [50 50 50 40 40 30 30];

% Constant life stage dispersal
sigma_j = 0.25; % juvenile dispersal rate
sigma_s = 0; % subadult dispersal rate
sigma_f = 0; % adult dispersal rate
sigma = diag([sigma_j, sigma_s, sigma_f]); % disp rate matrix
sigma_max = 0.5*diag(ones(num stages,1));

% Weights for a fully-connected system
alpha_full = ones(tot ponds,tot ponds);
% Weights for a fragmented system
alpha_fragment = [1 1 0 0 0 0 0; 1 1 0 1 1 1 1; 0 0 1 1 0 0 0; 0 1 1 1 0 0 0; 0 1 0 0 1 1 0; 0 1 0 0 0 0 1];

% Create vec-permutation matrix
V = zeros(tot ponds*num stages);
I = eye(tot ponds*num stages);
k = 1;
for i=1:num stages
    for j=i:num stages:tot ponds*num stages
        V(k,:) = I(j,:);
        k = k+1;
    end
end
% % Initial conditions
IC_all = [20 20 20]';
T = 2000;
r_all = [50,1000];
Beta_all = [-0.2 0.2];

% % local_dynamics
% Make diagonal block dispersal matrix
u_loc = repmat(IC_all,tot_ponds,1);
M = diag(ones(tot_ponds*num_stages,1));
[end, end, end, end, u_loc_end, end, end, end, end] = model_projection(tot_ponds, num_stages, u_loc, V,...
zeros(num_stages),1,DD_vars,mean_parms,kappa_x,T,M,zeros(tot_ponds));

% % 7–patch simulations for full and partial systems
for sims = 1:2
  if sims == 1
    alpha = alpha_full;
sims_tits = 'Full';
  else
    alpha = alpha_fragment;
sims_tits = 'Partial';
  end

% % loop through r and beta vals
for rr = 1:length(r_all)
  for bb = 1:length(Beta_all)
    % % Original model (no perturbations
    % % Create distance–based dispersal matrix
    D = make_distance_based_dispersal(tot_ponds, alpha, r_all(rr));
    % Make diagonal block dispersal matrix
    M = kron(eye(size(sigma))–sigma,eye(size(D))) + kron(sigma,D);
    [end_A, end_B, end_M, end_sigma_j, end_u(:,rr,bb), end_u_x(:,rr,bb), C(:,rr,bb),
    C_x(:,rr,bb), patch_Ox(:,rr,bb), patch_Ex(:,rr,bb), patch_Ix(:,rr,bb)] =
    ...
    model_projection(tot_ponds, num_stages, u_loc_end, V, sigma_max, Beta_all(bb),
    ...
    DD_vars,mean_parms,kappa_x,T,M,D);
    end_U(rr,bb) = sum(end_u(:,rr,bb),1);
  end

% % Calculate d/du
\( F = \text{mean\_parms}(1)\ast\text{mean\_parms}(9); \)
\( I_{s} = \text{eye}(\text{tot\_ponds}\ast\text{num\_stages}); \)

\% Calculate \( dA/du \)
% Calculate derivatives for each entry in pop vector
\( u_{\text{index}} = 1; \)
for \( kk = 1: \text{tot\_ponds} \)
for \( yy = 1: \text{num\_stages} \)

% Make diagonal block dispersal matrix
\( \text{dsigma} = \text{zeros}(1, \text{tot\_ponds}); \)
\( \text{dsigma}(kk) = \text{sigma\_max}(1, 1)\ast\text{Beta\_all}(bb)\ast\exp(\text{Beta\_all}(bb)\ast...) \)
\( \left(kappa_x(kk)\ast\text{end\_u\_x}(kk, rr, bb))\right)/(1+\exp(\text{Beta\_all}(bb)\ast(kappa_x(kk)\right)\ast\text{end\_u\_x}(kk, rr, bb)))\right)^{-2}; \)
\( dM_{du} = \text{zeros}(\text{tot\_ponds}\ast\text{num\_stages}, \text{tot\_ponds}\ast\text{num\_stages}); \)
\( dM_{du}(1: \text{tot\_ponds}, 1: \text{tot\_ponds}) = (D-\text{eye}(\text{tot\_ponds}))\ast\text{diag}(\text{dsigma}); \)
\( dB_{du} = \text{zeros}(\text{tot\_ponds}\ast\text{num\_stages}); \)
if \( yy == 3 \)
\( \% \text{num adults in previous time step} \)
\( a_{t} = \text{end\_u}(\text{num\_stages}\ast kk, rr, bb); \)
\( dH = -Sx(kk)\ast\gammaamma_x(kk)\ast\text{F.}/(Kx(kk)\ast(1+a_{t}\ast\text{F.}/Kx(kk)))^\left(\gammaamma_x(kk)\right); \)
\( dB_{du}(\text{num\_stages}\ast(kk-1)+1, \text{num\_stages}\ast kk) = F\ast\text{mean\_parms}(2)\ast dH; \)
end
\( dA_{du} = V'\ast dM_{du}\ast V\ast s\_end\_B + V'\ast s\_end\_M\ast V\ast dB_{du}; \)
\( \text{vecdA\_du}(:, u_{\text{index}}) = dA_{du}(:); \)
\( dA_{du\_all}(:, :, u_{\text{index}}) = dA_{du}; \)
\( u_{\text{index}} = u_{\text{index}}+1; \)
end
end

\% \text{d/dLx} \)
for \( LL = 1: \text{tot\_ponds} \)
\( dB_{dLx} = \text{zeros}(\text{tot\_ponds}\ast\text{num\_stages}); \)
\( \% \text{num egg masses in previous time step} \)
\( a_{t} = \text{end\_u}(\text{num\_stages}\ast LL, rr, bb); \)
\( dH = 1/((1+a_{t}\ast\text{F.}/Kx(LL)))^\left(\gammaamma_x(LL)\right); \)
\( dB_{dLx}(\text{num\_stages}\ast(LL-1)+1, \text{num\_stages}\ast LL) = F\ast\text{mean\_parms}(2)\ast dH; \)
\( dA_{dLx} = V'\ast s\_end\_M\ast V\ast dB_{dLx}; \)
\( \text{du\_dLx}(:, LL, rr, bb) = ((I_{s}-\text{end\_A}\ast\text{kron}(\text{end\_u}(:, rr, bb)')\ast I_{s})\ast... \ast\text{vecdA\_du})\ast(-1)\ast\text{kron}(\text{end\_u}(:, rr, bb)', I_{s})\ast dA_{dLx}(:); \)
dux\_dLx(:,LL,rr,bb) = sum(reshape(dux\_dLx(:,LL,rr,bb),num\_stages,tot\_ponds)) ;

dU\_dLx(LL,rr,bb) = sum(dux\_dLx(:,LL,rr,bb)) ;

% % Calculate perturbations of C\_x

dCx\_dLx(:,LL,rr,bb) = calculate\_dCx(tot\_ponds,num\_stages,da\_dLx,da\_du\_all,end\_u(:,rr,bb),...
end\_u\_x(:,rr,bb),du\_dLx(:,LL,rr,bb),dux\_dLx(:,LL,rr,bb),end\_A,C\_x(:,rr,bb)) ;

% % % Perturbation simulations

% dL = 0.001 ;
DD\_vars\_LL = DD\_vars ;
DD\_vars\_LL(1,LL) = DD\_vars\_LL(1,LL)+dL ;
[~,~,~,~,end\_u\_dL,end\_u\_x\_dL,C\_dL,Cx\_dL,~,~] = ...
model\_projection(tot\_ponds,num\_stages,end\_u(:,rr,bb),V,sigma\_max,Beta\_all(bb),...
DD\_vars\_LL,mean\_parms,kappa\_x,T,end\_M,D) ;
end\_du\_dLx(:,LL,rr,bb) = (end\_u\_dL-end\_u(:,rr,bb))/dL ;
end\_dux\_dLx(:,LL,rr,bb) = (end\_u\_x\_dL-end\_u\_x(:,rr,bb))/dL ;
end\_dCx\_dLx(:,LL,rr,bb) = (C\_dL-C\_x(:,rr,bb))/dL ;

% % \( d/dK_x \)

for KK = 1:tot\_ponds
dB\_dKx = zeros(tot\_ponds*num\_stages) ;
% num egg masses in previous time step
a\_t = end\_u(num\_stages*KK,rr,bb) ;

dH = (gamma\_x(KK)*Sx(KK)*F*a\_t)./((Kx(KK)-2*f(1+a\_t.*F./Kx(KK))).*(gamma\_x(KK)+1)) ;

dB\_dKx(num\_stages*(KK-1)+1:num\_stages*KK) = F*mean\_parms(2)*dH ;

% % % Perturbations of C\_x

dA\_dKx = V'*end\_M*V*dB\_dKx ;

du\_dKx(:,KK,rr,bb) = ((I_s-end\_A-kron(end\_u(:,rr,bb)\',I_s)...*vecdA\_du)'*(-1))*(kron(end\_u(:,rr,bb)\',I_s)*dA\_dKx(:,)) ;

dux\_dKx(:,KK,rr,bb) = sum(reshape(du\_dKx(:,KK,rr,bb),num\_stages,tot\_ponds)) ;

% % dU\_dKx(KK,rr,bb) = sum(dux\_dKx(:,KK,rr,bb)) ;

% % % Perturbations of C\_x

\[ d\text{Cx}_dKx(:,KK,rr,bb) = \text{calculate}_d\text{Cx}(\text{tot}_\text{ponds},\text{num}_\text{stages},dA_dKx,dA_du_{all},\text{end}_u(:,rr,bb)), ... \]
\[ \text{end}_u_x(:,rr,bb),du_dKx(:,KK,rr,bb),dux_dKx(:,KK,rr,bb),end_A,\text{C}_x(:,rr,bb)); \]

\% \% Perturbation simulations
\% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \% \%
Perturbation simulations

dgamma = 0.001;

\[ \text{DD_vars_gamma} = \text{DD_vars}; \]
\[ \text{DD_vars_gamma}(3,\text{GG}) = \text{DD_vars_gamma}(3,\text{GG}) + \text{dgamma}; \]
\[
\text{model_projection}(\text{tot_ponds}, \text{num_stages}, \text{end_u}(:, \text{rr}, \text{bb}), V, \text{sigma_max}, \\
\text{Beta_all}(\text{bb}), \ldots)
\]
\[ \text{end_du_dgamma}(:, \text{GG}, \text{rr}, \text{bb}) = (\text{end_u_dgamma} - \text{end_u}(:, \text{rr}, \text{bb}))/\text{dgamma}; \]
\[ \text{end_dC_dgamma}(:, \text{GG}, \text{rr}, \text{bb}) = (\text{C_dgamma} - \text{C_x}(:, \text{rr}, \text{bb}))/\text{dgamma}; \]

% Plot Results

figure

plot(0, 0, kx); hold on; plot(0, 0, k*); plot(0, 0, k+); % plot(0, 0, gx); plot(0, 0, cx)

plot(sign(end_du_dLx(:)).*log(1+abs(end_du_dLx(:))), ...
    sign(dudLx(:)).*log(1+abs(dudLx(:))), kx')
plot(sign(end_du_dKx(:)).*log10(1+abs(end_du_dKx(:))), ...
    sign(dudKx(:)).*log10(1+abs(dudKx(:))), k+')
plot(sign(end_du_dgamma(:)).*log10(1+abs(end_du_dgamma(:))), ...
    sign(dudgamma(:)).*log10(1+abs(dudgamma(:))), k+')

xlabel('$$\text{\texttt{\textbackslash x\texttt{\textbackslash l e f t ( \frac{\Delta u\ast}{\Delta \theta_x} \right) \texttt{\textbackslash r i g h t )}}}$$; sim_tits}, 'Interpreter', 'Latex')
ylabel('$$\text{\texttt{\textbackslash l e f t ( \frac{\Delta u\ast}{d\theta_x} \right) \texttt{\textbackslash r i g h t )}}}$$', 'Interpreter', 'Latex')

legend(Lx', 'Kx', gamma, 'Location', 'Northwest')

xlim([-5 6.5])
ylim([-5 6.5])

set(gcf, 'Position', [489 535 358 314])

% print(gcf,['\texttt{7patch_figs/du_validate}', sim_tits],'-dpsc','-r600')

% plot dux
figure
plot (0,0,'kx'); hold on; plot (0,0,'k*'); plot (0,0,'k+');% plot (0,0,'gx'); plot (0,0,'cx')

plot (sign(end_dux_dLx(:)).* log(1+abs(end_dux_dLx(:))) , ...
     sign(dux_dLx(:)).* log(1+abs(dux_dLx(:))), 'kx')
plot (sign(end_dux_dKx(:)).* log10(1+abs(end_dux_dKx(:))) , ...
     sign(dux_dKx(:)).* log10(1+abs(dux_dKx(:))) , 'k*')
plot (sign(end_dux_dgamma(:)).* log10(1+abs(end_dux_dgamma(:))) , ...
     sign(dux_dgamma(:)).* log10(1+abs(dux_dgamma(:))) , 'k+')
xlabel({ '$\text{plog}_x{10}\left(\frac{\Delta \gamma x^*}{\Delta \theta x}\right)$'; '
     \text{Interpreter}', 'LaTeX'});
ylabel({ '$\text{plog}_x{10}\left(\frac{\Delta \gamma x^*}{d\theta x}\right)$'; '
     \text{Interpreter}', 'LaTeX'});
legend ('L_x', 'K_x', '\gamma_x', 'Location', 'Northwest')
xlim([-5 6.5])
ylim([-5 6.5])
set(gcf, 'Position', [489 535 358 314])

% % print (gcf,['7 patch_figs/dux_validate ', sim_tits],'-depsc','-r600')

% % plot dC_x
figure
plot (0,0,'kx'); hold on; plot (0,0,'k*'); plot (0,0,'k+');% plot (0,0,'gx'); plot (0,0,'cx')

plot (sign(end_dC_x_dLx(:)).* log(1+abs(end_dC_x_dLx(:))) , ...
     sign(dC_x_dLx(:)).* log(1+abs(dC_x_dLx(:))), 'kx')
plot (sign(end_dC_x_dKx(:)).* log10(1+abs(end_dC_x_dKx(:))) , ...
     sign(dC_x_dKx(:)).* log10(1+abs(dC_x_dKx(:))) , 'k*')
plot (sign(end_dC_x_dgamma(:)).* log10(1+abs(end_dC_x_dgamma(:))) , ...
     sign(dC_x_dgamma(:)).* log10(1+abs(dC_x_dgamma(:))) , 'k+')
xlabel({ '$\text{plog}_x{10}\left(\frac{\Delta C_x^*}{\Delta \theta x}\right)$'; '
     \text{Interpreter}', 'LaTeX'});
ylabel({ '$\text{plog}_x{10}\left(\frac{\Delta C_x^*}{d\theta x}\right)$'; '
     \text{Interpreter}', 'LaTeX'});
legend ('L_x', 'K_x', '\gamma_x', 'Location', 'Northwest')
xlim([-2.5 2])
ylim([-2.5 2])
set(gcf, 'Position', [489 535 358 314])

% % print (gcf,['7 patch_figs/dCx_validate ', sim_tits],'-depsc','-r600')

end
function sensitivity_analysis

% % Validate sensitivity analysis results with simulations
% Define parameters
[mean_parms] = define_parms;
num_stages = 3;
tot_ponds = 7;

% % Larval survival parameters
Sx = [0.09857 0.09857 0.09857 0.29997 0.29997 0.05231 0.05231];
Kx = [425.23 425.23 425.23 371.45 371.45 998.29 998.29];
gamma_x = [0.6147 0.6147 0.6147 1.11 1.11 0.74556 0.74556];
DD_vars = [Sx; Kx; gamma_x];
kappa_x = [50 50 50 40 40 30 30];

% % Constant life stage dispersal
sigma_j = 0.25; % juvenile dispersal rate
sigma_s = 0; % subadult dispersal rate
sigma_f = 0; % adult dispersal rate
sigma = diag([sigma_j, sigma_s, sigma_f]); % disp rate matrix
sigma_max = 0.5*diag(ones(num_stages,1));

% % Weights for a fully-connected system
alpha_full = ones(tot_ponds,tot_ponds);
% % Weights for a fragmented system
alpha_fragment = [1 1 0 0 0 0 0; 1 1 0 1 1 1 1; 0 0 1 1 0 0 0; 0 1 1 1 0 0 0;...
0 1 0 0 1 1 0; 0 1 0 0 1 1 0; 0 1 0 0 0 0 1];

% Create vec-permutation matrix
V = zeros(tot_ponds*num_stages);
I = eye(tot_ponds*num_stages);
k = 1;
for i=1:num_stages
    for j=i:num_stages:tot_ponds*num_stages
        V(k,:) = I(j,:);
        k = k+1;
    end
end

% % Initial conditions
IC_all = [20 20 20];
T = 2000;
r_all = [50,1000];
Beta_all = [-0.2 0 0.2];

% % Initializations
patch_symbols = ['kx'; 'ko'; 'ks'; 'bx'; 'bo'; 'rx'; 'ro'];

% % local_dynamics

% Make diagonal block dispersal matrix
u_loc = repmat(IC_all, tot_ponds, 1);
M = diag(ones(tot_ponds*num_stages, 1));
[end, end, end, end, end, u_loc_end, end, end, end, end] = model_projection(tot_ponds, num_stages, u_loc, V, ...
    zeros(num_stages), 1, DD_vars, mean_parms, kappa_x, T, M, zeros(tot_ponds));

%% 7–patch simulations for full and partial systems

for sims = 1:2
    if sims == 1
        alpha = alpha_full;
        sim_tits = 'Full';
    else
        alpha = alpha_fragment;
        sim_tits = 'Partial';
    end

    % % loop through r and beta vals
    for rr = 1:length(r_all)
        % % Original model (no perturbations)
        % % Create distance–based dispersal matrix
        D = make_distance_based_dispersal(tot_ponds, alpha, r_all(rr));
        % Make diagonal block dispersal matrix
        M = kron(eye(size(sigma))–sigma, eye(size(D)) + kron(sigma, D);
        [end_A, end_B, end_M, end_sigma_j, end_u(:, rr, bb), end_u_x(:, rr, bb), C(:, rr, bb, sim_s), C_x(:, rr, bb, sim_s), patch_Ox(:, rr, bb), patch_Ex(:, rr, bb), patch_Ix(:, rr, bb)] = ...
            model_projection(tot_ponds, num_stages, u_loc_end, V, sigma_max, Beta_all(bb), ...
            DD_vars, mean_parms, kappa_x, T, M, D);
        end_U(rr, bb) = sum(end_u(:, rr, bb), 1);
    end
```matlab
%% Calculate d/du
F = mean_parms(1)*mean_parms(9);
I_s = eye(tot_ponds*num_stages);

%% Calculate dA/du
%% Calculate derivatives for each entry in pop vector
u_index = 1;
for kk = 1:tots_ponds
    for yy = 1:num_stages
        % Make diagonal block dispersal matrix
        dsigma = zeros(1,tot_ponds);
        dsigma(kk) = sigma_max(1,1)*Beta_all(bb)*exp(Beta_all(bb).*
            (kappa_x(kk)'-end_u_x(kk,rr,bb))/(1+exp(Beta_all(bb).*
            (kappa_x(kk)
            '-end_u_x(kk,rr,bb)))).^2;
        dM_du = zeros(tot_ponds*num_stages,tot_ponds*num_stages);
        dB_du = zeros(tot_ponds*num_stages);
        if yy ==3
            % num adults in previous time step
            a_t = end_u(num_stages*kk,rr,bb);
            dH = -Sx(kk).*gamma_x(kk).*F./(Kx(kk).*((1+a_t'*F./Kx(kk)).
                (gamma_x(kk)+1)));
            dB_du(num_stages*(kk-1)+1,num_stages*kk) = F*mean_parms(2)*dH;
        end
        dA_du = V'*dM_du*V*end_B +V'*end_M*V*dB_du;
        vecdA_du(:,u_index) = dA_du(:,);
        dA_du_all(:,u_index) = dA_du;
        u_index = u_index+1;
    end
end

%% d/dL_x
for LL = 1:tots_ponds
    dB_dLx = zeros(tot_ponds*num_stages);
    % num egg masses in previous time step
    a_t = end_u(num_stages*LL,rr,bb);
    dH = 1./((1+a_t'*F./Kx(LL)).
        (gamma_x(LL)));
    dB_dLx(num_stages*(LL-1)+1,num_stages*LL) = F*mean_parms(2)*dH;
    dA_dLx = V'*end_M*V*dB_dLx;
end```

\[
\begin{align*}
du_{dLx}(:,LL,rr,bb) &= ((I_s-end_A-kron(end_u( :, rr , bb)' , I_s)*\vec{A}du_dLx( : )) -1) ; \\
&* (kron(end_u( :, rr , bb)' , I_s)*dA_dLx( : )); \\
dUx_dLx(:,LL,rr,bb) &= \text{sum(reshape(du_dLx(:,LL,rr,bb),num}\_\text{stages},tot\_ponds }) ; \\
dU_dLx(LL,rr,bb) &= \text{sum(dUx_dLx(:,LL,rr,bb))} ; \\
 \% \% \text{Calculate perturbations of } \mathbf{C}_x \\
dCx_dLx(:,LL,rr,bb) &= \text{calculate_dCx(tot\_ponds,num}\_\text{stages},dA_dLx,dA_{du}\_\text{all} \\
&\quad, end_u( :, rr , bb), \ldots \end{align*}
\]

%% Elasticities
\[
\begin{align*}
du_{dLx}( :,LL,rr,bb) &= Sx(LL) .* end_u(:,rr,bb) .* du_dLx(:,LL,rr,bb) ; \\
dUx_dLx(:,LL,rr,bb) &= Sx(LL) .* end_u(:,rr,bb) .* dUx_dLx(:,LL,rr,bb) ; \\
dU_dLx(LL,rr,bb) &= Sx(LL) .* end_u(:,rr,bb) .* dU_dLx(LL,rr,bb) ; \\
dCx_dLx(:,LL,rr,bb) &= Sx(LL) .* C_x(:,rr,bb,\text{sim}) .* dCx_dLx(:,LL,rr,bb) ;
\end{align*}
\]

%% \text{d/dK}_x

\begin{verbatim}
for KK = 1:tot\_ponds \\
   dB_dKx = zeros(tot\_ponds*num\_stages) ; \\
   \% num egg masses in previous time step \\
   a\_t = end_u(num\_stages*KK,rr,bb) ; \\
   dH = (gamma_x(KK)*Sx(KK)*F*\text{a\_t} ./ (Kx(KK)^2*(1 + a\_t.*F./Kx(KK))) ./ ( gamma_x(KK)+1)) ; \\
   dB_dKx(num\_stages*(KK-1)+1,num\_stages*KK) = F*mean\_parms(2)*dH ; \\
   dA_dKx = V\'*end_M*Vs dB_dKx ; \\
   du_dKx(:,KK,rr,bb) = ((I_s-end_A-kron(end_u( :, rr , bb)' , I_s)*\vec{A}du) -1) ; \\
&* (kron(end_u( :, rr , bb)' , I_s)*dA_dKx( : )); \\
   dUx_dKx(:,KK,rr,bb) &= \text{sum(reshape(du_dKx(:,KK,rr,bb),num}\_\text{stages},tot\_ponds }) ; \\
   dU_dKx(KK,rr,bb) &= \text{sum(dUx_dKx(:,KK,rr,bb))} ; \\
 \% \% \text{Calculate perturbations of } \mathbf{C}_x \\
dCx_dKx(:,KK,rr,bb) &= \text{calculate_dCx(tot\_ponds,num}\_\text{stages},dA_dKx,dA_{du}\_\text{all} \\
&\quad, end_u( :, rr , bb), \ldots \end{verbatim
% dUx dKx ( : , KK, rr , bb ) = Kx(KK) . / end_u_x ( : , rr , bb ) . * dUx x dKx ( : , KK, rr , bb ) ;
% dU x dKx(KK, rr , bb ) = Kx(KK) . / end_U ( rr , bb ) . * dU x dKx(KK, rr , bb ) ;
% dCx x dKx ( : , KK, rr , bb ) = Kx(KK) . / C_x ( : , rr , bb , sims ) . * dCx x dKx ( : , KK, rr , bb ) ;
end

% % d/dgamma x
for GG = 1 : tot_ponds
    dB x dgamma = zeros( tot_ponds * num_ stages ) ;
    % num egg masses in previous time step
    a_t = end_u ( num_ stages * GG, rr , bb ) ;
    dH = -(Sx(GG) * log(1 + a_t . * F ./ Kx(GG))) ./ ((1 + a_t . * F ./ Kx(GG)) . ^ (gamma_x(GG))) ;
    dB x dgamma(num_ stages + (GG - 1) + 1) = dH ;
    dA x dgamma = V . * end_MS . * dB x dgamma ;
    du x dgamma( : , GG, rr , bb ) = ( ( I_s = end_A - kron( end_u ( : , rr , bb ) . ^ ' , I_s ) )... * vecdA . du ) ( -1 ) * ( kron( end_u ( : , rr , bb ) . ^ ' , I_s ) . * dA x dgamma ) ;
    dU x dgamma( : , GG, rr , bb ) = sum( reshape( du x dgamma( : , GG, rr , bb ) . num_ stages , tot_ponds ) ) ;
    dU x dgamma(GG, rr , bb ) = sum( dU x dgamma( : , GG, rr , bb ) ) ;
% % Calculate perturbations of C_x
    dCx x dgamma( : , GG, rr , bb ) = calculate dCx( tot_ponds , num_ stages , dA x dgamma ,....
        dA . du_all , end_u ( : , rr , bb ) ,...
        end_u_x ( : , rr , bb ) , du x dgamma( : , GG, rr , bb ) , dU x dgamma( : , GG, rr , bb ) , end_A ,
        C_x( : , rr , bb , sims ) ) ;
end
% Elasticities
    du x dgamma( : , GG, rr , bb ) = gamma_x(GG) ./ end_u ( : , rr , bb ) . * du x dgamma( : , GG, rr , bb ) ;
    dU x dgamma( : , GG, rr , bb ) = gamma_x(GG) ./ end_u_x ( : , rr , bb ) . * dU x dgamma( : , GG, rr , bb ) ;
    dU x dgamma(GG, rr , bb ) = gamma_x(GG) ./ end_U ( rr , bb ) . * dU x dgamma(GG, rr , bb ) ;
    dCx x dgamma( : , GG, rr , bb ) = gamma_x(GG) ./ C_x( : , rr , bb , sims ) . * dCx x dgamma( : , GG, rr , bb ) ;
end
end

% % Vital rates sensitivities
% % % Plot U sensitivities
figure
\[ dU_{all} = \text{zeros}(3,7,2,3); \]
\[ dU_{all}(1,:,:,:) = dU_{dLx}; \]
\[ dU_{all}(2,:,:,:) = dU_{dKx}; \]
\[ dU_{all}(3,:,:,:) = dU_{dgamma}; \]
\[ dU_{all}_{pos} = dU_{all}; \]
\[ dU_{all}_{pos}(dU_{all}_{pos} < 0) = 0; \]
\[ dU_{all}_{neg} = dU_{all}; \]
\[ dU_{all}_{neg}(dU_{all}_{neg} > 0) = 0; \]

\[ ba = \text{bar}([\text{squeeze}(dU_{all}_{pos}(1,:,:)) ; \text{zeros}(1,7); \text{squeeze}(dU_{all}_{pos}(1,:,:)) ... \text{zeros}(1,7); \text{squeeze}(dU_{all}_{pos}(1,:,:)) ; \text{zeros}(1,7); \text{squeeze}(dU_{all}_{pos}(2,:,:)) ... \text{zeros}(1,7); \text{squeeze}(dU_{all}_{neg}(1,:,:)) ; \text{zeros}(1,7); \text{squeeze}(dU_{all}_{neg}(1,:,:))], 'stacked', 'FaceColor', 'flat'); \]

\[ \text{hold on} \]
\[ ba2 = \text{bar}([\text{squeeze}(dU_{all}_{neg}(1,:,:)) ; \text{zeros}(1,7); \text{squeeze}(dU_{all}_{neg}(1,:,:)) ... \text{zeros}(1,7); \text{squeeze}(dU_{all}_{neg}(1,:,:)) ; \text{zeros}(1,7); \text{squeeze}(dU_{all}_{neg}(1,:,:))], 'stacked', 'FaceColor', 'flat'); \]

\[ \text{ba(1).CData} = [0 0 0]; ba(2).CData = [0 0 0]; ba(3).CData = [0 0 0]; \]
\[ \text{ba(4).CData} = [0 0 1]; ba(5).CData = [0 0 1]; ba(6).CData = [1 0 0]; ba(7).CData = [1 0 0]; \]
\[ \text{ba2(1).CData} = [0 0 0]; ba2(2).CData = [0 0 0]; ba2(3).CData = [0 0 0]; \]
\[ \text{ba2(4).CData} = [0 0 1]; ba2(5).CData = [0 0 1]; ba2(6).CData = [1 0 0]; ba2(7).CData = [1 0 0]; \]
\[ \text{ba(2).FaceAlpha} = 0.7; ba(3).FaceAlpha = 0.5; ba(5).FaceAlpha = 0.5; ba(7).FaceAlpha = 0.5; \]
\[ \text{ba2(2).FaceAlpha} = 0.7; ba2(3).FaceAlpha = 0.5; ba2(5).FaceAlpha = 0.5; ba2(7).FaceAlpha = 0.5; \]
\[ \text{xticks(0:23)} \]
\[ \text{xticklabels}(['' '',''L_{x}'',''K_{x}'',''\gamma_{x}''',''L_{x}'',''K_{x}'',''\gamma_{x}''',''L_{x}'',''K_{x}'',''\gamma_{x}''',''L_{x}'',''K_{x}'',''\gamma_{x}''']) \]
\[ \text{xlabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{\beta = -0.2'' , 'r=50'' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
\[ \text{ylabel}(''r=50'' , 'r=50' , 'r=50' , 'r=1000' , 'r=1000'); \]
xlim([0 27])

legend({'\theta_{P1}', '\theta_{P2}', '\theta_{P3}', '\theta_{S1}', '\theta_{S2}', '\theta_{E1}', '\theta_{E2}'})

set(gcf, 'Position', [188 508 856 304]);
%
print(gcf,['7patch_figs/dUdtheta_',',sim_tits','=-depsc','-r600'])

%% Plot C_x sensitivities

\[ \text{dCx\_all} = \text{zeros}(3,7,7,2,3); \]
\[ \text{dCx\_all}(1,:,:,:) = \text{dCx\_dLx}; \]
\[ \text{dCx\_all}(2,:,:,:) = \text{dCx\_dKx}; \]
\[ \text{dCx\_all}(3,:,:,:) = \text{dCx\_dgamma}; \]
\[ \text{dCx\_all\_pos} = \text{dCx\_all}; \]
\[ \text{dCx\_all\_pos}(<0) = 0; \]
\[ \text{dCx\_all\_neg} = \text{dCx\_all}; \]
\[ \text{dCx\_all\_neg}(>0) = 0; \]
\[ \text{for } r2 = 1:2 \]
\[ \text{for } b2 = 1:3 \]
\[ \text{figure} \]
\[ \text{ba} = \text{bar}([\text{squeeze(dCx\_all\_pos}(1,:,:,:r2,b2));\text{zeros}(1,7);\text{squeeze(dCx\_all\_pos}(2,:,:,:r2,b2))]); \]
\[ \text{hold on} \]
\[ \text{ba2} = \text{bar}([\text{squeeze(dCx\_all\_neg}(1,:,:,:r2,b2));\text{zeros}(1,7);\text{squeeze(dCx\_all\_neg}(2,:,:,:r2,b2))]); \]
\[ \text{ba}(1).\text{CData} = [0 0 0]; \text{ba}(2).\text{CData} = [0 0 0]; \text{ba}(3).\text{CData} = [0 0 0]; \]
\[ \text{ba}(4).\text{CData} = [0 0 1]; \text{ba}(5).\text{CData} = [0 0 1]; \text{ba}(6).\text{CData} = [1 0 0]; \text{ba}(7).\text{CData} = [1 0 0]; \]
\[ \text{ba}(1).\text{FaceAlpha} = 0.7; \text{ba}(2).\text{FaceAlpha} = 0.5; \text{ba}(3).\text{FaceAlpha} = 0.5; \text{ba}(4).\text{FaceAlpha} = 0.5; \]
\[ \text{xticks}(0:23) \]
\[ \text{xticklabels}({'','L_{P1}','L_{P2}','L_{P3}','L_{S1}','L_{S2}','L_{E1}','L_{E2}','K_{P1}','K_{P2}','K_{P3}','K_{S1}','K_{S2}','K_{E1}','K_{E2}'}); \]
\gamma\{P_1\}', '\gamma\{P_2\}', '\gamma\{P_3\}', '\gamma\{S_1\}', '\gamma\{S_2\}', '\gamma\{E_1\}', '\gamma\{E_2\}'}
ylabel({['$r=r$' num2str(r1); ' $\beta=\beta$' num2str(Beta);]...'
' $\frac{\theta_x}{C_x^*}$; $\frac{dC_x^*}{d\theta_x}$})
xlim([0 27])
ylim([-1.5 1.5])
set(gcf,'Position',[188 508 856 304]);
legend({'C\{P_1\}', 'C\{P_2\}', 'C\{P_3\}', 'C\{S_1\}', 'C\{S_2\}', 'C\{E_1\}', 'C\{E_2\}'}
xlabel(sim titles)

%% Plot U_x sensitivities
dUx all = zeros(3,7,7,2,3);
dUx all(1,:) = dUx dLx;
dUx all(2,:) = dUx dKx;
dUx all(3,:) = dUx dgamma;
dUx all pos = dUx all;
dUx all pos(dUx all pos<0) = 0;
dUx all neg = dUx all;
dUx all neg (dUx all neg>0) = 0;
for r2 = 1:2
for b2 = 1:3
figure
ba = bar([squeeze(dUx all pos(1,:,:r2,b2)); zeros(1,7); squeeze(dUx all pos(2,:,:r2, b2))]),
' stacked ', ' FaceColor ', ' flat ');
hold on
ba2 = bar([squeeze(dUx all neg(1,:,:r2,b2)); zeros(1,7); squeeze(dUx all neg(2,:,:r2, b2))],
' stacked ', ' FaceColor ', ' flat ');
ba(1).CData = [0 0 0]; ba(2).CData = [0 0 0]; ba(3).CData = [0 0 0];
b(4).CData = [0 0 1]; ba(5).CData = [0 0 1]; ba(6).CData = [1 0 0]; ba(7).CData = [1 0 0];
b2(1).CData = [0 0 0]; ba2(2).CData = [0 0 0]; ba2(3).CData = [0 0 0];
b2(4).CData = [0 0 1]; ba2(5).CData = [0 0 1]; ba2(6).CData = [1 0 0]; ba2(7).CData = [1 0 0];
ba(2).FaceAlpha = 0.7; ba(3).FaceAlpha = 0.5; ba(5).FaceAlpha = 0.5; ba(7).FaceAlpha = 0.5;
ba2(2).FaceAlpha = 0.7; ba2(3).FaceAlpha = 0.5; ba2(5).FaceAlpha = 0.5; ba2(7).FaceAlpha = 0.5;
xticks(0:23)
xticklabels({'','L_{P1}','L_{P2}','L_{P3}','L_{S1}','L_{S2}','L_{E1}','L_{E2}'
               ',...'
               'K_{P1}','K_{P2}','K_{P3}','K_{S1}','K_{S2}','K_{E1}','K_{E2}'
               ',...
               '\gamma_{P1}','\gamma_{P2}','\gamma_{P3}','\gamma_{S1}','\gamma_{S2}'
               ',...'
gamma_{E1}','\gamma_{E2}'
               )
ylabel({'[$8r=$8',num2str(r2),'$],' '$8\beta=$8',num2str(Beta_all(b2))';...'
               '$8\frac{\theta_x}{U_x^*}\frac{dU_x^*}{d\theta_x}$','Interpreter','Latex'}
xlim([0 27])
ylim([-5 3])
set(gcf,'Position',[188 508 1051 304]);
legend({'U_{P1}','U_{P2}','U_{P3}','U_{S1}','U_{S2}','U_{E1}','U_{E2}'})
xlabel(sim_tits)

% % print(gcf,'[7 patch figs/dU_x',num2str(r2),'b',num2str(b2),'dtheta_','sim_tits]','-
% depsc',''-r600')
end
end

% accumulate all metrics
dUx_dLx_all(:,:,,:,:,:,sims) = dUx_dLx;
dCx_dLx_all(:,:,,:,:,:,sims) = dCx_dLx;
dU_dLx_all(:,:,,:,:,:,sims) = dU_dLx;
dUx_dKx_all(:,:,,:,:,:,sims) = dUx_dKx;
dCx_dKx_all(:,:,,:,:,:,sims) = dCx_dKx;
dU_dKx_all(:,:,,:,:,:,sims) = dU_dKx;
dUx_dgamma_all(:,:,,:,:,:,sims) = dUx_dgamma;
dCx_dgamma_all(:, :, :, :, :, sims) = dCx_dgamma;
dU_dgamma_all(:, :, :, :, :, sims) = dU_dgamma;

close all
end

%% Plot Summary Results
% % Plot U sensitivities
for ee = 1:2
figure

\texttt{dU\_all} = \texttt{zeros(3,7,2,3,1)}; \\
\texttt{dU\_all(1,:,:,:,:) = squeeze(dU\_dLx\_all,:,:,:,:)}; \\
\texttt{dU\_all(2,:,:,:,:) = squeeze(dU\_dKx\_all,:,:,:,:)}; \\
\texttt{dU\_all(3,:,:,:,:) = squeeze(dU\_dgamma\_all,:,:,:,:)}; \\
\texttt{dU\_all = reshape(dU\_all,3,7,6)}; \\
\texttt{dU\_all\_mean = squeeze(mean(dU\_all,3))}; \\
\texttt{dU\_all\_pos = dU\_all\_mean}; \\
\texttt{dU\_all\_pos(dU\_all\_mean<0) = 0}; \\
\texttt{dU\_all\_neg = dU\_all\_mean}; \\
\texttt{dU\_all\_neg(dU\_all\_mean>0) = 0}; \\
\texttt{dU\_all\_std = squeeze(std(dU\_all,[],3))}; \\
\texttt{dU\_all\_std\_pos = dU\_all\_std}; \\
\texttt{dU\_all\_std\_pos(dU\_all\_mean<0) = 0}; \\
\texttt{dU\_all\_std\_neg = dU\_all\_std}; \\
\texttt{dU\_all\_std\_neg(dU\_all\_mean>0) = 0}; \\
\texttt{dU\_std\_pos = [squeeze(dU\_all\_std\_pos(1,:,:)),[0 0],squeeze(dU\_all\_std\_pos(2,:,:)),[0 0],} \\
\texttt{squeeze(dU\_all\_std\_pos(3,:,:))];} \\
\texttt{dU\_std\_neg = [squeeze(dU\_all\_std\_neg(1,:,:)),[0 0],squeeze(dU\_all\_std\_neg(2,:,:)),[0 0],} \\
\texttt{squeeze(dU\_all\_std\_neg(3,:,:))];} \\
\texttt{ba = bar(dU\_all\_pos,'FaceColor','flat');} \\
\texttt{hold on} \\
\texttt{eba = errorbar(.6667:.1111:3.3333,[squeeze(dU\_all\_pos(1,:,:)),[0 0],squeeze(dU\_all\_pos} \\
\texttt{(2,:,:)),[0 0],squeeze(dU\_all\_pos(3,:,:))],dU\_std\_pos,dU\_std\_pos);} \\
\texttt{eba\_Color = [0 0 0];} \\
\texttt{eba\_LineStyle = 'none';} \\
\texttt{ba2 = bar(dU\_all\_neg,'FaceColor','flat');} \\
\texttt{eba2 = errorbar(.6667:.1111:3.3333,[squeeze(dU\_all\_neg(1,:,:)),[0 0],squeeze(dU\_all\_neg} \\
\texttt{(2,:,:)),[0 0],squeeze(dU\_all\_neg(3,:,:))],dU\_std\_neg,dU\_std\_neg);} \\
\texttt{eba2\_Color = [0 0 0];} \\
\texttt{eba2\_LineStyle = 'none';} \\
\texttt{ba(1)\_CData = [0 0 0];ba(2)\_CData = [0 0 0];ba(3)\_CData = [0 0 0];} \\
\texttt{ba(4)\_CData = [0 0 1];ba(5)\_CData = [0 0 1];ba(6)\_CData = [1 0 0];ba(7)\_CData = [1 0 0];} \\
\texttt{ba2(1)\_CData = [0 0 0];ba2(2)\_CData = [0 0 0];ba2(3)\_CData = [0 0 0];} \\
\texttt{ba2(4)\_CData = [0 0 1];ba2(5)\_CData = [0 0 1];ba2(6)\_CData = [1 0 0];ba2(7)\_CData = [1 0 0];} \\
\texttt{ba(2)\_FaceAlpha = 0.7;ba(3)\_FaceAlpha = 0.5;ba(5)\_FaceAlpha = 0.5;ba(7)\_FaceAlpha = 0.5;} \\
\texttt{ba2(2)\_FaceAlpha = 0.7;ba2(3)\_FaceAlpha = 0.5;ba2(5)\_FaceAlpha = 0.5;ba2(7)\_FaceAlpha = 0.5;}
```
xticks(0:3)
xticklabels({'','L_x','K_x','\gamma_x'})
ylabel('$$\frac{\theta_x}{U^*}\frac{dU^*}{d\theta_x}$$','Interpreter','Latex')
xlim([0.5 4.2])
ylim([-0.8 .5])
set(gcf,'Position',[993 539 508 258]);
set(gca,'Fontsize',12)
legend({'\theta_{P_1}','\theta_{P_2}','\theta_{P_3}','\theta_{S_1}','\theta_{S_2}',
'\theta_{E_1}','\theta_{E_2}'],'fontsize',7)
if ee == 1
annotation(gcf,'textbox',...,[0.150383838383838 0.862995169082125 0.0495050505050505 0.0772946859903378],...,'String','A',...,'LineStyle','none',...,'FontWeight','bold',...,'FontSize',15,...,'FitBoxToText','off');
xlabel({'$$\theta_x$$';'Full'},'Interpreter','Latex')
end
else
annotation(gcf,'textbox',...,[0.150383838383838 0.862995169082125 0.0495050505050505 0.0772946859903378],...,'String','B',...,'LineStyle','none',...,'FontWeight','bold',...,'FontSize',15,...,'FitBoxToText','off');
xlabel({'$$\theta_x$$';'Partial'},'Interpreter','Latex')
end
end

% Plot all Cx sensitivities
dCx_all = zeros(3,7,7,2,3,2);
dCx_all(1,:,:,:,:,:) = dCx_dLx_all;
dCx_all(2,:,:,:,:,:) = dCx_dKx_all;
dCx_all(3,:,:,:,:,:) = dCx_dgamma_all;
title_labels = 'ABC';
for b3 = 1:3
figure
```
\documentclass{article}
\usepackage{amsmath}
\begin{document}
\begin{verbatim}
\dC_{\beta} = \text{squeeze}(\dC_{\text{all}}(:,:,\,:,:,\b3,:))
\dC_{\beta} = \text{reshape}(\dC_{\beta},3,7,7,4);
\dC_{\beta}\text{mean} = \text{squeeze}(\text{mean}(\dC_{\beta},4))
\dC_{\beta}\text{std} = \text{squeeze}(\text{std}(\dC_{\beta},[],4))
for \text{cc} = 1:\text{tot\_ponds}
    \text{plot}(1, \text{squeeze}(\dC_{\beta}\text{mean}(1,\text{cc},1)), \text{patch\_symbols}(\text{cc,:}), '\text{markersize}', 10, '\text{linewidtth}', 1.5)
    \text{hold on}
end
\text{plot}([0 25],[0 0], 'k--')
for \text{cc} = 1:\text{tot\_ponds}
    \text{ebl}=\text{errorbar}(1:7, \text{squeeze}(\dC_{\beta}\text{mean}(1,\text{cc,:})), \text{squeeze}(\dC_{\beta}\text{std}(1,\text{cc,:})))
    \text{ebl}\text{.Color} = [0 0 0];
    \text{ebl}\text{.LineStyle} = '\text{none}';
    \text{ebl}=\text{errorbar}(9:15, \text{squeeze}(\dC_{\beta}\text{mean}(2,\text{cc,:})), \text{squeeze}(\dC_{\beta}\text{std}(2,\text{cc,:})))
    \text{ebl}\text{.Color} = [0 0 0];
    \text{ebl}\text{.LineStyle} = '\text{none}';
    \text{ebl}=\text{errorbar}(17:23, \text{squeeze}(\dC_{\beta}\text{mean}(3,\text{cc,:})), \text{squeeze}(\dC_{\beta}\text{std}(3,\text{cc,:})))
    \text{ebl}\text{.Color} = [0 0 0];
    \text{ebl}\text{.LineStyle} = '\text{none}';
    \text{plot}(1:7, \text{squeeze}(\dC_{\beta}\text{mean}(1,\text{cc,:})), \text{patch\_symbols}(\text{cc,:}), '\text{markersize}', 10, '\text{linewidtth}', 1.5)
end
\text{xticks}(0:23)
\text{xticklabels}({'}\text{"L\_P\_1"}', '{'}\text{"L\_P\_2"}', '{'}\text{"L\_P\_3"}', '{'}\text{"L\_S\_1"}', '{'}\text{"L\_S\_2"}', '{'}\text{"L\_E\_1"}', '{'}\text{"L\_E\_2"}', '{'}\text{"K\_P\_1"}', '{'}\text{"K\_P\_2"}', '{'}\text{"K\_P\_3"}', '{'}\text{"K\_S\_1"}', '{'}\text{"K\_S\_2"}', '{'}\text{"K\_E\_1"}', '{'}\text{"K\_E\_2"}', '{'}\text{"\gamma\_P\_1"}', '{'}\text{"\gamma\_P\_2"}', '{'}\text{"\gamma\_P\_3"}', '{'}\text{"\gamma\_S\_1"}', '{'}\text{"\gamma\_S\_2"}', '{'}\text{"\gamma\_E\_1"}', '{'}\text{"\gamma\_E\_2"}')
\text{ylabel}({'}\text{"beta = \text{\textbackslash{}beta}}"', num2str(Beta\_all(b3))), '{'}\text{"\frac{\text{\theta\_x}}{\text{C\_x}}"}', '{'}\text{"\frac{\text{dC\_x}}{\text{\textbackslash{}theta\_x\_x}}"}', '{'}\text{\textbackslash{}text{Interpreter}}, '{'}\text{"Latex"}')
\text{xlabel}({'}\text{"\textbackslash{}text{\theta\_x}"}', '{'}\text{"\textbackslash{}text{Interpreter}}, '{'}\text{"Latex"}')
\text{xlim}([0 27])
\end{verbatim}
\end{document}
```matlab
ylim([-1 1])
set(gcf, 'Position', [188 497 1280 315]);
set(gca, 'fontsize', 14)
annotation(gcf, 'textbox', ...
    [0.130383838383838 0.862995169082125 0.0495050505050505 0.0772946859903378], ...
    'String', num2str(title_labels(b3)), ...
    'LineStyle', 'none', ...
    'FontWeight', 'bold', ...
    'FontSize', 14, ...
    'FitBoxToText', 'off');
legend({'C_{P_1}' , 'C_{P_2}' , 'C_{P_3}' , 'C_{S_1}' , 'C_{S_2}' , 'C_{E_1}' , 'C_{E_2}' }, ...
    'fontsize', 9);

end

%% Plot all Ux sensitivities

```

```matlab
dUx_all = zeros(3,7,7,2,3,2);
dUx_all(:,:,1,:,:,:) = dUx_dLx_all;
dUx_all(:,:,2,:,:,:) = dUx_dKx_all;
dUx_all(:,:,3,:,:,:) = dUx_dgamma_all;
for b3 = 1:3
    figure
    dUx_beta = squeeze(dUx_all(:,:,b3,:));
dUx_beta = reshape(dUx_beta,3,7,7,4);
dUx_beta_mean = squeeze(mean(dUx_beta,4));
dUx_beta_std = squeeze(std(dUx_beta,[]),4));
for cc = 1:tot_ponds
    plot(1, squeeze(dUx_beta_mean(1,cc,1)),patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
    hold on
end
plot([0 25],[0 0], 'k--')
for cc = 1:tot_ponds
    eb1=errorbar(1:7, squeeze(dUx_beta_mean(1,cc,:)), squeeze(dUx_beta_std(1,cc,:)));
    eb1.LineStyle = 'none';
    eb1=errorbar(9:15, squeeze(dUx_beta_mean(2,cc,:)), squeeze(dUx_beta_std(2,cc,:)));
    eb1.LineStyle = 'none';
end
```
eb1.LineStyle = 'none';
eb1=errorbar(17:23, squeeze(dUx_beta_mean(3,cc,:)), squeeze(dUx_beta_std(3,cc,:)) , squeeze(dUx_beta_std(3,cc,:)));
eb1.Color = [0 0 0];
eb1.LineStyle = 'none';
plot(1:7, squeeze(dUx_beta_mean(1,cc,:)), patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
plot(9:15, squeeze(dUx_beta_mean(2,cc,:)), patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
plot(17:23, squeeze(dUx_beta_mean(3,cc,:)), patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
end
xticks(0:23)
xticklabels({'', 'L_{p1}', 'L_{p2}', 'L_{p3}', 'L_{s1}', 'L_{s2}', 'L_{e1}', 'L_{e2}' , '', ...
'K_{p1}', 'K_{p2}', 'K_{p3}', 'K_{s1}', 'K_{s2}', 'K_{e1}', 'K_{e2}' , '' , ...
'\gamma_{p1}', '\gamma_{p2}', '\gamma_{p3}', '\gamma_{s1}', '\gamma_{s2}' , '' , ...
\gamma_{e1}', '\gamma_{e2}'})
ylabel([['$\beta = \beta$', num2str(Beta_all(b3))]], ['$\frac{\theta{x}}{U_{x}^{*}}$']$rac{dU_{x}^{*}}{d\theta{x}}$]
xlabel(['$\theta{x}$'], 'Interpreter', 'Latex')
xlim([0 27])
ylim([-3 3])
set(gcf, 'Position', [188 497 1280 315]);
set(gca, 'fontsize', 14)
annotation(gcf, 'textbox', ...[0.13038383838383 0.862995169082125 0.0495050505050505 0.0772946859903378] , ...
'String', num2str(title_labels(b3)), ...'
LineStyle', 'none', ...'
FontWeight', 'bold', ...'
FontSize', 14, ...'
FitBoxToText', 'off');
legend({'U_{p1}', 'U_{p2}', 'U_{p3}', 'U_{s1}', 'U_{s2}', 'U_{e1}', 'U_{e2}'}, ...'
fontsize', 9)
% % print(gcf,['7patch_figs/dUxdtheta_all_', num2str(b3)], '-depsc', '-r600')
end

% % Plot all Cx sensitivities
dCx_all = zeros(3,7,7,2,3,2);
dCx_all(1,:,:,:,:,:) = dCx_dLx_all;
\[ dC_x_{\text{all}}(2,:) = dC_x dK_x_{\text{all}}; \]
\[ dC_x_{\text{all}}(3,:) = dC_x d\gamma_{\text{all}}; \]

\textbf{figure}

\textbf{for} \( b3 = 1:3 \)
\[ dC_x_{\beta} = \text{squeeze} \left( dC_x_{\text{all}}(:,b3,:) \right); \]
\[ dC_x_{\beta} = \text{reshape} \left( dC_x_{\beta},3,7,7,4 \right); \]
\[ dC_x_{\beta \text{ mean}} = \text{squeeze} \left( \text{mean} \left( dC_x_{\beta},4 \right) \right); \]
\[ dC_x_{\beta \text{ std}} = \text{squeeze} \left( \text{std} \left( dC_x_{\beta},[],4 \right) \right); \]

\textbf{for} \( cc = 1:\text{tot} \_\text{ponds} \)
\[ \text{plot} \left(0, \text{squeeze} \left( dC_x_{\beta \text{ mean}}(1,cc,1) \right), \text{patch} \_\text{symbols}(cc,:), \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{hold on} \]
\[ \text{end} \]

\[ \text{plot} \left(0,0, 'k', \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left([0 25],[0 0], 'k--' \right) \]
\[ \text{plot} \left([1.5 1.5],[-2 1.5], 'k-', \text{linewidth}', 3 \right) \]
\[ \text{plot} \left([2.5 2.5],[-2 1.5], 'k-', \text{linewidth}', 3 \right) \]

\textbf{for} \( cc = 1:\text{tot} \_\text{ponds} \)
\[ \text{perturbed} \_\text{pond} = \text{squeeze} \left( dC_x_{\beta \text{ mean}}(:,cc,cc) \right); \]
\[ \text{all} \_\text{others} = \text{squeeze} \left( dC_x_{\beta \text{ mean}}(:,cc,:) \right); \]
\[ \text{all} \_\text{others}(:,cc) = []; \]
\[ \text{plot} \left(b3=0.35, \text{perturbed} \_\text{pond}(1), \text{patch} \_\text{symbols}(cc,:), \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left(b3=0.25, \text{all} \_\text{others}(1,:), 'k', \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left(b3=0.05, \text{perturbed} \_\text{pond}(2), \text{patch} \_\text{symbols}(cc,:), \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left(b3+0.05, \text{all} \_\text{others}(2,:), 'k', \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left(b3+0.25, \text{perturbed} \_\text{pond}(3), \text{patch} \_\text{symbols}(cc,:), \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{plot} \left(b3+0.35, \text{all} \_\text{others}(3,:), 'k', \text{markersize}', 10, \text{linewidth}', 1.5 \right) \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{xticks}([.7 1 1.3 1.7 2.3 2.7 3 3.3]) \]
\[ \text{xticklabels}('L_x', 'K_x', '{\gamma}_x', 'L_x', 'K_x', '{\gamma}_x', 'L_x', 'K_x', '{\gamma}_x') \]
\[ \text{ylabel}('\frac{\theta_x}{C_x}\frac{dC_x}{d\theta_x}', 'Interpreter', 'Interpreter', 'Latex') \]
\[ \text{xlabel}('\beta = -0.2 \quad \beta = 0 \quad \beta = 0.2 \quad \beta = 0.4') \]
\[ \text{xlim}([0.5 4.3]) \]
\[ \text{ylim}([-1 0.5]) \]
```matlab
set(gcf,'Position', [188 485 856 330]);
annotation(gcf,'textbox', ...
    [0.130383838383838 0.862995169082125 0.0495050505050505 0.0772946859903378], ...
    'String','A',...
    'LineStyle','none',...
    'FontSize',16,....
    'FitBoxToText','off');
set(gca,'fontSize',16)
legend({'dC\{P_1\}/d\theta\{P_1\}', 'dC\{P_2\}/d\theta\{P_2\}', 'dC\{P_3\}/d\theta\{P_3\}', 'dC\{S_1\}/d\theta\{S_1\}', 'dC\{S_2\}/d\theta\{S_2\}', 'dC\{E_1\}/d\theta\{E_1\}', 'dC\{E_2\}/d\theta\{E_2\}','Indirect effects','fontSize',9)

%% Plot all Ux sensitivities
dUx_all = zeros(3,7,7,2,3,2);
dUx_all(1,:,:,:,:,:) = dUx_dLx_all;
dUx_all(2,:,:,:,:,:) = dUx_dKx_all;
dUx_all(3,:,:,:,:,:) = dUx_dgamma_all;
figure
for b3 = 1:3
    dUx_beta = squeeze(dUx_all(:,,:,:,:,:,b3,:,:));
dUx_beta = reshape(dUx_beta,3,7,7,4);
dsUx_beta_mean = squeeze(mean(dUx_beta,4));
dsUx_beta_std = squeeze(std(dUx_beta,[],4));
    for cc = 1:tot_ponds
        plot(0, squeeze(dUx_beta_mean(1,cc,1)),patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
    hold on
end
plot(0,0,'k', 'markersize', 10, 'linewidth', 1.5)
plot([0 25],[0 0], 'k--')
plot([1.5 1.5],[0 3], 'k--', 'linewidth',3)
plot([2.5 2.5],[-3 0], 'k--', 'linewidth',3)
for cc = 1:tot_ponds
    perturbed_pond = squeeze(dUx_beta_mean(:,cc,cc));
    all_others = squeeze(dUx_beta_mean(:,cc,:));
    all_others(:,cc) = [];
plot(b3-0.35, perturbed_pond(1),patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
```
```matlab
plot(b3-0.25, all_others(1,:), 'k.', 'markersize', 10, 'linewidth', 1.5)
plot(b3-0.05, perturbed_pond(2).patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
plot(b3+.05, all_others(2,:), 'k.', 'markersize', 10, 'linewidth', 1.5)
plot(b3+.25, perturbed_pond(3).patch_symbols(cc,:), 'markersize', 10, 'linewidth', 1.5)
plot(b3+.35, all_others(3,:), 'k.', 'markersize', 10, 'linewidth', 1.5)
end
end
xticks([.7 1 1.3 1.7 2 2.3 2.7 3 3.3])
xticklabels({{'L_x'}, {'K_x'}, '\gamma_x', 'L_x', 'K_x', 'L_x', 'K_x', 'L_x', 'K_x', 'L_x', 'K_x'}); gamma_x
ylabel({'$\frac{dU_x(\theta)}{d\theta}$', 'Interpreter','Latex'}); $\frac{dU_x(\theta)}{d\theta}$
xlabel('beta = -0.2 \quad quad \quad beta = 0 \quad quad \quad beta = 0.2 \quad quad \quad quad$'); 'Interpreter','Latex')
xlim([0.5 4.3])
ylim([-4 2])
annotation(gcf,'textbox',... 
[0.130383838383838 0.862995169082125 0.0495050505050505 0.0772946859903378],....
'LineStyle','none',...
'FontSize',16,...
'FitBoxToText','off');
set(gca,'fontsize',16)
legend({'{du_{U(P_1)}~/d\theta}', '{du_{U(P_2)}~/d\theta}', '{du_{U(P_3)}~/d\theta}', '{du_{U(E_1)}~/d\theta}', '{du_{U(E_2)}~/d\theta}', 'Indirect effects'},'fontsize',9)
set(gcf,'Position', [188 469 856 330]);
%% print(gcf,'7patch_figs/dUxdttheta_all_summary','-depsc','-r600')

%% Determine best mgnt strategies for each scenario
C_x_reshape = reshape(C_x,7,12);
dCx_dgamma_reshape = reshape(dCx_dgamma_all,7,12);
for strat = 1:12
    for pond_type = 1:3
        if pond_type == 1
            [maxCx(pond_type),maxCx_ind(pond_type)] = max(C_x_reshape(1:3,strat));
        elseif pond_type == 2
```
\[
\begin{align*}
\text{max}_C \text{x(pond\_type)} & , \text{max}_C \text{x\_ind(pond\_type)} = \text{max}(C_x \text{\_reshape(4:5, strat)}); \\
\text{else} & \\
\text{max}_C \text{x(pond\_type)} & , \text{max}_C \text{x\_ind(pond\_type)} = \text{max}(C_x \text{\_reshape(6:7, strat)}); \\
\text{end} & \\
\text{end} & \\
\text{max}_C \text{x\_ind} = \text{max}_C \text{x\_ind} + [0, 3, 5]; \\
\% % \text{Identify optimal pond conservation} & \\
\text{degrade\_pond\_ind} = 1:7; & \\
\text{degrade\_pond\_ind}(\text{max}_C \text{x\_ind}) = []; & \\
\text{if sum( max}_C \text{x<0.999)==1 % Conserve only sink} & \\
\text{sink\_inds} = \text{find( max}_C \text{x<0.999)}; & \\
\text{max}_d\text{C}_x\_\text{ind}(\text{strat}) = \text{max}_C \text{x\_ind(sink\_inds)}; & \\
\text{elseif sum( max}_C \text{x<0.999)>1 % Degrade pond to increase sink contributions} & \\
\text{sink\_inds} = \text{find( max}_C \text{x<0.999)}; & \\
[\text{max}_d\text{C}_x, \text{max}_d\text{C}_x\_\text{ind}(\text{strat})] = \text{max}(\sum(d\text{C}_x \text{\_dgamma\_reshape...}) & \\
(\text{max}_C \text{x\_ind(sink\_inds)}, \text{degrade\_pond\_ind, strat}), 1)); & \\
\text{max}_d\text{C}_x\_\text{ind}(\text{strat}) = -\text{degrade\_pond\_ind(max}_d\text{C}_x\_\text{ind(strat)}); & \\
\text{else % Do nothing} & \\
\text{max}_d\text{C}_x\_\text{ind(strat)} = 0; & \\
\text{end} & \\
\text{end} & \\
\text{end} & 
\end{align*}
\]
function PRCC_model
% PRCC analysis for the density-dependent model
% Set seed
rng ('default')

%%% Initialization

% Define parameters
[mean_parms] = define_parms;
num_stages = 3;
tot_ponds = 7;

%%% Larval survival parameters
kappa_x = [50 50 50 40 40 30 30];

%%% Constant life stage dispersal
sigma_j = 0.25; % juvenile dispersal rate
sigma_s = 0; % subadult dispersal rate
sigma_f = 0; % adult dispersal rate

sigma = diag([sigma_j, sigma_s, sigma_f]); % disp rate matrix
sigma_max = 0.5*diag(ones(num_stages,1));

%%% Weights for a fully-connected system
alpha_full = ones(tot_ponds, tot_ponds);

%%% Weights for a fragmented system
alpha_fragment = [1 1 0 0 0 0 0; 1 1 0 1 1 1 1; 0 0 1 1 0 0 0; 0 1 1 1 0 0 0; ...
                 0 1 0 0 1 1 0; 0 1 0 0 1 1 0; 0 1 0 0 0 0 1];

% Create vec-permutation matrix
V = zeros(tot_ponds*num_stages);
I = eye(tot_ponds*num_stages);
k = 1;
for i=1:num_stages
    for j=i:num_stages:tot_ponds*num_stages
        V(k,:) = I(j,:);
        k = k+1;
    end
end
%% Initial conditions
IC_all = [20 20 20];
u_loc = repmat(IC_all, tot_ponds, 1);
T = 2000;

%% max and min vectors for dispersal probability weights
N = 500; % size of LHS
min_parms = [repmat(.01,1,7) repmat(10,1,7) repmat(0.01,1,7) 50 -0.2];
max_parms = [repmat(.3,1,7) repmat(1000,1,7) repmat(100,1,7) 1000 0.2];

%% PRCC
% Create LHS matrix
for yy = 1:length(min_parms)
    % uniform distribution (uses p_min and p_max)
    parm_samps = linspace(min_parms(yy), max_parms(yy), N);
    parm_samps = parm_samps(randperm(N)); % shuffles all values
    all_parms(:,yy) = parm_samps';
end

% Loop through both the full and partial systems
for sims = 1:2
    if sims == 1
        alpha = alpha_full;
        sim_tits = 'Full';
    else
        alpha = alpha_fragment;
        sim_tits = 'Partial';
    end

% Work through all sample combinations
for tt = 1:N
    % Larval survival parameters
    Sx = all_parms(tt,1:7);
    Kx = all_parms(tt,8:14);
    gamma_x = all_parms(tt,15:21);
    DD_vars = [Sx; Kx; gamma_x];
    r = all_parms(tt,22);
    beta = all_parms(tt,23);
```
%% Run model
%% Create distance-based dispersal matrix
D = make_distance_based_dispersal(tot_ponds, alpha, r);
%% Make diagonal block dispersal matrix
M = kron(eye(size(sigma))-sigma,eye(size(D)))+kron(sigma,D);
[end_A,end_B,end_M,end_sigma_i,end_u(:,tt,sims),end_u_x(:,tt,sims),C(:,tt,sims),...]
  C_x(:,tt,sims),"",""]=...
model_projection(tot_ponds,num_stages,u_loc,V,sigma_max,beta,...)
DD_vars,mean_parms,kappa_x,T,M,D);
end_U(tt,sims)=sum(end_u(:,tt,sims),1);

end

%% Results

%% Calculate and plot PRCCs
U_PRCC = PRCC(all_parms, end_U(:,sims));
U_PRCC_pos = U_PRCC;
U_PRCC_pos(U_PRCC_pos<0)=0;
U_PRCC_neg = U_PRCC;
U_PRCC_neg(U_PRCC_neg>0)=0;
figure
ba = bar(reshape(U_PRCC_pos(1:21),7,3)', 'FaceColor','flat');
hold on
ba2 = bar(reshape(U_PRCC_neg(1:21),7,3)', 'FaceColor','flat');
bar(3.7,U_PRCC_pos(22),1, 'FaceColor','flat')
bar(4,U_PRCC_pos(23),1, 'FaceColor','flat')
ba(1).CData = [0 0 0];ba(2).CData = [0 0 0];ba(3).CData = [0 0 0];
ba(4).CData = [0 0 1];ba(5).CData = [0 0 1];ba(6).CData = [1 0 0];ba(7).CData = [1 0 0];
ba2(1).CData = [0 0 0];ba2(2).CData = [0 0 0];ba2(3).CData = [0 0 0];
ba2(4).CData = [0 0 1];ba2(5).CData = [0 0 1];ba2(6).CData = [1 0 0];ba2(7).CData = [1 0 0];
ba(2).FaceAlpha = 0.7;ba(3).FaceAlpha = 0.5;ba(5).FaceAlpha = 0.5;ba(7).FaceAlpha = 0.5;
ba2(2).FaceAlpha = 0.7;ba2(3).FaceAlpha = 0.5;ba2(5).FaceAlpha = 0.5;ba2(7).FaceAlpha = 0.5;
xticks([1:3,3.7,4])
xticklabels({'L_x','K_x','\gamma_x','r','\beta'})
ylabel({'$8\frac{\theta_x}{U^*}\frac{dU^*}{d\theta_x}$88'},'Interpreter','Latex')
```
xlabel(sim_tits)
legend({'\theta_{\{P_1\}}', '\theta_{\{P_2\}}', '\theta_{\{P_3\}}', '\theta_{\{S_1\}}',...
   '\theta_{\{S_2\}}', '\theta_{\{E_1\}}', '\theta_{\{E_2\}}', 'r', '\beta'})
xlim([0.5 4.5])
ylim([0.3 0.3])
set(gcf,'Position', [188 508 1051 304]);
legend({'U_{\{P_1\}}', 'U_{\{P_2\}}', 'U_{\{P_3\}}', 'U_{\{S_1\}}', 'U_{\{S_2\}}', 'U_{\{E_1\}}', 'U_{\{E_2\}}'})
print(gcf,['save_figs/U_PRCC',sim_tits],'-depsc','-r600')
for qq = 1:tot_ponds
    C_x_PRCC(:,qq) = PRCC(all_parms, squeeze(C_x(qq,:,sims))');
    u_x_PRCC(:,qq) = PRCC(all_parms, squeeze(end_u_x(qq,:,sims))');
end
C_x_PRCC_pos = C_x_PRCC;
C_x_PRCC_pos(C_x_PRCC_pos<0)=0;
C_x_PRCC_neg = C_x_PRCC;
C_x_PRCC_neg(C_x_PRCC_neg>0)=0;
figure
ba = bar([squeeze(C_x_PRCC_pos(1:7,:))'; zeros(1,7); squeeze(C_x_PRCC_pos(8:14,:))']...
   ; zeros(1,7); squeeze(C_x_PRCC_pos(15:21,:))'; zeros(1,7); squeeze(C_x_PRCC_pos
   (22:23,:))], 'stacked', 'FaceColor', 'flat');
hold on
ba2 = bar([squeeze(C_x_PRCC_neg(1:7,:))'; zeros(1,7); squeeze(C_x_PRCC_neg(8:14,:))']...
   ; zeros(1,7); squeeze(C_x_PRCC_neg(15:21,:))'; zeros(1,7); squeeze(C_x_PRCC_neg
   (22:23,:))], 'stacked', 'FaceColor', 'flat');
ba(1).Color = [0 0 0]; ba(2).Color = [0 0 0]; ba(3).Color = [0 0 0];
ba(4).Color = [0 0 1]; ba(5).Color = [0 0 1]; ba(6).Color = [1 0 0]; ba(7).Color = [1 0 0];
ba2(1).Color = [0 0 0]; ba2(2).Color = [0 0 0]; ba2(3).Color = [0 0 0];
ba2(4).Color = [0 0 1]; ba2(5).Color = [0 0 1]; ba2(6).Color = [1 0 0]; ba2(7).Color = [1 0 0];
ba(2).FaceAlpha = 0.7; ba(3).FaceAlpha = 0.5; ba(4).FaceAlpha = 0.5; ba(5).FaceAlpha = 0.5; ba(6).FaceAlpha = 0.5; ba(7).FaceAlpha = 0.5;
ba2(2).FaceAlpha = 0.7; ba2(3).FaceAlpha = 0.5; ba2(4).FaceAlpha = 0.5; ba2(5).FaceAlpha = 0.5; ba2(6).FaceAlpha = 0.5; ba2(7).FaceAlpha = 0.5;
xticks(0:26)
xticklabels({''',''L_{\{P_1\}}''',''L_{\{P_2\}}''',''L_{\{P_3\}}''',''L_{\{S_1\}}''',''L_{\{S_2\}}''',''L_{\{E_1\}}''',''L_{\{E_2\}}''
   '';''
   '',''K_{\{P_1\}}''',''K_{\{P_2\}}''',''K_{\{P_3\}}''',''K_{\{S_1\}}''',''K_{\{S_2\}}''',''K_{\{E_1\}}''',''K_{\{E_2\}}''
   '';...
ux_PRCC_pos = ux_PRCC;
ux_PRCC_pos(ux_PRCC_pos<0)=0;
ux_PRCC_neg = ux_PRCC;
ux_PRCC_neg(ux_PRCC_neg>0)=0;

figure
ba = bar ( squeeze(ux_PRCC_pos(1:7,:))' ; zeros(1,7) ; squeeze(ux_PRCC_pos(8:14,:))'...
    ; zeros(1,7) ; squeeze(ux_PRCC_pos(15:21,:))') ; zeros(1,7) ; squeeze(ux_PRCC_pos
    (22:23,:))', 'stacked' , 'FaceColor' , 'flat' );
hold on
ba2 = bar ( squeeze(ux_PRCC_neg(1:7,:))' ; zeros(1,7) ; squeeze(ux_PRCC_neg(8:14,:))'...
    ; zeros(1,7) ; squeeze(ux_PRCC_neg(15:21,:)) ) ; zeros(1,7) ; squeeze(ux_PRCC_neg
    (22:23,:))', 'stacked' , 'FaceColor' , 'flat' );
ba(1).CData = [0 0 0]; ba(2).CData = [0 0 0]; ba(3).CData = [0 0 0];
ba(4).CData = [0 0 1]; ba(5).CData = [0 0 1]; ba(6).CData = [1 0 0]; ba(7).CData = [1 0 0];
ba2(1).CData = [0 0 0]; ba2(2).CData = [0 0 0]; ba2(3).CData = [0 0 0];
ba2(4).CData = [0 0 1]; ba2(5).CData = [0 0 1]; ba2(6).CData = [1 0 0]; ba2(7).CData = [1 0 0];
ba(2).FaceAlpha = 0.7; ba(3).FaceAlpha = 0.5; ba(5).FaceAlpha = 0.5; ba(7).FaceAlpha = 0.5;
ba2(2).FaceAlpha = 0.7; ba2(3).FaceAlpha = 0.5; ba2(5).FaceAlpha = 0.5; ba2(7).FaceAlpha = 0.5;
xticks(0:26)
xticklabels({'','L_z(P_1)','L_z(P_2)','L_z(P_3)','L_z(S_1)','L_z(S_2)','#L_z(E_1)#','#L_z(E_2)#'
    ,'','
    '#K_z(P_1)#','#K_z(P_2)#','#K_z(P_3)#','#K_z(S_1)#','#K_z(S_2)#','#K_z(E_1)#','#K_z(E_2)#'
    ,'','
    '#\gamma_(P_1)#','#\gamma_(P_2)#','#\gamma_(P_3)#','#\gamma_(S_1)#','#\gamma_(S_2)#','#\gamma_(S_3)#'
    ', '#\gamma_(E_1)#','#\gamma_(E_2)#', ',',',',',',',',',',');'

ylabel({'$\frac{\theta_x(C,x^*)}{\theta_x}$','$\frac{dU_x(x^*)}{d\theta_x}$'},'Interpreter','Latex')
xlim([0 30])

ylim([-1.5 1.5])
set(gcf,'Position',[188 508 1051 304]);

legend({'U_z(P_1)', 'U_z(P_2)', 'U_z(P_3)', 'U_z(S_1)', 'U_z(S_2)', 'U_z(E_1)', 'U_z(E_2)'})
xlabel(sims_tits)
print(gcf,['save_figs/Cx.PRCC_','sims_tits'],'-depsec','-r600')
ylim([-1.5 1.5])
set(gca,'Position',[188 508 1051 304]);
legend({'U_{P1}','U_{P2}','U_{P3}','U_{S1}','U_{S2}','U_{E1}','U_{E2}'});
xlabel(sim_tits)

% % print(gcf,['save_figs/ux_PRCC-',sim_tits,'.depse','r600'])
end

% Plot PRCC for all patch distributions
figure
for pp = 1:7
  % Habitat PRCC
  w_PRCCs(pp,:) = PRCC(all_parms, sum(Y_PRCC_w(:,(pp-1)*3+1:pp*3),2));
  % percap contribution PRCC
  percap_PRCCs(pp,:) = PRCC(all_parms, Y_PRCC_percap_CR(:,pp));

  % plot bar graph of sensitivities
  subplot(totponds,1,pp)
  hB = bar(1:7,reshape(percap_PRCCs(pp,:),6,7)));
  hAx=gca; % get a variable for the current axes handle
  hT=[]; % placeholder for text object handles
  labels = zeros(totponds,1);
  for i=1:length(hB) % iterate over number of bar objects
    labels = labels + ones(totponds,1);
    labels(i) = labels(i)+1;
    hT=[hT text(hB(i).XData+hB(i).XOffset,hB(i).YData,num2str(labels),... 'VerticalAlignment','bottom','horizontalalign','center')];
  end
  if pp == 1
    title('Per Capita Patch Contribution PRCC')
  end
  ylabel('PRCC value')
end
xlabel('Source patch')
end

function PRCCs = PRCC parms_in, output)
% % Adapted from http://malthus.micro.med.umich.edu/lab/usadata/
% % by Simeone Marino, May 29 2007 % %

LHSmatrix = parms_in;
Y=output;  % Define the output. Comment out if the Y is already
% a subset of all the time points and it already comprises
% ONLY the s rows of interest
[~, k]=size(LHSmatrix);  % Define the size of LHS matrix
[~, out]=size(Y);

for i=1:k  % Loop for the whole submatrices
    c=['LHStemp=LHSmatrix;LHStemp(:,',num2str(i),')=[];Z',num2str(i),'=LHStemp;
        LHStemp=[];';]
    eval(c);
    % Loop to calculate PRCCs and significances
    c1=['[LHSmatrix(:,',num2str(i),'),Y];';
        c2=['Z',num2str(i)];
    [rho, p]=partialcorr(eval(c1),eval(c2),'type','Spearman');
    for j=1:out
        c3=['prcc_=',num2str(i),')(',num2str(j),')=rho(1,','num2str(j+1),');';
        c4=['prcc_sign_=',num2str(i),')(',num2str(j),')=p(1,','num2str(j+1),');';
        eval(c3);
        eval(c4);
    end
    c5=['clear Z',num2str(i),';';
        eval(c5);
end
prcc=[];
prcc_sign=[];

for i=1:k
    d1=['prcc=[prcc ; prcc_',num2str(i),'];';
        eval(d1);
    d2=['prcc_sign=[prcc_sign ; prcc_sign_',num2str(i),'];';
        eval(d2);
end

% [length(s) k out];
PRCCs=prcc';
uncorrected_sign=prcc_sign';
prcc=PRCCs;
sign=uncorrected_sign;
end
Curriculum Vitae

Guenchik ‘Guen’ Grosklos, Ph.D.
Mathematical Sciences Ph.D.
Utah State University
guen.grosklos@usu.edu
717-332-9812

Education
Ph.D., Mathematical Sciences - Utah State University 2015-2021
Research - Theoretical Ecology, Mathematical Modeling, Dynamical Systems, Interdisciplinary Sciences
Committee: Jia Zhao, Ph.D. (advisor), Michael Cortez, Ph.D., Joe Koebbe, Ph.D., Luis Gordillo, Ph.D., Noelle Beckman, Ph.D.

B.S., Applied Mathematics - Millersville University 2009-2013

Publications


3. Grosklos, G., and M. Cortez. 2020. Evolutionary and plastic phenotypic change can be just as fast as changes in population densities. The American Naturalist 97(1)


6. Theiler, J., and G. Grosklos. 2016. Problematic projection to the in-sample sub-
space for a kernelized anomaly detector. IEEE Geoscience and Remote Sensing Letters
13

sensing imagery. Proc. SPIE 9472

algorithms to find unique sample classes in ChemCam data from Curiosity’s first 90
Sols in Gale Crater, Mars. American Geophysical Union

Funding and Awards
USU Mathematics and Statistics Travel Award, SIAM NSS Conference Sep 2019
NIMBioS Network Theory Tutorial Travel Award Feb 2019
NSF Fellowship for Climate Adaptation Science Research 2018-2019
SIAM Student Travel Award, Life Sciences Conference (LS18) Aug 2018
USU Mathematics and Statistics Travel Award, Dynamical Systems Conference May 2017
Utah State University Award of Excellence 2015-2016
Board of Governors Science and Mathematics Scholarship 2009-2013
Research Experience

Graduate Research Assistant - Utah State University  
Summer 2020  
Mentor: Kezia Manlove, Ph.D.

- Designed and developed a mechanistic model that analyzes the underlying processes of disease spread between bighorn sheep populations in the intermountain west

Graduate Research Assistant - National Park Service (GrYN), Bozeman, MT  
Summer 2019  
Mentor: Andrew Ray, Ph.D.

- Analyzed the effects of dispersal on the population dynamics of a montane Columbia spotted frog metapopulation

Climate Adaptation Science Research Fellow - Utah State University  
2018-2019  
Mentors: Patrick Belmont, Ph.D., and Courtney Flint, Ph.D.

- Designed and developed interdisciplinary research on climate science and adaptation strategies for decision makers and project managers. Collaboration with a multidisciplinary team of graduate students and faculty, including wildlife ecologists, sociologists, and biologists

Graduate Research Assistant - National Park Service (GrYN), Bozeman, MT  
Summer 2018  
Mentor: Andrew Ray, Ph.D.

- Designed and developed a mechanistic model to understand the effects of dispersal on the population dynamics of a montane Columbia spotted frog metapopulation

Graduate Research Assistant - Los Alamos National Laboratory  
Summer 2017  
Mentor: Chonggang Xu, Ph.D.

- Collaborated with plant experts in the Earth and Environmental Sciences division to develop a functional integral projection model for the Ecosystem Demography Model

Graduate Research Assistant - Utah State University  
2016-2017  
Mentor: Michael Cortez, Ph.D.

- Developed a metric to measure and compare rates of change between ecological and phenotypic time series data and whether or not the phenotypic rates differed between evolutionary and plastic modes of adaptation

Graduate Research Assistant - Los Alamos National Laboratory  
Summer 2016  
Mentor: James Theiler, Ph.D.

- Developed and implemented a unique kernelized machine learning algorithm for anomaly detection in hyperspectral image data
Mentor: James Theiler, Ph.D.

- Developed new kernelized machine learning algorithms and implemented them with existing algorithms for anomaly detection in hyperspectral image data

Software Developer - Dynofit Inc., Los Alamos, NM Fall 2013
Mentor: Rohan Loveland, Ph.D.

- Developed software to be used for a gaming environment accessible to patients with cerebral palsy

Summer SULI Intern - Los Alamos National Laboratory Summer 2013
Mentor: Rohan Loveland, Ph.D.

- Developed a unique semi-supervised machine learning algorithm that used a hierarchical structure to cluster data sets obtained from the Curiosity Mars rover
Teaching Experience

Main Lecturer - Utah State University 2019 - 2020

- Calculus 2 (Fall 2020, Spring 2021)
- Trigonometry (Spring 2020)
- Calculus 1 (Fall 2019)

Undergraduate Research Mentor - Utah State University 2017-2018

- Co-created an undergraduate research program to introduce research topics in math biology for a team of undergraduates

Teaching Assistant - Utah State University 2015–2016

- Recitation leader, College Algebra (Fall 2015)
- Recitation leader, Calculus 1 (Spring 2016)

Presentations

Invited Oral

- Grosklos, G. 2021. An innovative approach to evaluating effects of cervid population management. CWD Tribal Nations and Stakeholder Meeting. Online

Contributed Oral

- Grosklos, G. 2020. Lesson planning for a diverse classroom. Together We Teach Conference. USU, Logan, UT
- Grosklos, G., and J. Zhao. 2019. The effects of dispersal on the dynamics of a montane amphibian metapopulation. SIAM Northern States Section. UW, Laramie, WY


• Grosklos, G., and R. Loveland. 2013. Using rare category detection machine learning algorithms to find unique sample classes in ChemCam data. SULI Student Talks. Los Alamos, NM

Poster

• Grosklos, G. 2016. One ring to enclose them all: minimum volume ellipsoids for data periphery characterization, Los Alamos National Laboratory Student Symposium, Los Alamos, NM

Programming Proficiencies

• Matlab, Python, R

Professional Memberships/Activities

SIAM Northern States Section Mini-symposium                  Sep 2019
  - Co-organizer - ‘From Genome to Biome: Mathematical Models in Biology’

NIMBioS Network Theory Tutorial                           Feb 2019
  - Participant

Climate Adaptation Traineeship Program               2017-2019
  - NSF Trainee

USU Chapter SIAM Club                                2016-2019
  - Web Designer

  • Maintained USU chapter’s website, including updating events, member profiles, and web design while also organizing curricular events such as the Wasatch Student Chapter Conference USU (2019), club trip to Idaho National Lab (2017), ‘Integration Bee’ (2016), and ‘Calculus Boot Camp’ (2016)

Wilderness First Aid Certification                   March 2019
  - Participant
Community Outreach

Volunteer Judge, Undergraduate Research Symposium, USU 2019

Adopt-a-trail Program Volunteer, Uinta-Wasatch-Cache National Forest Service 2018-2020

Volunteer, Cache Valley Trail Crew 2016-2020

Volunteer Presenter, USU Science Unwrapped 2016-2019

Volunteer, Los Alamos Tuff Riders Trail Crew 2015-2016

Member, Los Alamos Mountaineers Club 2014-2015