Diffusion Models for Species-Specific Functional Habitat Connectivity

UC Santa Cruz Technical Report UCSC-SOE-23-10

October 2, 2023

Natalia Ocampo-Peñuela¹*, Coen Adler²†, Artie Nazarov²†, Jasmine Tai²†, Natalie Valett²†, Luca de Alfaro²*

¹Environmental Studies, University of California, Santa Cruz, 1156 High St, Santa Cruz, 95064, California, United States.
²Computer Science and Engineering, University of California, Santa Cruz, 1156 High St, Santa Cruz, 95064, California, United States.

*Corresponding author(s). E-mail(s): nocampop@ucsc.edu; luca@ucsc.edu;
Contributing authors: ctadler@ucsc.edu; anazarov@ucsc.edu; cjtai@ucsc.edu; nvalett@ucsc.edu;
†These authors contributed equally to this work.

Abstract
Habitat connectivity is essential to maintain viable and healthy populations due to increased genetic flow, enhanced functional habitat, and repopulation potential [1–4]. Connected habitats will be key to support species resilience to climate change given already documented upslope and pole-ward range shifts [5–7]. Conserving, enhancing, and restoring habitat connectivity has been identified as a priority to achieve global targets of halting extinctions and safeguarding 30% of our planet [8]. Computational constraints of existing connectivity models have restricted these analyses to coarse scales and to a few charismatic species and/or landscapes [9–11].

We introduce EcoScape, an efficient algorithm for computing the pixel-level functional connectivity of species habitats. Ecoscape estimates the connectivity as the probability that a pixel will be repopulated after a major extinction event by simulating a diffusion process that takes into account the species' ecological

1
preferences and dispersal abilities [12]. We leverage Graphic Processing Units (GPUs) to compute habitat connectivity with similar accuracy, and hundreds of times faster, than widely accepted connectivity models. We show that for two forest/woodland bird species in California, EcoScape habitat connectivity values are highly correlated with relative abundances reported in eBird [13]. The speed and accuracy of our algorithm allows mapping habitat connectivity at large scale and fine resolutions for multiple species, offering guidance for current and future conservation efforts.

1 Main

Habitat connectivity is essential for the long-term viability of animal and plant populations because it enhances species movement in the landscape, increasing the amount of functional habitat, and ultimately maintaining genetic diversity [14–17]. Conserving and enhancing habitat connectivity has become increasingly important to protect biodiversity under future climatic scenarios [7, 18] and has been prioritized in global frameworks such as the Convention of Biological Diversity (CBD) [19]. Species are already shifting their ranges in response to warming temperatures, seeking suitable climatic conditions towards the poles or at higher elevations [20, 21]. Conservation efforts and actions must prioritize enhancing landscape connectivity to allow species movement to areas with favorable conditions [6, 18].

Conservation planning benefits from spatially-explicit models of habitat connectivity to inform decisions and evaluate outcomes [22]. To reach the CBD goal of protecting 30% of the planet to effectively safeguard biodiversity, we need to prioritize areas that protect and enhance landscape connectivity for as many species as possible, while considering future climatic scenarios [7, 18]. However, current efforts to map habitat connectivity in current and future scenarios have focused on a few charismatic species (e.g. large mammals), limited spatial scales (e.g. specific landscapes), or only within protected areas [23–25].

Computational constraints have limited the scope and scale at which habitat connectivity models are applied. Currently, widely used connectivity models are based on electrical circuit theory [9, 26–29] and graph theory [10, 11]. These models are limited in the scale and magnitude of data they can compute, and the speed at which they can compute it [30, 31]; see recent advances using the Julia language [28, 31]. These limitations have led to conservation ecologists making decisions at very coarse scales, based on a few charismatic species, or ignoring species-specific data on habitat preference and dispersal ability [24]. Notably, even the most advanced recent models require extensive input from the users [31], making them inaccessible to practitioners and continuing a narrow focus on charismatic and well-known species. There is a need for a modeling approach that allows computation of species-specific habitat connectivity based on habitat, ecological preferences, and dispersal abilities, while taking advantage of open fine resolution spatial datasets and the wealth of citizen science observations available [32, 33].
We present here EcoScape, a novel way of computing habitat connectivity, directly rooted in metapopulation theory [34]. In our approach, we directly estimate the probability with which each habitat pixel can be repopulated following a mass extinction event. We do so by running many simulations. In each simulation, we assume that the animal population is extirpated except in randomly chosen “seed” locations, and we compute the habitat that can be repopulated by simulating a diffusion process from the seed locations. The diffusion process takes into account species-specific habitat requirements, dispersal distance and gap-crossing ability, and affinity to move through different types of landcover in the landscape matrix. Each diffusion simulation is a simple, regular computation on a two-dimensional lattice of geographical pixels, of a kind that is very well suited to be optimized via Graphic Processing Units (GPUs).

We validate the approach on two California forest/woodland birds: Acorn Woodpecker (*Melanerpes formicivorus*) and Steller’s Jay (*Cyanocitta stelleri*). These species are non-migratory, have close association with forest/woodland, and contain large numbers of recorded observations in California, making them ideal for the validation. We show that for these species, the repopulation computed via our diffusion-based approach is highly correlated with bird sightings logged in the citizen-science platform eBird [13]. To the best of our knowledge, this represents the first time a connectivity model has been validated via fine-grained, large-scale observations over an extended region.

By removing the computational limitations of connectivity models at fine scales and for many species, we are unlocking the potential of the wealth of biodiversity data that exists, while harnessing the power of computer science methodologies. Our approach draws on the power of open biodiversity data from global and citizen-science platforms such as the International Union for the Conservation of Nature (IUCN) [35], eBird [13], and Avonet [36]. The algorithm we present is accompanied by Python packages that are user friendly and allow input of as much or as little information as is available for a certain species.

2 EcoScape: Modeling Habitat Connectivity via Diffusion

We model the functional habitat connectivity for a species at the pixel level by simulating an extinction event followed by repopulation from randomly placed survivors in the landscape. The connectivity of a habitat pixel is the average probability that the pixel is repopulated after several hundred simulations.

Our model takes as input two spatial layers: a layer \( h \) of habitat pixels \( h_{ij} \in \{0, 1\} \) for the species (derived as the Area of Habitat map [37]), and a landscape matrix resistance layer \( p \), with pixel \( p_{ij} \in [0, 1] \) indicating how permeable the land cover is at \( ij \) to the species’ movement (i.e. a forest bird might be reluctant to cross an open field or built surface). We assume \( p_{ij} = 1 \) in the habitat. We develop our results under the standard assumption of all pixels being equal-sized. The model also takes as input the gap-crossing distance \( g \), and the dispersal distance \( d = g \cdot n \) of the species, both measured in number of pixels. The dispersal consists of \( n \) rounds. In each round, a bird with gap-crossing distance \( g \) can land up to \( g \) pixels away, flying over up to \( g - 1 \)
pixels of the landscape regardless of its permeability. The total distance covered is thus at most \( d = g \cdot n \).

EcoScape performs a large number of extinction-repopulation simulations, and outputs a *connectivity* layer \( c \) and a *flow* layer \( f \). The *connectivity* layer indicates the average extent \( c_{ij} \) with which each pixel \( ij \) was repopulated; this indicates the extent to which each habitat pixel can support the species. The sum \( q = a \cdot \sum_{ij} c_{ij} \) is the *functionally connected habitat* (FCA), where \( a \) is the area of a pixel. The *flow* layer indicates how important each pixel is for habitat connectivity: the flow \( f_{ij} \) at pixel \( ij \) indicates the amount of functionally connected area that was repopulated via \( ij \).

Each extinction-repopulation simulation starts by selecting random *seeds* from which the repopulation starts; each habitat pixel is selected independently (see Methods). From the seeds, we create a *diffusion* layer \( s \) with \( s_{ij} = 1 \) at pixels \( ij \) selected as seeds, and \( s_{ij} = 0 \) elsewhere. During the simulation, the algorithm updates the diffusion layer \( n \) times, simulating a diffusion process [12] with gap-crossing distance \( g \) via:

\[
s_{ij} := p_{ij} \cdot \max_{kl \in N(ij)} \left( X_{kl} \cdot s_{kl} \right),
\]

where \( p_{ij} \in [0, 1] \) is the landscape matrix permeability of pixel \( ij \), \( N(ij) \) is a pixel neighborhood of \( ij \), \( s_{kl} \) is the (pre-update) diffusion at source pixel \( kl \), and \( X_{kl} \) is a random number chosen independently for each source pixel \( kl \). The neighborhood \( N(ij) \) consists of the pixels that are within gap-distance \( g \) from \( ij \). To obtain an efficient GPU-based implementation, we let \( N(ij) \) be the square centered around \( ij \) and with edge \( 2g + 1 \), that is, \( N(ij) = \{ kl \mid i - g \leq k \leq i + g, j - g \leq l \leq j + g \} \). The random factor \( X_{kl} \) models the probability that the population at pixel \( kl \) will diffuse in the round. The random factor acts as a tie-breaker for the max operator. Further, the randomness at each simulation step ensures that multiple diffusion paths are chosen, to avoid highlighting a single least-cost path [9, 17, 38]. The permeability \( p_{ij} \) of pixel \( ij \) is proportional to the affinity between the species and the land cover, or 1 in the habitat. Permeability is inversely related to the resistance layer used in circuit-theory based approaches [9, 27].

Each simulation proceeds for a number \( n \) of updates (1), corresponding to the number \( n \) of gap-crossing the individual can do during dispersal. The connectivity layer \( c \) is finally obtained by clipping the diffusion to the habitat via \( c = hs \) (i.e., \( c_{ij} = h_{ij} s_{ij} \) at every pixel \( ij \)). The algorithm outputs the connectivity thus computed as the average over many simulations. The regular nature of the update (1) allows the computation to be encoded as tensor operations in the Machine-Learning (ML) framework PyTorch [39], and run with great efficiency on GPUs. The flow layer is computed at pixel \( ij \) via

\[
f_{ij} = p_{ij} \frac{\partial q}{\partial p_{ij}},
\]

where \( q = a \cdot \sum_{ij} c_{ij} \) is the *functionally connected habitat* (FCA). The flow is not clipped to the habitat, yielding information on the role of terrain outside the habitat in maintaining connectivity. In the methods, we show that the flow \( f_{ij} \) indicates the amount of functionally connected area that depends on diffusion through pixel \( ij \). By taking the total connectivity \( q \) as model output, and the landscape permeability \( p \) as
a model parameter, we can compute \( \frac{\partial q}{\partial p} \) via back-propagation, again relying on the ML framework PyTorch; multiplication by \( p \) yields the flow \( f \).

The EcoScape computation of connectivity and flow is illustrated on an artificial landscape in Figure 1. We have implemented the algorithms in a Python package [40]; we also provide sample Python notebooks that demonstrate how to produce connectivity and flow layers for example species.

**Functionally Connected Habitat and Equivalent Connected Area.**

The *functionally connected habitat* (FCA) \( q = a \cdot \sum_{i,j} c_{ij} \) computed by EcoScape is closely related to the Equivalent Connected Area (ECA) of the habitat [41]. To see this, consider habitat patches \( H_1, \ldots, H_n \), where patch \( H_k \) has area \( A_k \), \( 1 \leq k \leq n \), and assume as a simplification that the probability with which a bird can move from a patch \( H_k \) to patch \( H_l \) in the EcoScape diffusion model is equal to \( p_{kl} \), and is independent on the precise source and destination location of the bird in the patches. For a location to be repopulated, birds need to be able to diffuse from a seed to the destination patch. In EcoScape, a seed occurs in patch \( H_k \) with probability proportional to \( A_k \), and such seed can repopulate a location in \( H_l \) with probability \( p_{kl} \). As there are \( A_l \) location in \( H_l \), the contribution of seeds in \( H_k \) for the repopulation of \( H_l \) is \( A_l A_k p_{kl} \). The total habitat connectivity, which essentially equals functionally connected habitat, is thus proportional to the ECA \( \sum_{kl} A_k A_l p_{kl} \). Thus, the *functionally connected habitat* (FCA) computed by EcoScape is consistent with established ideas in landscape connectivity and metapopulation theory.

### 3 Results

We used EcoScape to compute the connectivity and flow layers for two non-migratory bird species: Acorn Woodpecker (*Melanerpes formicivorus*) and Steller’s
Fig. 2: Connectivity (A, B) and flow (C) layers for the Acorn Woodpecker (*Melanerpes formicivorus*) in California using a 600m gap-crossing distance and 12 km dispersal distance calculated using EcoScape.

*Jay (Cyanocitta stelleri).* The species were selected for their association with forest/woodland, and ease of detection, facilitating the validation with citizen-science data. We studied these species in California at a resolution of 300m pixels. We obtained the habitat layer by refining eBird range maps [42] based on habitat preference data from IUCN [43, 44] to produce an Area of Habitat map [37]. We produced the landscape matrix permeability layer using a global habitat class map [43], species habitat preferences from IUCN, and habitat use from eBird observations [13] (see Methods for details). We used 600m (2 pixels) as the gap-crossing distance for both species. The dispersal distances were derived from the species’ Hand-Wing Index (HWI) [36], which is
Ecoscape and Omniscape

Species | N.sim | Time(s) | %std>0.02 | FCA(km$^2$) | $R^2$ | Time(s) | $R^2$
---|---|---|---|---|---|---|---
Acorn Woodpecker | 400 | 16 | 0.75 | 50,668 | 0.95 | 1866 | 0.79
Steller’s Jay | 400 | 5 | 0.64 | 68,915 | 0.90 | 772 | 0.97

Table 1: Experimental results on EcoScape and Omniscape for Acorn Woodpecker (12 km dispersal) and Steller’s Jay (1.8 km dispersal). N.sim is the number of simulations, %std>0.02 is the percentage of pixels with standard deviation greater than 0.02 (or 2% of range), FCA is the functionally connected habitat area, and $R^2$ is the coefficient of determination of eBird sightings vs. connectivity (EcoScape) or current flow (Omniscape). EcoScape was run on Google Colab with A100 GPU; Omniscape on Intel i7 2.3GHz.

Speed and accuracy.

EcoScape computed the connectivity and flow layers as the average of multiple simulations. We estimated the accuracy of using 400 simulations, by computing the standard deviation at every pixel. As reported in column %std>0.02 of Table 1, fewer than 1% of pixels had standard deviation greater than 0.02, or 2% of the overall connectivity value range [0, 1]; no pixel had standard deviation greater than 0.05. As the connectivity is computed as the average of statistically independent simulations, its standard deviation at a pixel is inversely related to the square root of the number of simulations being averaged. Henceforth, we used 400 simulations unless otherwise noted. Table 1 also reports the time used by EcoScape to compute the connectivity layer and functionally connected area for each species; we used Colab notebooks with NVidia A100 GPU [46]. If the flow layer is also desired, the running time increases by a factor of 3-4.

Correlation with citizen-science data.

We validated the connectivity layers computed by EcoScape by comparing them with bird sightings reported in eBird [13]. We only considered complete eBird checklists with a maximum traveling distance of 2 km, during the breeding season (April to June), from 2012 to 2018 to match the 2015 landcover used [43] and exclude the influence of the 2020 California wildfires (n=185,996 checklists). We grouped the checklists using a 1 km grid, and computed the average number of individuals of the species reported per checklist at each location within the species’ California habitat (number of locations: 6,818 for Acorn Woodpecker, 5,806 for Steller’s Jay). We then subdivided the
connectivity values $[0, 1]$ into 10 intervals $[0, 0.1), [0.1, 0.2), \ldots, [0.9, 1]$, and computed the average number of individuals per checklist in each interval. The results, obtained by averaging 10,000 simulations, indicate that average bird sightings are strongly correlated with connectivity, with a coefficient of determination $R^2$ above 0.9 (Table 1). The relationship between sightings and connectivity for the two species is illustrated in Figure 3.

In the supplementary material, we provide measurements of $R^2$ for many values of the gap-crossing distance $g = 600m$, and for dispersal distance of 12 km for Acorn Woodpecker and 1.8 km for Steller’s Jay.

### 3.1 Comparison with Previous Approaches

OmniScape [29] is the habitat connectivity tool whose output can be most directly compared with the one of EcoScape. Like EcoScape, OmnisciScape computes a measure of how much each pixel is connected, namely, the cumulative current; the analysis radius is analogous to our dispersal distance. The algorithms in OmnisciScape and EcoScape are conceptually related, differing in the direction of exploration and the underlying analogy. OmnisciScape asks, for each pixel, how much current can flow into it; conversely in EcoScape, we simulate the repopulation originating from habitat pixels. OmnisciScape uses current, while EcoScape uses diffusion, as a physical analogy for bird movement.
We have run Omniscape on the study area, using conductance values proportional to our landscape matrix permeability (see the Methods, Section 5.5). The results (Table 1) indicate that Omniscape’s and EcoScape’s outputs are both highly correlated with eBird bird sighting frequencies. We view the fact that EcoScape’s results roughly match those of a widely-used tool such as Omniscape [47] as encouraging. We note that the parameters used by both tools can be fine-tuned, and the correlation values vary with the precise values of landscape matrix permeability and conductance.

The potential of diffusion to represent animal movement has been explored in [12, 48], where the results are validated on a species of butterflies at a landscape scale.

Several habitat connectivity models rely on circuit and graph-theory and are computationally intensive, when compared to EcoScape. Efforts to speed them up have included coding in Julia language [28, 31], and stand-alone packages such as Gflow [49] and ConScape [31]. The current fastest approach is ConScape [31], which can process a 10,000 pixel territory in 72 seconds; EcoScape can process the same extent in 0.02 seconds, or over 1,000 times faster.

The outputs of EcoScape represent two layers that usually require running two different pieces of software. As the connectivity of a pixel is directly its repopulation probability, EcoScape allows the computation of the Functionally Connected Area for the entire range of a species in a small fraction of the time, and much higher resolution, than the Equivalent Connected Area produced by graph-theory algorithms such as CONEFORE [11] at the patch level. The flow layer produced by EcoScape is akin to least cost path corridors identified by software such as LinkageMapper [50] and Omniscape [29], but it is produced during the same run as the connectivity layer. We expand this approach to include the importance of pixels for connectivity for habitat and non-habitat pixels.

4 Discussion

EcoScape is a novel, diffusion-based, pixel-level, and species-specific algorithm to map functional habitat connectivity and flow at fine resolutions and large scales.

Conserving, enhancing, and increasing habitat connectivity has been identified as an important mechanism to combat habitat loss, fragmentation, and climate change impacts [7, 51]. Current efforts to address this challenge have fallen short because they only account for a few charismatic and well-studied species (e.g. pumas [52], jaguars [53]), or are limited to recommendations only for protected areas [25]. To achieve global targets of increased connectivity at landscape scales, we must produce spatially-explicit models that highlight areas of high value for conserving and restoring connectivity for multiple species [54, 55].

EcoScape contributes to this task by significantly speeding up the computation of functional habitat connectivity, making it practical to map hundreds of species at once. The speed of EcoScape also facilitates large-scale scenario exploration for connectivity, e.g., with respect to future climate conditions [6] by replacing the habitat and landcover layers with projected species distributions, to understand how current decisions will play out in the future (for example see [56]).
The workflow we developed relies on open data from IUCN, eBird, and AVONET, and provides users Python packages for every step of the process, providing a starting point for the study of any bird species globally. Dispersal distances, in particular, are challenging to estimate for most species and the lack of information has limited the application of connectivity models for lesser-known species. Though not perfect, we propose using the Hand-Wing index \cite{45} to estimate dispersal as an input for EcoScape for birds. A similar idea has been explored with body mass for mammals \cite{57}, and applied to existing connectivity models \cite{58}.

We present the first validation of a habitat connectivity model via large-scale, freely available citizen science data. Existing connectivity models have been tested via simulation experiments \cite{59}, using animal tracking data \cite{60, 61}, and more rarely with genetic data \cite{62}. The limited availability of tracking data, which currently exists mostly for large and charismatic fauna \cite{63}, has been an obstacle to the validation of connectivity models for lesser-known species. In contrast, the validation framework we introduced can be used for any bird species with sufficient eBird data \cite{13}, and potentially for other taxa using data from iNaturalist \cite{64} or other taxon-specific citizen science platforms (e.g. eMammal). Accurate connectivity maps will result in more effective conservation measures \cite{65}.

As ecology and conservation enter the world of big data \cite{30}, technological challenges arise from processing and taking full advantage of the fine resolution and large quantity of data. EcoScape addresses this challenge by providing users an efficient, easy to use, and open way to map habitat connectivity and flow at fine resolution over large scales. These advances will allow ecologists and conservation planners to fully consider a spectrum of lesser-known species and diverse scenarios in their conservation plans, while taking advantage of the wealth of citizen science and expert-derived data available.

Acknowledgments.
We thank Dr. Tyler Sorensen for helpful discussions on adapting algorithms to GPU computation, and Dr. Santiago Claramunt for helpful discussions to parameterize dispersal distances using the Hand-Wing Index.

Funding statement.
This work was partly funded by a Google Research gift.

5 Methods
The overall data and processing flow used in this paper is illustrated in Figure 4. First, we prepare the inputs used by the EcoScape algorithm, namely, the spatial layers encoding the habitat and landscape matrix permeability. Next, we run the EcoScape algorithm on these data, using as parameters a given gap-crossing and dispersal distance. Lastly, we analyze the connectivity layer produced by EcoScape, correlating it with bird observations from the eBird citizen-science project \cite{13, 66}, yielding the validation for our proposed connectivity approach. Everything needed to reproduce the results of this paper is provided in an open GitHub repository \cite{67}, accompanied by
5.1 The EcoScape Algorithm And Its Implementation

**EcoScape inputs.**

The EcoScape algorithm takes three inputs:
• A boolean (binary) habitat layer $h$, so that for a pixel $ij$, $h_{ij} = 1$ if the pixel is in the habitat, and $h_{ij} = 0$ if it is outside.

• A landscape matrix permeability layer $p$, with $0 \leq p_{ij} \leq 1$: we assume $p_{ij} = 1$ in the habitat, and $p_{ij} = 0$ for landcovers with high resistance for the species; intermediate values correspond to suitable but not ideal landcover for the species.

• A gap-crossing distance $g$, expressed in number of pixels, and a number of gaps $n$, so that the dispersal distance $d$ is $d = n \cdot g$.

Note that $h \leq p$, in the sense that at every pixel $ij$, we have $h_{ij} \leq p_{ij}$.

In practice, in our implementation, we do not provide $p$ directly. Rather, we provide a landcover layer $t$, where $t_{ij}$ is an integer, describing the landcover type at pixel $ij$, along with a permeability table $D$, giving the permeability $D(t) \in [0,1]$ for every landcover type $t$. The permeability layer is then obtained as $p_{ij} = D(t_{ij})$. The advantage of this approach is that, when studying multiple species on the same study area (as done in this paper), we can use the same landcover layer; only the permeability table needs to be modified to achieve the appropriate landscape matrix layer. Further, one can experiment with different permeability values without having to rebuild a raster layer.

**The EcoScape diffusion algorithm.**

The EcoScape algorithm computes the connectivity and flow layers as the average of a large number of simulations (400, in the results presented in this paper). The algorithm models the repopulation of habitat pixels from a few seeds after a mass extinction event. At the beginning of each simulation, we initialize the diffusion layer $s$ via

$$s_{ij} = h_{ij} \cdot \delta(Y_{ij} < \alpha/(4d^2)),$$

where $Y_{ij}$ is a random variable with value uniformly distributed in $[0,1]$ sampled independently at each pixel, $d$ is the dispersal distance, and $\delta$ is a characteristic function so that $\delta(\phi) = 1$ if $\phi$ is true, and 0 otherwise. Above, the constant $\alpha$ regulates the density of seeds. A bird, starting from a seed location, can repopulate at most an area $4d^2$, as it can spread by $d$ in all directions; the seed density $\alpha$ says how many seeds there are, on average, in such a $4d^2$ area. We chose in our experiments $\alpha = 4$; this corresponds to one seed every $d^2$ pixels. We found that smaller values of $\alpha$ simply scaled the output values for connectivity down, while requiring a larger number of simulations for a similar accuracy (as each simulation involved fewer seeds). Larger values of $\alpha$ resulted in too much overlapped spread, losing some of the informativeness in the connectivity layer for well-connected habitat regions.

Once the diffusion layer $s$ is initialized, EcoScape performs $n$ iterations of the diffusion update (1). In the update, as mentioned, the random variable $X$ is used to break ties at random between source locations where $c$ has the same value. If we did not use such a random tie-break, the nature of the GPU implementation would favor a fixed origin for the birds (in our experiments, the upper-left corner), leading to large artifacts and invalidating the output. The randomness also ensures that the diffusion is multi-path, rather than using exclusively the path of least resistance. We let $X$ assume values uniformly at random in the open interval $(0.9,1)$, sampled independently at
every pixel. The fact that $X < 1$ can be interpreted as the fact that birds at a pixel have a small probability, at each diffusion step, of choosing not to diffuse.

The random tie-break can also be understood as follows. Let $s^{(0)}$ be the repopulation layer after seed initialization, and let $s^{(m)}$ be the layer after the $m$-th iteration of (1), so that (1) can be rewritten as

$$s_{ij}^{(m+1)} = p_{ij} \cdot \max_{kl \in N(ij)} \left( X_{kl} \cdot s_{kl}^{(m)} \right).$$

(3)

Then, it is easy to see that the sequence $s^{(0)}, s^{(1)}, s^{(2)}, \ldots$ is monotonically increasing, so that at each $ij$, $s_{ij}^{(0)} \leq s_{ij}^{(1)} \leq s_{ij}^{(2)} \leq \cdots$.

To track where the repopulation came from, we can define a source mapping $S$, so that the repopulation at pixel $ij$ came from neighboring pixel $S(ij)$. We define this mapping as follows. Initially, $S(ij) = \bot$ at all $ij$, where $\bot$ is a special symbol indicating that the source is undefined (the population did not come from another location). When we update the value at $ij$, that is, when $s_{ij}^{(m)} < s_{ij}^{(m+1)}$, there is a unique neighbor $kl \in N(ij)$ that provides the argmax in (3): this $kl$ can be thought of as the source of the increased bird population at $ij$, and therefore, we set $S(ij) := kl$. When the value at $ij$ is unchanged, we leave $S(ij)$ unchanged.

The source mapping $S$ defines a graph, with an edge from $ij$ to $kl$ when $S(ij) = kl$.

It is easy to see that this graph is a collection of trees, each tree having root in a repopulation seed. Indeed, there can be no loops in the graph: $S(ij) = kl$ implies $s_{ij}^{(m+1)} < s_{kl}^{(m)}$ at the time $m + 1$ of the last update of $kl$, and by monotonicity, $s_{ij}^{(n)} < s_{kl}^{(n)}$ after the last update $n$. Thus, the EcoScape algorithm can be understood as a way to generate random diffusion trees rooted in the repopulation seeds. This alternate characterization is essential to the computation of the flow layer. Finally, the connectivity layer is obtained by clipping the diffusion to the habitat via $c = h s^{(n)}$, where multiplication is pixel-wise.

The flow layer.

The flow layer relates to the importance of each pixel for the connectivity of the species’ habitat. Let $c_{ij} = c_{ij}^{(n)}$ be the output connectivity, and let $q = a \sum c_{ij}$ be its sum over all the raster, where $a$ is the area of a pixel (expressed, for example, in km$^2$). If $c_{ij} > 0$, we can ask: how much repopulation passed through $ij$? The locations that owe their repopulation to $ij$ consists of $ij$, its immediate descendants in the diffusion trees, or the $kl$ with $S(kl) = ij$, and their descendants of any level: we call this the downstream region of $ij$ (see Figure 5). Precisely, the downstream region of $ij$ is the set of pixels $U(ij)$ that have a path in the graph $S$ to $ij$. Pixels in $U(ij)$ were repopulated through $ij$.

If we change $p_{ij}$ by a tiny amount $\Delta$ (tiny, so as not to change the source mapping), the connectivity in the downstream region of $ij$ will change correspondingly, by $\Delta(q/dp_{ij})$. If we keep $S$ fixed, that is, if we fix the flow taken by the updates (3), it is easy to see that the total habitat repopulation of the downstream region of $ij$ is
Fig. 5: A diffusion layer \( s \). The value \( s_{ij} \) is indicated in each pixel \( ij \); pixel in the habitat are depicted in green, and the dark-colored pixel \((1, 3)\) is the (only) seed, with \( s_{1,3} = 1 \). The source mapping \( S \) is indicated with arrows: for instance, we have \( S(1, 2) = (1, 3) \) (pixel \((1, 2)\) points to the sink \((1, 3)\)) and \( S(2, 2) = (1, 2) \). The pixels in the downstream region of \((2, 2)\) are \( U(2, 2) = \{(2, 2), (3, 1), (3, 2), (4, 1), (4, 2), (4, 3)\} \), and are shown with a bold border around them. We have \( f_{2,2} = 0.87 + 0.85 + 0.83 + 0.82 = 3.37 \).

\[
p_{ij}(\partial q/\partial p_{ij}), \text{ or:} \quad f_{ij} = a \sum_{kl \in U(ij)} c_{kl} = p_{ij} \frac{\partial q}{\partial p_{ij}}. \tag{4}
\]

This provides a way of computing the flow \( f_{ij} \) through a pixel \( ij \) using back-propagation in a machine-learning framework.

**Implementation.**

The EcoScape connectivity algorithm is implemented on top of PyTorch, one of the most widely-used and powerful machine-learning frameworks [39]. We encode the EcoScape computation in PyTorch by having the initial diffusion layer \( s^{(0)} \), consisting of the repopulation seeds, be the input. The landscape matrix permeability \( p \) is a model parameter. The PyTorch model computes \( n \) multiplication by a random layer followed by max-pool operations, and finally a multiplication by \( p \), corresponding to the updates (3). The output is the connectivity layer \( c = h s^{(n)} \). As \( p \) is a model parameter in PyTorch, we can let \( q = a \sum_{ij} c_{ij} \) and compute \( \partial q/\partial p_{ij} \) via back-propagation; the flow layer is obtained via a multiplication by \( p_{ij} \), according to (4).

When the input layers are sufficiently small, they can be fed at once to the GPU; otherwise, they can be fed in separate tiles.

When interested in both connectivity and flow layers, we divided our \( 3716 \times 5377 \) study area in \( 2048 \times 2048 \) tiles, each with a border of 256 pixels around it (some tiles were smaller, at the extremes of the area). We fed these tiles, of overall size \((2048 + 2 \cdot 256) \times (2048 + 2 \cdot 256) \) pixels, to an A100 NVidia GPU. The GPU processed these tiles, and we uses the output results only from the core part, excluding the border. The purpose of the border was to provide context for the processing of the
core: in particular, seeds in the border region can cause repopulation in the core. Thus, it is necessary for the border region to be larger than the dispersal distance: this was easily true in our setting, as the longest dispersal distance we considered was of $2 \cdot 20 = 40$ pixels.

When only the connectivity layer is desired, the PyTorch implementation can be run in forward-mode only, or no-grad, without storing the information required for back-propagation. This speeds the computation and reduces the memory requirements. In this case, our study area of $3716 \times 5377$ pixels could be sent at once to an NVidia A100 GPU.

Connectivity and flow layers are obtained as the average of many simulations. As the values produced by each simulation are statistically independent, the standard deviation of the average decreases with the square root of the number of simulations, as usual.

While connectivity is bounded to the $[0, 1]$ interval, the flow through a pixel is not bounded (see Figure 5), and can assume large values if the pixel is helpful in repopulating a large territory. To accommodate for the value range, the tool outputs $20 \log_{10}(1 + f_{ij})$ rather than $f_{ij}$ at each pixel $ij$, and such logarithmic scale has also been used in our figures.

**Limitations.**

A limitation of the current algorithm consists in the use of a neighborhood $N(ij)$ that is square in shape: $kl \in N(ij)$ if $|k - i| \leq g$ and $|l - j| < g$, where $g$ is the gap-crossing distance in pixels. We chose to use square neighborhoods for the sake of efficiency, as they matched well the max-pool operators available in PyTorch. For larger values of the gap-distance $g$, a round neighborhood may be preferable. We experimented with variants, but for the small values of $g$ considered in this paper more complex algorithms did not lead to superior results, at least as measured by the fit with citizen-scientist species observations.

**5.2 Generation of the Input Layers**

EcoScape requires as input two spatial layers in GeoTIFF format (rasters), as well as gap-crossing and dispersal distances. We create a Python package, `ecoscope_layers` [68], that generates these layers using freely available data products. To use the package, it is necessary to have API tokens for eBird https://science.ebird.org/en/status-and-trends/download-data and IUCN https://apiv3.iucnredlist.org/api/v3/token. In our code repository for reproduction, we provide Python notebooks that produce the inputs used for this paper [67]. Further, as part of the documentation for the `ecoscope_layers` package, we provide sample code for generating input layers for other species and regions.

To run the package, users need to provide the Jung et al. [43] habitat types map; the desired map projection depending on their study area; the bounds used to define the study area as coordinates; and the six-letter eBird codes of the species of interest. These can be obtained as documented in the software packages.
We note that EcoScape can work with any permeability and habitat layers; the only requirement being that they are specified in the same coordinate reference system (CRS).

**Landscape matrix layer.**

We prepare the landscape matrix layer using the Jung et al. Level 2 habitat map from 2015 [43]. This map classifies global land covers into the habitat types as described by the IUCN Red List Habitats Classification Scheme [44].

**Habitat layer.**

The EcoScape habitat layer is similar to an Area of Habitat map [37] which depicts the suitable area within a species’ range based on habitat and elevation preferences. In this study we do not consider elevation and only refined the range map by suitable habitat due to inconsistencies in elevational limits.

To create the habitat layer, the *ecoscape_layers* package uses the landscape matrix layer produced above in conjunction with the range maps that can be downloaded from eBird at https://science.ebird.org/en/status-and-trends/range-maps. For methodological details on how eBird range maps are computed, see Fink and collaborators [42].

We refine the eBird ranges according to landcover type, using the habitat preference categories downloaded from the IUCN Red List [35], in combination with the landscape matrix layers produced above. Users can decide what classifies as a habitat; the options available are: selecting all forest habitat types (for forest specialist birds), selecting only habitat types classified as “major importance” by the IUCN, or selecting habitat types classified as “major importance” and/or “suitable” by the IUCN. Based on the user-defined refining method, *ecoscape_layers* will create a binary habitat layer of pixels that fall within the species’ range map and are classified as habitat (value 1), and those in the range map but not classified as habitat (value 0).

**Landscape matrix permeability.**

The IUCN Red List habitat preferences provided for a bird species include the habitat code, major importance, suitability, and seasonality for each habitat type listed for a species. For each species, the *ecoscape_layers* package builds a table that associates each landcover type with its permeability. Landcover classes deemed “suitable” and of “major importance” as habitats are assigned a permeability of 1 (no resistance), landcover classes that are “suitable” (but not “major importance”) get a permeability of 0.9, and all other landcovers receive a permeability of 0.

We then fine-tuned these values according to eBird sightings. This step was necessary, as IUCN occasionally did not list as suitable some landcover classes where in fact the species are often sighted, and that thus, are clearly permeable to the species (even when they may not be suitable as habitat proper). To do so, we considered the maximum number of sightings $K$ reported in the period of study by eBird in any landcover class that is part of the habitat. For a landcover class, let $p_{IUCN}$ be the permeability determined via IUCN, and let $N$ the number of species sightings reported by eBird;
we let \( p = \min(1, \max(p_{\text{IUCN}}, N/K)) \), thus giving high permeability to the landcovers where a species is often sighted.

**Layers for Acorn Woodpecker and Steller’s Jay.**
For the study, we selected two forest/woodland associated bird species: Acorn Woodpecker (*Melanerpes formicivorus*), and Steller’s Jay (*Cyanocitta stelleri*). These species were chosen due to their resident status, affinity to wooded habitats, abundant eBird sighting records, and high detectability.

Our study region encompasses California, plus a 100-mile buffer zone around it to provide surrounding information for all California location. We use a spatial resolution of 300m pixels, reasoning that the gap crossing distance of all these birds is likely to be greater than 300m. The layers used for the experiments were of \( 3716 \times 5377 \) pixels; they are provided as part of the reproducibility dataset [67].

For Acorn Woodpecker, we added to the list of suitable landcover types the landcover type 308, *Mediterranean type shrubby vegetation*. In California, this landcover type typically corresponds to sparse oak woodland, which is prime habitat for Acorn Woodpeckers.

### 5.3 Gap-crossing and dispersal distance
To run EcoScape, users must provide values of species movement as gap-crossing distance and dispersal distance, both expressed in pixels. As the name implies, the gap-crossing distance is the longest amount of unfavorable landcover that a bird is willing to fly over during dispersal. A bird with gap-crossing distance \( g \) can fly over up to \( g - 1 \) pixels of the landscape matrix regardless of its permeability. The dispersal distance \( d \) is the distance a species most commonly travels when dispersing. We do not specify \( d \) directly, but rather, we specify the number \( n \) of gap-crossings, so that \( d = ng \).

If users have data on gap-crossing and/or dispersal distance for their study species, such as from bird banding or movement studies, this is the ideal data to input into EcoScape. However, these values are hard to encounter in the literature and only exist for very few well-studied birds [69]. The recent release of the global AVONET database [36], and proofs-of-concept for the Hand-Wing Index and its relationship to gap-crossing abilities, such as those by Claramunt and collaborators in Amazonian forest fragments [70], provide a framework for using the HWI as a proxy for bird dispersal [45, 71]. If dispersal distance for the species of interest is not known, we provide a way to calculate it from the HWI (provided freely in the global database AVONET [36]). Using a dataset of known bird HWI and average dispersal distance, we trained a linear regression model to predict dispersal distance. The trained linear model predicts the dispersal distance \( d \) via the formula:

\[
    d = 0.734 \cdot \text{HWI} - 8.777
\]

The gap-crossing distance has also shown to have a relationship to the HWI, but this has only been explored for road crossing between forest fragments [70]. Given this uncertainty, we use a gap-crossing of 2 pixels (600m) for all species in this study, except
in the sensitivity analysis reported below. We report the gap-crossing and dispersal distance used in our example species in Table 2 and note that the dispersal distance was calculated based on the HWI. As our knowledge of bird dispersal expands with increasingly smaller and cheaper GPS transmitters [63], EcoScape will be ready to receive parameters on both gap-crossing and dispersal distance.

### 5.4 Validation with Citizen-Science Data

We validate the EcoScape connectivity layer for our example species by measuring the correlation between the connectivity at a pixel, and the observed bird frequency at that pixel. As a source of bird observations, we rely on the eBird citizen-science database [13] downloaded January 2023 [66].

**eBird checklists.**

eBird data consists of a collection of checklists. Each checklist is compiled by a birder visiting a location, and counting how many birds of each species they were able to identify. Among other data, a checklist contains:

- a location expressed as latitude/longitude pair,
- a date,
- the type of checklist, which can be stationary, traveling, or incidental (the latter used for observations when the main goal was not birding),
- for traveling checklists, a length of travel,
- whether the checklist was a complete or partial list of observed species,
- and finally, a list of observed species, each with the count of individuals observed.

As a first step, we filtered the checklists, keeping only those that:

- were located in California (our area of study),
- took place in 2012-2018, so as to match the time-frame of our landscape matrix layer, which was derived from satellite observations in 2015,
- took place in the months April-May-June, as this is breeding season for the species we considered,
- were traveling, as incidental checklists are unreliable, and we did not want to consider stationary checklists, which are a minority and might behave differently from traveling ones,
- were of length not exceeding 2 km,
- were complete.

<table>
<thead>
<tr>
<th>Species</th>
<th>Gap Distance</th>
<th>N. gaps</th>
<th>Dispersal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acorn Woodpecker</td>
<td>2px (600m)</td>
<td>20</td>
<td>12 km</td>
</tr>
<tr>
<td>Steller’s Jay</td>
<td>2px (600m)</td>
<td>3</td>
<td>1.8 km</td>
</tr>
</tbody>
</table>

**Table 2:** Parameters for the EcoScape runs that correspond to the dispersal distance estimated based on the Hand-Wing Index (HWI).
The choice of limiting the analysis to checklists not exceeding 2 km was done to balance data availability with the location accuracy of sightings. The shorter the distance traveled, the more accurately we know the locations of the birds sighted, but the fewer the checklists we select. The above criteria selected a total of 185,996 checklists.

Once filtered, we clustered the checklists into a discrete set of locations according to their geographical coordinates. For each checklist, we truncated latitude and longitude to two decimal digits, forming a string: each location would correspond to one such string. For instance, an observation occurring at coordinates (37.428992, -122.015710) would be represented by the string “37.42;-122.01”, and all locations with this string would be associated with the same location. Due to the rounding, a location corresponds to an area of about 1 km^2, which was an approximate match for our checklist length upper-bound of 2 km (during a checklist, observers do not always travel in straight lines). This clustering resulted in 18,336 locations in California.

**Computing the correlation between connectivity and bird sightings.**

We validated the connectivity layers for both EcoScape and existing algorithm Omniscape [29] using eBird observations, and detail the methods below.

Given a habitat layer \( h \) with values in \{0, 1\} and a connectivity layer \( c \) computed via EcoScape or Omniscape, we compute the correlation between \( c \) and eBird sightings as follows. We consider the 18,336 locations with checklists in California. From these, we exclude locations with over 100 checklists; we judge that these are locations with high human visitation and often high anthropogenic impact (close to roads, disturbed areas, etc). This criterion excluded only 296 locations (1.6% of the total checklists). For each of the remaining 18,040 locations, we compute the average number of birds per checklist \( b_l \) for each location \( l \).

Next, we sampled the values of the habitat and connectivity layers at each location. We identified the pixel coordinates \( ij \) corresponding to the location center, and we computed \( c_l \) (the connectivity of the location) as the maximum connectivity in the habitat, in the 3 × 3 pixels (thus, 900 m × 900 m) around \( ij \). Precisely, remembering that the habitat has values in \{0, 1\}:

\[
c_l = \max_{i-1 \leq k \leq i+1} \max_{j-1 \leq m \leq j+1} h_{km} c_{km}.
\]

Like most citizen-science data, eBird observations are affected by various types of noise [72, 73]. Bird sightings are recorded by volunteers with variable skills [74], under variable conditions (length of outing, time of day, and so on), and in locations that have different characteristics. Thus, we do not directly compare the connectivity with the eBird-reported sightings at each location. Rather, we de-noise the data by binning locations according to their connectivity. A detailed statistical justification for this is reported in Section A.1 in the supplementary material.

We divide the 18,040 locations into 10 bins \([0, 0.1), [0.1, 0.2), \ldots, [0.9, 1)\), according to their connectivity. Let the bins be \( B_0, B_{0.1}, \ldots, B_{0.9} \), labeled according to their lower bound. A location \( l \) with \( c_l = 0.12 \), \( l \) will be put in \( B_{0.1} \). Let \(|B|\) be the number of locations that are in bin \( B \). For each bin \( B \), we compute the average number of birds
Species | Radius | Blocksize
--- | --- | ---
Acorn Woodpecker | 40px (12 km) | 5
Steller's Jay | 6px (1.8 km) | 1

**Table 3:** Parameters used in the Omniscape runs that we used in the comparison with EcoScape.

per checklist at locations in the bin: $b_B = E[b_l | l \in B]$. For $x = 0, 0.1, 0.2, \ldots, 0.9$, we measure the correlation between $x$ and $b_{B_x}$, that is, between the bin’s connectivity $x$, and the average number $b_{B_x}$ of bird sightings per checklist in the bin. Specifically, we compute a linear fit between $x$ and $b_{B_x}$, where point $(x, b_{B_x})$ has weight $|B_x|$ equal to the number of locations in $B_x$, and we report the coefficient of determination $R^2$ of such linear fit. This provides a measure of the predictive power of connectivity with regards to bird observations.

**Sensitivity analyses.**

We performed a sensitivity analysis, reported in Section A.2 of the supplementary material, to examine how the results depend on the choice of gap-crossing and dispersal distances. The analysis shows that for Acorn Woodpecker and Steller’s Jay, $R^2$ is high for dispersal distances up to those computed via the hand-wing index (see Section 5.3).

### 5.5 Detailed Comparison with Omniscape

Table 3 provides the details of the Omniscape runs that have been used in our comparison with EcoScape. The block size reported in the table corresponds to the granularity with which Omniscape considers sources for the current flow. A block size of 5 implies that only the central pixel in each $5 \times 5$ tile is used as a source. Omniscape’s running time is roughly proportional to the number of sources, and hence, to the inverse of the square of the block size. A block size of 5 is perhaps a bit coarse, and favors Omniscape in the timing comparison. In EcoScape, a seed density of 4 corresponds for the Acorn Woodpecker to 1 pixel every $40^2 = 1600$ for each simulation, and hence, every pixel has a probability $1/1600$ of being selected in each simulation, and so roughly $1/4$ probability of being selected in a simulation; this compares with a $1/25$ sampling probability in Omniscape. We used a block size of 5 for the Acorn Woodpecker, and of 1 for the Steller’s Jay, where the small radius requires the use of more frequent seeds. To match it, we could have likely reduced the number of simulations in EcoScape, to 100 rather than 400. Due to these considerations, and more in general, due to the large number of parameters involved in these simulations, the comparison we present between Omniscape and EcoScape has qualitative value; the precise results vary according to parameter choices.

For Omniscape, we used Julia 1.5.5 and Omniscape 0.5.8, run on Intel i7 2.3GHz processor. For EcoScape, we relied on an A100 GPU on Google Colab. To obtain the conductivity values EcoScape needs, we have taken our permeability layer $p$, with values in $[0, 1]$, and rescaled it to the interval $[1,11]$ via $1 + 10p$. Table 1 gives a detailed comparison of the running time of EcoScape and Omniscape, in runs with
parameters according to Tables 2 and 3. The results indicate that the output of both tools correlates closely with eBird bird frequency observations.
References


[64] INaturalist.


A Supplementary Material

A.1 De-noising eBird observations

We provide here some statistical justification for the procedure we have employed to de-noise eBird checklist data. We recall that we first cluster eBird checklists into locations according to their geographical coordinates, using rounded-off values of latitude and longitude to attribute each checklist to a discrete location.

Once this is done, a simple way to de-noise the observations consists in restricting the consideration to locations with at least a certain number of checklists, say 10 or 20, and computing the average number of birds per checklist in each location. However, it turns out that relying on locations with many checklists for the validation is rather ineffective, for two reasons. First, there are relatively few locations with many checklists, and they tend not to occur in the middle of the forest habitats of our species, but at the borders, where roads and cities occur. Among our total of 18,336 locations in California, only 3,169 had more than 10 checklists.

Second, and more interestingly, it turns out that even from a statistical point of view, it is not useful to restrict consideration to locations with many checklists. We can think of the noise associated with observations as the sum of two kinds of noise: checklist noise and location noise.

Checklist noise is the noise in checklists occurring at the same location; in other words, it is the standard deviation of the number of birds per checklist, in checklists taken at the same location. To measure the checklist noise, we compute at each location the standard deviation of the number of birds per checklist at that location. We then average the standard deviation at all locations with at least 10 checklists.

Location noise is the standard deviation in average number of birds per checklist measured on locations that are similar, for some measure of similarity. To measure the location noise, we considered only locations with at least 10 checklists. For these locations, we computed the average number of birds per checklist. We then considered two location similar if their connectivity value were similar, computing the connectivity with the parameters chosen for Figure 3. Hence, we divided the locations in 10 bins $[0,0.1), [0.1,0.2),\ldots,[0.9,]$ of connectivity. For each bin, we computed the standard deviation of the locations’ number of birds per checklist. We then computed the average over all bins.

It turns out that for our species, location noise is greater than checklist noise; the values are reported in Figure 4. In other words, there is stronger agreement between the number of birds seen in individual checklists compiled at the same location, than there is about the average number of birds per checklist compiled at equivalent locations from the point of view of connectivity. This most likely happens because we have focused on species that are very easy to detect: if Acorn Woodpeckers are breeding in a
location, they are very likely to be detected and correctly recorded by most observers, lowering checklist noise.

Averaging the checklists taken at the same location is effective for dealing with checklist noise, but ineffective for location noise. As location noise is predominant, we are better off considering all locations: this far improves coverage (from 3,169 locations with at least 10 checklists, to 18,336 locations), and more importantly, improves coverage in the middle of the forested regions that constitute our habitats.

Figure 6 provides another view into this phenomenon. The figure reports the standard deviation of the reported species sightings across locations that have the same connectivity, according to the location’s connectivity, and according to the number of checklists available at the location. As we can see, more checklists per location does not translate into lower standard deviation; the variability of sightings across locations is not primarily a function of the number of checklists available. It is likely that other factors, from availability of nutrients, to characteristics of the location (sun exposure, humidity, amount of human traffic, and so forth) play a larger role.

A.2 EcoScape Runs: Sensitivity Analysis

To analyze the sensitivity of EcoScape runs with respect to the values of gap and dispersal distance, we have run it for our species for gap distances \( g = 1, 2, \ldots, 6 \), and for numbers of gap crossings \( n \) ranging from 2 to 40. Figures 7A and 7B report the coefficient of determination \( R^2 \) between connectivity and sightings for each parameter combination for the species considered in the study.

The results present a threshold effect: the coefficient of determination is high until the dispersal distance \( d = ng \) reaches a critical threshold, then decreases. We hypothesize that the distance threshold corresponds to the effective dispersal distance for the species, that is, the distance at which dispersal routinely contributes to population abundance. The threshold corresponds roughly to the dispersal distance obtained via the Hand-Wing Index approach (see Section 5.3).

A.3 Omniscape Runs: Current Flow vs. Sightings

In Figure 8, we report the average number of sightings vs. cumulative current in habitat locations, as computed by Omniscape. This figure is the equivalent of Figure 3 for Ecoscape. To produce Figure 8, we set the 0.9 value to the 90% percentile of the current range in the habitat; this renormalization makes the graphs comparable with the ones for EcoScape.
Fig. 6: Standard deviation of bird sightings at each location (measured as average over checklists at that location) according to the connectivity of the habitat, and the number of checklists, at the location.
Fig. 7: Coefficient of determination for connectivity as a predictor of average bird sightings per checklist, according to the gap crossing distance $g$ (expressed in km; every pixel is 300m) and the number of gap crossings in a dispersal.
Fig. 8: Average sightings per checklist vs. cumulative current in habitat as computed by Omniscape.