Sampling design for monitoring boreal birds in Quebec

Willian Vieira Bruno Drolet

Table of contents

Welcome

This webpage contains the technical report detailing the Sampling design for monitoring boreal birds in Quebec.

The online version of this report is hosted at willvieira.github.io/sampling_BMS and is automatically built whenever the code repository at github.com/sampling_BMS is modified.

This version of the report was built on GitHub Actions on 2023-05-21.

License

TBD

1 Introduction

This project is part of a nationwide effort to monitor the status of birds in the underrepresented boreal region. In this report we describe the Quebec adaptation of the Boreal Optimal Sampling Strategy (BOSS). The BOSS design is a hierachical sampling approach stratified by ecoregions, habitat types, and cost constraits (Van Wilgenburg 2020). This structured design provides a spatially balanced coverage while accounting for rare habitats and sample cost. Here, we focus on the adaptation of the design for the Quebec province; for a thorough explanation and discussion of the national strategy, see Van Wilgenburg (2020).

In addition to stratifying the sampling based on habitat distribution and cost constraits, the BOSS design includes a function to take legacy sites and iconic sites into account. Legacy sites are existing or historical surveys with data extracted from randomly selected sites, whereas iconic sites are from non-randomly selected sites. The key reason for integrating legacy or iconic sites in the sampling design is to keep a representative sample of the community while reducing the sample cost. This is especially important in Quebec, as there are many historical data in the southern part of the province. Considering legacy sites in well-covered regions allows us to allocate ressources to remote areas with less data and higher sampling costs. In Chapter 5, we detail a novel approach accounting for the number and distribution of legacy and iconic sites to reduce sample size and maintain a representative sample of habitat types.

Once habitat types, cost constraints, and legacy sites are defined, the BOSS design uses the Generalized Random Tessellation Stratified Sampling (GRTS; Stevens Jr and Olsen (2004)) method to perform the random sampling. This is a widely used approach to ensure spatially balanced samples in a region. The GRTS uses a mapping function to transform two-dimensional space into one-dimensional space with an ordered spatial address. This one-dimensional ordered space is then randomly reordered before the sampling. This random reordering of the linear, one-dimensional space ensures a spatially well-balanced sample, whatever the sample size. After being sampled, this one-dimensional space is then mapped back to the original two-dimensional space.

This report is divided into two main sections. The first section details the spatial layers to feed the GRTS algorithm. We begin by describing the study area in the Quebec province, the selected ecoregions, and the Primary Sample Unit (PSU). We then detail de habitat and cost layers to weight the inclusion probabilities. Finally, we dedicate a complete section to describe the simulations used to create the new method to account for legacy and iconic sites. The first section contains most of the steps that have been regionalized for Quebec. The second section details the sample steps using the GRTS algorithm. In this second part, we will begin

by describing the method used to calculate the stratified sample size for each of the ecoregions. We then detail the use of the GRTS to sample the PSUs and the Secondary Sample Unit (SSU).

2 Sampling frame

2.1 Area of study

The study area for Quebec is outlined in Figure 2.1. It was expanded beyond the Boreal boundary to include the Arctic ecosystems. The study area contains a total of 7 ecozones, and their sizes and proportions are described in Table 2.1:

Table 2.1: Area (in hectares) and proportion of ecozones covered by the study area.

Ecozone	Area (% prop)
Taiga Shield	56275939 (35.74)
Boreal Shield	51342357 (32.61)
Southern Arctic	27378100 (17.39)
Northern Arctic	12918116 (8.2)
Hudson Plain	6235409 (3.96)
Arctic Cordillera	1713974 (1.09)
Atlantic Maritime	1587877 (1.01)

In order to accommodate habitat heterogeneity, the study area was hierarchically stratified into different levels of spatial aggregation. Below, we will provide a brief description of each of these strata, ranging from the ecoregion level to the specific sampling point level. For a more comprehensive explanation of the reasoning behind each stratification, please refer to Van Wilgenburg (2020).

2.2 Ecoregion

The ecoregion is the first level of aggregation in the sampling design. The sample size and habitat inclusion probability (described in the next chapter) are defined for each separate ecoregion. There are a total of 26 ecoregions in the study area (Figure 2.2), and their details are described in Table 2.2. Ecoregion 131 was excluded from the study area because it was too small to support enough sampling points for the random sampling design.

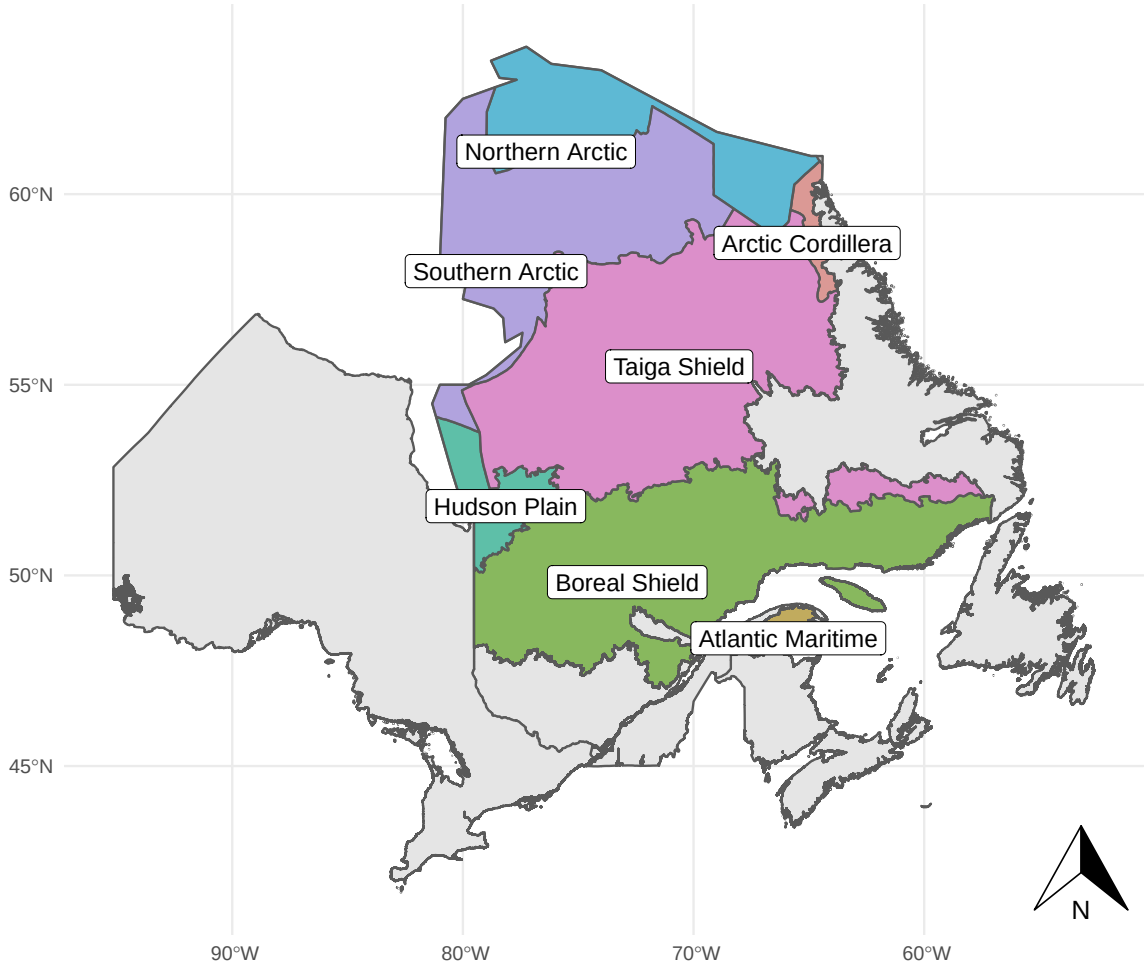


Figure 2.1: Study area (colored polygons) and its ecozone. The ecozone map was extracted from the Terrestrial Ecoregions of Canada data product - Government of Canada; Agriculture and Agri-Food Canada.

Table 2.2: Area (in hectares) and proportion of ecozones covered by the study area.

Code	Name	Area (% prop)
101	Central Laurentians	19431698 (12.35)
47	Central Ungava Peninsula	18114106 (11.51)
74	New Quebec Central Plateau	17262947 (10.97)
72	La Grande Hills	12929820 (8.22)
75	Ungava Bay Basin	9671186 (6.15)
103	Mecatina Plateau	9388108 (5.97)
100	Riviere Rupert Plateau	9083301 (5.77)
73	Southern Ungava Peninsula	8247511 (5.24)
96	Abitibi Plains	6787969 (4.31)
99	Southern Laurentians	5853410 (3.72)
48	Ottawa Islands	5766770 (3.67)
31	Northern Ungava Peninsula	5714604 (3.63)
28	Meta Incognita Peninsula	5295229 (3.37)
217	James Bay Lowlands	4482672 (2.85)
86	Mecatina River	2552186 (1.62)
46	Southampton Island Plain	2550507 (1.62)
76	George Plateau	2501318 (1.59)
30	Wager Bay Plateau	1908283 (1.21)
77	Kingarutuk-Fraser River	1861164 (1.18)
216	Hudson Bay Lowland	1752738 (1.11)
7	Torngat Mountains	1713974 (1.09)
117	Appalachians	1553719 (0.99)
78	Smallwood Reservoir-Michikamau	1148044 (0.73)
49	Belcher Islands	946717 (0.6)
102	Anticosti Island	790375 (0.5)
131	Iles-de-la-Madeleine	34149 (0.02)

The sample size for this study was determined solely based on the size of the ecoregion. While the BOSS design considered bird species richness to increase sampling in regions with more bird species, we chose not to use this metric because it may be biased by sampling efforts in the southern region, potentially increasing sampling bias in already well-covered regions. Our goal was to sample 2% of the available hexagons (PSU described below) in each ecoregion. We defined a hexagon as available for sampling if at least 20% of it contained natural habitat types.

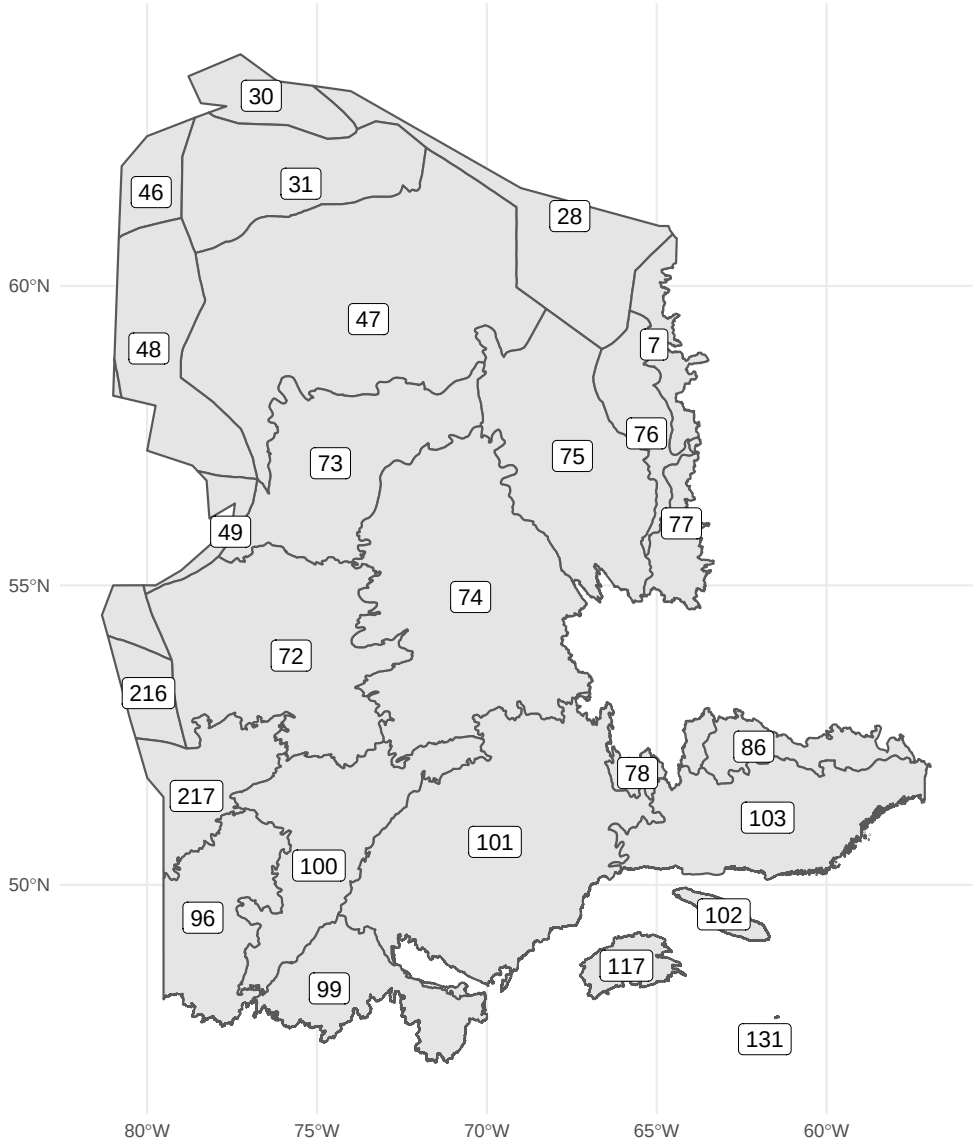


Figure 2.2: Ecoregions (code) of the study area. Code description is detailed in Table 2.2.

2.3 Primary Sampling Unit (PSU)

We followed the BOSS design by using a 5 km diameter hexagon (Figure 2.3) as the Primary Sampling Unit (PSU). This was the lowest level of aggregation before performing the stratified sample with the GRTS algorithm. We selected only the hexagons whose centroid fell within the study area. Similarly, each hexagon was classified into one of the ecoregions using the same centroid rule. The number of hexagons, available hexagones, and sample size is described in Table 2.3.

Table 2.3: Distribution of total hexagons, hexagons with at least 20% of natural habitats, and sample size (2%) across the ecoregions.

Ecoregion code	Ecoregion name	Total Hexagons	Available Hexagons	Sample size
7	Torngat Mountains	866	866	17
28	Meta Incognita Peninsula	65	65	1
30	Wager Bay Plateau	174	174	3
31	Northern Ungava Peninsula	2307	2307	46
46	Southampton Island Plain	206	206	4
47	Central Ungava Peninsula	9580	9580	192
48	Ottawa Islands	36	36	1
49	Belcher Islands	4	4	0
72	La Grande Hills	7490	7490	150
73	Southern Ungava Peninsula	5019	5019	100
74	New Quebec Central Plateau	10773	10773	215
75	Ungava Bay Basin	5549	5549	111
76	George Plateau	1482	1482	30
77	Kingarutuk-Fraser River	1184	1184	24
78	Smallwood Reservoir-Michikamau	715	715	14
86	Mecatina River	1607	1607	32
96	Abitibi Plains	4139	4139	83
99	Southern Laurentians	3563	3563	71
100	Riviere Rupert Plateau	5539	5539	111
101	Central Laurentians	11970	11970	239
102	Anticosti Island	487	487	10
103	Mecatina Plateau	5711	5711	114
117	Appalachians	957	957	19
131	Iles-de-la-Madeleine	10	10	0
216	Hudson Bay Lowland	23	23	0

Ecoregion code	Ecoregion name	Total Hexagons	Available Hexagons	Sample size
217	James Bay Lowlands	2293	2293	46

2.4 Secondary Sampling Unit (SSU)

For each PSU hexagon, we created a grid of Secondary Sampling Units (SSUs). The SSU represents the ultimate sampling locations to be utilized in the field. Instead of using the proposed 300-meter distance from the BOSS design, we followed the approach used in the Ontario regionalization design, where each SSU was separated by 294 meters. We made this small reduction in distance to ensure that the same number of SSUs were present across all PSU hexagons. Figure 2.3 displays the distribution of Secondary Sampling Units (SSUs) within a hexagon.

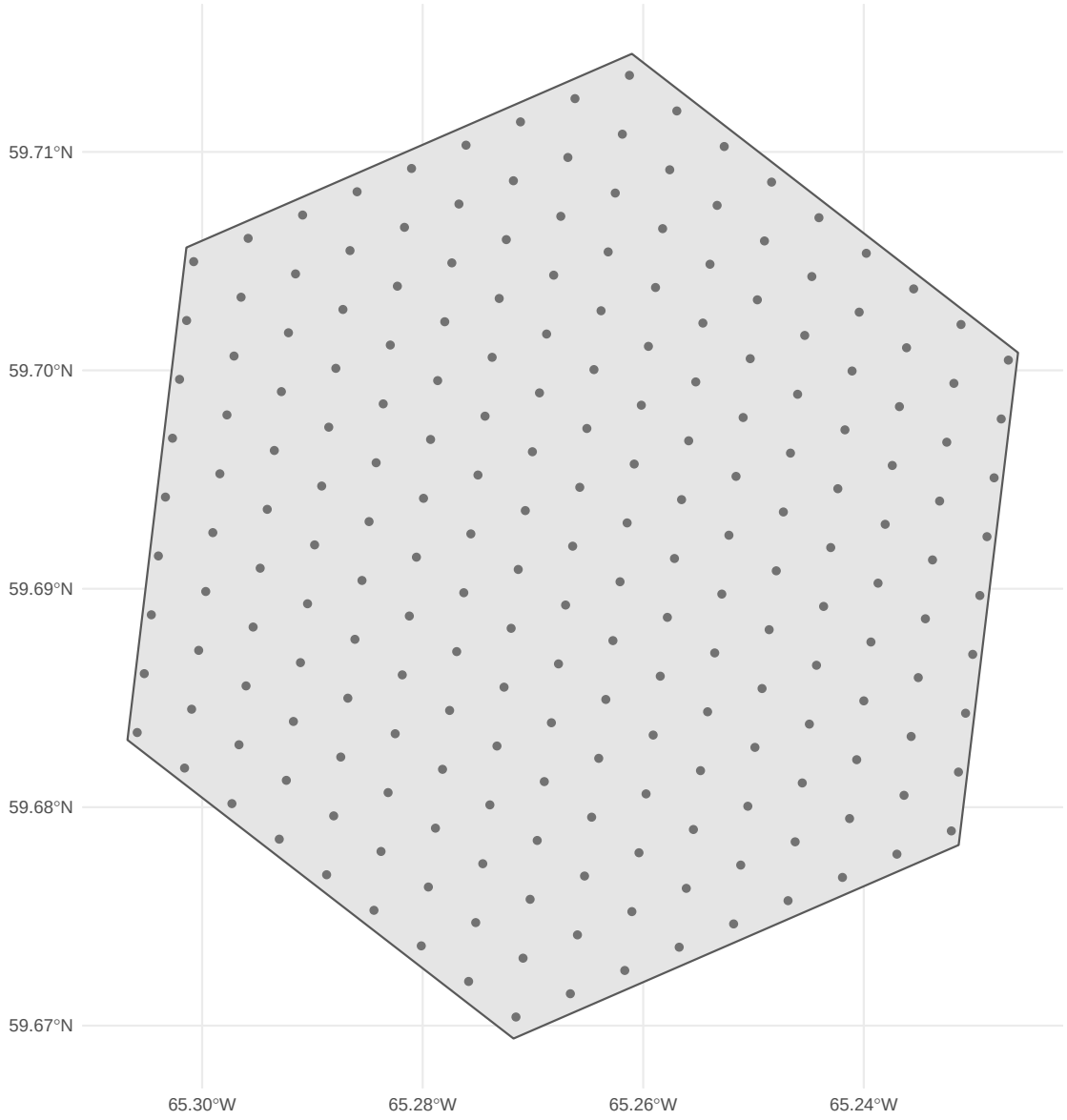


Figure 2.3: Distribution of Secondary Sampling Units (SSUs) spaced 294 meters apart within an Primary Sampling Unit (PSU) hexagon.

Part I

PSU inclusion probability

3 Habitat information

3.1 Land cover

To control for relative frequency of habitats within each ecoregion, we used the Land cover Canada and Land cover Quebec. Both layers have habitat classes at a resolution of 30 meter pixel. The Land cover of Quebec (*source?*) has 10 extra classes of habitat when compared to the Land cover of Canada and is, therefore, more precise to control for habitat heterogeneity. However, the spatial extension of this layer does not cover all the study area. To avoid conflit of different sources of information within a ecoregion, the Land cover of Quebec was only used when it covered the total area of the ecoregion (Figure 3.1). For the remaining ecoregions, we used the Land cover of Canada, version 2015 (Latifovic et al. 2016). Following the BOSS design, for both Land cover layers the Snow and ice, water, Urban, and cropland classes were excluded to keep only the classes of interest for the sampling design.

3.2 Inclusion probability

Inclusion probability based on habitat type was calculated for each ecoregion individually. Within an ecoregion, it considers the number of habitats and their relative frequency. Let $C(i, e)$ be the number of pixels from an ecoregion e that are equal to the habitat i and $\#H$ be the number of habitat classes. Then, the inclusion probability of a habitat within an ecoregion ($P_{i,e}$) is given by

$$P_{i,e} = \frac{\#H^{-1}}{C(i, e)}$$

As a result, the likelihood of a habitat being included decreases as the number of pixels increases. This weighted inclusion probability ensures that rare and abundant habitats are equally likely to be chosen.

Finally, the inclusion probability of each hexagon is calculated by taking into account the inclusion probability and the relative frequency of each habitat found within the hexagon. For each hexagon h from an ecoregion e, their habitat inclusion probability is calculated from all habitat types i following the equation:

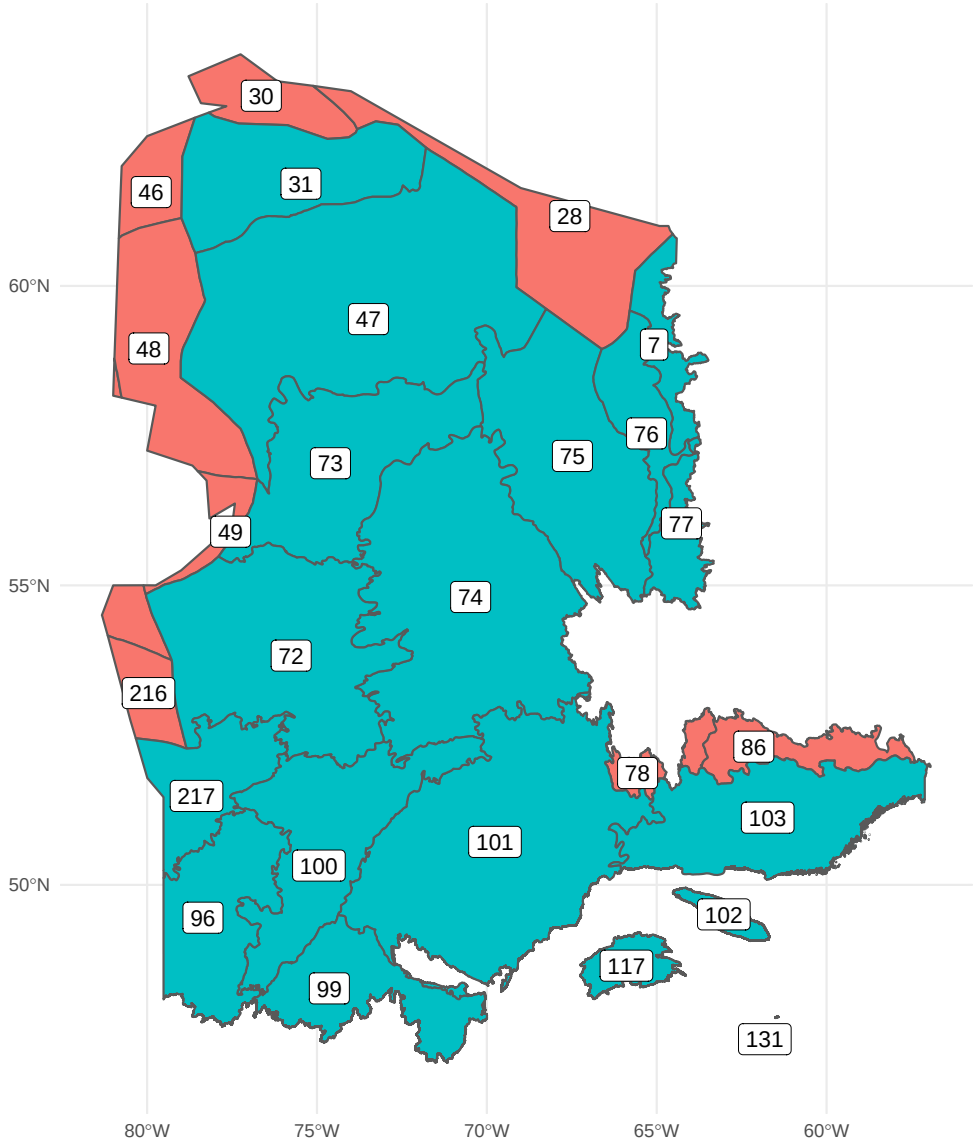


Figure 3.1: Ecoregions using Land cover Canada (red) or Land cover Quebec (blue).

$$P_{habitat_{h,e}} = \sum_{i=H_1}^{H} C(i,e) \times P_{i,e}$$

3.3 Example: Ecoregion 102

Take, for instance, the frequency of pixels per habitat type for the ecoregion 102:

Habitat type	Frequency
1	5462771
2	884
5	102195
6	250493
8	165165
10	123548
12	32971
13	3762
14	1683501
16	9437
25	31889
26	1863
27	280462
28	224313
29	18783

The habitat 1 is the most frequent while the habitat 2 the least. Following the equation above, the inclusion probability of these two habitats are 1.220382e-08 and 7.541478e-05, respectively. This means that although habitat 1 is 6180 times more frequent than habitat 2, they are equally likely to be sampled. For a visual example, we show the relative proportion of habitat from a sample of 10 hexagons with and without inclusion probability Figure 3.2.

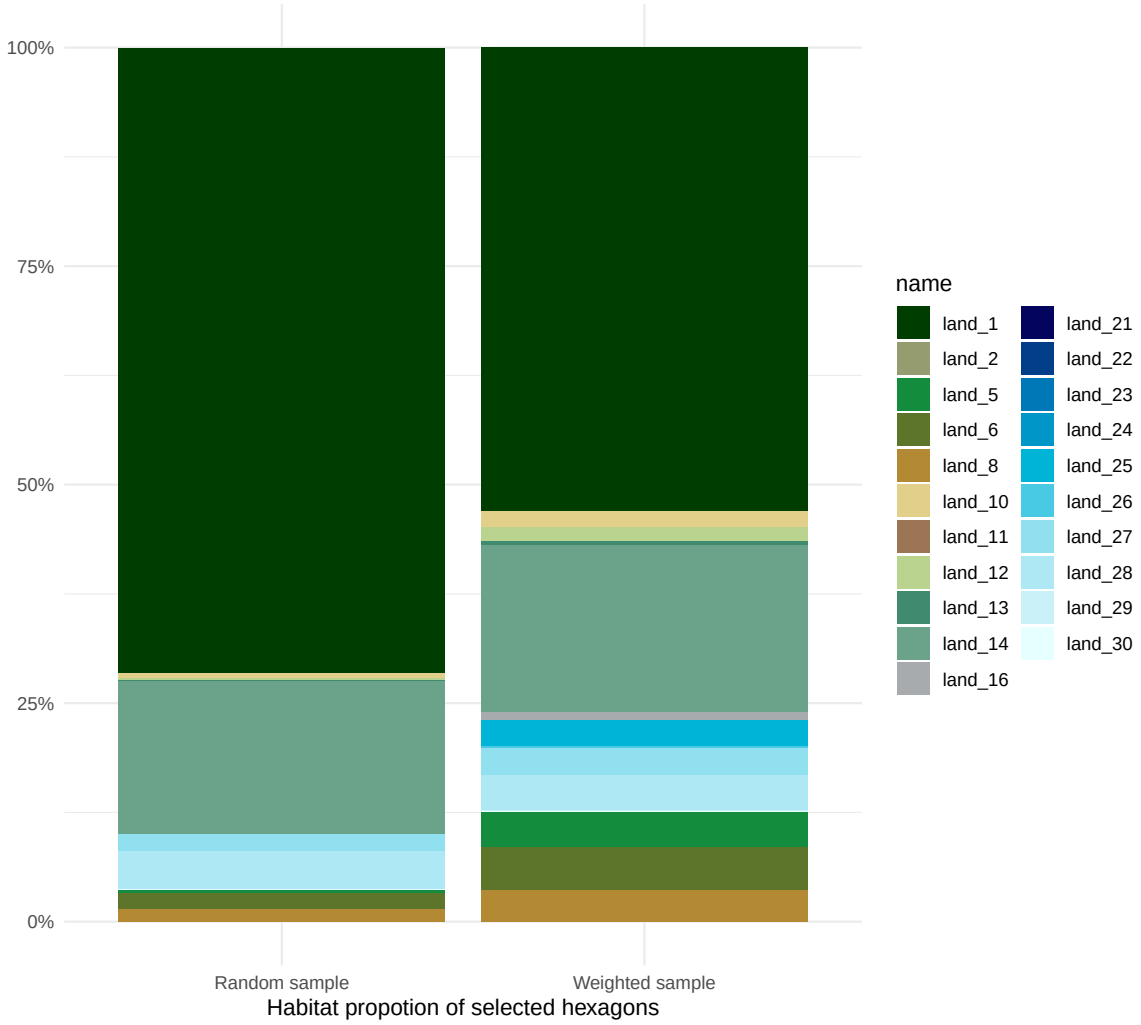


Figure 3.2: Example of sampling 10 (2%) hexagons in ecoregion 102 with (left bar) and without (right) weighted probability accounting for habitat heterogenity. Eeach bar represents the relative proportion of habitat classes from the 10 selected hexagons.

4 Cost information

In this section we define our approach to estimate the average access cost for each hexagon. We begin by describing the information layers used to define transport methods. We then define our approach to calculate the access cost for each hexagon, given the transport method and its respective parameters. Finally, we present the equations to calculate the inclusion probability given the average cost of access.

4.1 Transport layers and access cost

We used four layers of information to calculate the cheapest cost of access of a site: roads, trails, trains, and airports. Roads are used for truck transport and are mainly available in the southern part of the study area. Trails are available in more remote areas to be acessed by ATVs, but is relatively irrelevant given the amout of trails over the study area. Trails are used by ATVs to access remote areas, but their distribution overlaps with roads. For roads and trails, a buffer of size specified in Table 4.1 is defined around each line of access to create an accessible region by these methods of transport. The final access cost (AC) for a method of transport x is constant inside the buffer and is calculated as:

$$AC_x = \frac{x_{cost\_per\_day} \times nb_{arus}}{x_{arus\_per\_crew\_per\_day}}$$

The parameters are defined in Table 4.1, x is either roads or trails. We then rasterized the road and trail buffer polygons to a 30-meter resolution raster to calualte the minimum cost of access for each pixel across the study area.

For transport by helicopter, we used the airport and train layers. Since train lines provide a source of fuel for the helicopters, they are classified as a pseudo-airport for the purposes of determining the cost of access. Among all airports from Quebec and Labrador, we filtered for airport classified as `Aéroport`, `Héliport` or `Aérodrome`. We also filtered for airports that have available fuel or were from either `Hydro-Quebec` or `Administration Régionale Kativik`.

Different from roads and trails, the cost of access using helicopters was calculated at the level of the hexagon as there was little variation within a hexagon. The first step was to calculate the distance between each hexagon centroid and the closest airport or train line. Given the

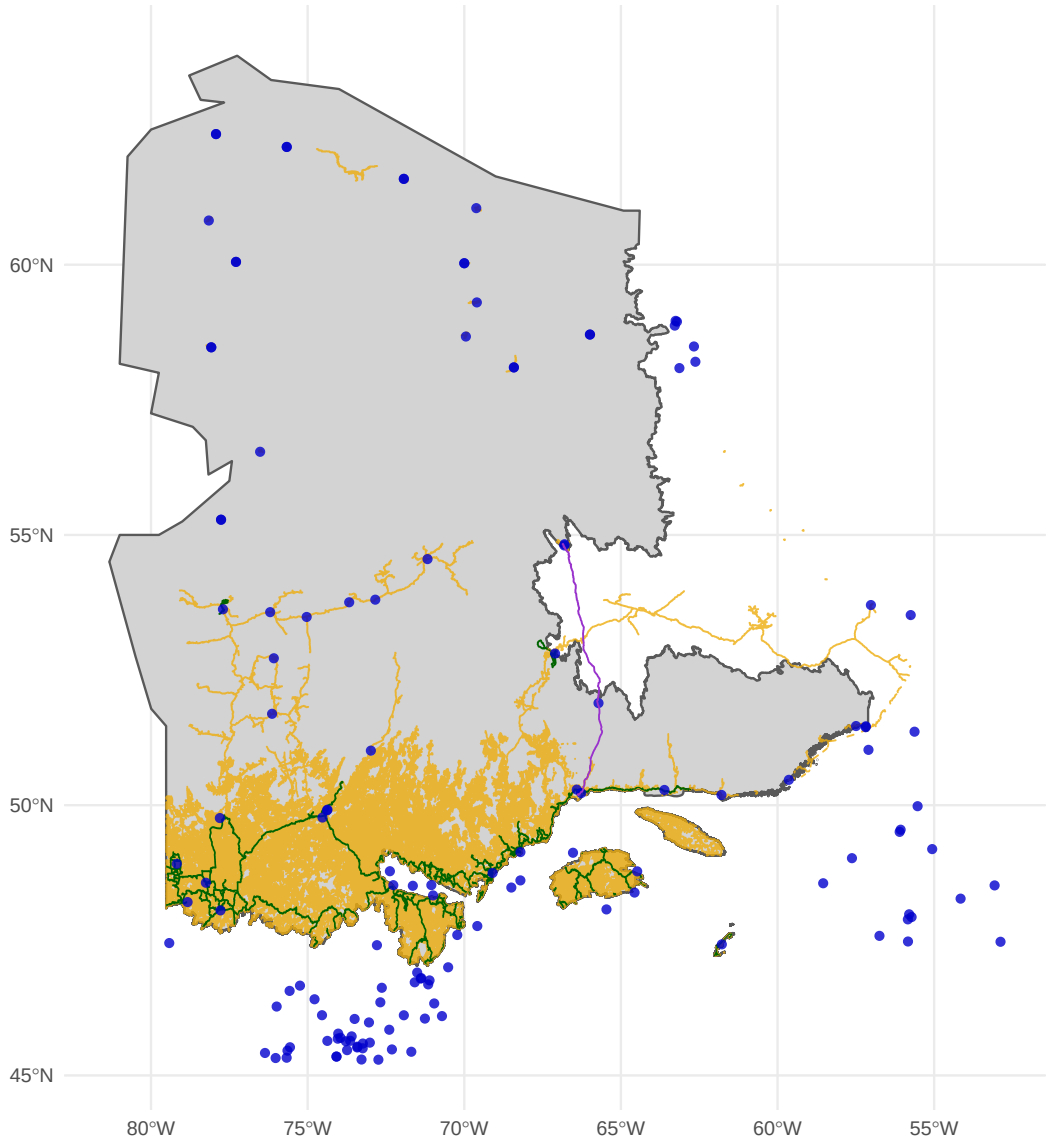


Figure 4.1: Transport layers used to calculate the access cost across the study area. Yellow for roads, green for trails, blue dots for airports, and purple for trains.

distance between a hexagon h and the closest airport or train line, the average access cost by helicopter (ACH) was defined as follows:

$$ACH_h = CS_h + CF_h + CB_h$$

Where CS is the cost of deplyoing the Autonomous recording unit (ARU) within a hexagon, CF is the cost of flying to the hexagon, and CB the cost of flying if base camp is needed. CS is calculated in function of the number of ARUs deployed in a hexagon, the time it takes to deploy an ARU, and the cost per hour of the helicopter:

$$CS_h = nb_{ARUs} \times H_{ARU} \times CH_{ARU}$$

$$H_{ARU} = \frac{helicopter_{hours\_flying\_within\_sa\_per\_day}}{helicopter_{crew\_size} \times helicopter_{aru\_per\_person\_per\_day}}$$

$$CH_{ARU} = helicopter_{l\_per\_hour} \times C_l + helicopter_{cost\_per\_hour}$$

If the distance (d) between the hexagon and the nearest airport exceeds the range of the helicopter, additional flights will be necessary, increasing the cost per litre of fuel (C_l):

$$C_l = \begin{cases} helicopter_{airport\_cost\_per\_l} & \text{if } d < helicopter_{max\_km\_from\_base} \\ helicopter_{base\_cost\_per\_l} & \text{if } d < 2 \times helicopter_{max\_km\_from\_base} \\ helicopter_{2nd\_base\_cost\_per\_l} & \text{otherwise} \end{cases}$$

The cost of flying (CF) to the hexagon from an airport is two times the distance between them, multiplied by the cost per kilometre:

$$CF_h = 2 \times d \times C_d$$

$$C_d = \frac{CH_{ARU}}{helicopter_{relocation\_speed}}$$

In case a base camp is required due to the long distance of the hexagon, the parameter d from the equation above becomes the distance between the hexagon and the base camp. Then, the cost of flying from the airport to base camp (CB) is defined as:

$$CB_h = d \times helicopter_{base\_setup\_cost\_per\_km} + \frac{2 \times d}{helicopter_{relocation\_speed} \times helicopter_{l\_per\_hour} \times C_l}$$

Table 4.1: Parameters define buffer, price, and crew support for the different methods of tranportation.

Parameter	Value
nb_arus	5
truck_buffer	1000
truck_cost_per_day	600
truck_n_crews	2
truck_arus_per_crew_per_day	5
atv_buffer	1000
atv_cost_per_day	1200
atv_n_crews	2
atv_arus_per_crew_per_day	3
helicopter_cost_per_hour	1250
helicopter_max_km_from_base	150
helicopter_base_setup_cost_per_km	9
helicopter_l_per_hour	160
helicopter_crew_size	4
helicopter_aru_per_person_per_day	5
helicopter_relocation_speed	180
helicopter_airport_cost_per_l	1.3
helicopter_base_cost_per_l	5
helicopter_2nd_base_cost_per_l	10
helicopter_hours_flying_within_sa_per_day	5

4.2 Inclusion probability

To compute the inclusion probability of a hexagon in function of its accessibility, we calculated a weighted access cost based on the proportions of each access method used in the hexagon. We used the raster with the minimum cost among roads and trails to estimate the proportion of the hexagon that is accessible by land. The rest of the hexagon that is not covered by the road and trail buffer is then only accessible by helicopter. Given all available methods of transport in a hexagon, the weighted average cost is defined as the sum of the cost of each of these methods, weighted by their proportion in the hexagon:

$$W_{\text{averge cost}_h} = \frac{\sum_{i=1}^{n} w_i Cost_i}{\sum_{i=1}^{n} w_i}$$

Then, for a hexagon h and each method of transport i, its cost inclusion probability P_{cost_h} is given by:

$$P_{cost_h} = \frac{1}{\sqrt{W_{\text{averge cost}_h}}}$$

Where w_i is the weight given by the proportion of the hexagon accessible by the method i.

5 Legacy and iconic sites

In this section we discuss our approach to account for legacy and iconic sites in the sampling design. Legacy sites are historical or current surveys in which locations were randomly chosen, in contrast to iconic sites, which are obtained from non-random surveys. In this report, we will refer to both iconic and legacy sites as historical sites. These sites are particularly important in Quebec, where there are a large number of legacy and iconic sites (Figure 5.1). Given the high cost of sampling in remote regions and the limited resources available to sample, it is essential to take historical sites into account in order to make the most efficient use of limited resources to extend spatial coverage to under-sampled areas.

The latest method to integrate legacy and iconic sites is developed in Foster (2021). They use the position of each historical site to reduce the inclusion probability of neighboring sites following a kernel distribution, where the legacy effect decreases with distance from a historical site. However, their approach to incorporate legacy and iconic sites do not explicitly consider the amount of historical sites, nor the spatial randomness of their distribution. In fact, their approach naturally ensures randomnly selected sites, as they use only legacy sites while excluding iconic ones. They justify the exclusion of iconic sites, since these sites are usually special cases that may not represent the population and therefore generate a biased sample. Although we agree with the authors, we prioritised expanding the spatial coverage of samples by reducing the sample size of regions that have been well covered by historical sites.

In the following section we describe the historical sites for the study area. Then, we use two ecoregions as examples to discuss the decision to also consider sample size while accounting for historical sites. In the next chapter we describe our novel approach to account for the spatial randomness of historical sites to adjust the inclusion probability and the sample size. Finally, we briefly describe previous experimentations to incorporate historical sites.

5.1 Historical sites in the study area

TODO: describe the different sources

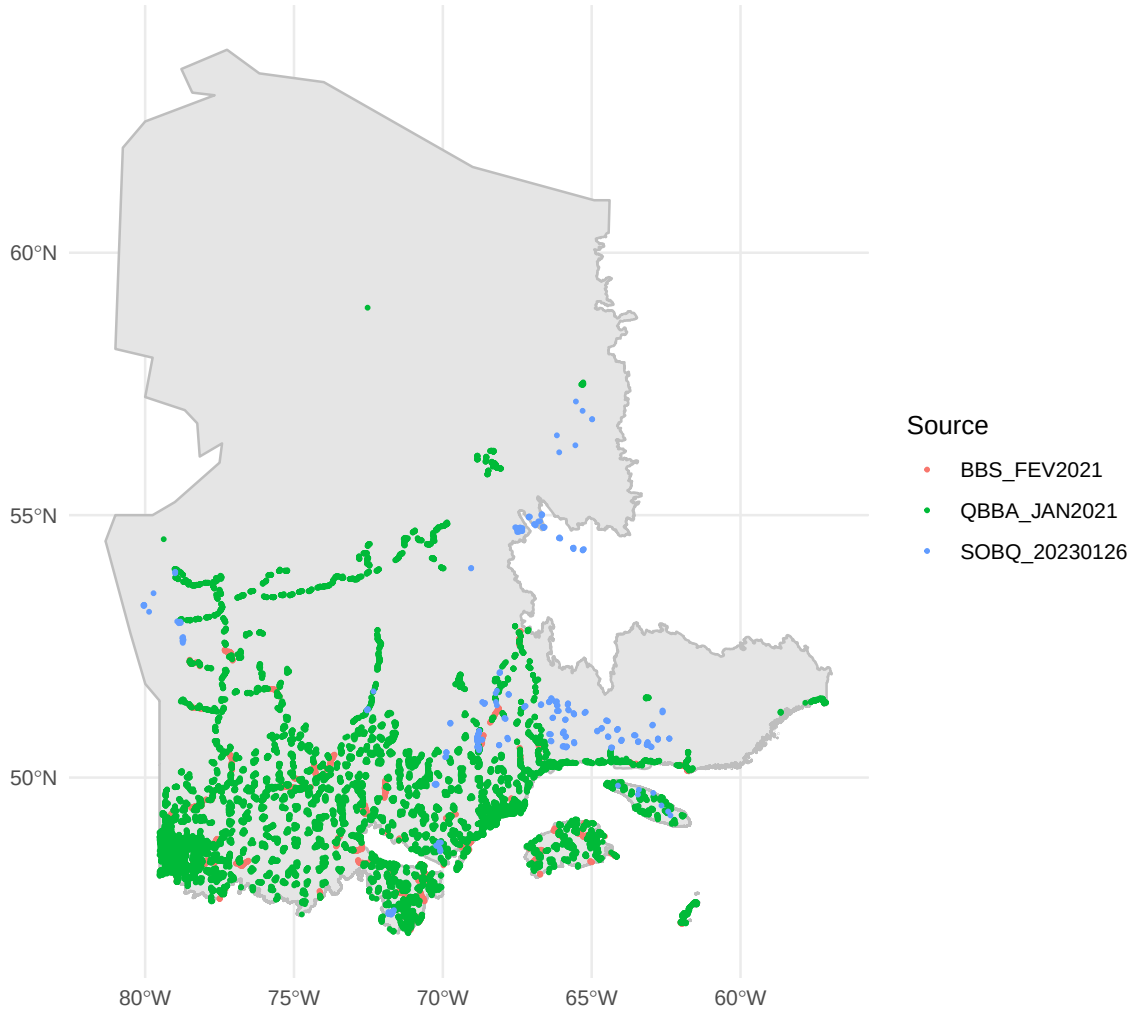


Figure 5.1: Legacy and historical sites from the study area.

5.2 Why should we consider sample size?

Here, we discuss why changing inclusion probability alone is insufficient by comparing two ecoregions with oposite spatial distributions of historical sites. While the ecoregion 99 has evenly distributed historical sites over space, the ecoregion 217 present most of its historical sites around roads. We begin by outlining the method for reducing the inclusion probability of sites near legacy sites from Foster (2021), which we then apply to the two contrasting ecoregions.

Every PSU hexagon is assigned an inclusion probability as a function of habitat and access cost. The effect of a legacy site (l) on the inclusion probability of neighbouring hexagons (h) is derived from the Euclidean distance between h and l ($d(h, l)$), and the amplitude effect σ using a Gaussian function. Figure 5.2 shows the one-dimensional effect of increasing values of σ on the inclusion probability of neighbouring sites.

The method to reduce the inclusion probability of neighbouring sites near legacy sites developed in Foster (2021) is implemented in the R package `MBHdesign`. We redrew their illustration using a simulated grid landscape with three legacy sites to ilustrate the effect of increasing the σ parameter on the inclusion probability of adjacent sites (Figure 5.3).

We applied the same approach for two ecoregion with contrasting spatial distribution of historical sites. By using the spatially balanced design to sample new sites while omitting historical sites, the selected hexagons are evenly distributed throughout the ecoregion, some of which overlap with the historical sites (Figure 5.4). Adjusting the probability of inclusion of hexagons, given their distance to historical locations, increases the spatial coverage of underrepresented areas (Figure 5.5).

The issue is that when an ecoregion is well covered by historical sites, the sampled hexagons are driven to cluster in smaller available areas. When the inclusion probability was adjusted for ecoregion 99, almost a third of the sampled hexagons were clustered in a 20 km buffer. In the next section, we will build on this approach so the sample size may also be adjusted in response to historical sites.

5.3 A new approach to integrate legacy and iconic sites in spatially balanced designs

Current methods to include historical surveys are limited to randomly selected sites. While they adjust the inclusion probability to avoid sampling near historical sites, there is no consideration for sample size adjustment. Here we propose a novel approach where, in addition to modifying the inclusion probability, we also adjust the sample size according to the distribution of historical sites.

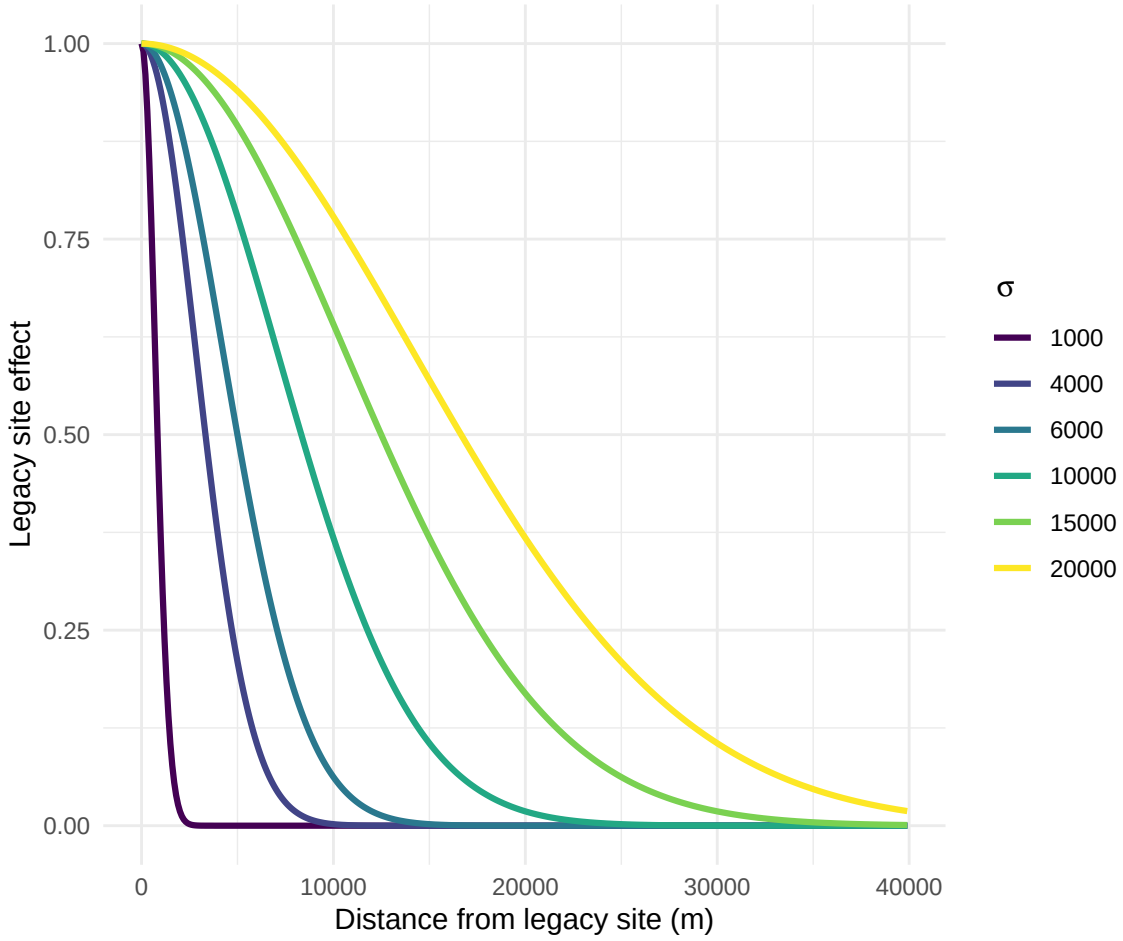


Figure 5.2: Legacy effect in the inclusion probabiity of neighbouring sites in function their Euclian distance. Each line represents the effect range of legacy sites. Figure adapted from the vignette 'An Introduction to MBHdesign' found in Foster (2021).

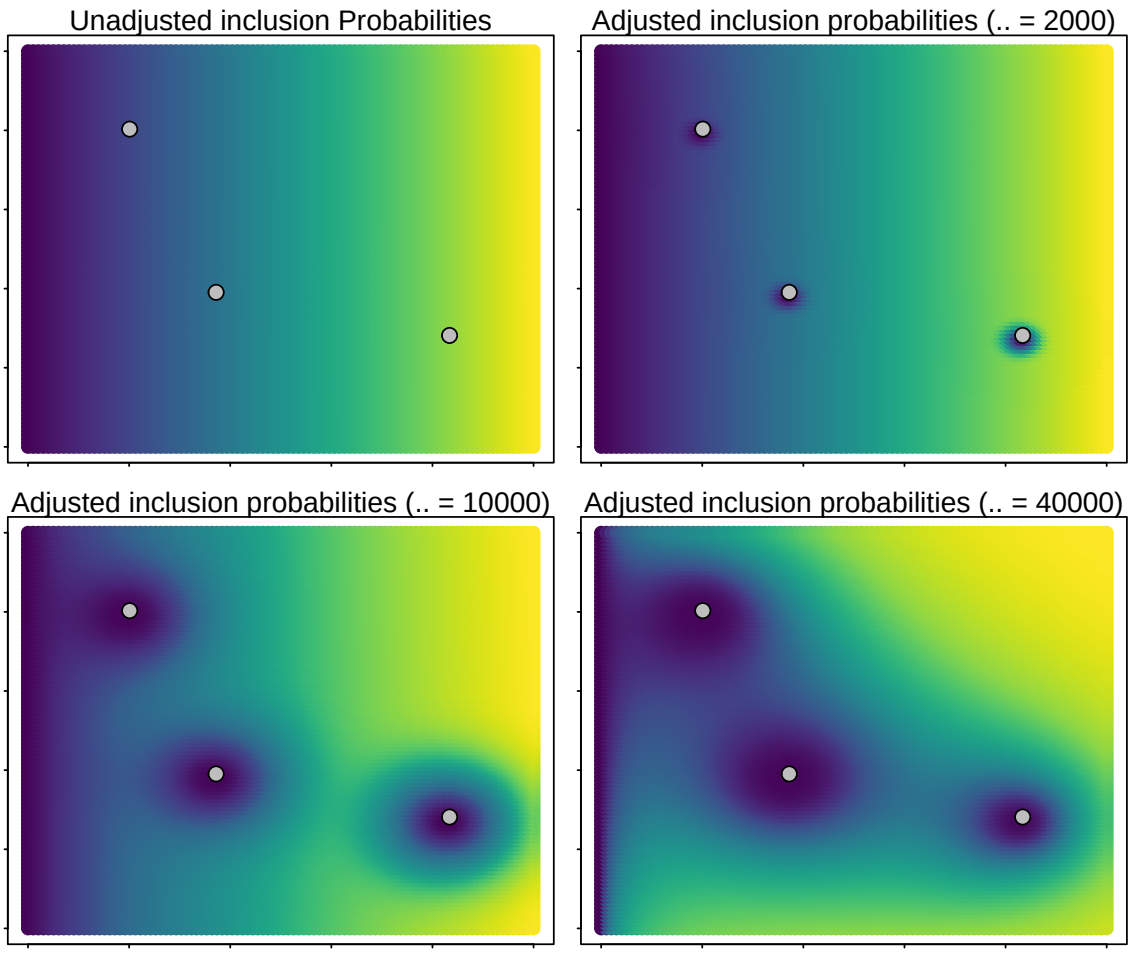


Figure 5.3: Effect of increasing the effect (sigma) of three legacy sites (white dots) on the inclusion probability of neighbouring sites. Each square is a simulated sampling grid with different inclusion probability described by the color gradient. Inclusion probability increases from dark blue (low) to yellow (high).

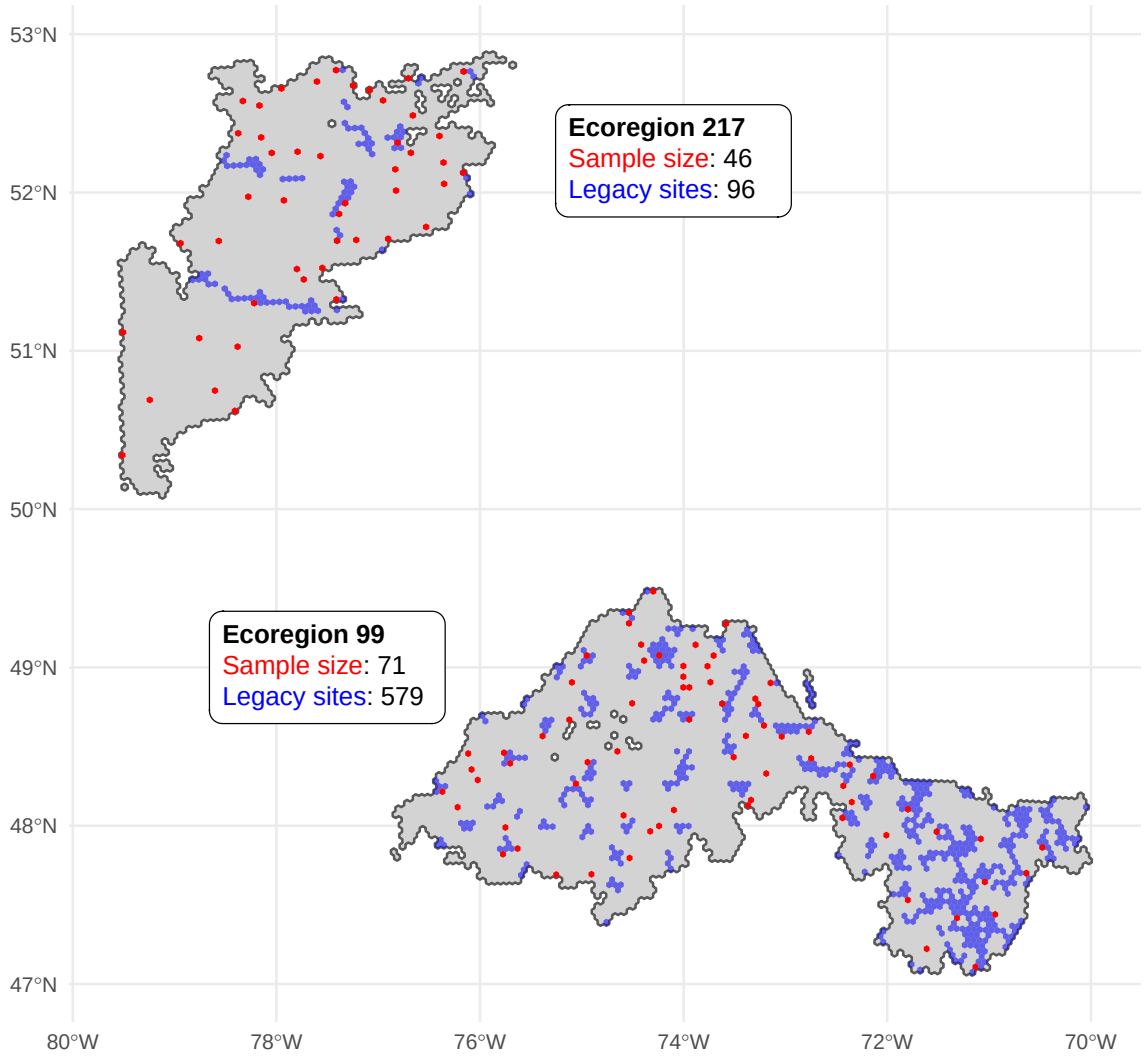


Figure 5.4: Example of when sampling sites (red dots) using the spatially balanced design without taking historical sites into account. Ecoregion 99 and 207 are two examples of legacy sites in which the legacy sites (blue dots) are evenly and non-uniformly distributed in space, respectively.

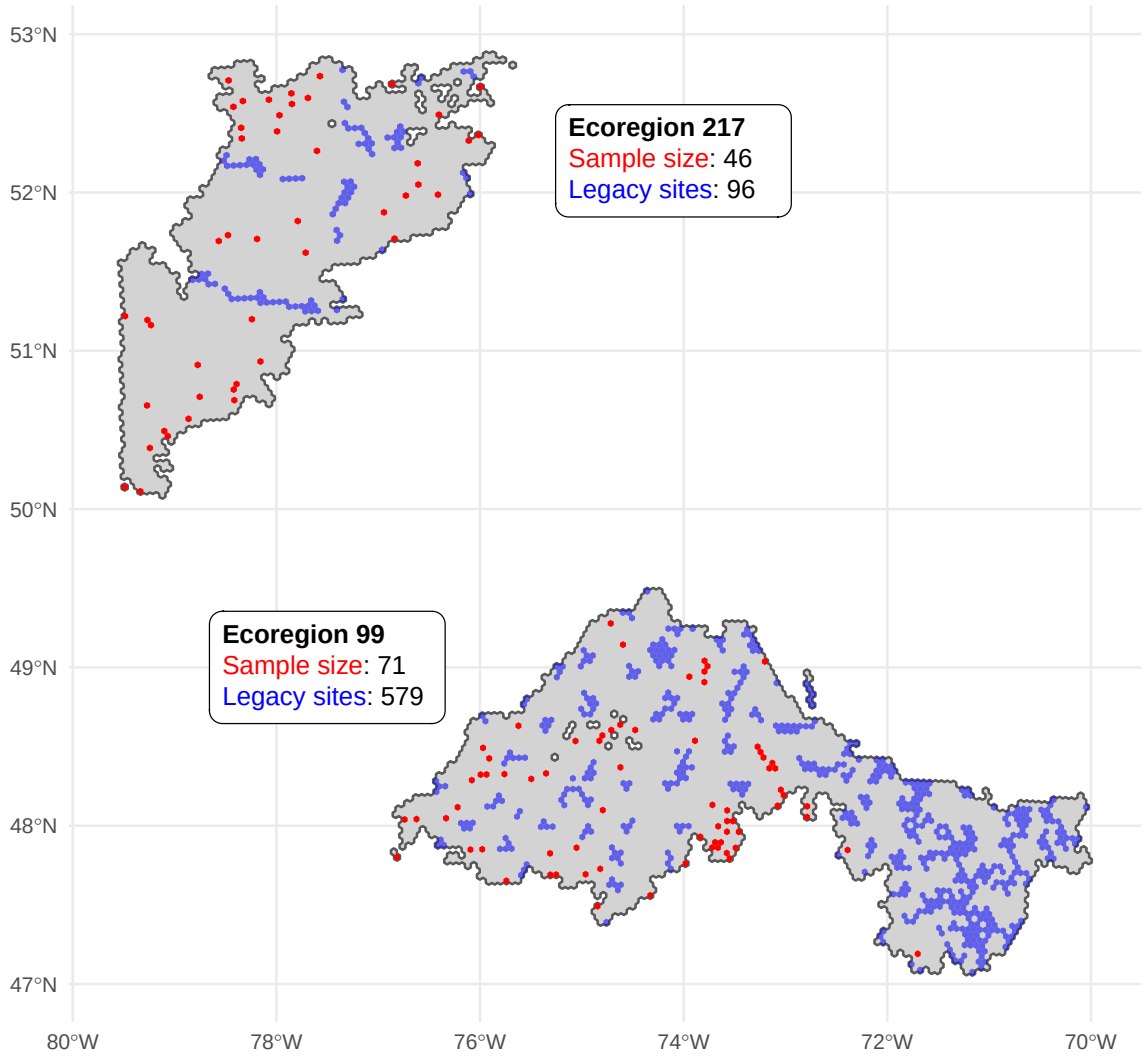


Figure 5.5: Example of when sites are sampled from a design in which only the inclusion probability is adjusted when incorporating historical sites. Ecoregion 99 and 207 are two examples of legacy sites in which the legacy sites (blue dots) are evenly and non-uniformly distributed in space, respectively.

Theoretical description

We use a simulated grid with equal inclusion probabilities for all the 7661 hexagons to illustrate our approach. Given a 2% sample effort, the sample size for the model grid is 153 hexagons. We used two scenarios with contrasting spatial distributions of historical sites to show how their distribution affect sample size (Figure 5.6). The first scenario (red dots) describe evenly distributed historical sites, while the second (blue dots) describes historical sites grouped in two spatial clusters.

The main idea of adjusting the inclusion probability is that the population present in the region around historical sites are well represented by the existing sample. We build on this rationale to propose that the sample size of an ecoregion can be similarly adjusted given the spatial coverage of historical sites within the ecoregion. The total coverage of a historical site can be defined by a buffer of size s. Then, the total area of the ecoregion is subtracted by the total area covered by historical sites to determine the final sample size for an ecoregion. By adding a buffer size to the historical locations in our grid example, the balanced scenario covered 14% of the total area, whereas the unbalanced scenario only covered 7.5% (Figure 5.7). Due to their clustered distribution, the unbalanced historical sites in this example are only 50% as effective at reducing the sample size as the balanced sites. This approach hopes to solve the two issues that emerged when only inclusion probability was adjusted. First, it offers the opportunity to use both legacy and iconic sites, regardless of the randomness of their distribution. Finally, it enables us to accommodate the number and distribution of historical sites in order to adjust the sample size and avoid clustered samples in ecoregions with adequate coverage.

The unsolved issue is how to define the optimal value for the coverage of a historical site (s). In a perfect spatial balanced distribution of historical sites, the adjusted sample size of an ecoreregion should be reduced by the total number of historical sites. Take, for instance, an ecoregion with initial sample size of 80 hexagons. The presence of 10 spatially balanced historical sites should yields an adjusted sample size of 70 hexagons. Then, the optimal buffer size s must produce an adjusted sample size (N_{adj}) that equals the difference between the initial sample size (N_{base}) and the number of historical sites (N_{hist}), given the historical sites are spatially balanced across the ecoregion (Figure 5.8).

With this approach, we are able to determine the ideal buffer size that will allow us to reduce the sample size without compromising the spatial representation of the initial sampling effort. This simulation-based strategy, however, is sensible for the fixed number of historical sites determined a priori. In the example grid used in Figure 5.8, we simulated 20 historical sites, which represents a 0.2% of the total hexagons in the sample grid. We simulated different sets of historical sites ranging from 5 (0.07%) to 1400 (20%) to illustrate its impact on the effect of buffer size on the adjusted sample size (Figure 5.9).

As expected, increasing the proportion of historical sites increases the rate of change (slope) of the buffer size effects on the adjusted sample size. Using the same simulations, we computed the optimal buffer size for each of the proportion of historical sites ($\frac{N_{hist}}{N_{total}}$) so the adjusted

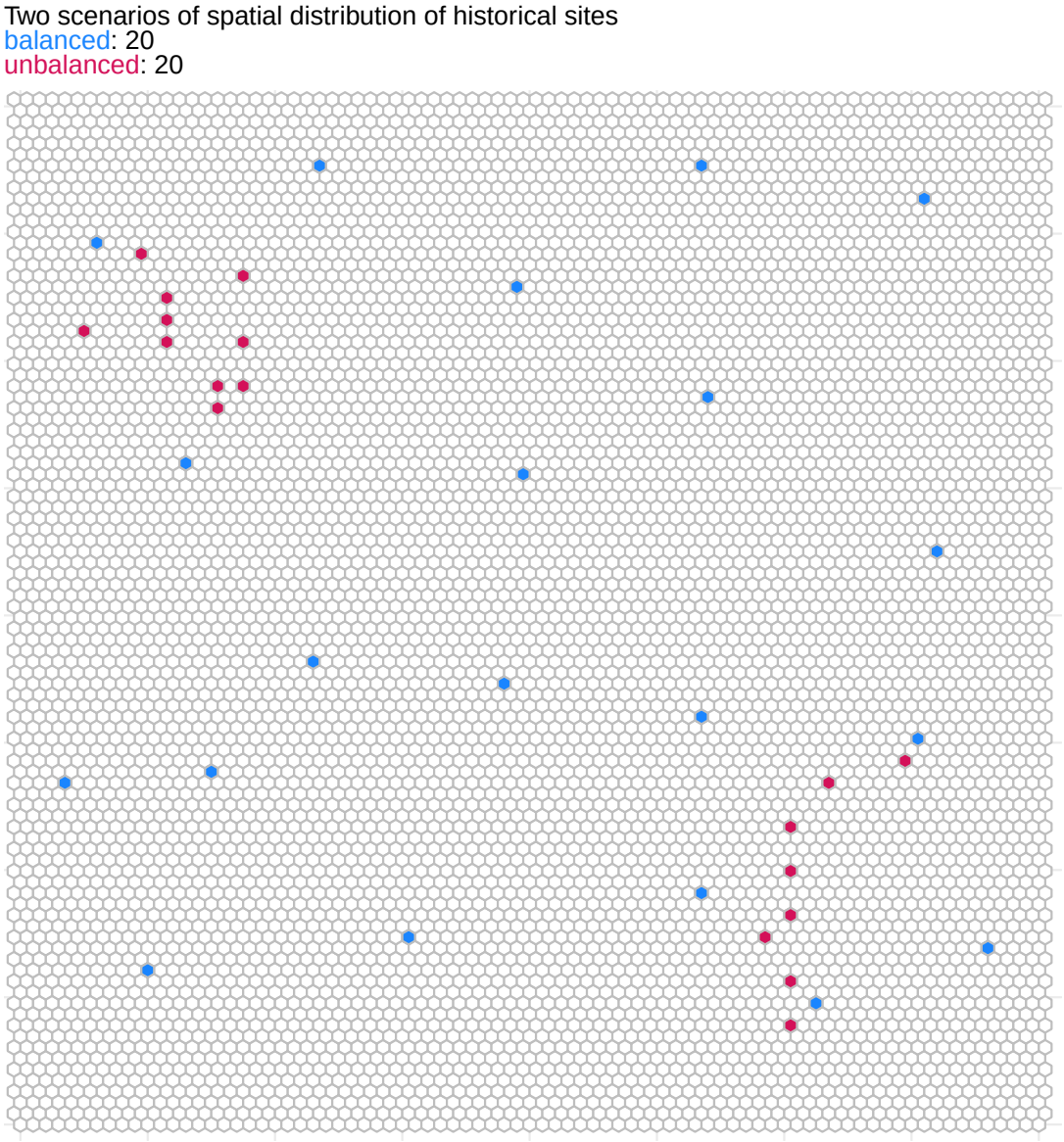


Figure 5.6: Simulated grid with equal inclusion probabilities among the 7661 hexagons. Two scenarios of historical sites consisting of 20 sites each illustrate the effect of evenly distributed (blue) and clustered historical sites on the sample size.

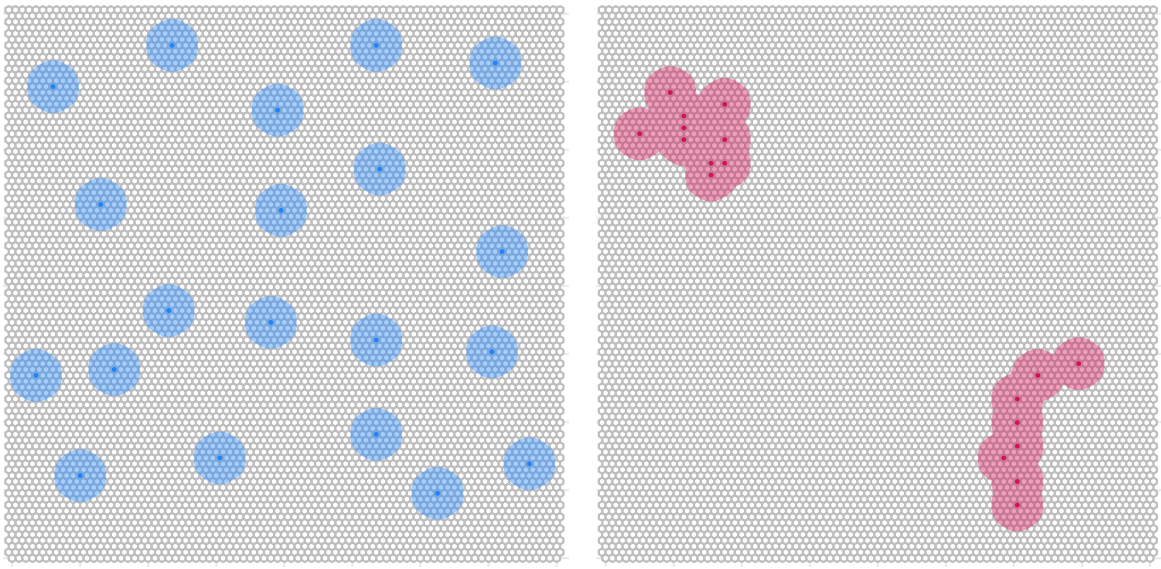


Figure 5.7: Grid of sample units and historical sites spatially balanced (blue) and unbalanced (red) for a simulated region. Using the same approach for adjusting the inclusion probability, a buffer around each historical site is created to calculate the adjusted sampled size.

sample size (N_{adj}) equals the difference between the initial sample size (N_{base}) and the number of historical sites (N_{hist}) (Figure 5.10). Interestingly, increasing the proportion of historical sites increases the optimal buffer size up to a point until it reaches the sample effort (specified to 2% for the figure). The optimal buffer size reduces as the proportion of historical sites increases for proportions greater than the specified sampling effort.

To illustrate the effect of different sampling efforts on the relationship between the proportion of historical sites and the optimal buffer size, we simulated sampling efforts ranging from 1 to 20%. (Figure 5.11).

The fact that the optimal buffer size changes with the number of historical sites in a region may be related to buffer coverage overlapping between historical sites and the border effects where coverage does not have effect on sample size. As a result, the optimal buffer size may also depend on the size and shape of the sample grid.However, given that the shape and size of the ecoregions are not adjustable, we decided to not test for these variables.

Results

Following the theoretical development, we performed the identical simulations using actual ecoregion boundaries rather than simulated grids. We simulated spatially balanced historical

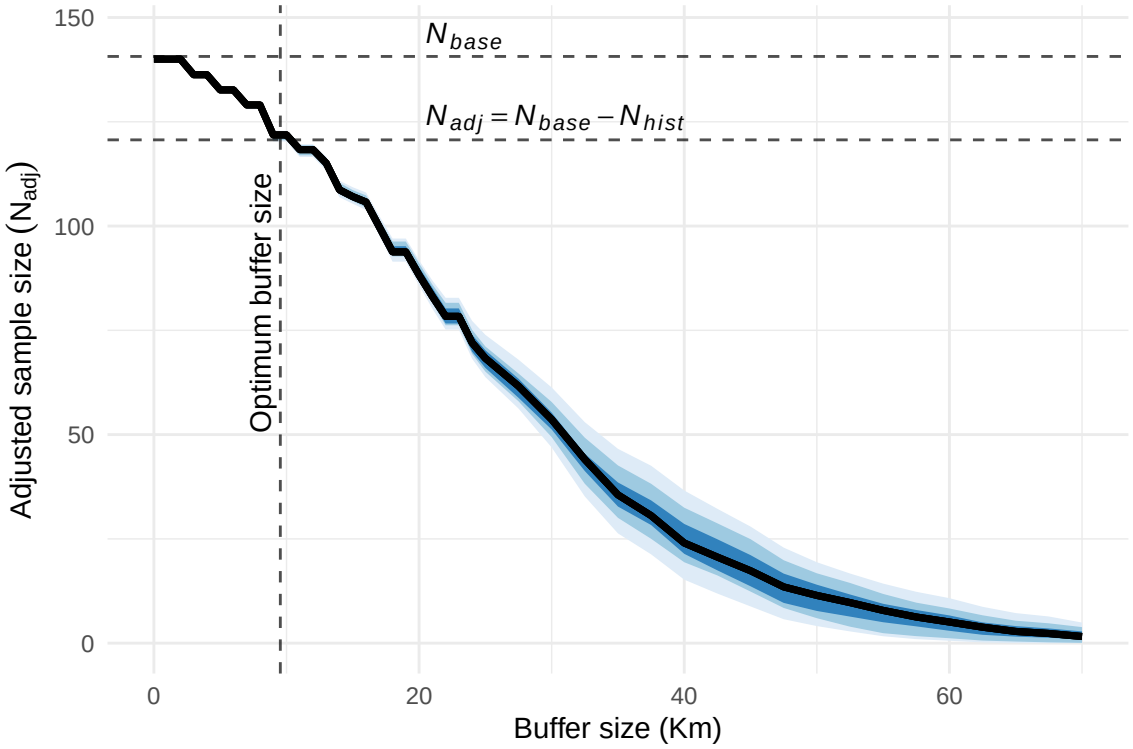


Figure 5.8: Effect of the buffer size, determining the coverage of historical sites, on the adjusted sample size for a simulated grid with equal inclusion probabilities. Given a spatially balanced distribution of historical sites, the optimal buffer size is determined when the adjusted sample size equals the difference between the initial sample size (N base) and the number of historical sites (N hist). The color gradient around the mean black line represents the 95%, 80%, and 50% confidence intervals for 50 replications of random generated spatially balanced historical sites.

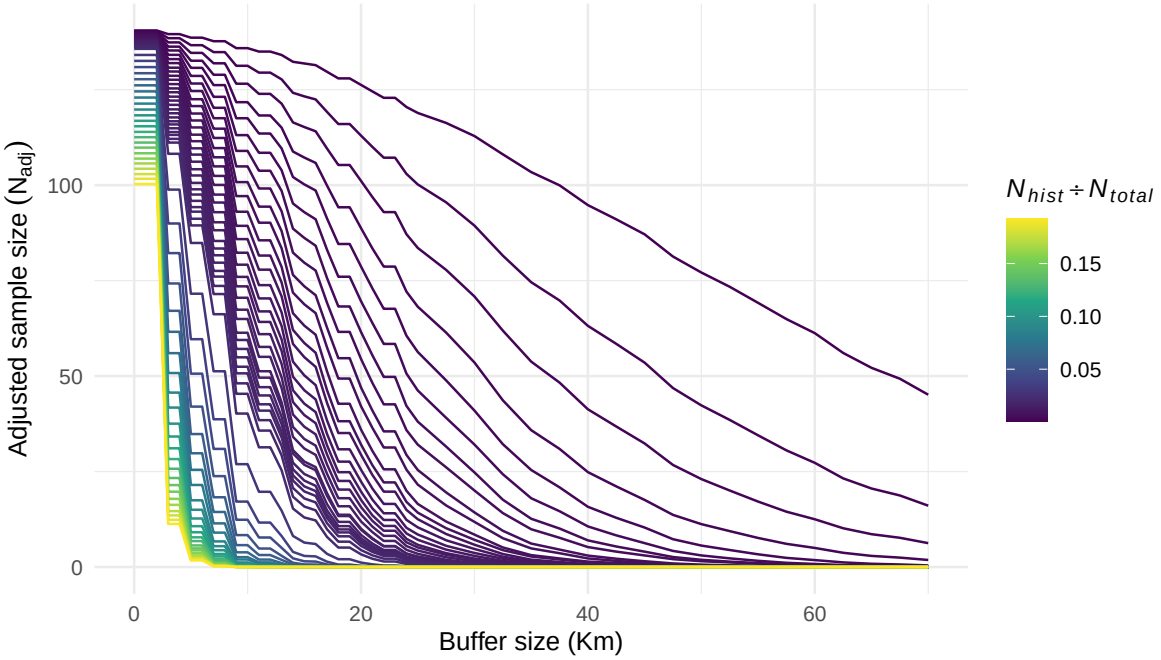


Figure 5.9: Effect of the buffer size on the adjusted sample size while controling for relative abundance of historical sites in the sample grid. Relative proportion of historical sites in the sample grid ranged from 0.07 to 20%.

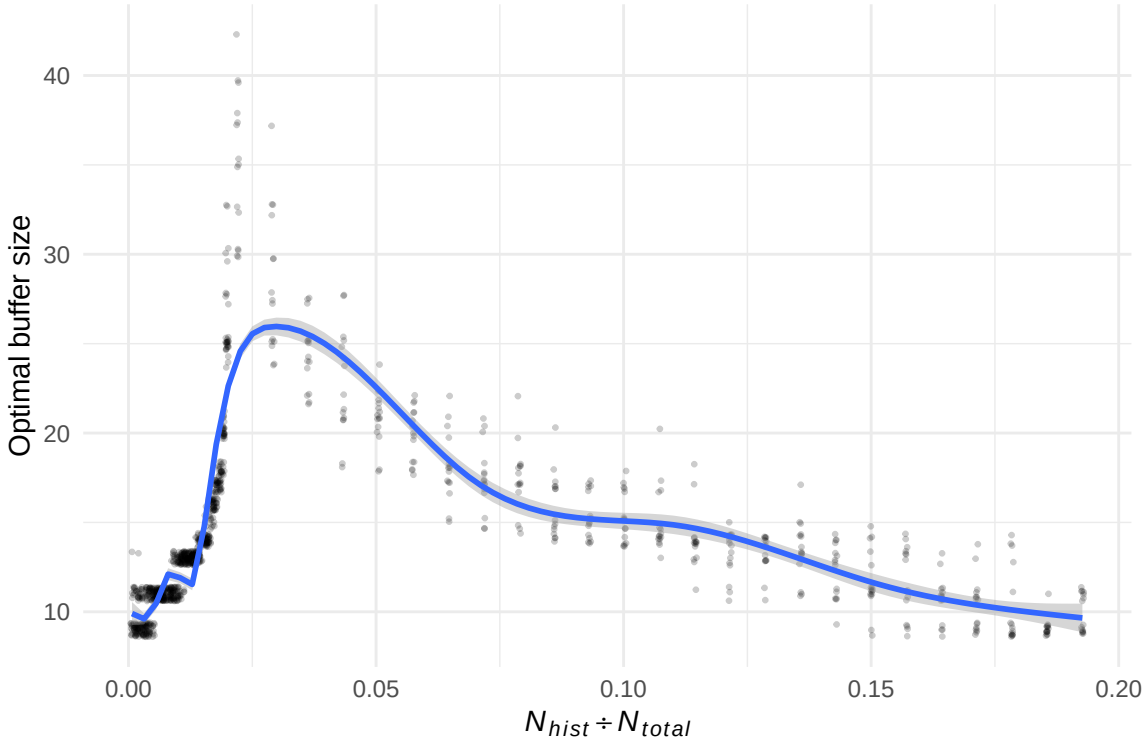


Figure 5.10: Optimal buffer size in function of the proportion of historical sites to the total number of hexagons. Each dot is a replication of the simulated increase of proportion of historical sites. A small amount of noise around each dot was added for visualization purposes. Blue line is smooth estimation using Generalized additive model to demonstrate the exponential effect of the proportion of historical sites on the optimal buffer size.

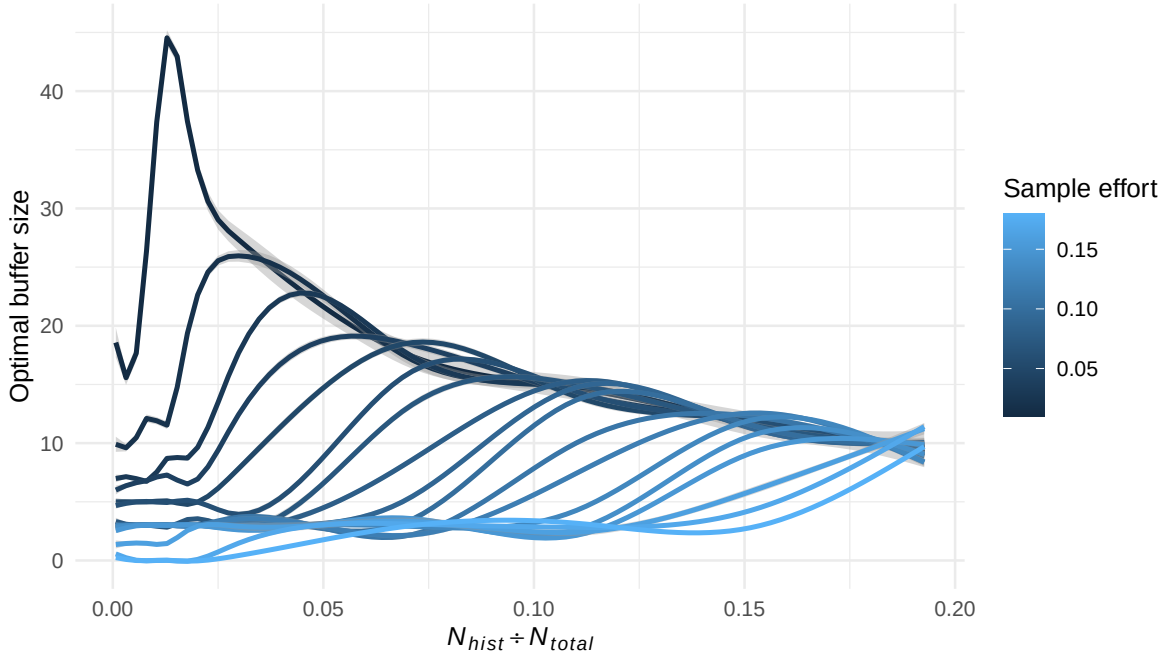


Figure 5.11: Optimal buffer size in function of the proportion of historical sites for different sampling efforts ranging from 1 to 20%. For clarity, we choose to omit the data points and illustrate only the smooth estimation using Generalized additive model.

sites across the ecoregions with relative proportions to the total number of hexagons ranging from 0.2 to 12%. For each value of proportion, we calculated the adjusted sample size for different buffer sizes with radius ranging from 4 to 40 km. Ecoregions 131, 28, 30, 46, 48, 49, and 216 were excluded due to the very small size that would result in zero generated historical size for most of the simulation sets.

The effect of buffer size and the proportion of historical sites on the adjusted sample size using the ecoregion boundaries were similar to those performed using a simulated grid (Figure 5.12). The effect of increasing the buffer size on the reduction of the adjusted sample size is non-linear, with the rate of change increasing as the number of historical sites rises.

This non-linear relationship can be observed in the effect of the number of historical sites on the optimal buffer size when calculating the optimal buffer size. (Figure 5.13). The optimal buffer size increases with the number of historical sites up to a point where it decreases for most ecoregions as the proportion of historical sites increases. Only the ecoregion 131 show a slight decrease in optimal buffer size with the number of historical sites.

Across the simulations and replications, the distribution of optimal buffer size follows a bimodal distribution.(Figure 5.14).

The case of ecoregions 99 and 217

We apply our approach to reduce the same size in function of the number and distribution of historical sites for two contrasting ecoregions. The ecoregions 99 and 217 have 579 (16.2%) and 97 (3.5%) hexagons classified as historical sites, respectively. While the number of historical sites outweighed the sampling effort (2%), the spatial distribution of these sites varied depending on the ecoregion. Spatial distribution of historical sites was balanced in ecoregion 99, but clustered in ecoregion 217. As a result, the adjusted sample size for these two ecoregions should also be different.

The optimal buffer size in fuction of the proportion of historical sites for the ecoregions 99 and 217 was 19.2 and 30.8 Km, respectively. The adjusted sample size taking into account the distribution and number of historical sites was reduced from 72 to 3 for ecoregion 99, and from 56 to 22 for ecoregion 217 (Figure 5.15).

Limitations

In order to achieve an optimal design, additional simulations may be required to fine-tune the parameters. While we have accounted for spatial coverage in adjusting the sample size, this approach could be also extended to include habitat distribution in order to better account the effect of legacy sites in the sampling design.

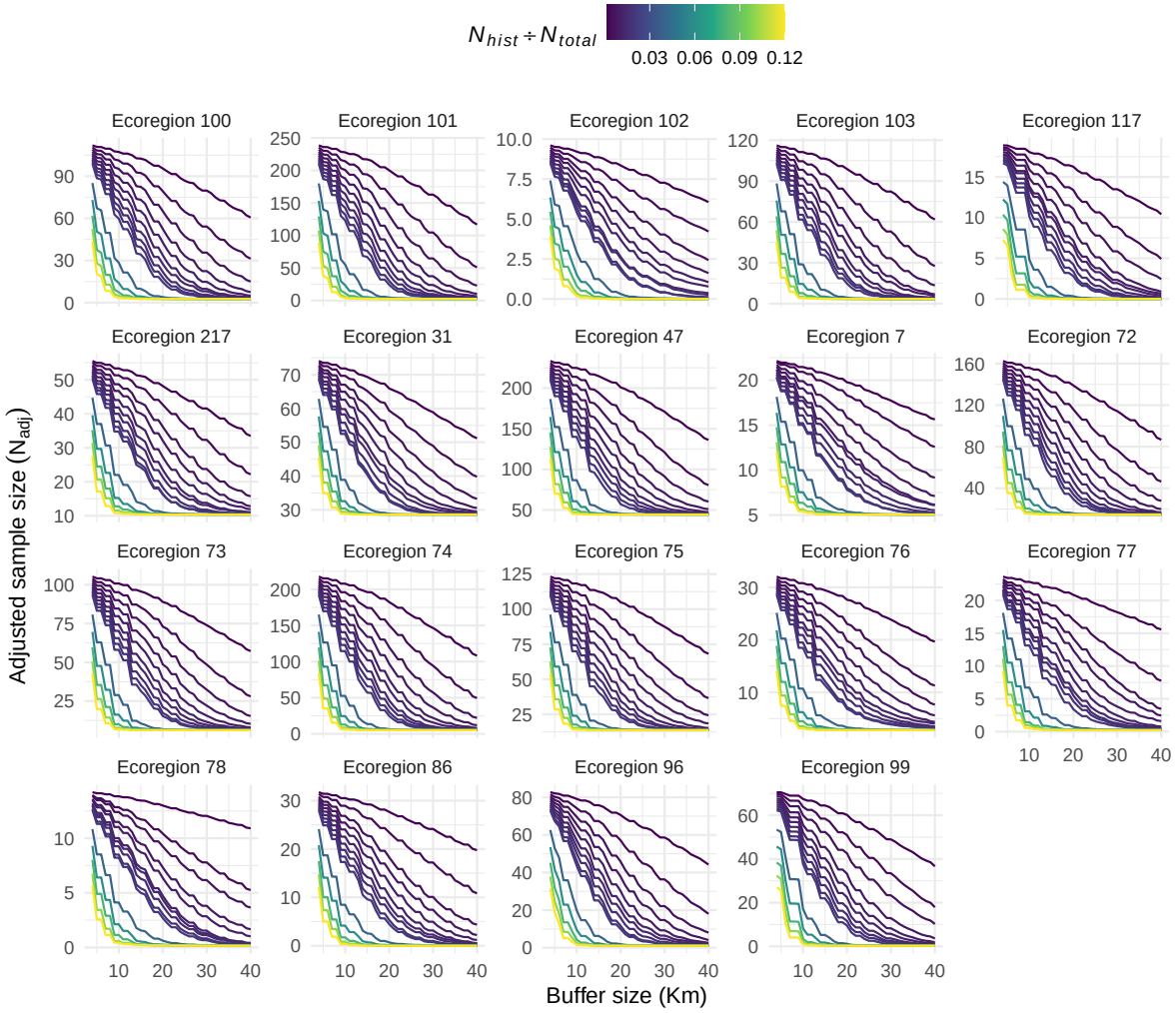


Figure 5.12: Adjusted sample size in function of buffer size for different proportion of historical sites compared to initial sample size. When buffer size and proportion of historical sites equals zero, adjusted sample size is equal to the initial sample size.

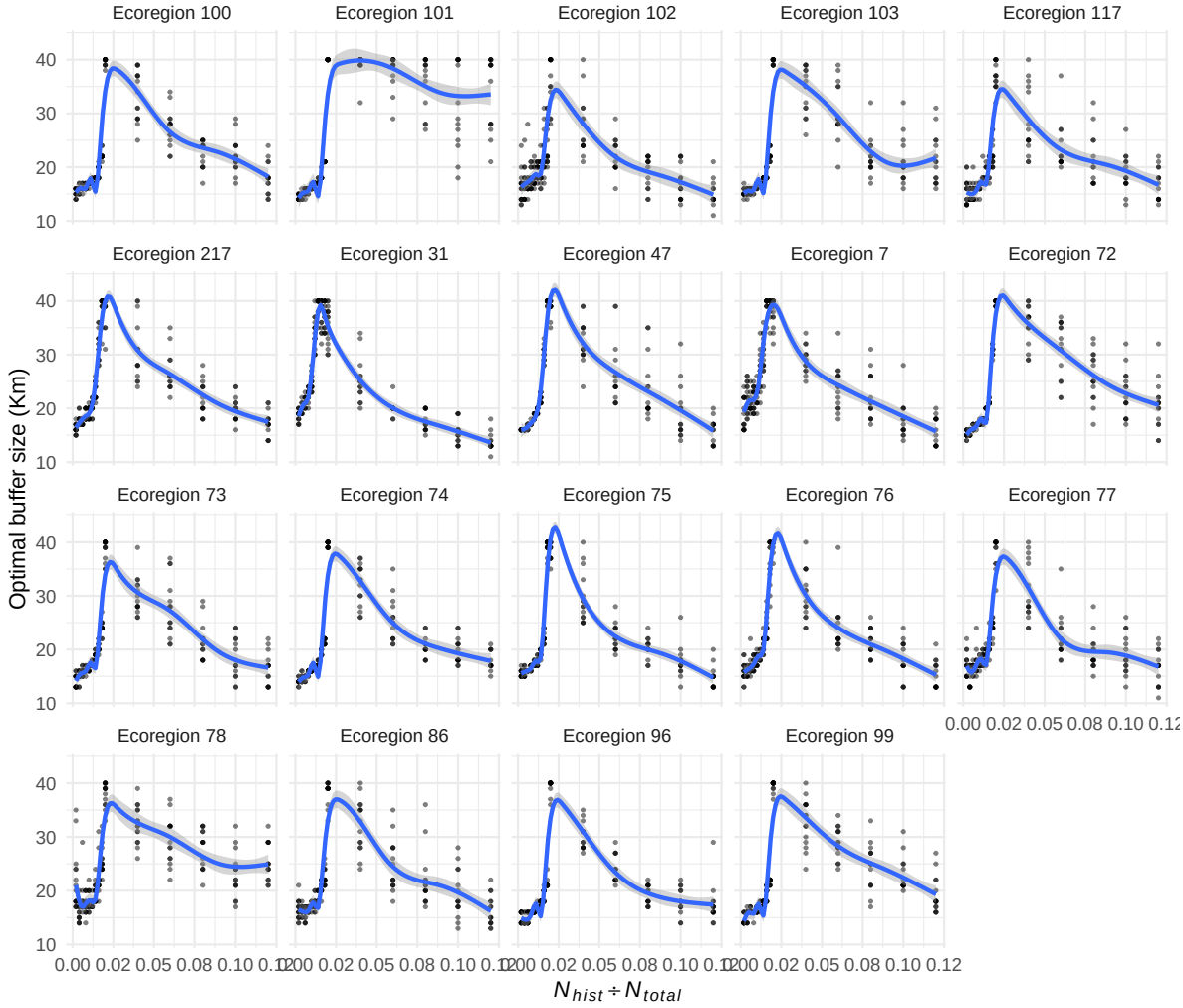


Figure 5.13: Optimal buffer size in function of the proportion of historical sites to the initial sample size. Each dot is a replication of the simulated increase of proportion of historical sites. Blue line is smooth estimation using Generalized additive model to demonstrate the exponential effect of the proportion of historical sites on the optimal buffer size.

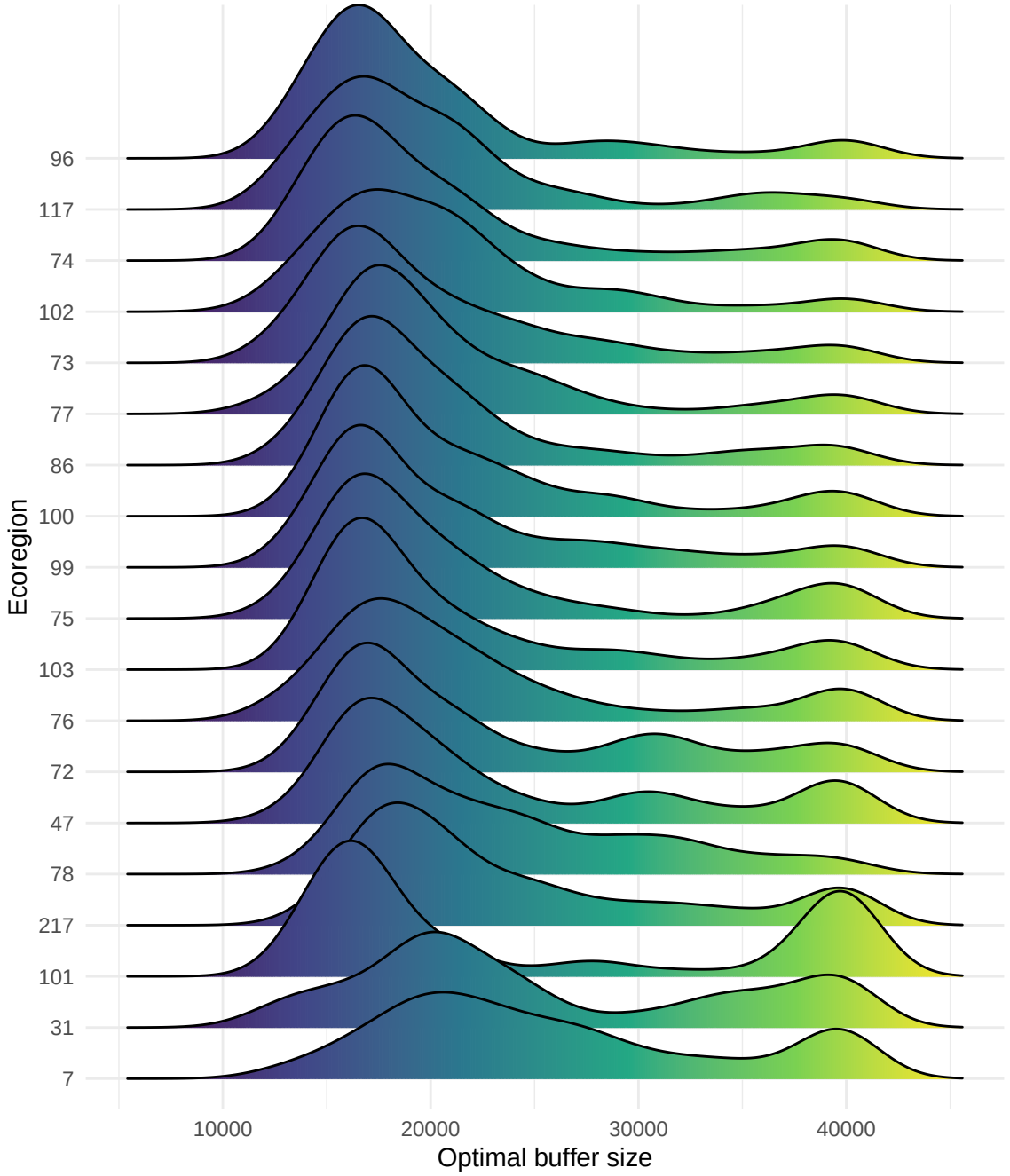


Figure 5.14: Distribution of optimal buffer size across all simulation variables and 15 replications for each ecoregion.

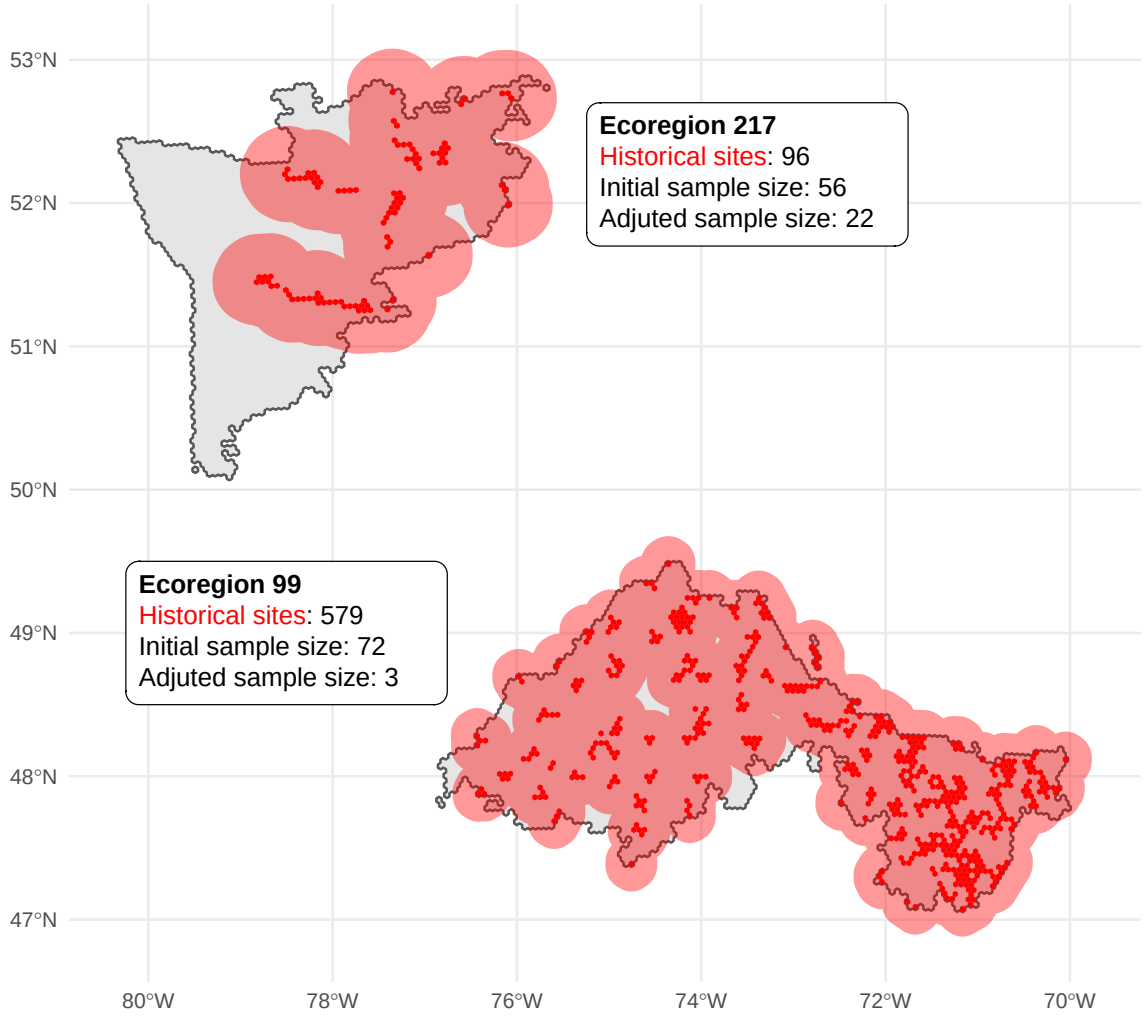


Figure 5.15: Distribution of historical sites and their coverage according to the optimal buffer size specific for the ecoregion.

Part II

Sampling design

6 Primary sampling unity

The GRTS algorithm is used for the stratified selection of hexagons (PSUs) to ensure spatially balanced samples in a region. In our case, the GRTS sampling is stratified at the ecoregion level, meaning that the random tessellation for spatially balanced hexagons is carried out separately for each ecoregion. Thus, the sample size and inclusion probabilities are determined at the ecoregion stratum level. In the previous section, we discussed in detail the process to define the inclusion probability of each PSU hexagon within an ecoregion using three layers of information: habitat, cost, and historical sites. In this section, we explain how we combine these three layers of information into a single value of inclusion probability, which weighs the selected PSU according to the frequency of habitat, the cost of sampling, and the proximity to a historical site. We then describe the necessary parameters for the GRTS. Finally, we detail the GRTS output layers and the export process. While this section provides an overview of our approach, along with key decision, we have also developed a detailed guide that includes step-by-step instructions that accompany the R code to demonstrate how to implement the sampling design.

6.1 Inclusion probabilities

Following the BOSS design (Van Wilgenburg 2020), the first step in defining the hexagon inclusion probability is to merge the habitat and cost probabilities. Given they have similar importance weight, the inclusion probability of a hexagon h from an ecoregion e is defined by:

$$P_{h,e} = \frac{P_{habitat_{h,e}} \times P_{cost_h}}{\sum^{h_n} (P_{habitat_{h,e}} \times P_{cost_h})}$$

Where $P_{habitat}$ is the habitat inclusion probability (Chapter 3) and P_{cost} is the cost inclusion probability (Chapter 4). The product of inclusion probabilities between habitat and cost layers is normalized so that it adds up to 1 across all available hexagons within the study area.

The final stage involves modifying the inclusion probability $P_{h,e}$ for neighboring hexagons located near historical sites. To achieve this, we used the `MBHdesign` R package, which adjusts the inclusion probability according to a predefined `bufferSize_p` parameter (discuted bellow).

The method employed in the R package uses a Gaussian function to reduce the inclusion probability on neighboring hexagons, with its impact diminishing as distance from the historical site increases (more details in Chapter 5).

6.2 User paramaters

To run the GRTS, the first required parameter is the list of ecoregions to be included in the sampling design, which enables us to concentrate on particular ecoregions of interest. In order to ensure that the selected hexagons have natural habitats suitable for sampling, we apply a filter to keep only the hexagons with a specific proportion of non-NA pixels. The `prop_na` parameter is used to define this threshold, our default value was defined to 0.8 which means that a hexagon must have at least 20% of natural habitat to be available for sampling.

After filtering the ecoregions and hexagons following the above rules, the next step is to determine the sample effort. As discuted in Chapter 2, we determined the sample size solely based on the size of the ecoregion, with target of sampling 2% of the available hexagons (Table 2.1). We then adjusted the sample size for each ecoregion based on the number and distribution of historical sites present in that ecoregion (Chapter 5). To do this, we need a list of historical sites and their coordinates to adjust the sample size. We also need the `bufferSize_N` parameter to determine the range of influence of historical sites on reducing the sample size. This parameter can be either a list of buffer sizes for each ecoregion or a single distance value applied to all ecoregions. Similarly, we used the `bufferSize_p` parameter to determine the range effect of the historical sites on the inclusion probability. Finally, we set the number of replications for the GRTS algorithm to run.

6.3 Selected layers

We used the `grts()` function available in the R package `spsurvey` version v5.0 or above. To run the function, we have to provide few arguments. The first argument is the `sframe`, which consists of the point grid and included the coordinates of the hexagon centroid. The second argument is the `n_base`, which specifies the sample size for each ecoregion stratum. We used the same stratum object for the `n_over` argument, which samples a supplementary layer of hexagons (herein called the over layer) available if any of the main hexagons were unavailable for any reason. The final two arguments are the column names for the stratum name (ecoregion code) and the inclusion probability of each hexagon point sample.

After defining the arguments, we execute the GRTS function with `nb_rep` replications. For each replication, we compute the total cost of sampling all selected main hexagons in the study area and select the replication with the lowest cost. Since the over layer obtained by the grts sampling is randomly selected in the stratum space, we also create an additional layer of hexagons that are obligatory next to the selected main hexagons. For each selected main

hexagon, we extract all neighboring hexagons and choose the one with the highest inclusion probability. If a selected main hexagon has no available neighboring hexagon, we will randomly select one within the ecoregion.

After running the GRTS sampling, we obtain three layers of PSU hexagons: main, over, and extra. We export these layers of selected hexagons in two different ways. The first one is a single shapefile for each layer of selected hexagons. The second way is shared the three layers of selected hexagons into a folder specific to the ecoregion. After defining the output folder to save, and the prefix arguments, the final tree of exported files follows:

Once the GRTS sampling is completed, we obtain three layers of selected PSU hexagons called *main*, *over*, and *extra*, plus the layer with all available hexagons of the study area called *ALL*. We export these layers of selected hexagons in two different ways. The first way is to export each layer as a separate shapefile and all ecoregion together. The second way, the three layers of selected hexagons are shared into a folder dedicated to the specific ecoregion. To export these files, we need to define the output folder to save the files and the prefix arguments to be added at the end of each shapefile. For instance, let's define the parameter `outputFolder` to *selection* and the parameter `fileSuffix` to *V2023*. The exported PSU file tree for the hypothetical ecoregions 101, 102, 10‘, and 104 will have the following format:

```
/tmp/RtmpOWjmDm/selection
+-- allEcoregion
|   +-- PSU-SOQB_ALL-V2023.shp
|   +-- PSU-SOQB_extra-V2023.shp
|   +-- PSU-SOQB_main-V2023.shp
|   \-- PSU-SOQB_over-V2023.shp
\-- byEcoregion
    +-- ecoregion_101
    |   +-- PSU-SOBQ_eco101_ALL-V2023.shp
    |   +-- PSU-SOBQ_eco101_extra-V2023.shp
    |   +-- PSU-SOBQ_eco101_main-V2023.shp
    |   \-- PSU-SOBQ_eco101_over-V2023.shp
    +-- ecoregion_102
    |   +-- PSU-SOBQ_eco102_ALL-V2023.shp
    |   +-- PSU-SOBQ_eco102_extra-V2023.shp
    |   +-- PSU-SOBQ_eco102_main-V2023.shp
    |   \-- PSU-SOBQ_eco102_over-V2023.shp
    +-- ecoregion_103
    |   +-- PSU-SOBQ_eco103_ALL-V2023.shp
    |   +-- PSU-SOBQ_eco103_extra-V2023.shp
    |   +-- PSU-SOBQ_eco103_main-V2023.shp
    |   \-- PSU-SOBQ_eco103_over-V2023.shp
    \-- ecoregion_104
```

```
+-- PSU-SOBQ_eco104_ALL-V2023.shp
+-- PSU-SOBQ_eco104_extra-V2023.shp
+-- PSU-SOBQ_eco104_main-V2023.shp
\-- PSU-SOBQ_eco104_over-V2023.shp
```

7 Secondary sampling unity

In this section we provide a detail our approach for selecting the Secondary Sampling Units (SSUs). Unlike the PSU sampling, the SSU sampling approach differs significantly from the BOSS design. For each selected hexagon, we create a grid of SSU points spaced 294 meters apart to perfrom the sampling (Figure 7.1). The chosen spacing value was optimized using the Ontario regionalization method to ensure that each hexagon contains an equal number of SSUs.

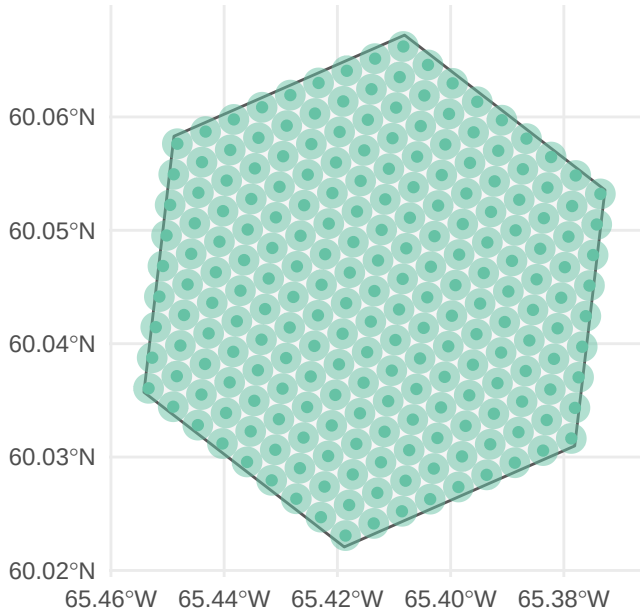


Figure 7.1: The green dots represent the Secondary Sampling Units (SSUs), which are spatially separated from each other by a distance of 294 meters. A buffer with a diameter of 147 meters was created around each SSU to extract the habitat pixels.

For each SSU point within the PSU, we generate a buffer with a distance of 147 meters. We then extract the habitat pixels to compute the inclusion probability of each SSU point following the same approach used for the PSU (Chapter 3). This process allows us to calculate the habitat inclusion probability and the proportion of non-empty habitats pixels for each SSU hexagon.

The final Step in the process of prepareing the SSU for sampling is to assign a matrix of neighouring SSU points for each focal SSU. This is important because each SSU is classified as available to be sampled in function of its proportion of non-empty habitats pixels, the number of neighbours SSU, and the proportion of non-empty habitats pixels of the neighbours SSUs. Specifically, for a SSU to be available for sampling it must have:

The next step is to assign a matrix of neighboring SSU points for each focal SSU. This is crucial because the availability of each SSU for sampling is determined based on its proportion of non-empty habitat pixels, the number of neighboring SSUs, and the proportion of non-empty habitat pixels of those neighbors. Specifically, to be considered available for sampling, a SSU must meet the following criteria:

- Have exactly 6 neighbors (fewer indicates being on the border of the PSU hexagon)
- Possess at least 80% non-empty pixels (prop_na = 0.8), similar to the PSU sampling
- Have at least 4 out of the 6 neighboring SSUs with at least 80% non-empty pixels

After defining the availability of each SSU, we proceed with a random sampling of SSU points, weighted by their habitat inclusion probability. The SSU sampling is performed sequentially, meaning that for each SSU selected within a PSU hexagon, its six neighbors are automatically marked as unavailable for the next sampling round. This procedure prevents the clustering of SSU samples in a single region. The SSU sampling continues until it reaches the desired SSU sample size (defined by `ssu_N`), or until there are no more available SSUs remaining. The selected SSU points, along with their respective neighbors, are classified using the AB code pattern (Figure 7.2). Here, A represents the SSU sample ID, ranging from 1 to `ssu_N`, while B represents the neighbor ID, ranging from 0 to 6. In this pattern, 0 denotes the focal point, while 1 to 6 represent the six neighbors, starting from the North point and moving clockwise.

Once the sampling process is completed, we obtain multiple SSU points for each hexagon from one of the three layers of selected PSU hexagons: *main*, *over*, or *extra*. We export both all SSU points and the selected SSU points for each selected hexagon from the three PSU layers. The export of SSU points follows a similar approach to the export of PSU hexagons. We first export each layer as a separate shapefile, including all ecoregions together, and then create separate shapefiles for each ecoregion. Using the same export parameters utilized in the PSU sampling, the SSU shapefiles will be exported as follows:

```
/tmp/RtmprDkHJk/selection
+-- allEcoregion
|   +-- SSU-SOQB_ALL_extra-V2023.shp
|   +-- SSU-SOQB_ALL_main-V2023.shp
|   +-- SSU-SOQB_ALL_over-V2023.shp
|   +-- SSU-SOQB_selected_extra-V2023.shp
|   +-- SSU-SOQB_selected_main-V2023.shp
|   \-- SSU-SOQB_selected_over-V2023.shp
```

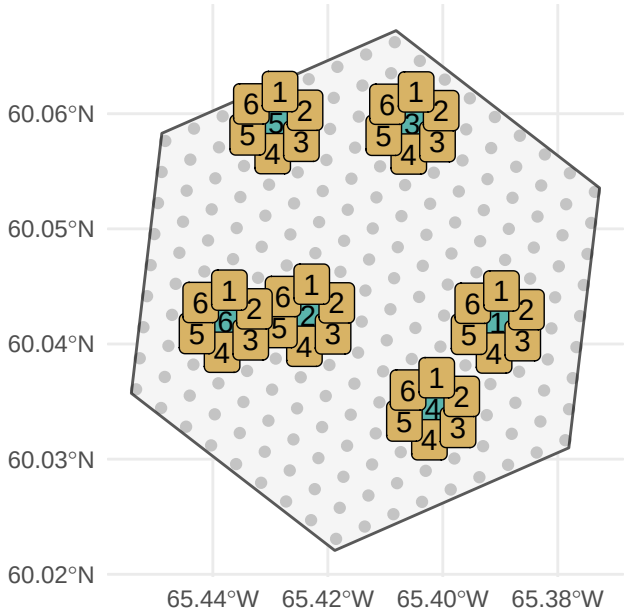


Figure 7.2: Selected SSU points (green) and their respectice neighboring (yellow).

```
\-- byEcoregion
    +-- ecoregion_101
    |   +-- SSU-SOBQ_eco101_ALL_extra-V2023.shp
    |   +-- SSU-SOBQ_eco101_ALL_main-V2023.shp
    |   +-- SSU-SOBQ_eco101_ALL_over-V2023.shp
    |   +-- SSU-SOBQ_eco101_selected_extra-V2023.shp
    |   +-- SSU-SOBQ_eco101_selected_main-V2023.shp
    |   \-- SSU-SOBQ_eco101_selected_over-V2023.shp
    +-- ecoregion_102
    |   +-- SSU-SOBQ_eco102_ALL_extra-V2023.shp
    |   +-- SSU-SOBQ_eco102_ALL_main-V2023.shp
    |   +-- SSU-SOBQ_eco102_ALL_over-V2023.shp
    |   +-- SSU-SOBQ_eco102_selected_extra-V2023.shp
    |   +-- SSU-SOBQ_eco102_selected_main-V2023.shp
    |   \-- SSU-SOBQ_eco102_selected_over-V2023.shp
    +-- ecoregion_103
    |   +-- SSU-SOBQ_eco103_ALL_extra-V2023.shp
    |   +-- SSU-SOBQ_eco103_ALL_main-V2023.shp
    |   +-- SSU-SOBQ_eco103_ALL_over-V2023.shp
    |   +-- SSU-SOBQ_eco103_selected_extra-V2023.shp
```

```
|   +-- SSU-SOBQ_eco103_selected_main-V2023.shp
|   \-- SSU-SOBQ_eco103_selected_over-V2023.shp
\-- ecoregion_104
    +-- SSU-SOBQ_eco104_ALL_extra-V2023.shp
    +-- SSU-SOBQ_eco104_ALL_main-V2023.shp
    +-- SSU-SOBQ_eco104_ALL_over-V2023.shp
    +-- SSU-SOBQ_eco104_selected_extra-V2023.shp
    +-- SSU-SOBQ_eco104_selected_main-V2023.shp
    \-- SSU-SOBQ_eco104_selected_over-V2023.shp
```

Part III

Appendices

8 A guide for BMS sampling design with R

In this section, we will provide an overview and explanation of the code used for the final step of the BMS sampling design. This report is designed to be self-contained and provides all the necessary steps for processing the data layers in conjunction with the GRTS algorithm to perform the sampling. The complete procedure, including code for data preparation and analysis, is organized in sequential R scripts and can be found in the GitHub repository. This specific report refers to the R script `08_runGRTS.R`.

8.1 Setup

Bellow we list the packages used in the sampling process. Note that the code described in the report is only compatible with spsurvey R package versions v5.0 and above. For consistent results, we recommend loading the same environment that was used in this project, which is saved in the `renv.lock` file. The following steps can be used to load the same environment:

```
# install `renv` package if necessary
if (!require(renv)) install.packages('renv')
# Restore the project environment
renv::activate(project = "/path/to/project")
```

The first line of code will check load (and install if necessary) the R pacakge `renv` to manage the version of the dependences. The `activate()` function will install all necessary packages with their respective version into a local container accessible only by this project. If you're not already in the path of the project, make sure to replace `/path/to/project` with the path to the directory where the project is stored.

```
library(raster)
library(exactextractr)
library(tidyverse)
library(spatstat)
library(sf)
library(spsurvey)
library(MBHdesign)
```

```
# to reproduce the same results
set.seed(0.0)
```

8.2 Custom parameters

Here is the main section the user will need to interact. Below is a list of all the variables that can be used to customize the sampling process, each accompanied by detailed comments.

```
# Threshold to determine whether a hexagon is suitable for sampling is based on
# the proportion of NA pixels (non-natural habitat).
# `0.8` means a hexagon has to have at least 20% of habitat pixels to be
# suitable for sampling
prop_na = 0.8

# Sample effort in percentage
sample_effort = 0.02

# Total sample size for SSU within a hexagon (Main + Over)
# It must be an even numbers
ssu_N = 6

# Number of replications when selecting the PSU with the GRTS algorithm
nb_rep = 15

# Code of ecoregions to sample
eco_sim = c(
    '7', '28', '30', '31',
    '46', '47', '48', '49',
    '73', '77', '78', '86',
    '72', '74', '75', '76',
    '96', '99', '100', '101',
    '102', '103', '117',
    '216', '217'
)

# File path of the csv file used to store the legacy sites
# E.g. https://github.com/willvieira/sampling_BMS/blob/main/data/legacySites.csv
# `lat` and `lon` arguments are used to define the character name of the
# columns in the csv
and columns to extract legacy sites
```

```
legacyFile = file.path('..', 'data', 'legacySites.csv')
lat = 'latitude'
lon = 'longitude'

# Buffer size (in Km) to adjust inclusion probability of hexagons around the
# legacy sites using the MBHdesign R package
bufferSize_p = 10

# Buffer size (in Km) to adjust the sample size of an ecoregion as a function of
# the number and distribution of its legacy sites
# The value provided can either be a single number applied to all ecoregions or
# a file path containing the buffer size information for each ecoregion.
# E.g.https://github.com/willvieira/sampling_BMS/blob/main/data/bufferSize_N.csv
bufferSize_N = file.path('..', 'data', 'bufferSize_N.csv')
# If you want to assign the same buffer size (in Km) for all ecoregion:
# bufferSize_N = 15

# Distance between SSU centroid (in meters)
ssu_dist = 294

# Path to save the shapefiles with the selected PSU and SSU
outputFolder = file.path('output', 'selection2023')

# suffix to add for each output layer
# e.g.: PSU-SOQB_ALL-SUFFIX.shp
fileSuffix = 'V2023'
```

8.3 Prepare layers for sampling

The first step is loading the complete hexagons list within the study area. While loading, we keep only the hexagons for the ecoregions of interest (`eco_sim`) and remove the hexagons with too many NA habitats. We then compute the hexagon inclusion probability from the habitat and cost layers normalized for the study area.

```
hexas <- readRDS(file.path('..', 'data', 'hexa_complete.RDS')) |>
    filter(propNA <= prop_na) |>
    filter(ecoregion %in% eco_sim) |>
    mutate(
        p = (hab_prob * cost_prob) / sum(hab_prob * cost_prob)
    ) |>
```

```
    filter(p != 0)
```

The legacy sites are the second layer of information needed to run the GRTS. Bellow the code is written to load the list of coordinate points from the legacy sites and create a data frame describing the count number of legacy sites per hexagon. This procedure scales the point data to the hexagon level, allowing us to modify the inclusion probability of the neighboring hexagons and adjust the ecoregion sample size.

```
# function to transform Latitude & longitude legacy site points in a table
# with the number of points per hexagon ID (ET_Index)
import_legacySites <- function(File, lat_name, lon_name)
{
    # transform hexagons projection to the same of the legacy points
    hx <- hexas |>
        st_transform(4326)

    # read legacy csv file
    lg <- read_csv(File, show_col_types = FALSE) |>
        rename(
            lat = all_of(lat_name),
            lon = all_of(lon_name)
        ) |>
        st_as_sf(
            coords = c('lon', 'lat'),
            crs = st_crs(hx)
        )

    # intersect legacy points with hexagon polygons
    nbLegacy <- hx |>
        st_contains(lg, sparse = FALSE) |>
        apply(1, sum)

    # Return a transformed data to data.frame
    # and keep only the hexagons that contains legacy sites
    tibble(
        ET_Index = hx$ET_Index,
        nbLegacySites = nbLegacy
    ) |>
    filter(nbLegacySites > 0)
}
```

```r
# load and transform legacy sites (slow function)
legacySites <- import_legacySites(
    File = legacyFile,
    lat_name = lat,
    lon_name = lon
)

# merge legacy sites data.frame into hexagon object
hexas <- hexas |>
    left_join(legacySites) |>
    mutate(nbLegacySites = replace_na(nbLegacySites, 0))
```

8.4 Adjust sample size and inclusion probability as a function of legacy sites

After merging the hexagons and legacy sites, the next step is to adjust the sample size and modify the inclusion probabilities. The sample size, defined in the number of hexagons, is defined in function of the size of the ecoregion, the sampling effort (`sample_effort`), and the number of legacy sites. The input `bufferSize_N` quantify the effect of each legacy site in reducing the final sample size. The value provided can either be a single number that will be applied to all ecoregions or a file path that contains the buffer size information for each individual ecoregion. For more details on defining the buffer size to adjust the sample size, please refer to Chapter 5.

```r
# function to get sample size for a specific ecoregion given:
# number of hexagons, number of legacy sites, bufferSize, and sample effort
get_sampleSize <- function(eco, hx, bf_N, sample_e)
{
    # get the hexagons centroid for a specific ecoregion
    hexa_eco <- hx |>
        filter(ecoregion == eco) |>
        st_centroid()

    if(nrow(subset(bf_N, ecoregion == eco)) > 0) {
        # create a buffer of size `BufferSize_N` around the legacy site centroid
        hexa_legacy_bf <- hexa_eco |>
            filter(nbLegacySites > 0) |>
            st_buffer(subset(bf_N, ecoregion == eco)$bufferSize * 1000) |>
            st_union()
```

```
        # Compute the number of hexagons from the ecoregion in which their
        # centroid falls within the buffer around the legacy sites
        nbHexas_legacy <- hexa_eco |>
            st_intersects(hexa_legacy_bf) |>
            unlist() |>
            sum()
    }else{
        nbHexas_legacy = 0
    }

    # Compute the adjusted sample size using only the total amount of hexagons
    # that do not fall within the legacy buffer
    adj_sampleSize <- round((nrow(hexa_eco) - nbHexas_legacy) * sample_e, 0)

    # If the ecoregion is too small to accommodate 2 hexagons with the defined
    # sampling effort, ensure to sample at least two hexagons.
    if((nrow(hexa_eco) * sample_e) < 2 & nbHexas_legacy < 1)
        adj_sampleSize = 2

    return(adj_sampleSize)
}
```

```
# Load buffer size information regardless of the `byfferSize_N` class
if(is.character(bufferSize_N)) {
    if(file.exists(bufferSize_N)) {
        buffSizeN <- read_csv(bufferSize_N, show_col_types = FALSE) |>
            mutate(ecoregion = as.character(ecoregion))
    }else{
        stop(
            paste0('File "', bufferSize_N,
            '" does not exist. Please check if the name is correct.')
        )
    }
}else if(is.numeric(bufferSize_N)){
    buffSizeN <- tibble(
        ecoregion = eco_sim,
        bufferSize = bufferSize_N
    )
}else{
    stop('The type of `bufferSize_N` must be either numeric or a character')
}
```

```r
# Run the function to define sample size for all selected ecoregions
sampleSize <- map_dbl(
    setNames(eco_sim, paste0('eco_', eco_sim)),
    get_sampleSize,
    hx = hexas,
    bf_N = buffSizeN,
    sample_e = sample_effort
)

# Define the stratum object with ecoregion that have
# at least a sample size of 1
Stratdsgn  <- sampleSize[sampleSize > 0]

# Because we are sampling an extra hexagon for each selected one,
# make sure that each ecoregion has at least 2 times more available
# hexagons than sample N
eco_to_remove <- hexas |>
    st_drop_geometry() |>
    group_by(ecoregion) |>
    summarise(nbHexas = n()) |>
    left_join(
        sampleSize |>
            enframe() |>
            mutate(
                ecoregion = as.character(parse_number(name)),
                sampleSize_extra = value * 2
            ) |>
            select(ecoregion, sampleSize_extra)
    ) |>
    mutate(
        diff = nbHexas - sampleSize_extra
    ) |>
    filter(diff < 0) |>
    pull(ecoregion)

Stratdsgn <- Stratdsgn[!names(Stratdsgn) %in% paste0('eco_', eco_to_remove)]

# Prepare sample frame to be used in GRTS
# scale inclusion probability to the total sample size
# Use only the centroid of the hexagon as a sample point
sampleFrame <- hexas |>
```

```
    filter(ecoregion %in% parse_number(names(Stratdsgn))) |>
    mutate(
        eco_name = paste0('eco_', ecoregion), # to match design name
        mdcaty  = sum(Stratdsgn) * p/sum(p),
        geometry = sf::st_geometry(sf::st_centroid(geometry))
)
```

The last step involves adjusting inclusion probabilities for neighboring hexagons around legacy sites. To do this, we used the `MBHdesign` R package and adjusted the inclusion probability based on the `bufferSize_p` parameter.

```
# Coordinates of all hexagons in matrix format for MBHdesign
coord_mt <- sampleFrame |>
    st_coordinates()

# Coordinates of legacy hexagons
legacySites <- sampleFrame |>
    filter(nbLegacySites > 0) |>
    st_coordinates()

# Adjust inclusion probability of neighbour hexagons in function of legacy sites
sampleFrame$adj_p <- MBHdesign::alterInclProbs(
    legacy.sites = legacySites,
    potential.sites = coord_mt,
    inclusion.probs = sampleFrame$mdcaty,
    sigma = bufferSize_p * 1000
)
```

8.5 Primary Sampling Unit (PSU)

The `spsurvey` R package was used to implement the GRTS algorithm for sampling the PSUs of each ecoregion stratum. The sample size for each stratum, as defined in `Stratdsn`, is used to sample the hexagon points from the `sampleFrame` object. Note that the same sample size stratum is used for sampling the replacement (`n_over`) sites for each ecoregion stratum. Within each stratum, the sample selection is weighted based on the inclusion probability `adj_p`, which is derived from the habitat, cost, and legacy site layers. We replicate this sampling process `nb_rep` times, sum the total cost of sampling for the selected hexagons, and choose the replication with the lowest cost.

```r
# list to store the hexagons ID (ET_Index) for the main and the replacement
grts_main <- grts_over <- list()
# Run GRTS selection over nb_rep times
for(Rep in 1:nb_rep)
{
    out_sample <- spsurvey::grts(
        sframe = sampleFrame,
        n_base = Stratdsgn,
        n_over = Stratdsgn,
        stratum_var = 'eco_name',
        aux_var = 'adj_p'
    )

    grts_main[[paste0('Rep_', Rep)]] <- out_sample$sites_base$ET_Index
    grts_over[[paste0('Rep_', Rep)]] <- out_sample$sites_over$ET_Index
}

# Select cheapest replication (based on 'main' selection)
cheapest_rep <- map_df(
    grts_main,
    ~ hexas |>
        st_drop_geometry() |>
        filter(ET_Index %in% .x) |>
        summarise(totalCost = sum(costSum))
) |>
pull(totalCost) |>
which.min()

# Save selected main and replacement hexagons
selected_main <- hexas |>
    filter(ET_Index %in% grts_main[[cheapest_rep]])
selected_over <- hexas |>
    filter(ET_Index %in% grts_over[[cheapest_rep]])
```

Since the replacement hexagons are randomly selected in the stratum space, we also selected an extra layer of replacement hexagons that are obligatory neighbors of each main hexagon. For each selected main hexagon, we extract all neighboring hexagons and select the one with the highest inclusion probability. If a selected main hexagon has no available neighbor hexagon, we will select a random one over the ecoregion.

```r
# object to store extra layer of selected hexagons
selected_extra <- hexas[0, ]

# loop over ecoregions to make sure neighbours are from the same ecoregion
for(eco in parse_number(names(Stratdsgn)))
{
    hexas_eco <- subset(hexas, ecoregion == eco)
    hexas_sel_eco <- subset(selected_main, ecoregion == eco)

    # Extract neighbours hexagons
    neigh_mt <- hexas_eco |>
        st_centroid() |>
        st_intersects(
            y = st_buffer(hexas_sel_eco, dist = 3500),
            sparse = FALSE
        )

    # Remove the focus hexagon (the selected one)
    rr = match(hexas_sel_eco$ET_Index, hexas_eco$ET_Index)
    cc = seq_along(rr)
    neigh_mt[rr + nrow(neigh_mt) * (cc - 1)] <- FALSE

    # Select the extra hexagon based on the highest p
    best_p <- apply(
        neigh_mt,
        2,
        function(x)
            hexas_eco$ET_Index[x][which.max(hexas_eco$p[x])]
    )

    # if a selected hexagon has no neighbour, select a random from the ecoregion
    toCheck <- unlist(lapply(best_p, length))
    if(any(toCheck == 0)) {
        # which hexagons were not selected?
        nonSelected_hexas <- setdiff(hexas_eco$ET_Index, hexas_sel_eco$ET_Index)

        # sample from non selected hexas
        best_p[which(toCheck == 0)] <- sample(
            nonSelected_hexas, sum(toCheck == 0)
        )
    }
```

```
    selected_extra <- rbind(
        selected_extra,
        hexas_eco[match(best_p, hexas_eco$ET_Index), ]
    )
}
```

8.6 Secondary Sampling Unit (SSU)

For each of the selected PSU hexagons (`main`, `over`, and `extra`), we generate a grid of SSU points equally spaced with a distance defined by `ssu_dist`. The following function to generate SSUs is from the Ontario's sampling approach (code source).

```
genSSU <- function(h, spacing)
{
    ch <- as_tibble(st_coordinates(h))
    top_point <- ch[which.max(ch$Y),]
    bottom_point <- ch[which.min(ch$Y),]
    gridsize <- 2*floor(abs(top_point$Y-bottom_point$Y)/spacing)+3
    rowAngle <- tanh((top_point$X-bottom_point$X)/(top_point$Y-bottom_point$Y))

    cent <- st_centroid(h) %>%
        bind_cols(as_tibble(st_coordinates(.))) %>%
        st_drop_geometry %>%
        dplyr::select(ET_Index, X, Y)

    genRow <- function(cX, cY, sp,...){
        tibble(rowid = seq(-gridsize,gridsize)) %>%
        mutate(X = sin(60*pi/180+rowAngle) *sp*rowid + {{cX}},
                Y = cos(60*pi/180+rowAngle) *sp*rowid  + {{cY}})
    }

    centroids <- tibble(crowid=seq(-gridsize,gridsize)) %>%
        mutate(cY = cos(rowAngle) *spacing*crowid + cent$Y,
            #spacing/2*crowid + cent$Y,
            cX =  sin(rowAngle) *spacing*crowid + cent$X) %>%
        #cent$X + crowid* sqrt(spacing**2-(spacing/2)**2)) %>%
        rowwise() |>
        mutate(row = list(genRow(cX = cX,cY = cY,sp = spacing))) %>%
        unnest(row) |>
        dplyr::select(X,Y) %>%
```

```
        st_as_sf(coords = c("X", "Y"), crs = st_crs(h)) %>%
        st_filter(h) %>%
        mutate(
            ET_Index = h$ET_Index,
            ecoregion = h$ecoregion,
            ssuID = row_number()
        )
    return(centroids)
}
```

The following step involves defining a function to sample the SSUs within each selected PSU hexagon. SSU sampling is done sequentially, so for each selected SSU, the first and second layers of neighbour points (6 and 12, respectively) are made unavailable for sampling the next SSU. We repeat this process until we reach the total sampling size (`ssu_N`) or when there are no more available SSUs left to sample.

```
sample_SSU <- function(ssuid, prob, geom, filtered, ssuDist, N)
{
    # check if N is even
    if(N %% 2 != 0)
        stop('`ssu_N` must be a even number.')

    filtered_1 <- filtered
    out <- rep(0, length(filtered))

    # loop to sample N SSUs
    count = 1
    while(
        count <= N &
        sum(get(paste0('filtered_', count))) > 1
    ){
        # sample point
        assign(
            paste0('sample_', count),
            sample(
                ssuid[get(paste0('filtered_', count))],
                size = 1,
                prob = prob[get(paste0('filtered_', count))]
            )
        )
```

```
        # remove points around the first sample for second point
        toKeep <- !st_intersects(
            geom,
            st_buffer(
                geom[which(get(paste0('sample_', count)) == ssuid)],
                dist = ssuDist * 2 + ssuDist * 0.1),
                sparse = FALSE
        )[, 1]

        # update available points
        assign(
            paste0('filtered_', count + 1),
            get(paste0('filtered_', count)) & toKeep
        )

        # assign point to output vector
        out[get(paste0('sample_', count))] = count

        count = count + 1
    }
    return( out )
}
```

With these two functions, we can loop over the `main`, `over`, and `extra` selected hexagons to create and sample SSU points. First, we create the SSU points for all selected hexagons. Then, we generate a buffer around each SSU point with a distance of `ssu_dist/2`. We use the habitat pixels within each buffer to calculate the inclusion probability and the proportion of NA pixels. To save computation time, we compute and store the six neighbour points around each SSU point. Before sampling the SSU points, we classify each point as available or not following these rules:

- it must have 6 neighbours (less than that means it's a border SSU)
- it must have at least 1 - `prop_na` of non-empty pixels
- 4 out of 6 neighbours must also meet these rules

In the final step, we assign a specific code to each selected SSU point and its neighbors, following the `A_B` pattern, where `A` represents the SSU sample ID (ranging from 1 to `ssu_N`), and `B` represents the neighbor ID (ranging from 0 to 6). Here, 0 represents the focal point, and 1 to 6 represent the six neighbors, starting from the top point and moving clockwise. please refer to Chapter 7 for more details.

```r
# Habitat shapefile to calculate inclusion probability
land_ca <- raster(file.path('..', 'data', 'landcover_ca_30m.tif'))
# Distribution of habitat types per ecoregion
prev_all <- readRDS(file.path('..', 'data', 'prev_all.RDS'))

for(lyr in c('main', 'over', 'extra'))
{
    sel_lyr <- get(paste0('selected_', lyr))

    # Generate SSU points
    SSUs <- map_dfr(
        seq_len(nrow(sel_lyr)),
        ~ genSSU(sel_lyr[.x, ], spacing = ssu_dist)
    )

    # Buffer of half `ssu_dist` to compute habitat inclusion prob
    SSU_bf <- st_buffer(SSUs, dist = ssu_dist/2)

    # extract pixels for each SSU polygon
    hab_pixels <- exactextractr::exact_extract(
        land_ca,
        SSU_bf,
        progress = FALSE
    )
    rm(SSU_bf)

    # get frequence of each class of habitat
    count_hab <- Map(
                function(x, y) {
                    freq <- table(x$value)
                    if(length(freq) > 0) {
                        data.frame(freq, ecoregion = y)
                    }else{
                        NA
                    }
                },
                x = hab_pixels,
                y = SSUs$ecoregion
            )

    # merge with inclusion probability
```

```
# and calculate inclusion probability for each NON empty polygon
SSUs$incl_prob <- unlist(
        lapply(
            count_hab,
            function(x) {
                if(is.data.frame(x)) {
                    mg_df <- merge(
                        x,
                        subset(
                            prev_all, ID_ecoregion == x$ecoregion[1]
                        ),
                        by.x = "Var1",
                        by.y = "code",
                        all.x = TRUE
                    )
                    sum(mg_df$Freq * mg_df$incl_prob)
                }else{
                    NA
                }
            }
        )
    )
rm(count_hab)

SSUs <- SSUs |>
    group_by(ET_Index) |>
    mutate(
        incl_prob = incl_prob/sum(incl_prob, na.rm = TRUE)
    ) |>
    ungroup()

# Calculate proportion of NA
SSUs$propNA <- map_dbl(
    hab_pixels,
    ~ sum(is.na(.x$value))/nrow(.x)
)
rm(hab_pixels)

# neighbours matrix
neighbour_ls <- list()
for(hx in unique(SSUs$ET_Index))
```

```
{
    ssuhx <- subset(SSUs, ET_Index == hx)

    neighbour_ls[[hx]] <- st_intersects(
        ssuhx,
        st_buffer(ssuhx, dist = ssu_dist + ssu_dist * 0.1),
        sparse = FALSE
    )
}

# These are the following rules to a SSU be suitable for sampling:
# - Must have 6 neighbours (less than that means it's a border SSU)
# - Must have at least 1 - `prop_na` of non empty pixels
# - 4 out 6 neighbours must also respect the above rule
SSU_selected = SSUs |>
    group_by(ET_Index) |>
    mutate(
        nbNeighb = map_dbl(
            ssuID,
            ~ sum(neighbour_ls[[unique(ET_Index)]][, .x]) - 1
        ),
        nbPropNA = map_int(
            ssuID,
            .f = function(x, pNA, ETI)
                sum(
                    pNA[
                        setdiff(which(neighbour_ls[[ETI]][, x]), x)
                    ] <= prop_na
                ),
            pNA = propNA,
            ETI = unique(ET_Index)
        ),
        sampled = sample_SSU(
            ssuid = ssuID,
            prob = incl_prob,
            geom = geometry,
            filtered = propNA <= prop_na & nbNeighb == 6  & nbPropNA >= 4,
            ssuDist = ssu_dist,
            N = ssu_N
        )
    ) |>
```

```
        ungroup()

    # Prepare selected SSU and their specific neighbours with code like `A_B`
    # `A` is for the SSU sample ID (1:`ssu_N`)
    # `B` is for the neighbour ID (0:6 where zero is the focal point, and
    # 1:6 are the 6 neighbours starting from the top point moving clockwise)
    SSU_lyr <- subset(SSU_selected, sampled > 0)
    SSU_lyr_ls <- list()

    for(i in 1:nrow(SSU_lyr))
    {
        # get neighbours for specific row
        nei_hx <- SSU_selected |>
            filter(ET_Index == SSU_lyr$ET_Index[i]) |>
            filter(
                ssuID %in% which(
                    neighbour_ls[[SSU_lyr$ET_Index[i]]][, SSU_lyr$ssuID[i]]
                )
            )

        # code A
        nei_hx$codeA <- SSU_lyr$sampled[i]

        # code B
        nei_hx$codeB <- c(4, 3, 5, 0, 2, 6, 1)

        SSU_lyr_ls[[i]] <- nei_hx
    }

    # save
    assign(
        paste0('SSU_all_', lyr),
        SSU_selected
    )
    assign(
        paste0('SSU_', lyr),
        do.call(rbind, SSU_lyr_ls)
    )
}
```

8.7 Export PSU and SSU samples in shapefiles

The purpose of this final section is to export all PSU and SSU points in a shapefile format. The format was chosen to facilitate post-processing, but it can be modified to fit different needs. The first code chunk saves all PSU and SSU points into a single file, while the second code chunk separates the same sampled points by ecoregion.

Grouped ecoregions

```
savePath = file.path(outputFolder, 'allEcoregion')
dir.create(savePath, recursive = TRUE)

varsToRm = c(
    'OBJECTID', 'Join_Count', 'TARGET_FID',
    'ET_ID', 'ET_ID_Old', 'ET_IDX_Old'
)

hexas_save <- hexas |>
    select(-all_of(varsToRm))

# rename attributes table so it has a maximum of 10 characters
names(hexas_save) <- abbreviate(names(hexas_save), minlength = 10)

# add coordinates
coords <- hexas_save |>
    sf::st_centroid() |>
    sf::st_transform(4326) |>
    sf::st_coordinates() |>
    as.data.frame()

hexas_save <- hexas_save |>
    mutate(
        latitude = coords$Y,
        longitude = coords$X
    )

# save all PSU
hexas_save |>
    write_sf(
        file.path(
```

```
            savePath,
            paste0('PSU-SOBQ_ALL-', fileSuffix, '.shp')
        )
    )

# Selected PSUs
for(lyr in c('main', 'over', 'extra'))
{
    hexas_save |>
        filter(
            ET_Index %in% get(paste0('selected_', lyr))$ET_Index
        ) |>
        write_sf(
            file.path(
                savePath,
                paste0('PSU-SOBQ_', lyr, '-', fileSuffix, '.shp')
            )
        )
}

# Save all SSU
for(lyr in c('main', 'over', 'extra'))
{
    # SSU main
    SSU_lyr <- get(paste0('SSU_all_', lyr)) |>
        select(-c('nbNeighb', 'nbPropNA', 'sampled'))

    coords <- SSU_lyr |>
        sf::st_transform(4326) |>
        sf::st_coordinates() |>
        as.data.frame()

    SSU_lyr |>
        mutate(
            latitude = coords$Y,
            longitude = coords$X
        ) |>
        write_sf(
            file.path(
                savePath,
                paste0('SSU-SOBQ_ALL_', lyr, '-', fileSuffix, '.shp')
```

```
            )
        )
}

# Save selected SSU
for(lyr in c('main', 'over', 'extra'))
{
    # SSU main
    SSU_lyr <- get(paste0('SSU_', lyr)) |>
        select(-c('nbNeighb', 'nbPropNA', 'sampled'))

    coords <- SSU_lyr |>
        sf::st_transform(4326) |>
        sf::st_coordinates() |>
        as.data.frame()

    SSU_lyr |>
        mutate(
            latitude = coords$Y,
            longitude = coords$X
        ) |>
        write_sf(
            file.path(
                savePath,
                paste0('SSU-SOBQ_selected_', lyr, '-', fileSuffix, '.shp')
            )
        )
}
```

Splited ecoregions

```
for(eco in eco_sim)
{
    # create folder
    eco_path <- file.path(
        outputFolder,
        'byEcoregion',
        paste0('ecoregion_', eco)
    )
    dir.create(eco_path, recursive = TRUE)
```

```
# PSU
##########################################
hexas_save |>
    filter(ecoregion == eco) |>
    write_sf(
        file.path(
            eco_path,
            paste0('PSU-SOBQ_eco', eco, '_ALL-', fileSuffix, '.shp')
        )
    )

for(lyr in c('main', 'over', 'extra'))
{
    hexas_save |>
        filter(
            ET_Index %in% subset(
                get(paste0('selected_', lyr)),
                ecoregion == eco
            )$ET_Index
        ) |>
        write_sf(
            file.path(
                eco_path,
                paste0(
                    'PSU-SOBQ_eco',
                    eco, '_',
                    lyr, '-',
                    fileSuffix,
                    '.shp'
                )
            )
        )
}

# SSU
##########################################

# Save all SSU
for(lyr in c('main', 'over', 'extra'))
{
    # SSU main
```

```
    SSU_lyr <- get(paste0('SSU_all_', lyr)) |>
        filter(ecoregion == eco) |>
        select(-c('nbNeighb', 'nbPropNA', 'sampled'))

    coords <- SSU_lyr |>
        sf::st_transform(4326) |>
        sf::st_coordinates() |>
        as.data.frame()

    SSU_lyr |>
        mutate(
            latitude = coords$Y,
            longitude = coords$X
        ) |>
        write_sf(
            file.path(
                eco_path,
                paste0(
                    'SSU-SOBQ_eco',
                    eco,
                    '_ALL_',
                    lyr, '-',
                    fileSuffix,
                    '.shp'
                )
            )
        )
}

# Save selected SSU
for(lyr in c('main', 'over', 'extra'))
{
    # SSU main
    SSU_lyr <- get(paste0('SSU_', lyr)) |>
        filter(ecoregion == eco) |>
        select(-c('nbNeighb', 'nbPropNA', 'sampled'))

    coords <- SSU_lyr |>
        sf::st_transform(4326) |>
        sf::st_coordinates() |>
        as.data.frame()
```

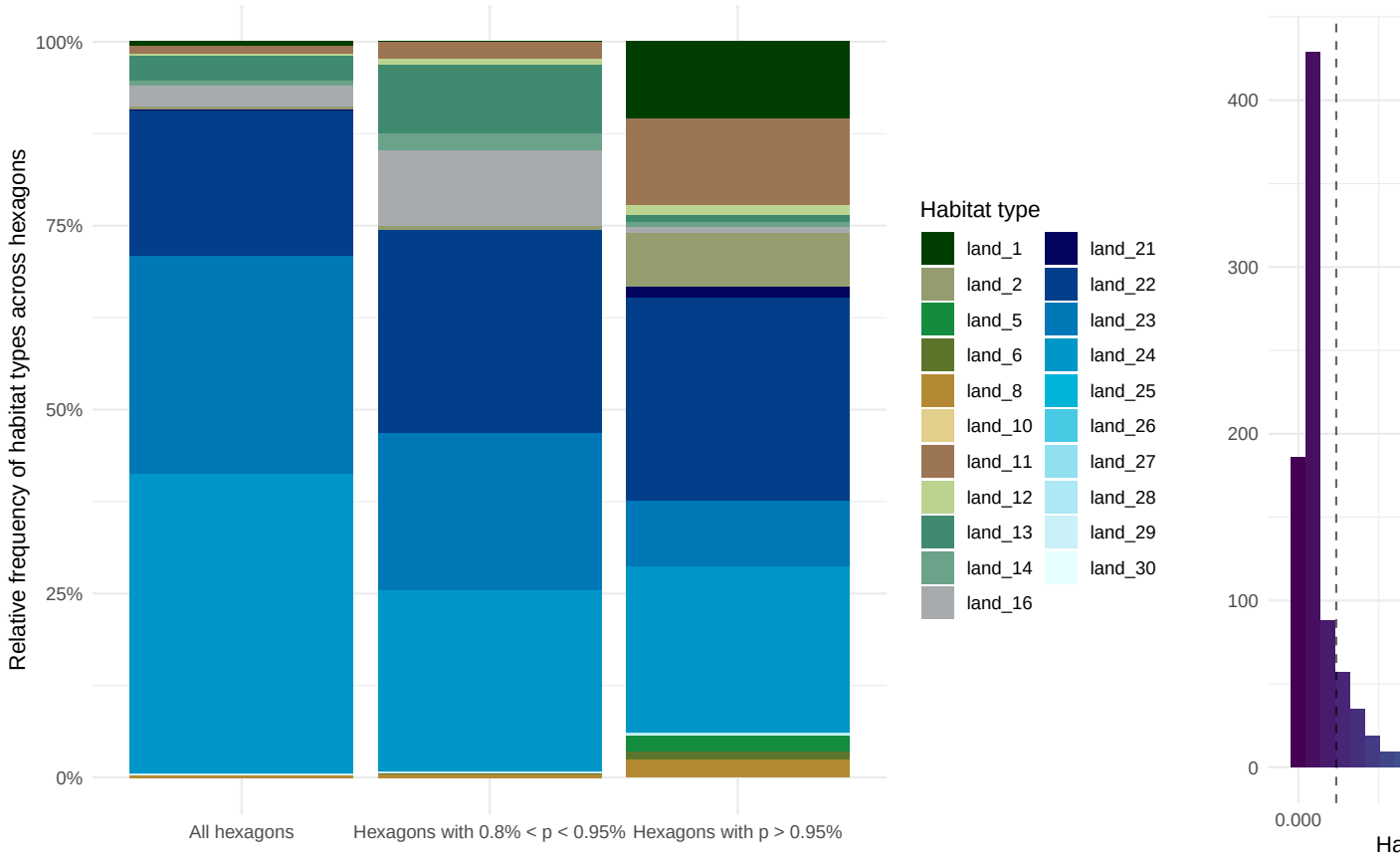
```
        SSU_lyr |>
            mutate(
                latitude = coords$Y,
                longitude = coords$X
            ) |>
            write_sf(
                file.path(
                    eco_path,
                    paste0(
                        'SSU-SOBQ_eco',
                        eco,
                        '_selected_',
                        lyr, '-',
                        fileSuffix,
                        '.shp'
                    )
                )
            )
    }
}
```

9 Ecoregion summary: habitat

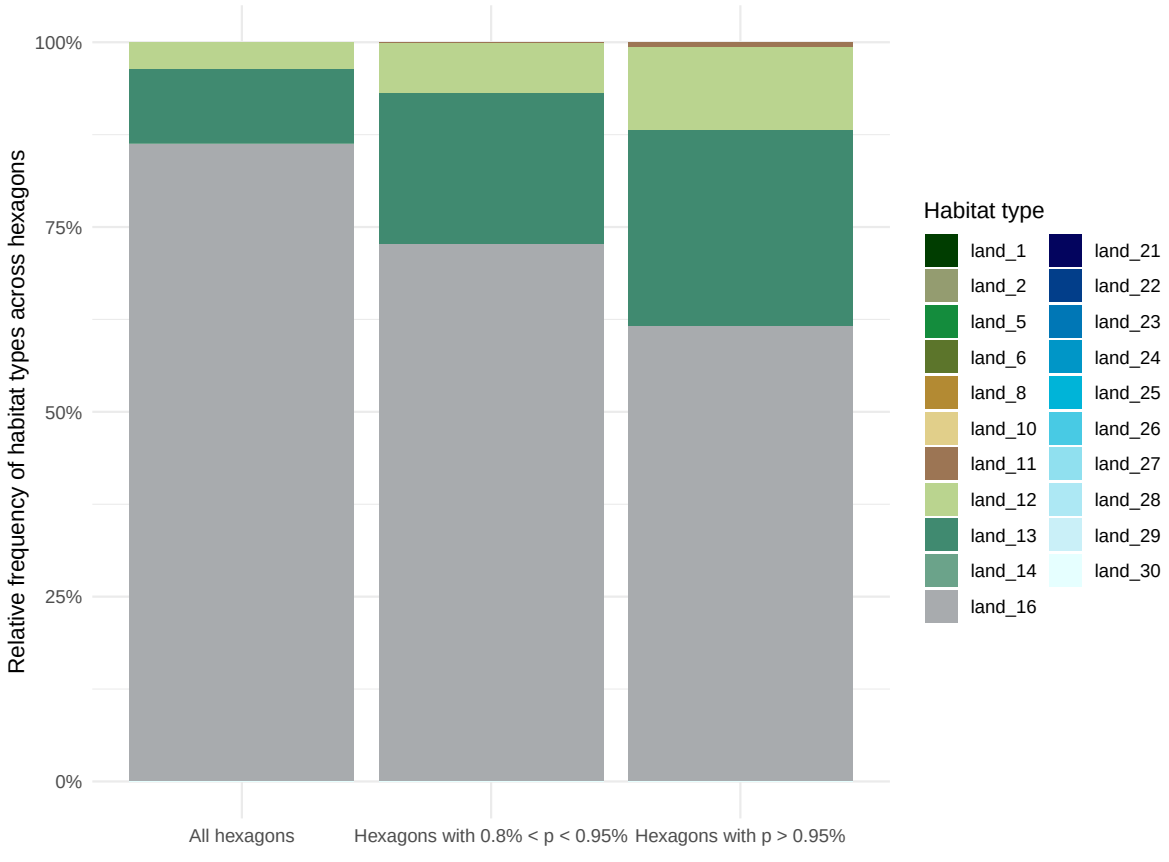
This section provides a summary of the information obtained from the habitat type layer for each ecoregion (detailed in Chapter 3). The frequency of each habitat type is extracted for every hexagon within the study area. The initial figure displays the relative frequency of habitat types for three classes of hexagons categorized based on their habitat inclusion probability. The first class consists of all hexagons in the ecoregion, while the other two classes only contain hexagons with the highest inclusion probability according to their quantile distribution.

The subsequent histograms illustrate the distribution of habitat inclusion probability and the proportion of NA pixels within each hexagon. For the habitat inclusion probability histogram, the two dashed lines are used to describe the 80% and 95% quantile distribution illustrated in the first figure. The final figure displays the spatial distribution of habitat inclusion probability across the ecoregion. It's worth noting that in most ecoregions, the distribution of habitat inclusion probability is highly skewed, with only a few hexagons having a very large inclusion probability. This skewed distribution can make it challenging to visualize the gradient of inclusion probability. To avoid this issue, we removed the hexagons with the top 0.1% inclusion probability to improve the clarity of the inclusion probability gradient.

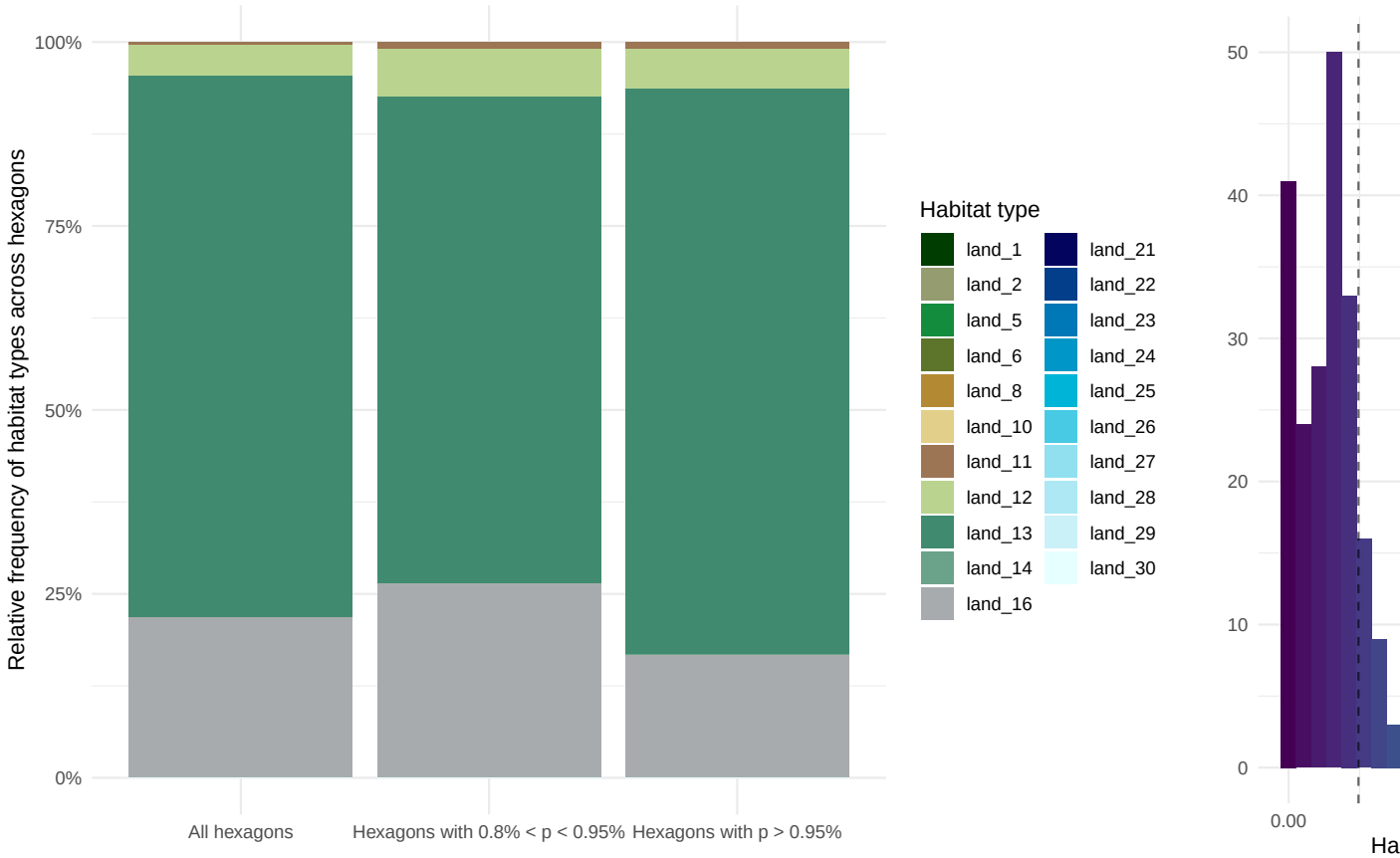
9.1 Ecoregion 7



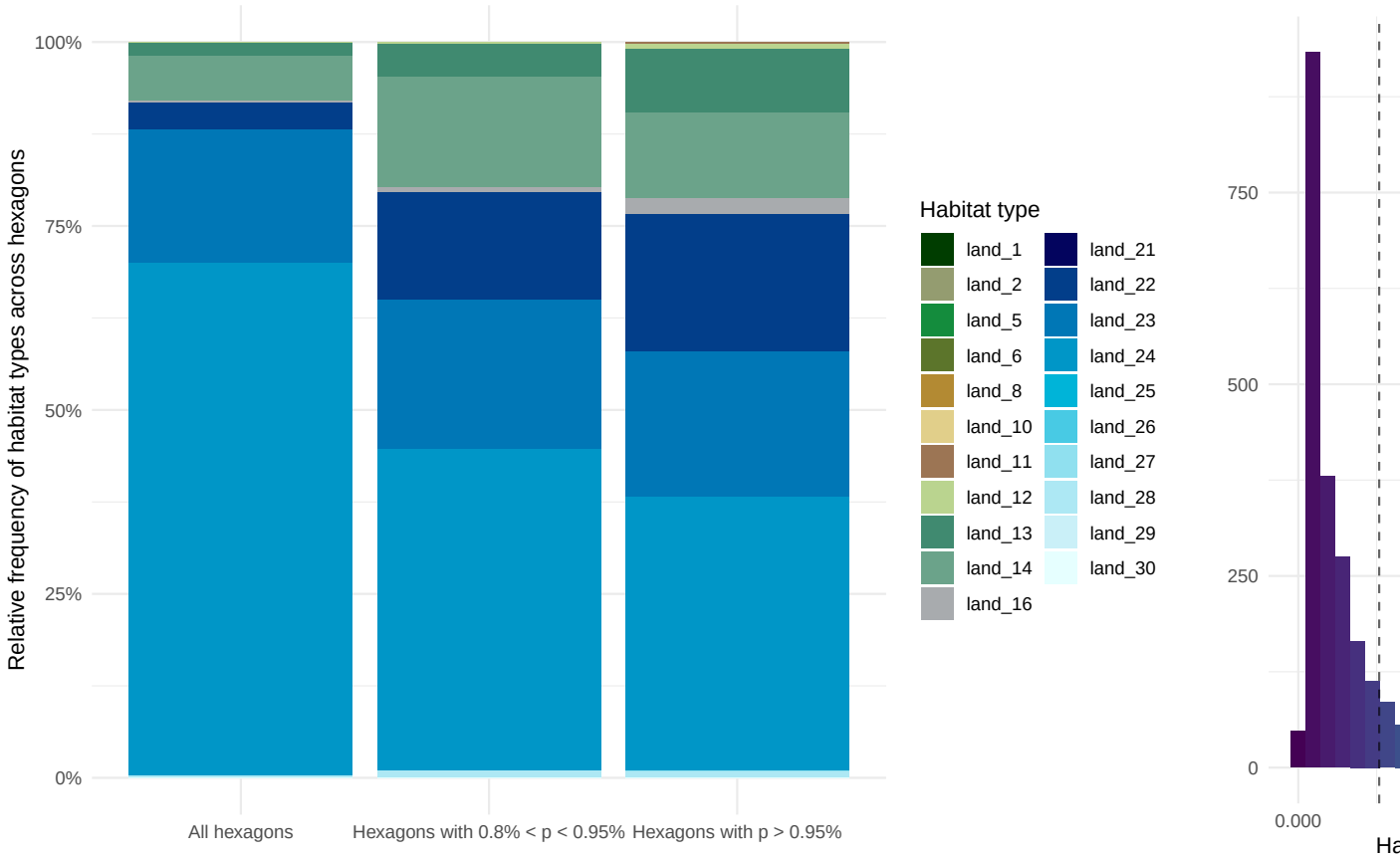
Ecoregion 28



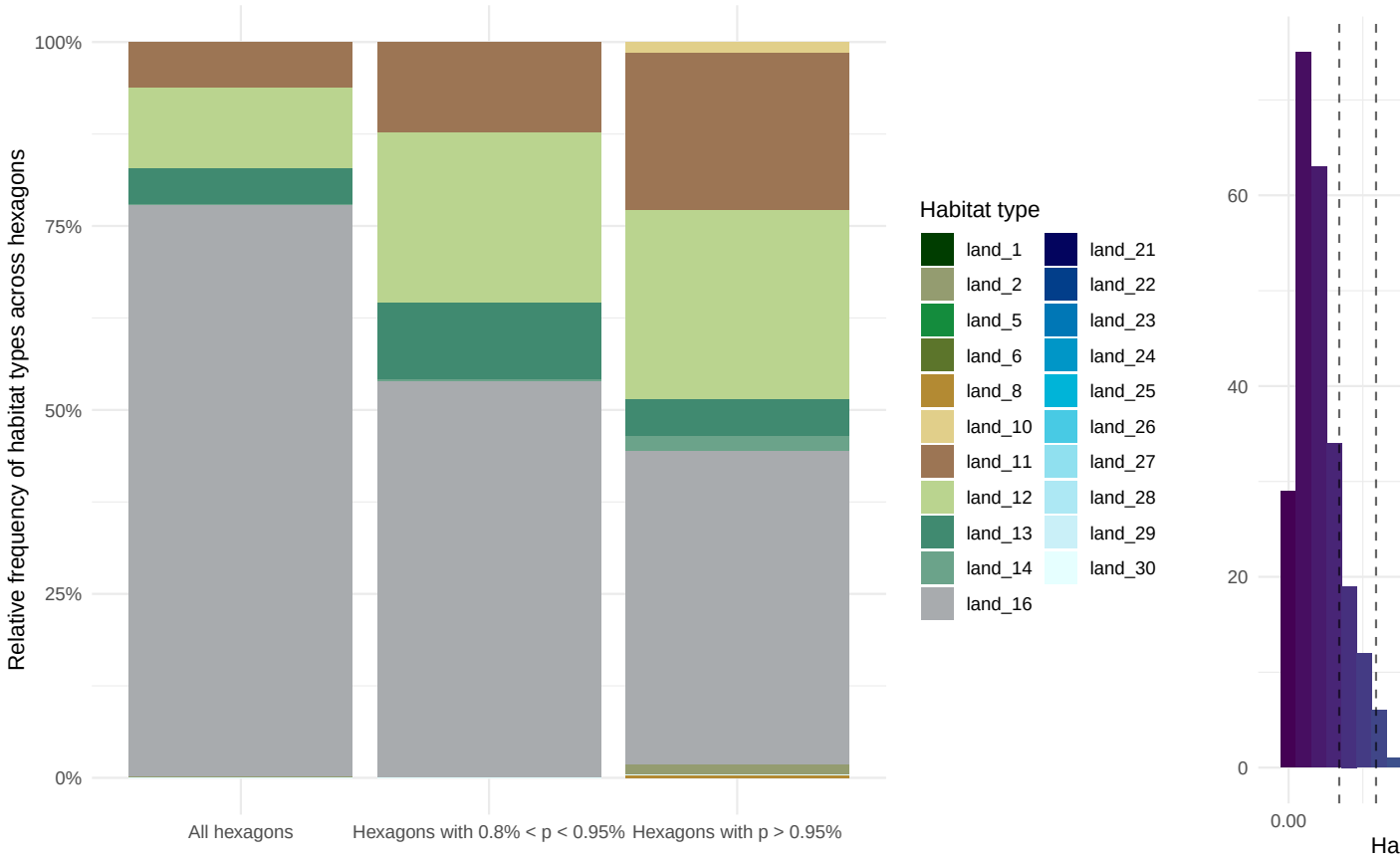
Ecoregion 30



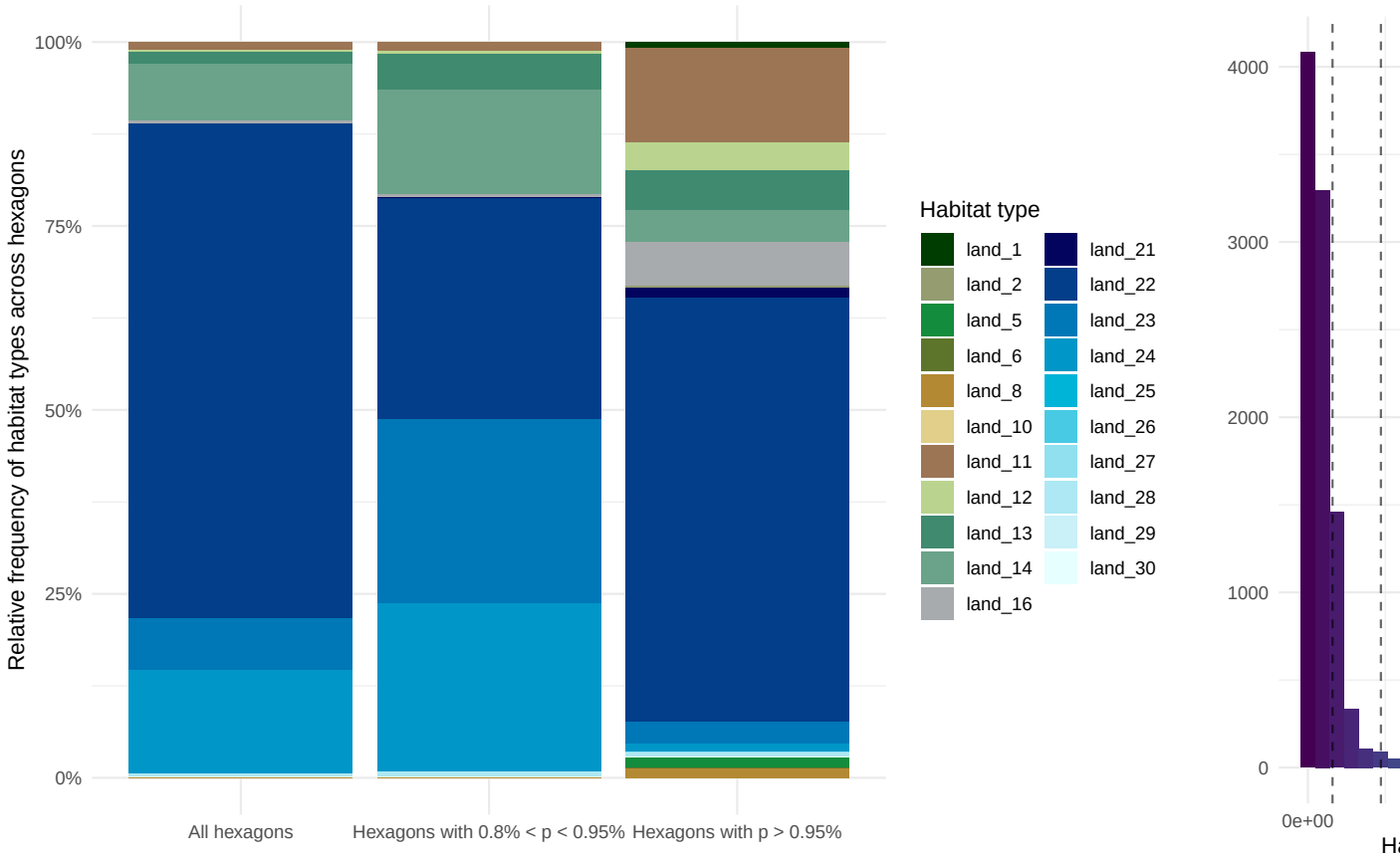
Ecoregion 31



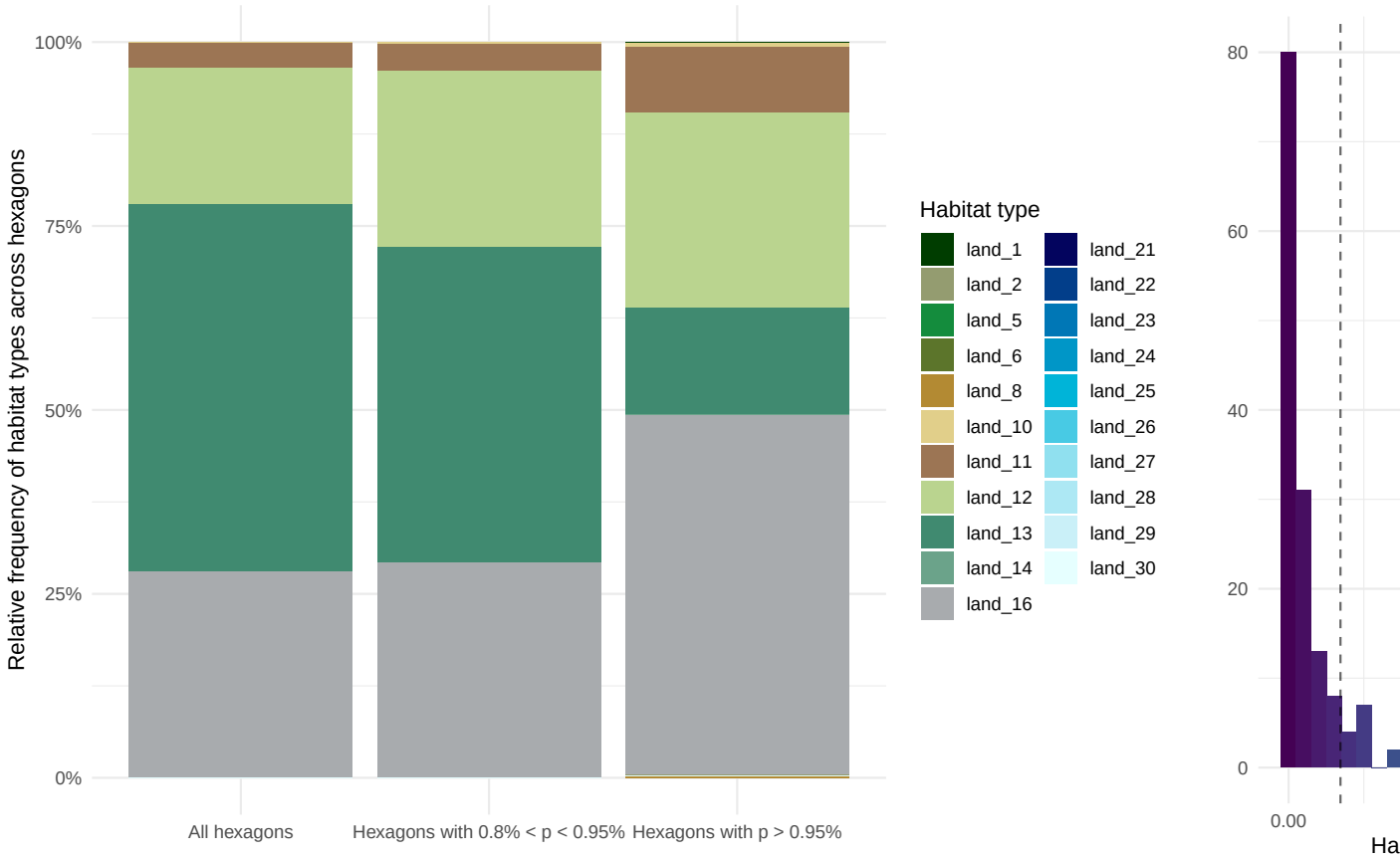
Ecoregion 46



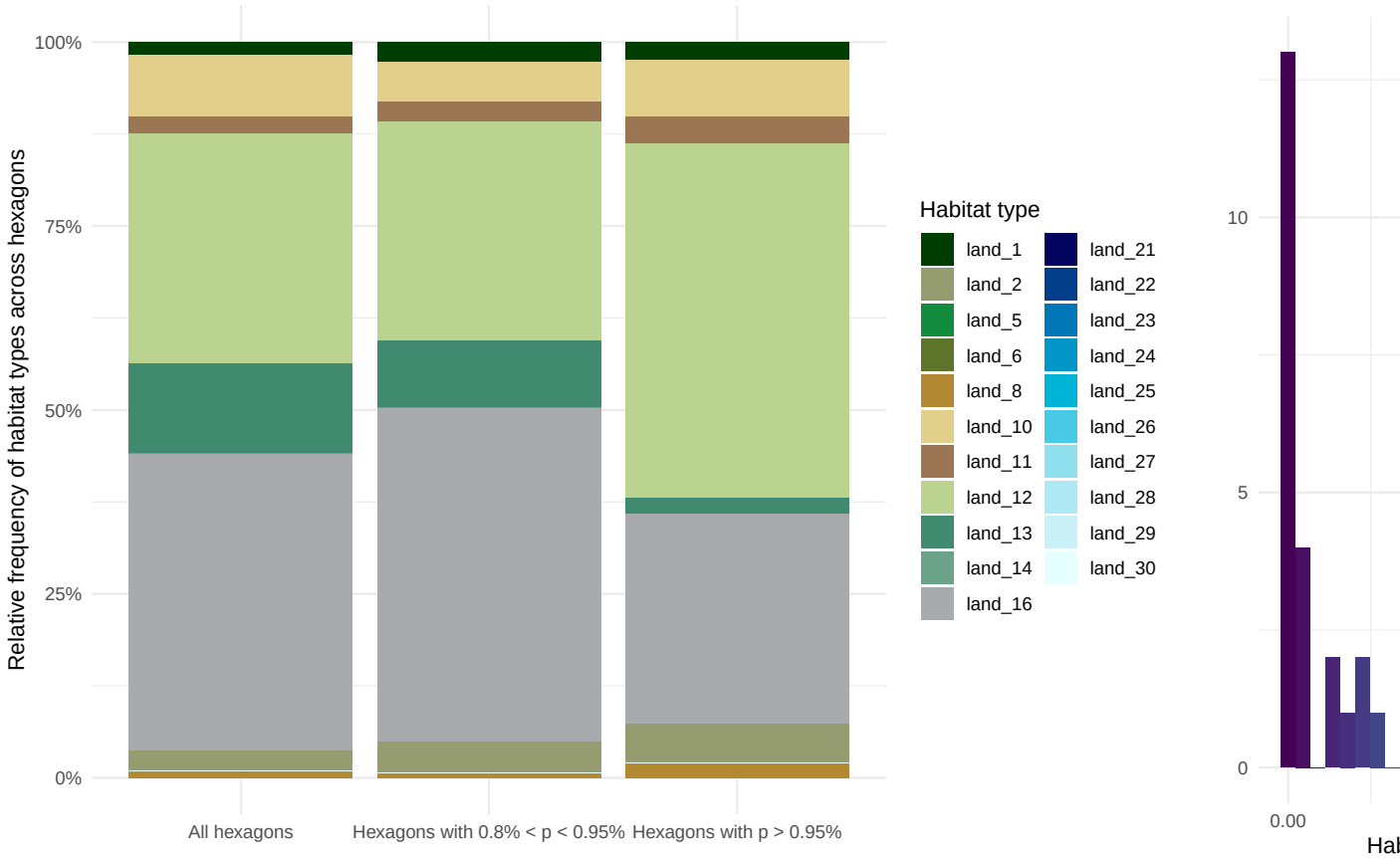
Ecoregion 47



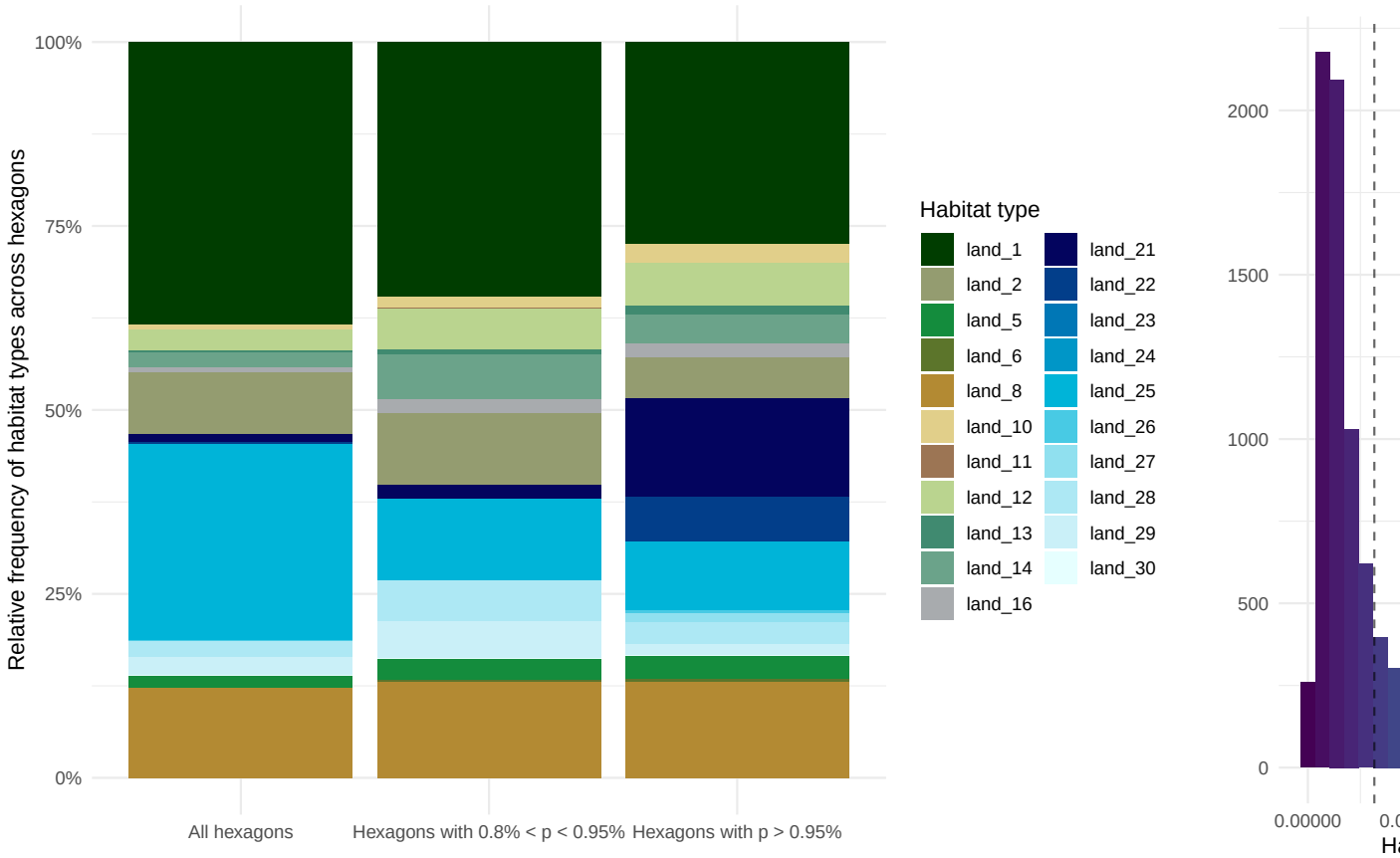
Ecoregion 48



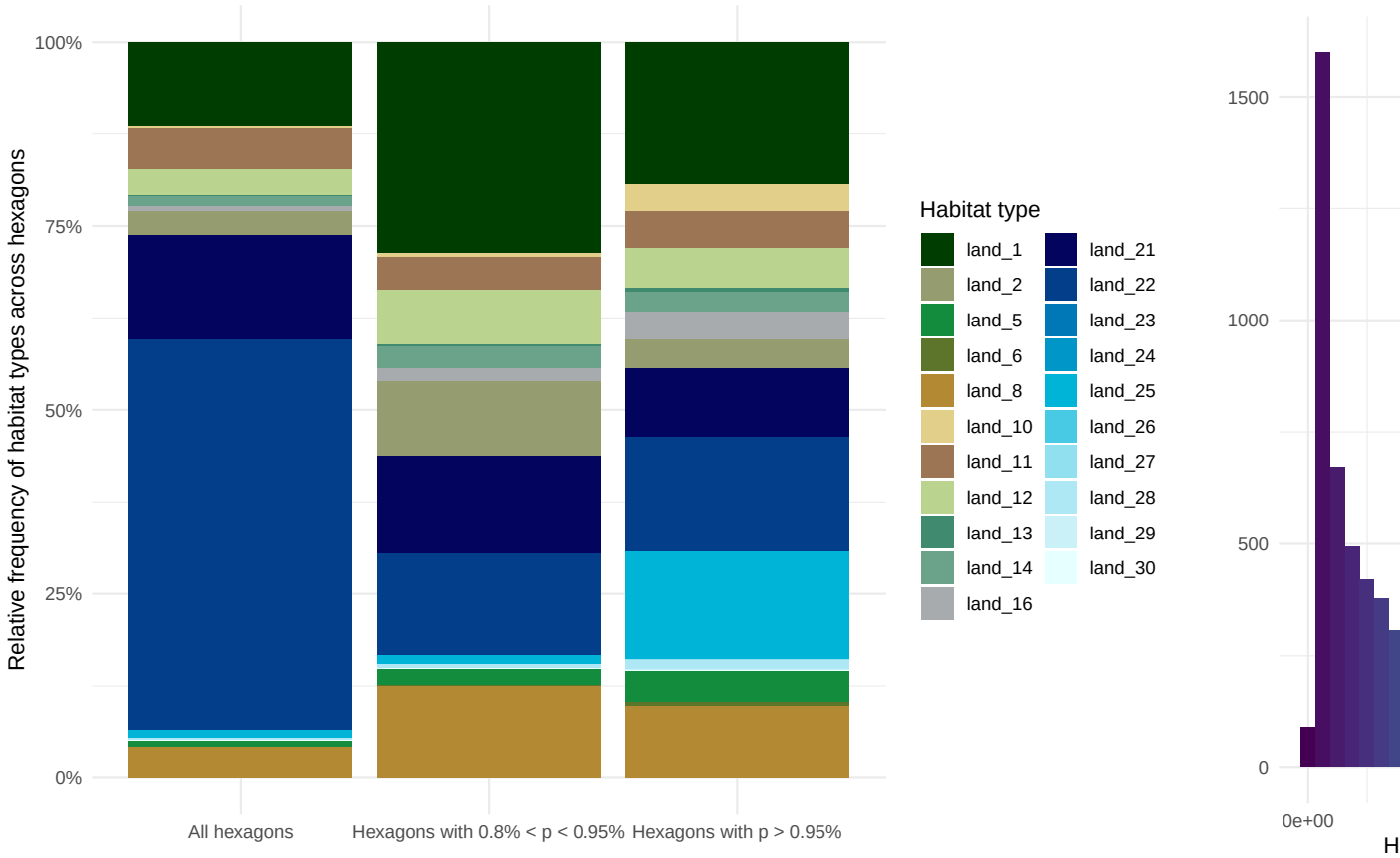
Ecoregion 49



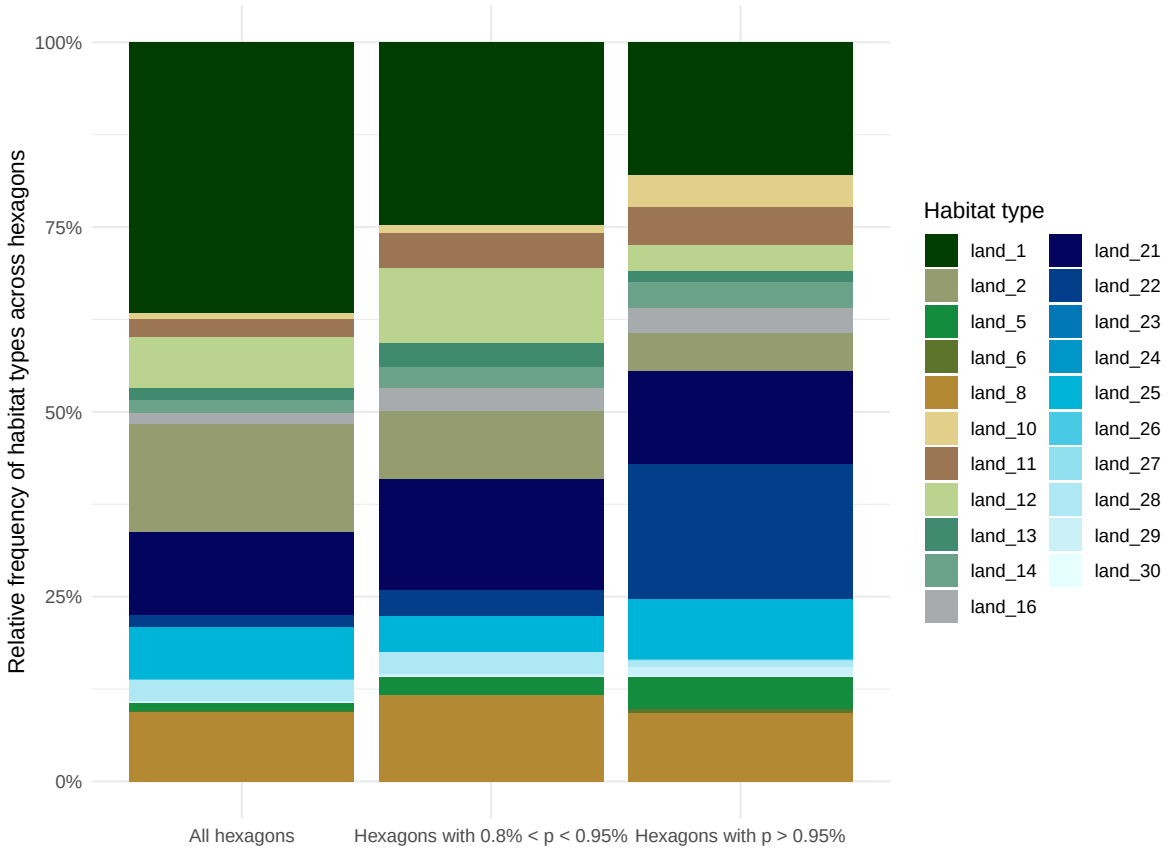
Ecoregion 72



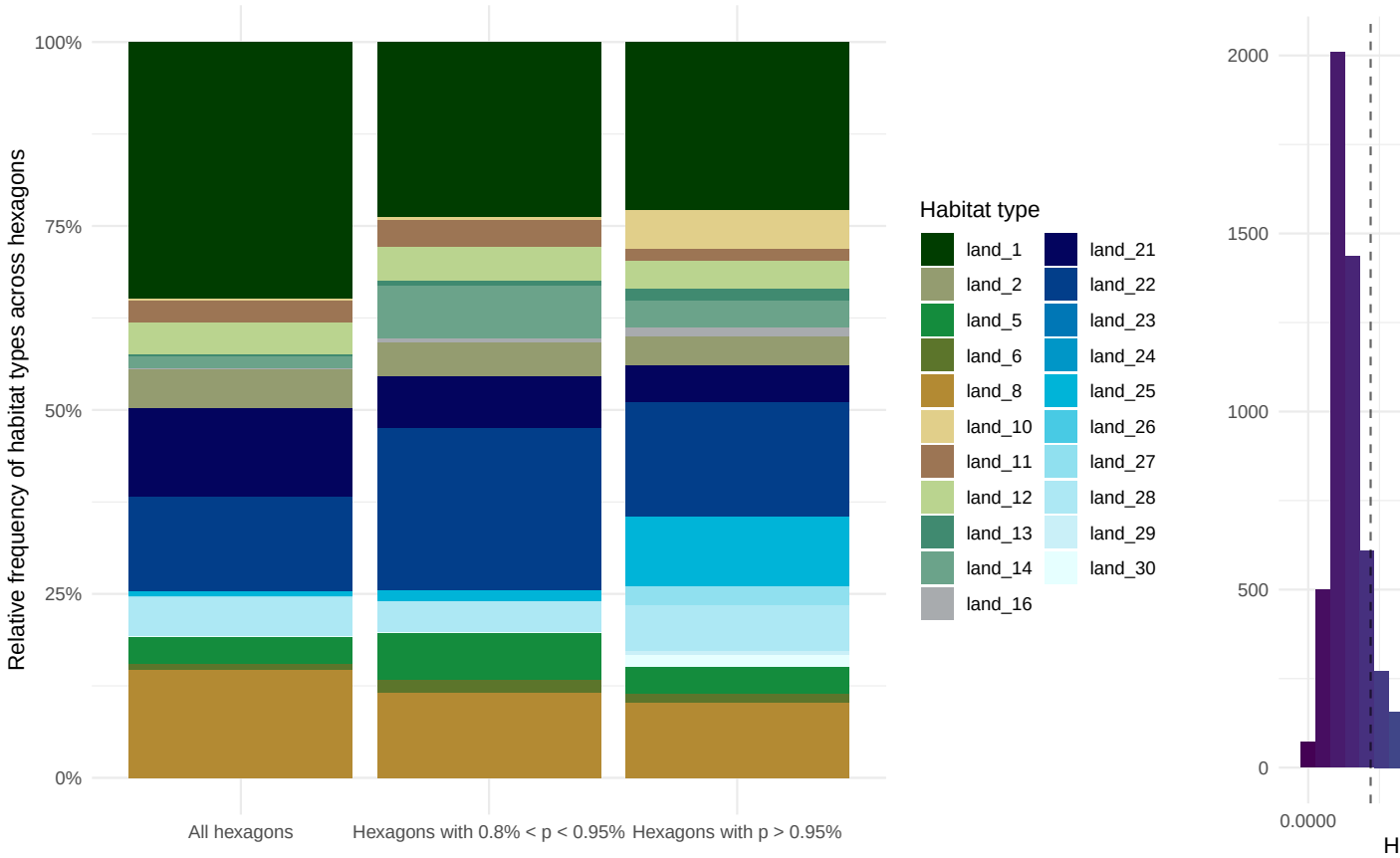
Ecoregion 73



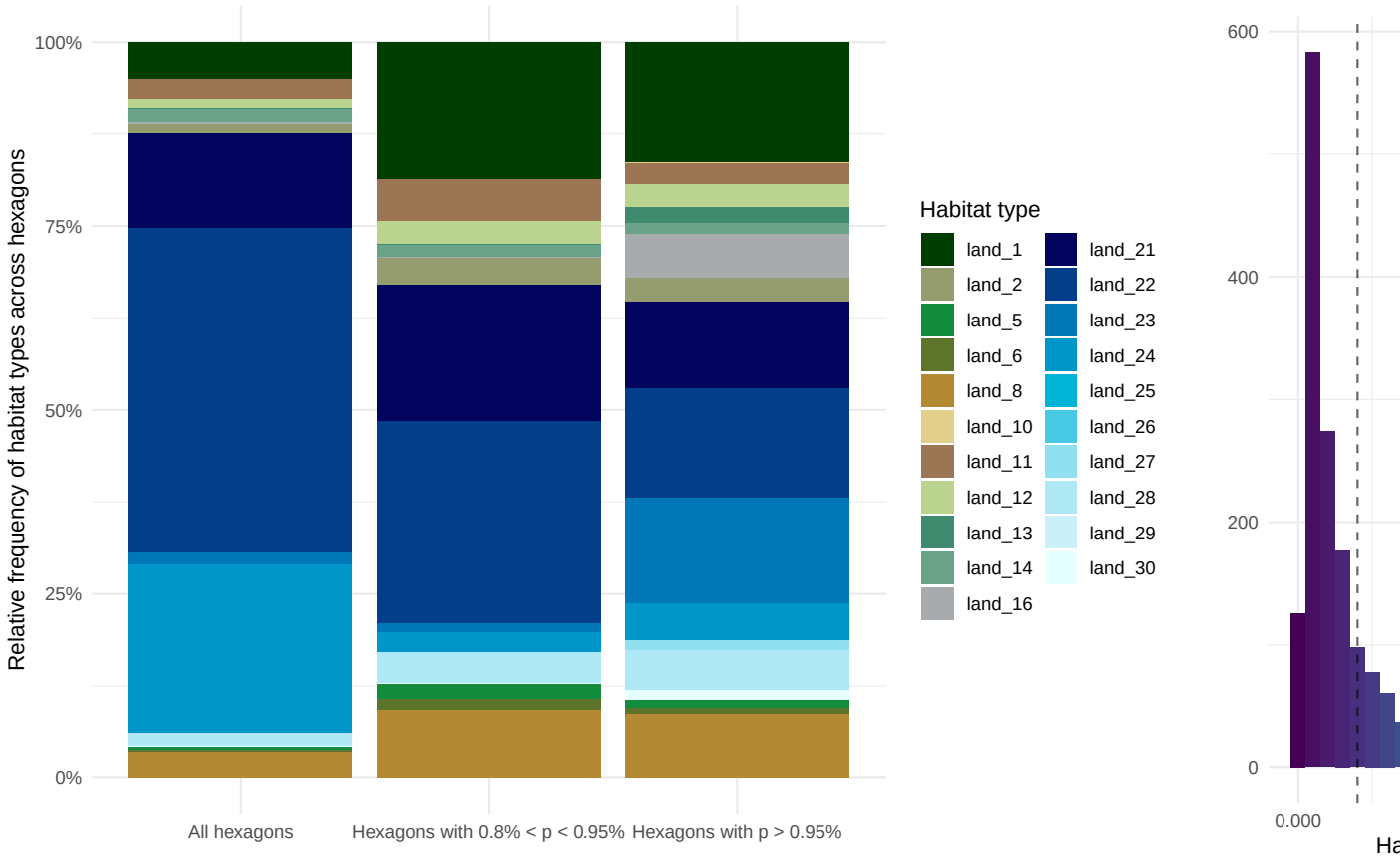
Ecoregion 74



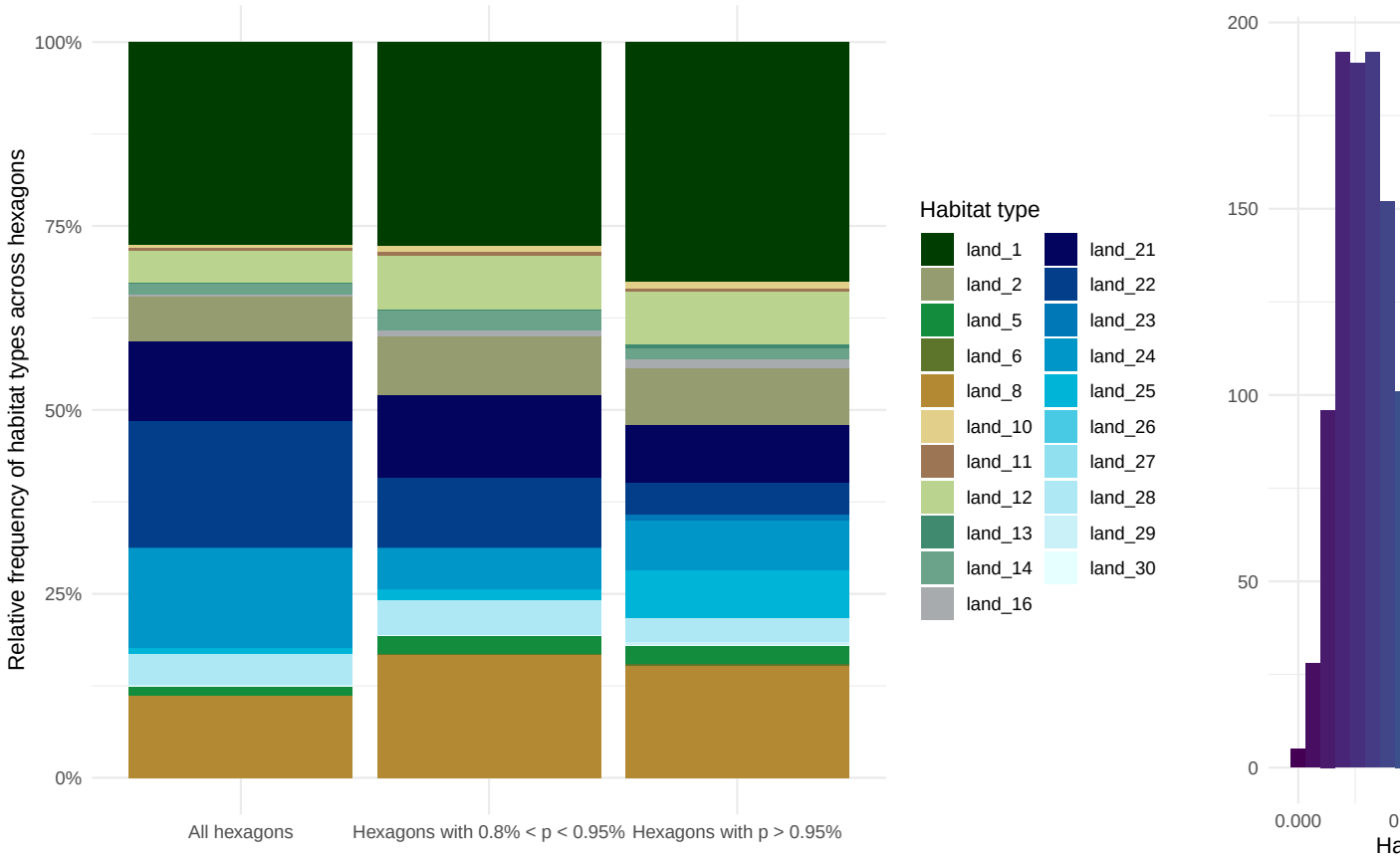
Ecoregion 75



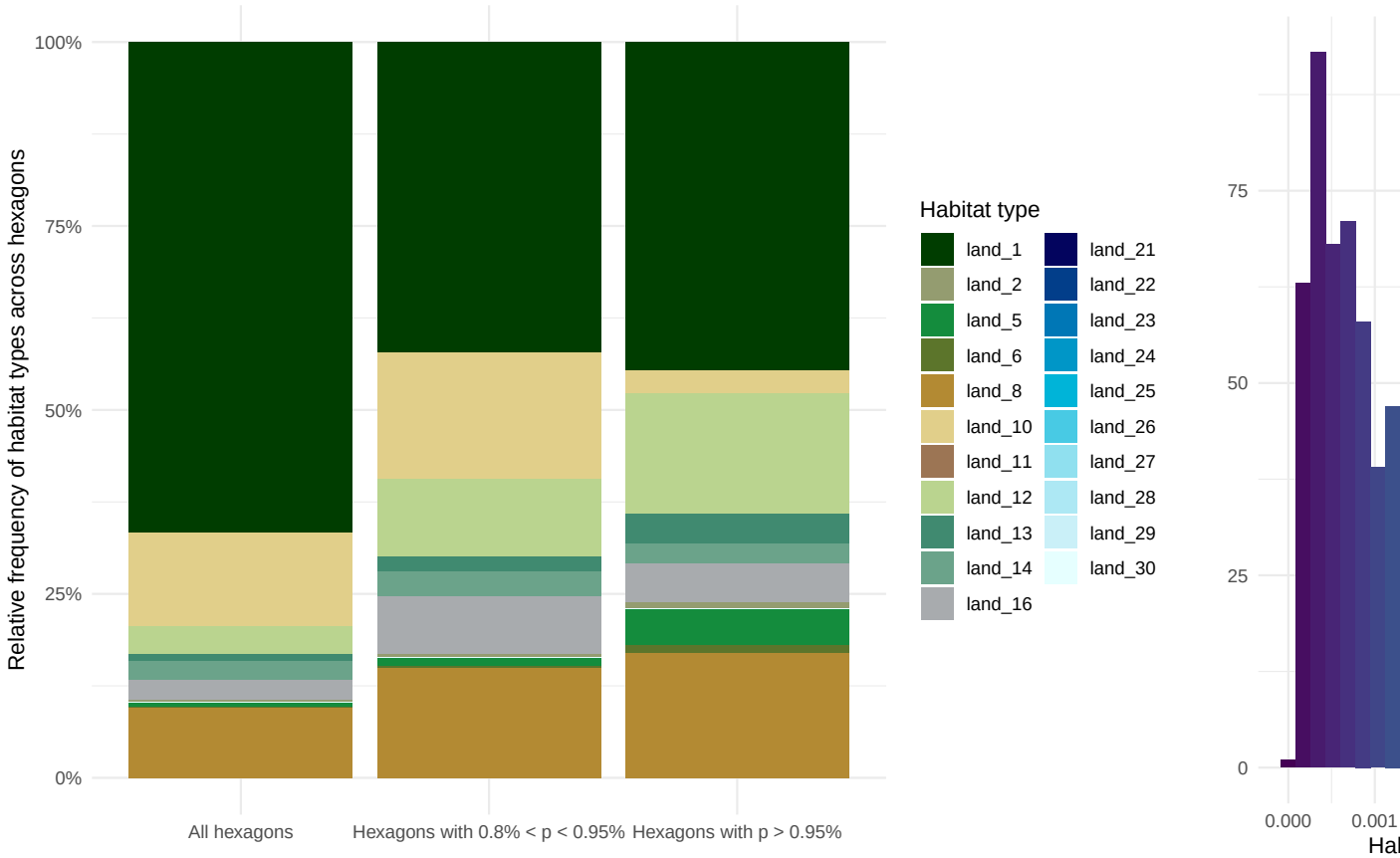
Ecoregion 76



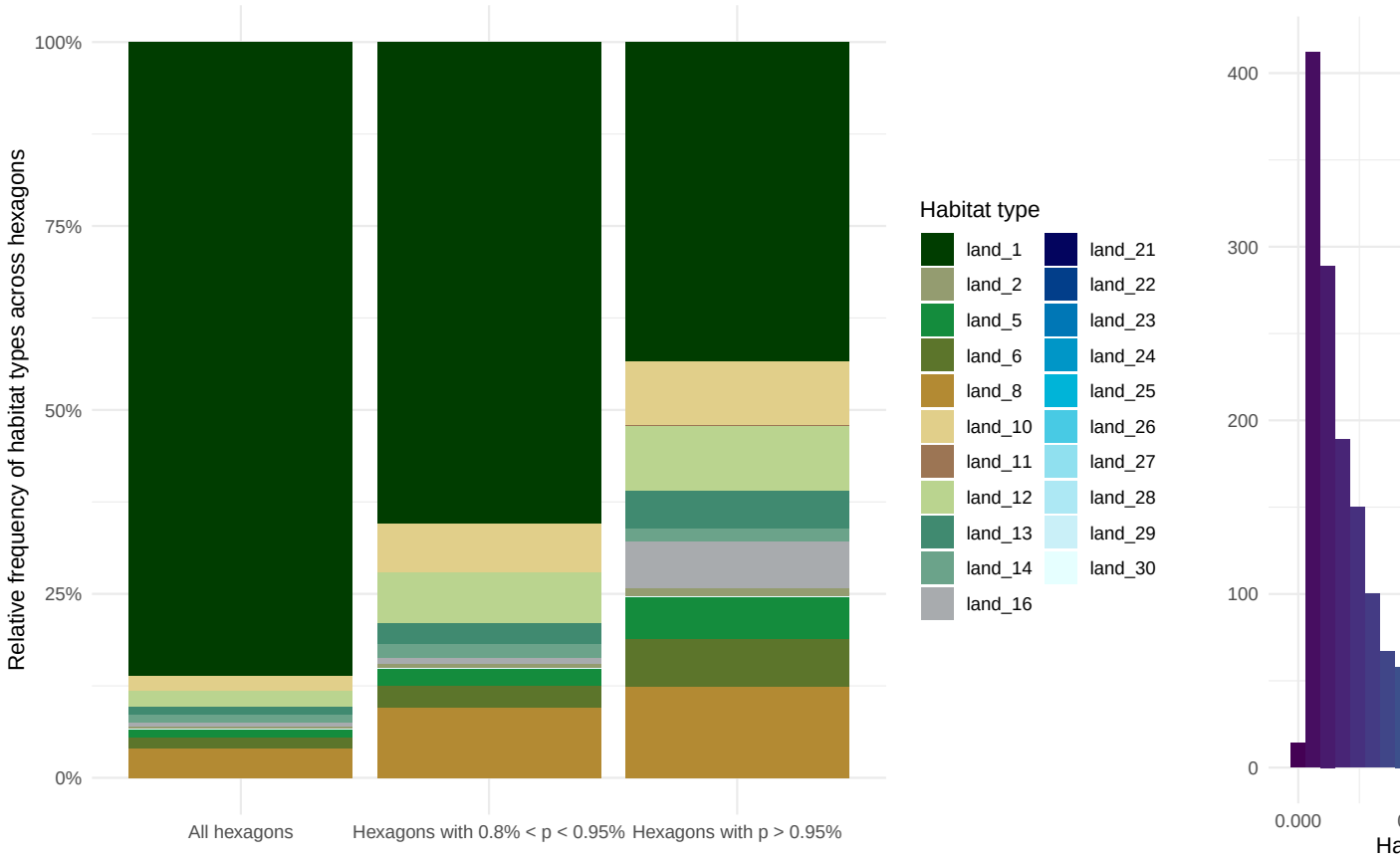
Ecoregion 77



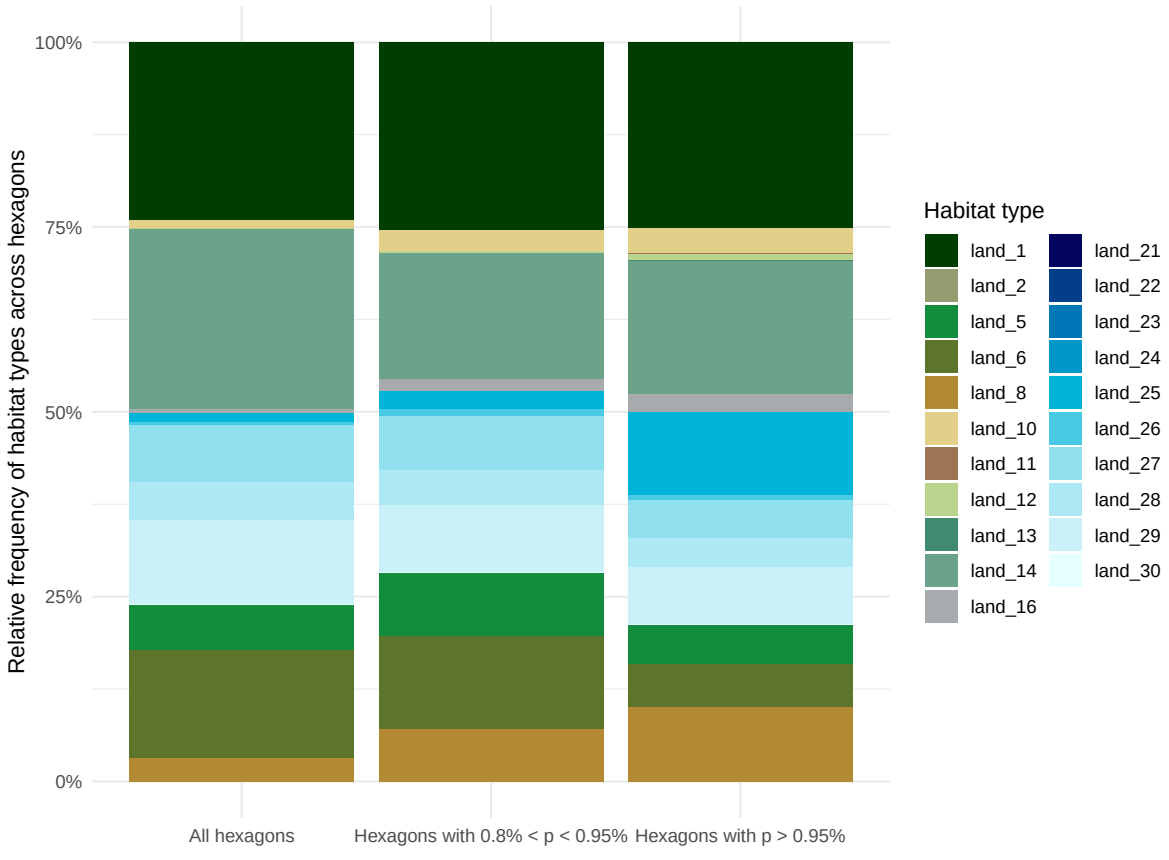
Ecoregion 78



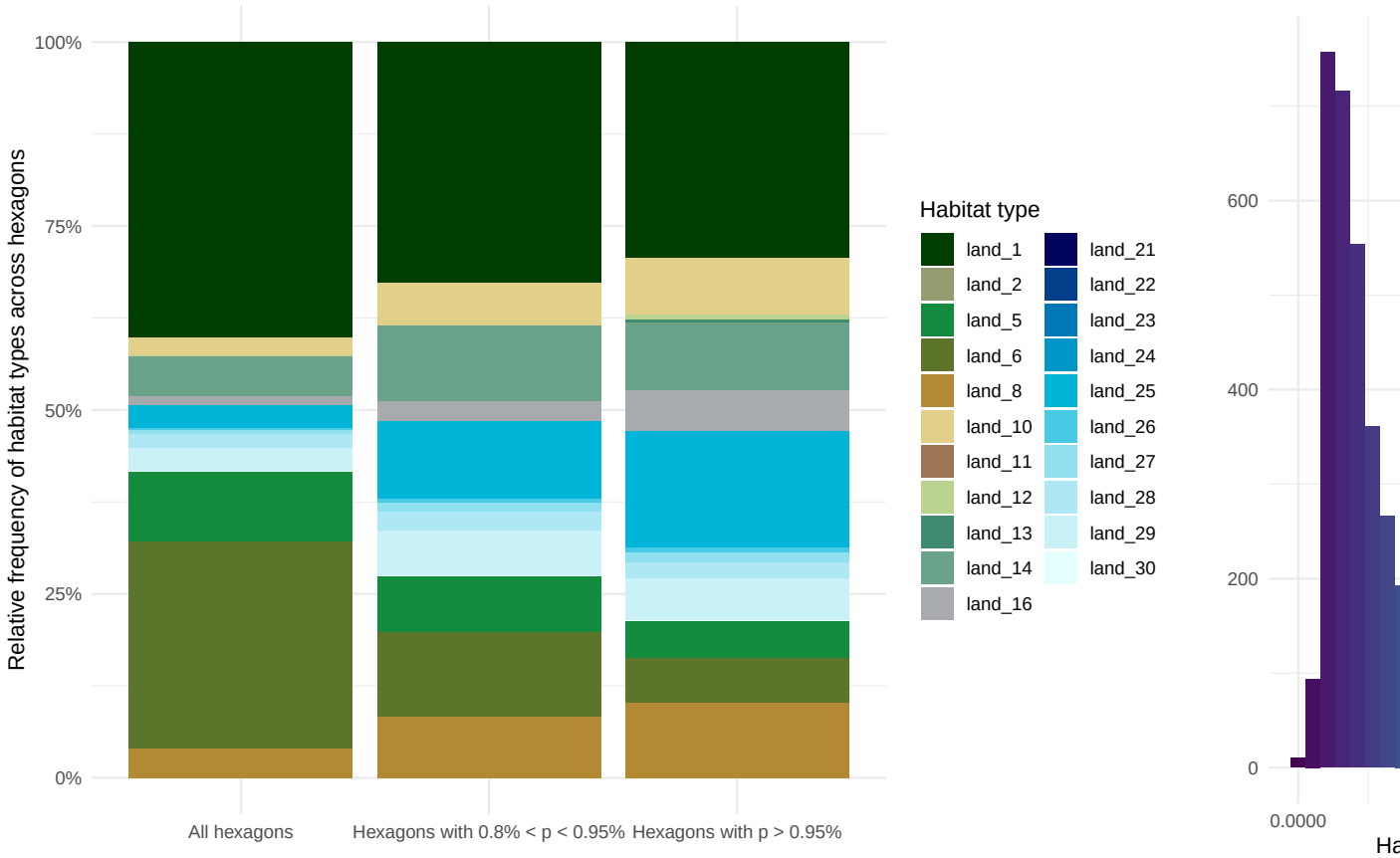
Ecoregion 86



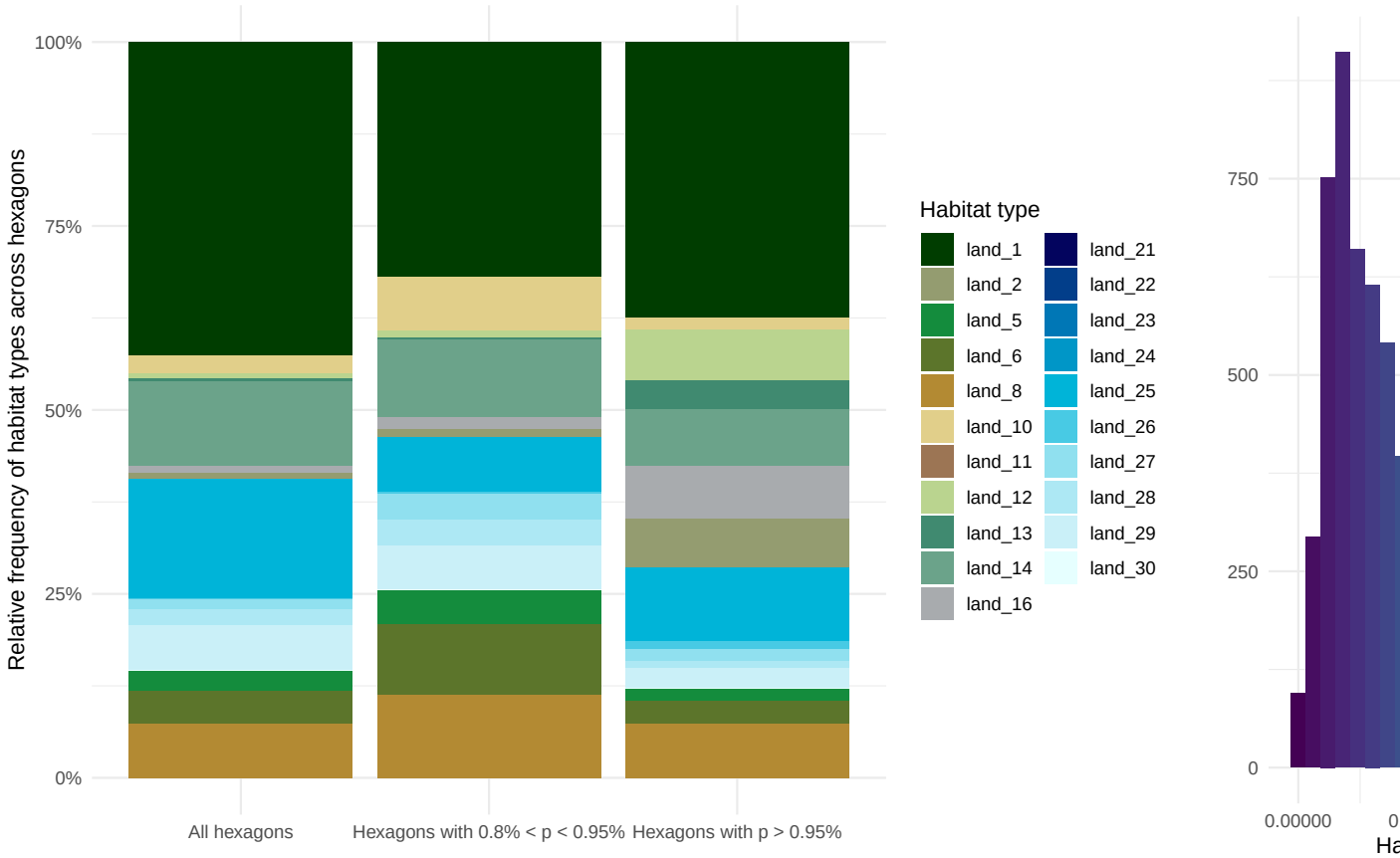
Ecoregion 96



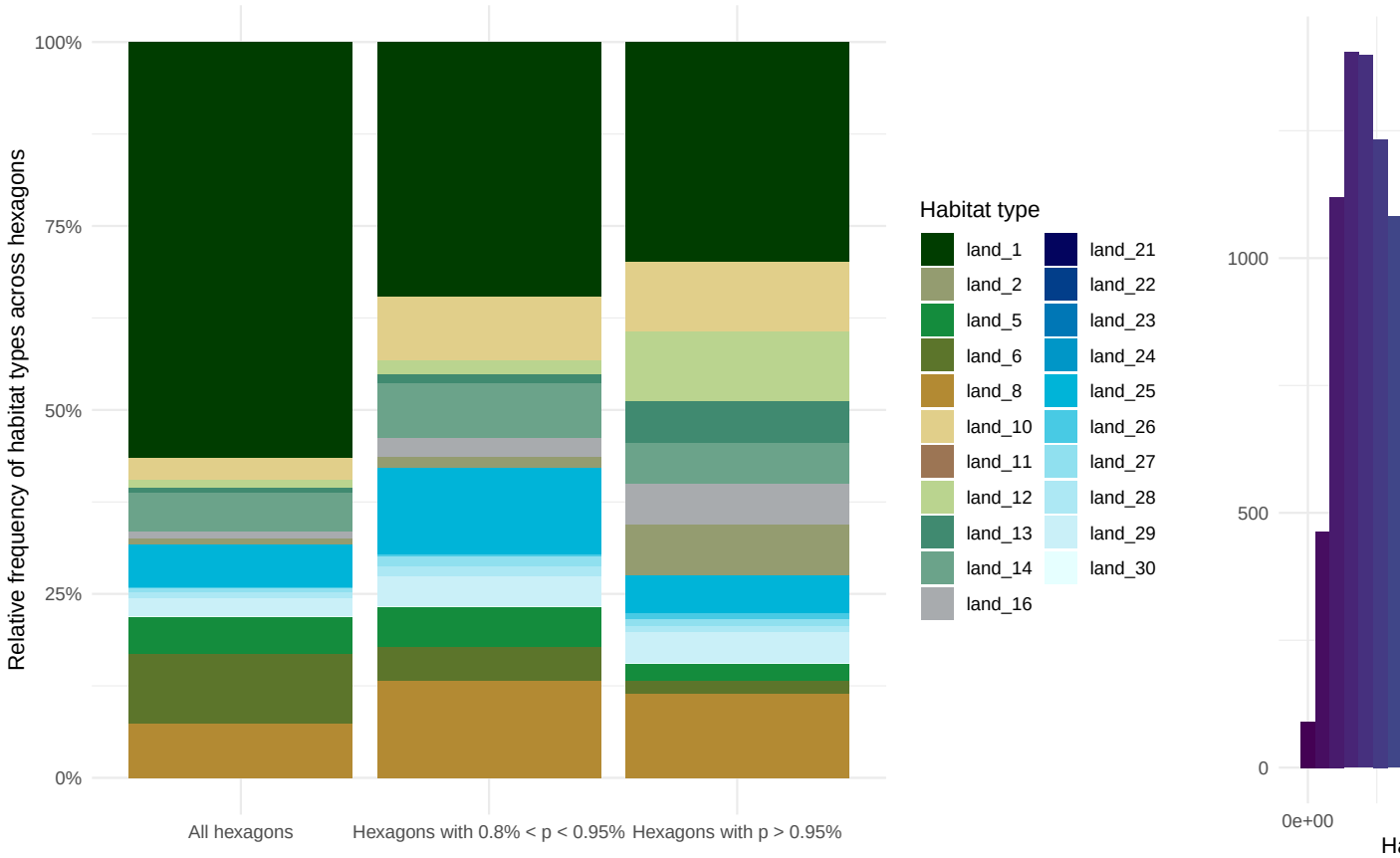
Ecoregion 99



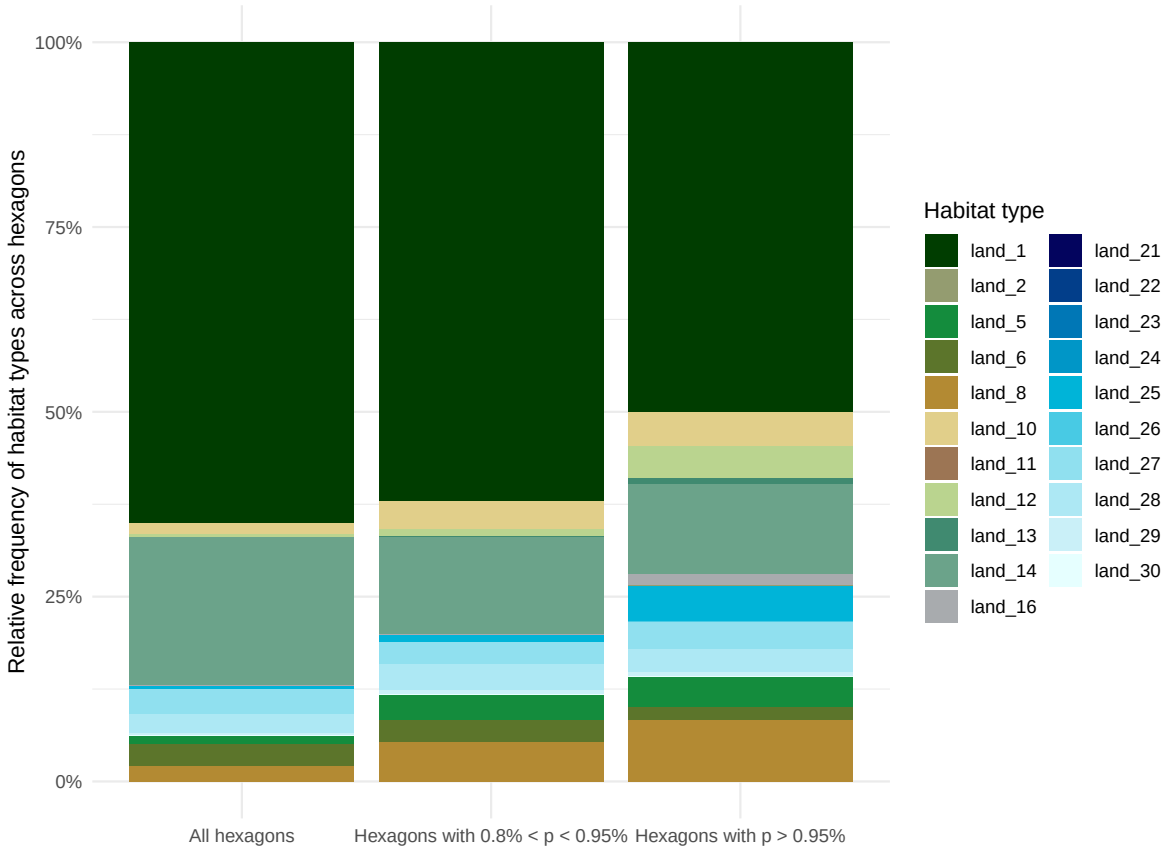
Ecoregion 100



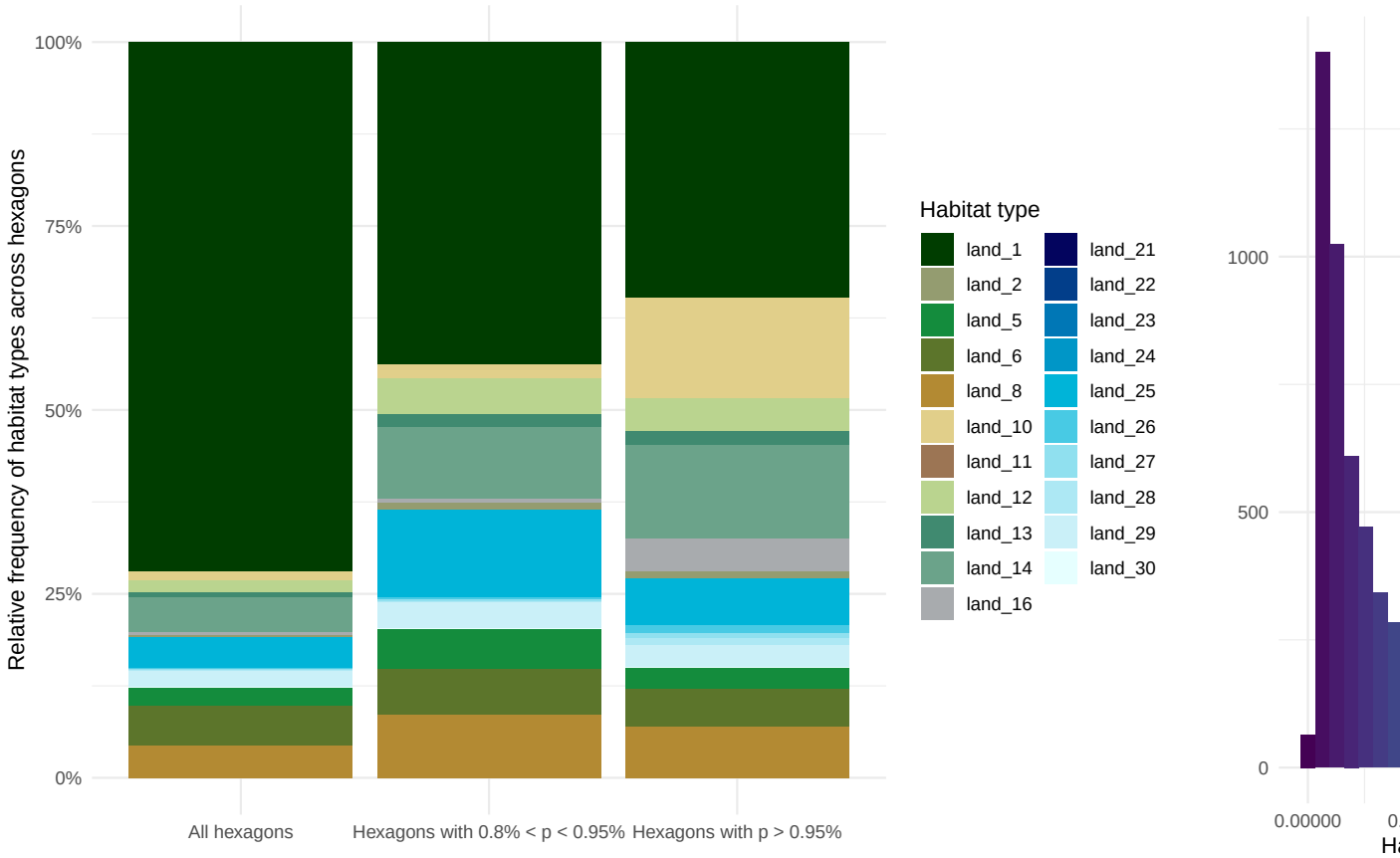
Ecoregion 101



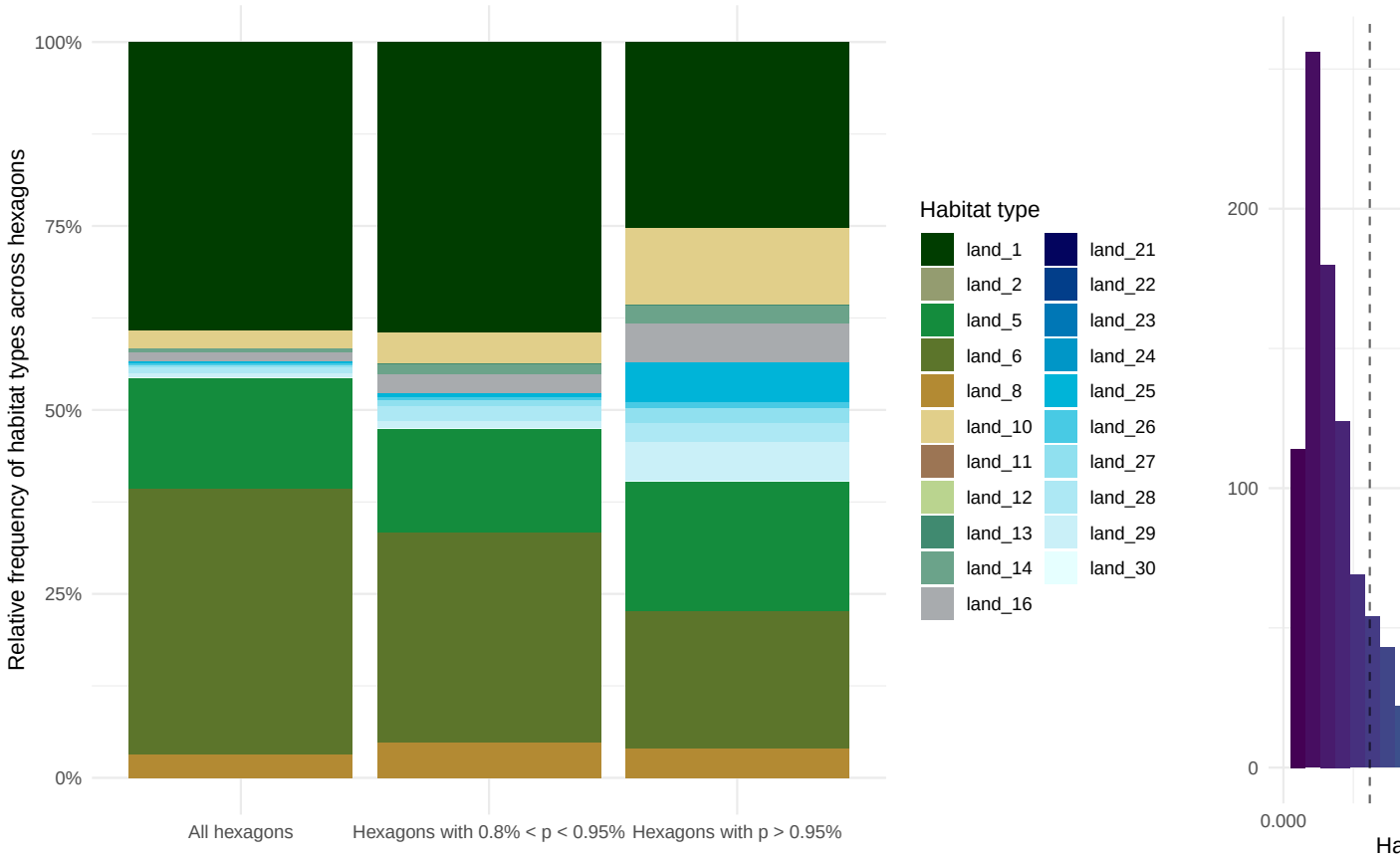
Ecoregion 102



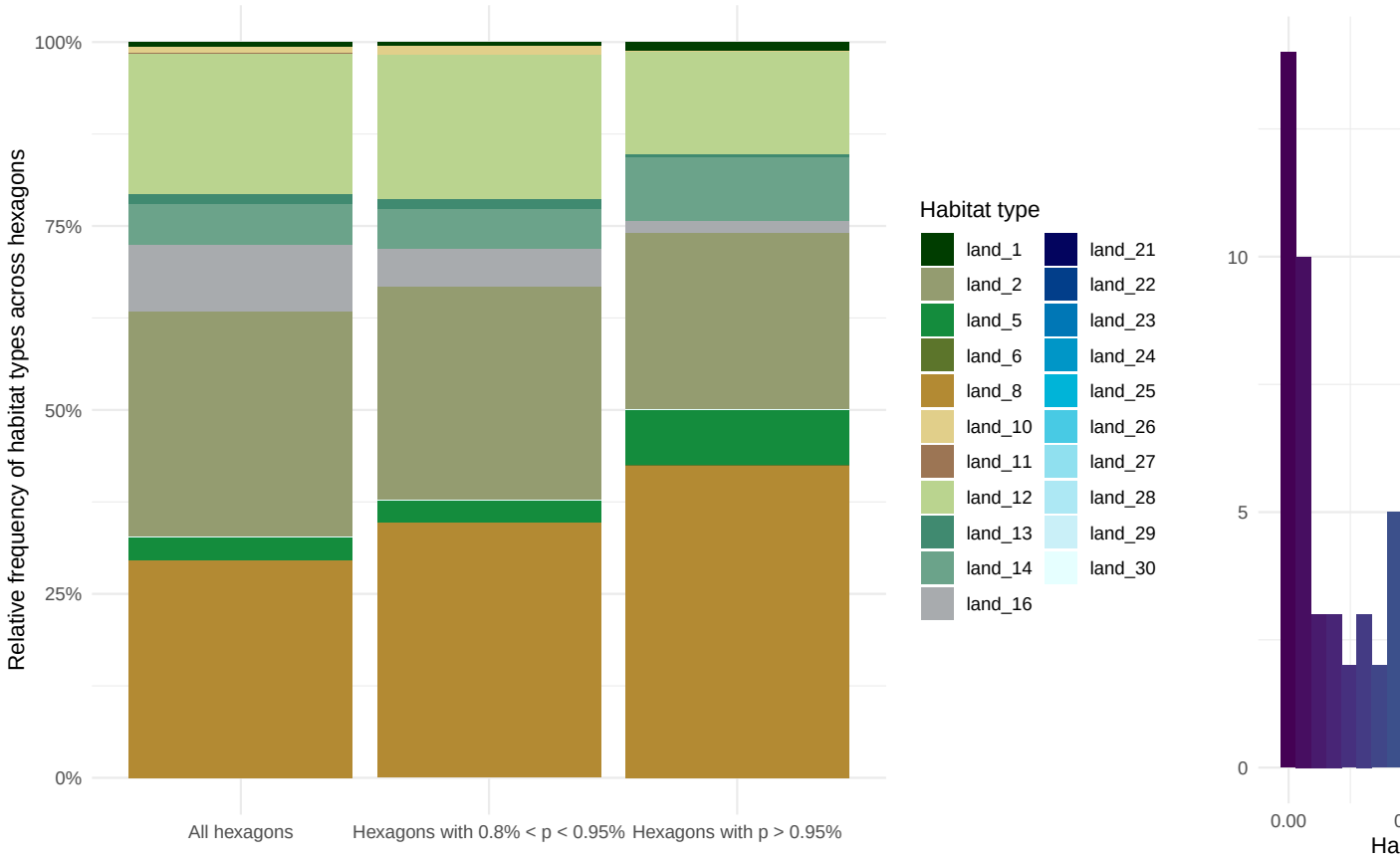
Ecoregion 103



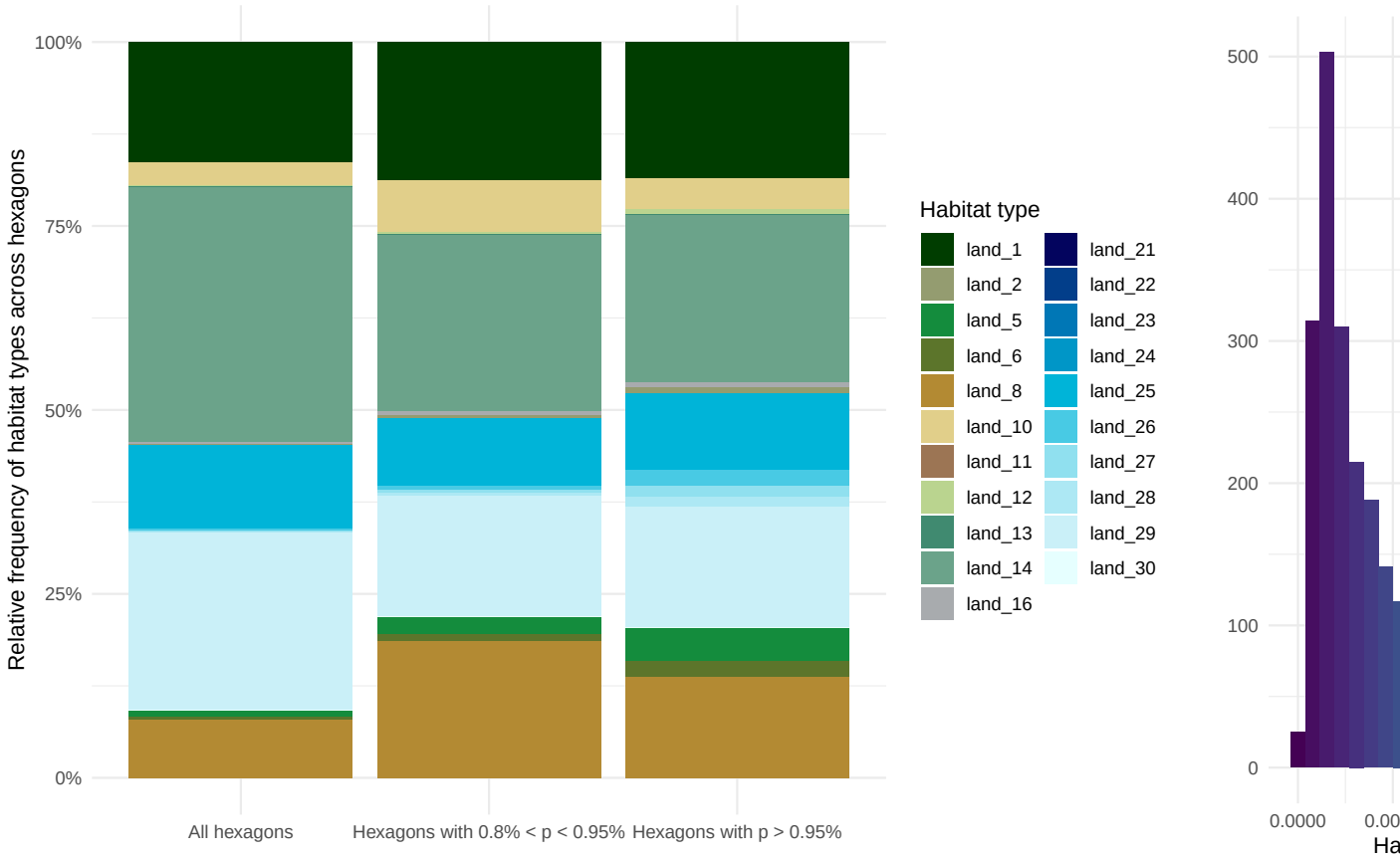
Ecoregion 117



Ecoregion 216



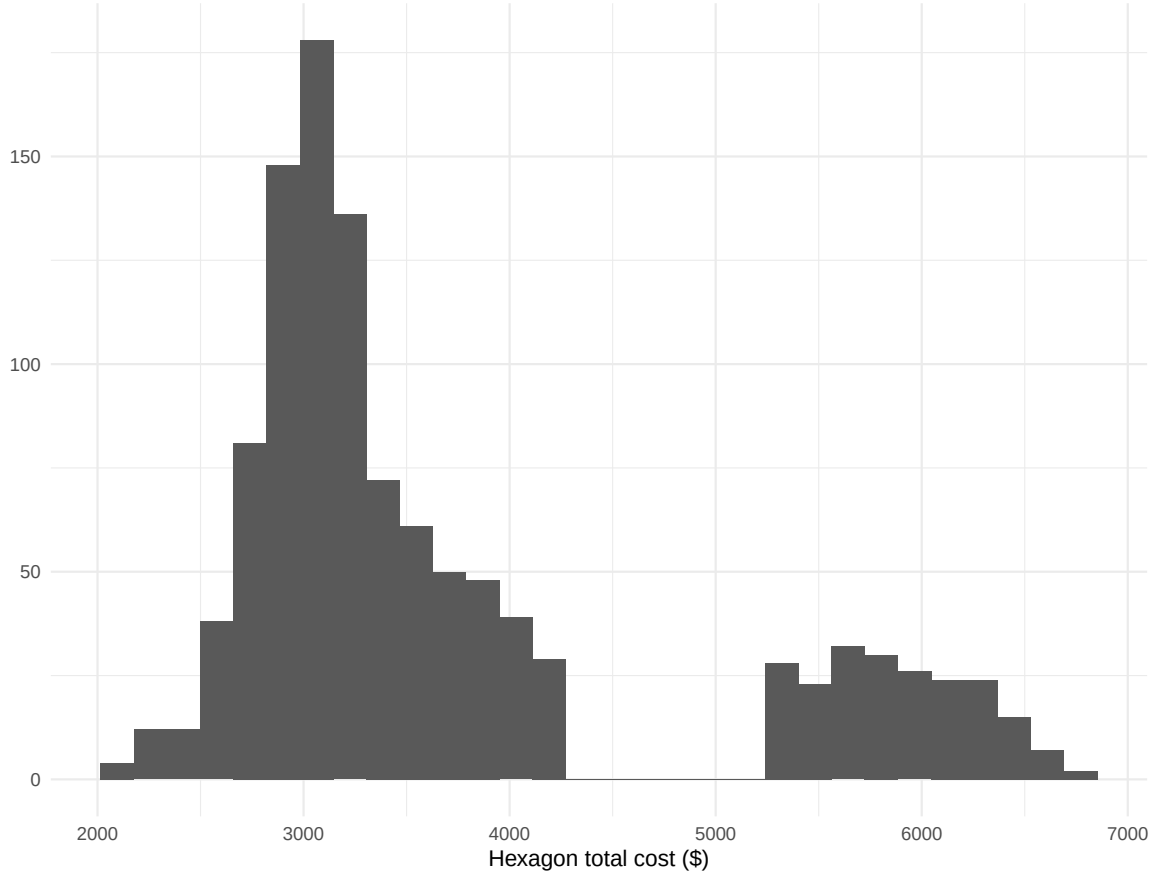
Ecoregion 217


Relative frequency of habitat types across hexagons
100%
75%
50%
25%
0%
All hexagons
Hexagons with 0.8% < p < 0.95%
Hexagons with p > 0.95%
Habitat type
land_1
land_2
land_5
land_6
land_8
land_10
land_11
land_12
land_13
land_14
land_16
land_21
land_22
land_23
land_24
land_25
land_26
land_27
land_28
land_29
land_30
500
400
300
200
100
0
0.0000
0.00
Ha

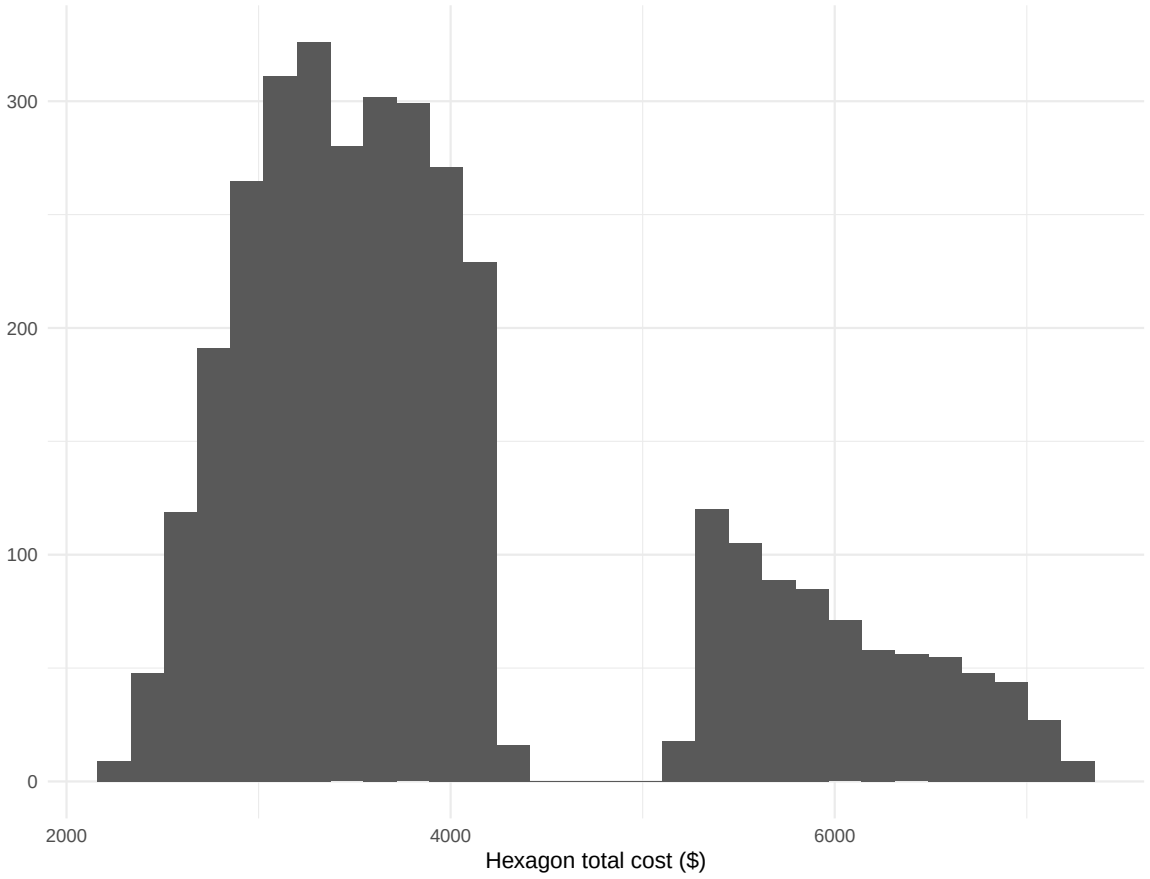
10 Ecoregion summary: cost

This section provides a summary of the total cost and cost inclusion probability obtained from the different layers described in Chapter 4. The first figure is the histogram of total cost distribution across the hexagons of an ecoregion. The second figure illustrates the spatial distribution of cost inclusion probability within the ecoregion.

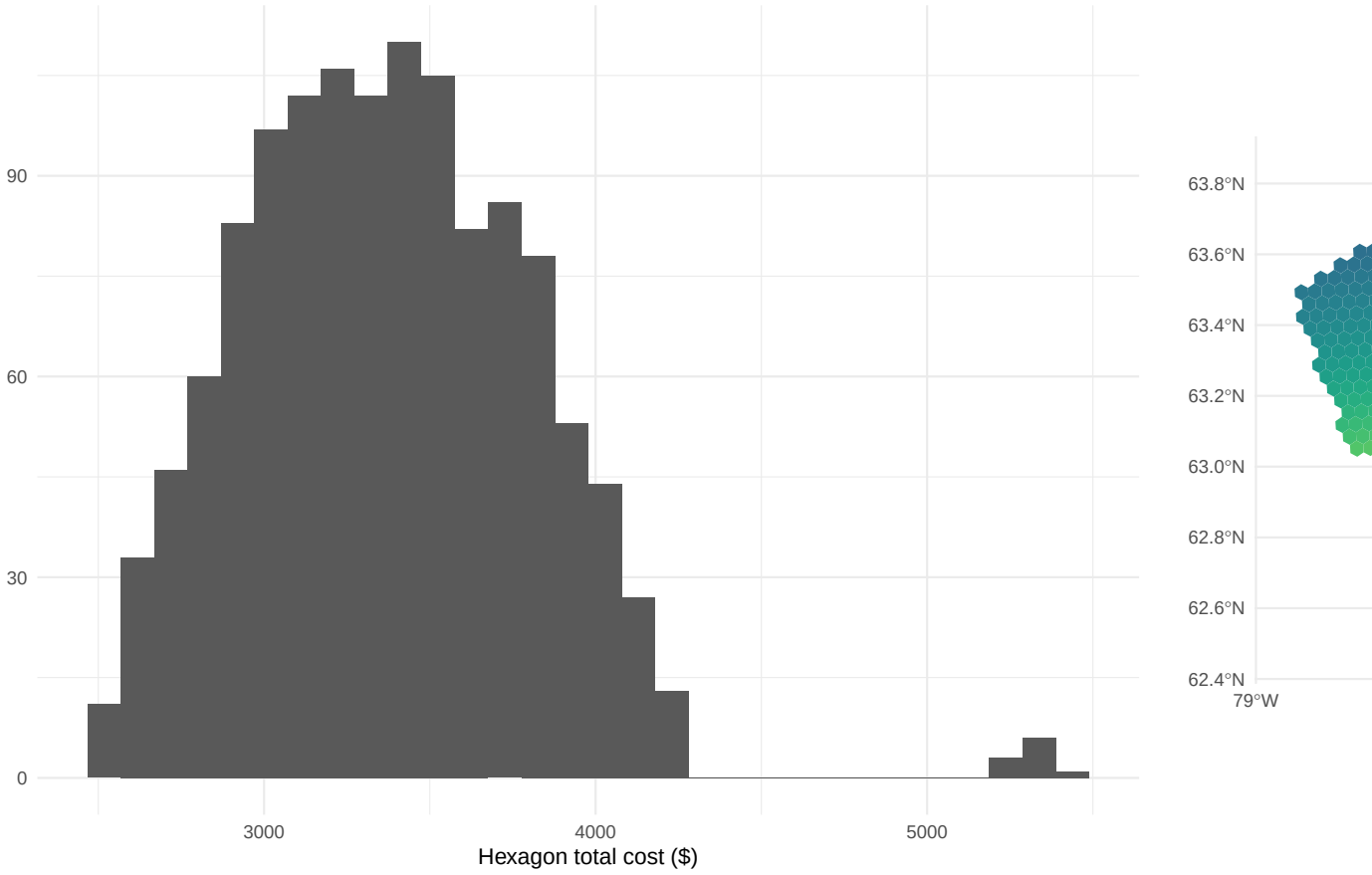
10.1 Ecoregion 7



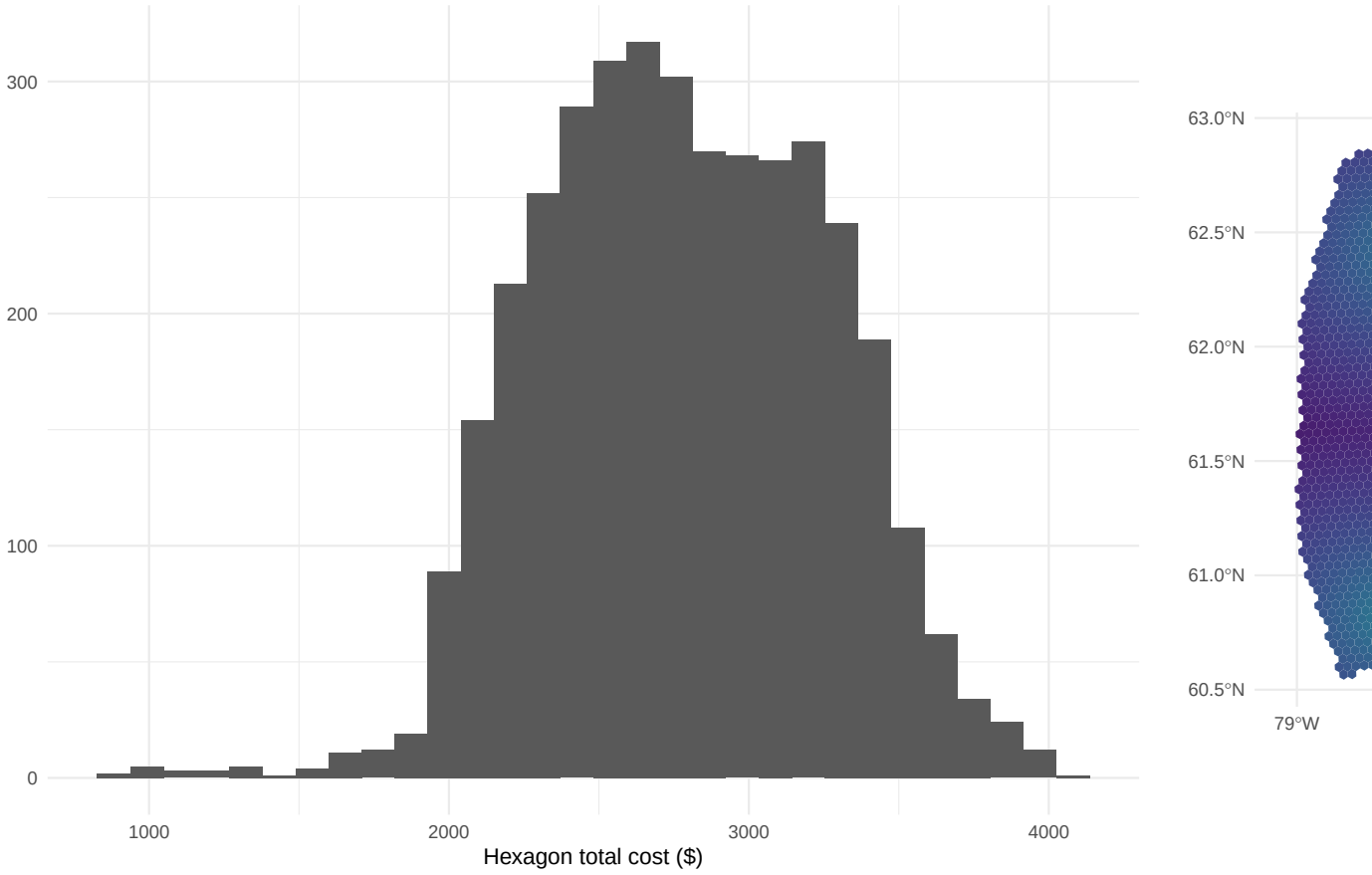
Ecoregion 28



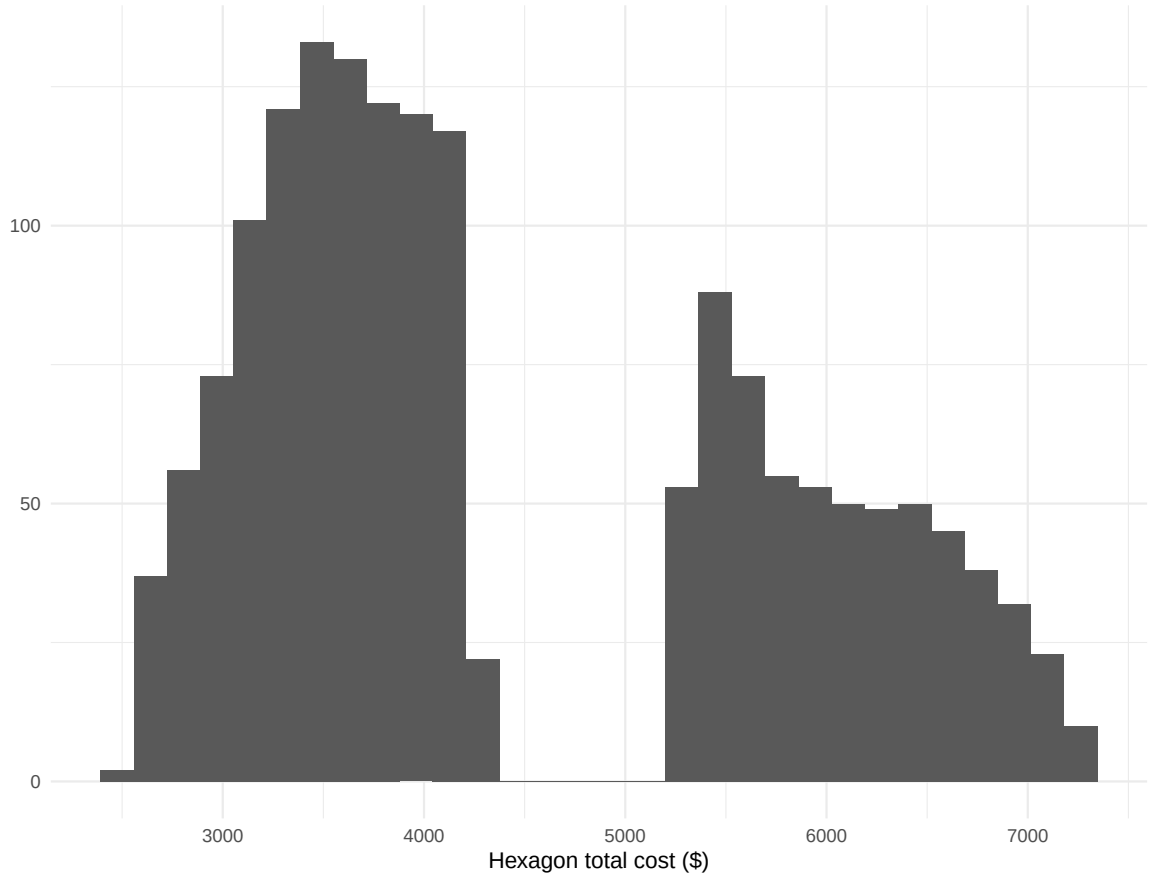
Ecoregion 30



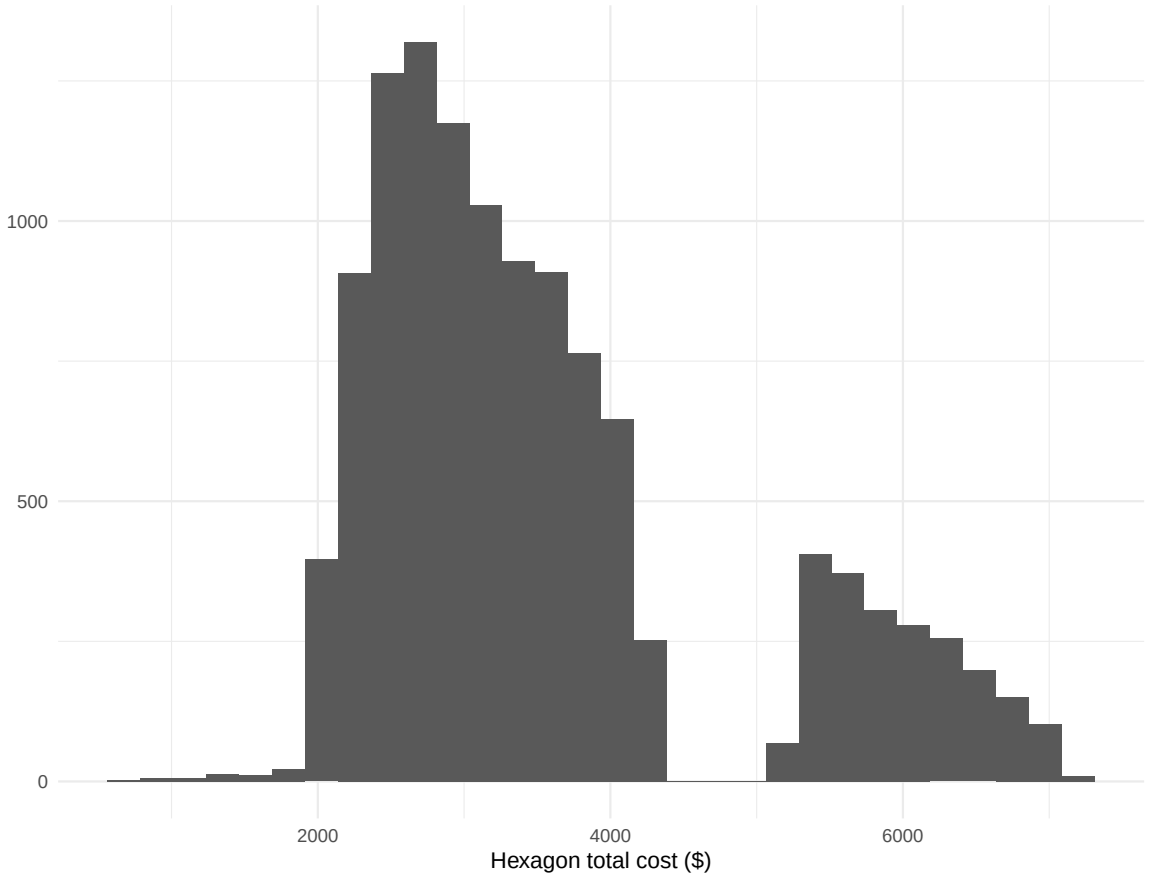
Ecoregion 31



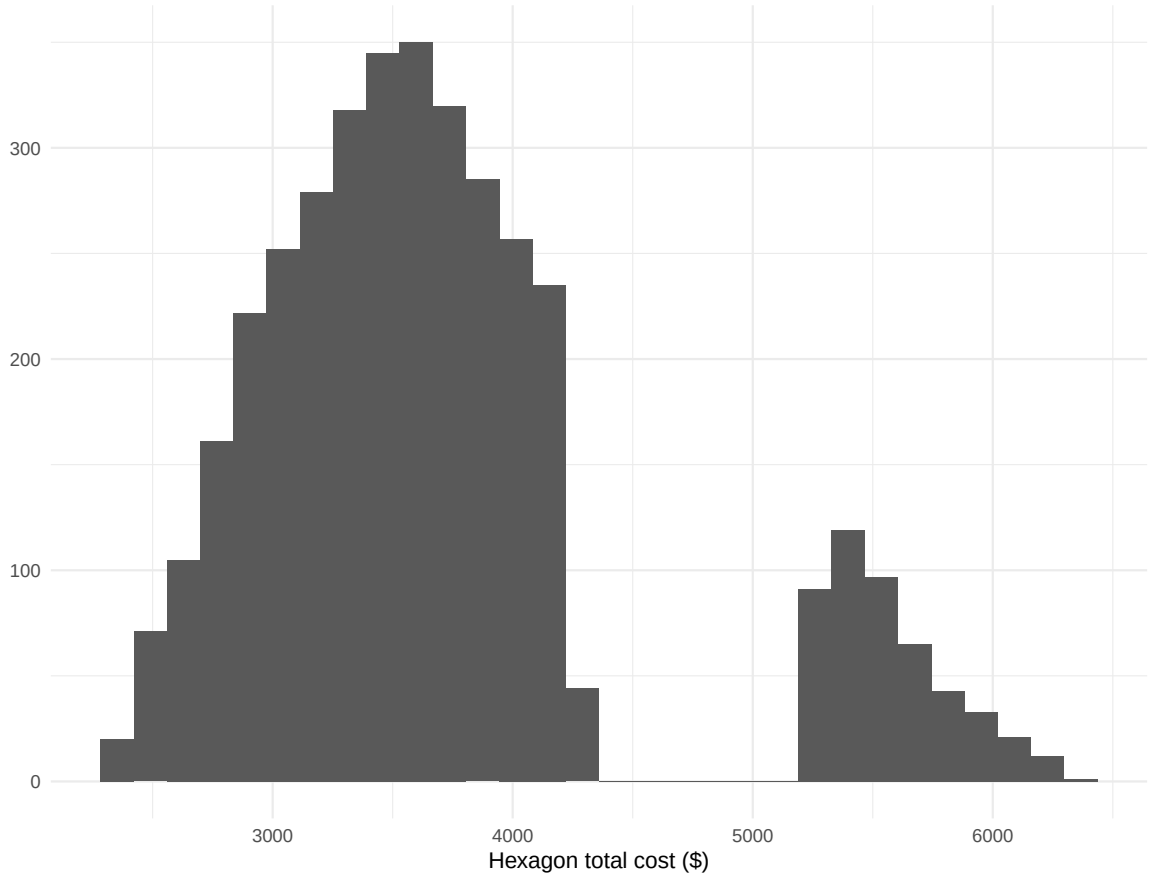
Ecoregion 46



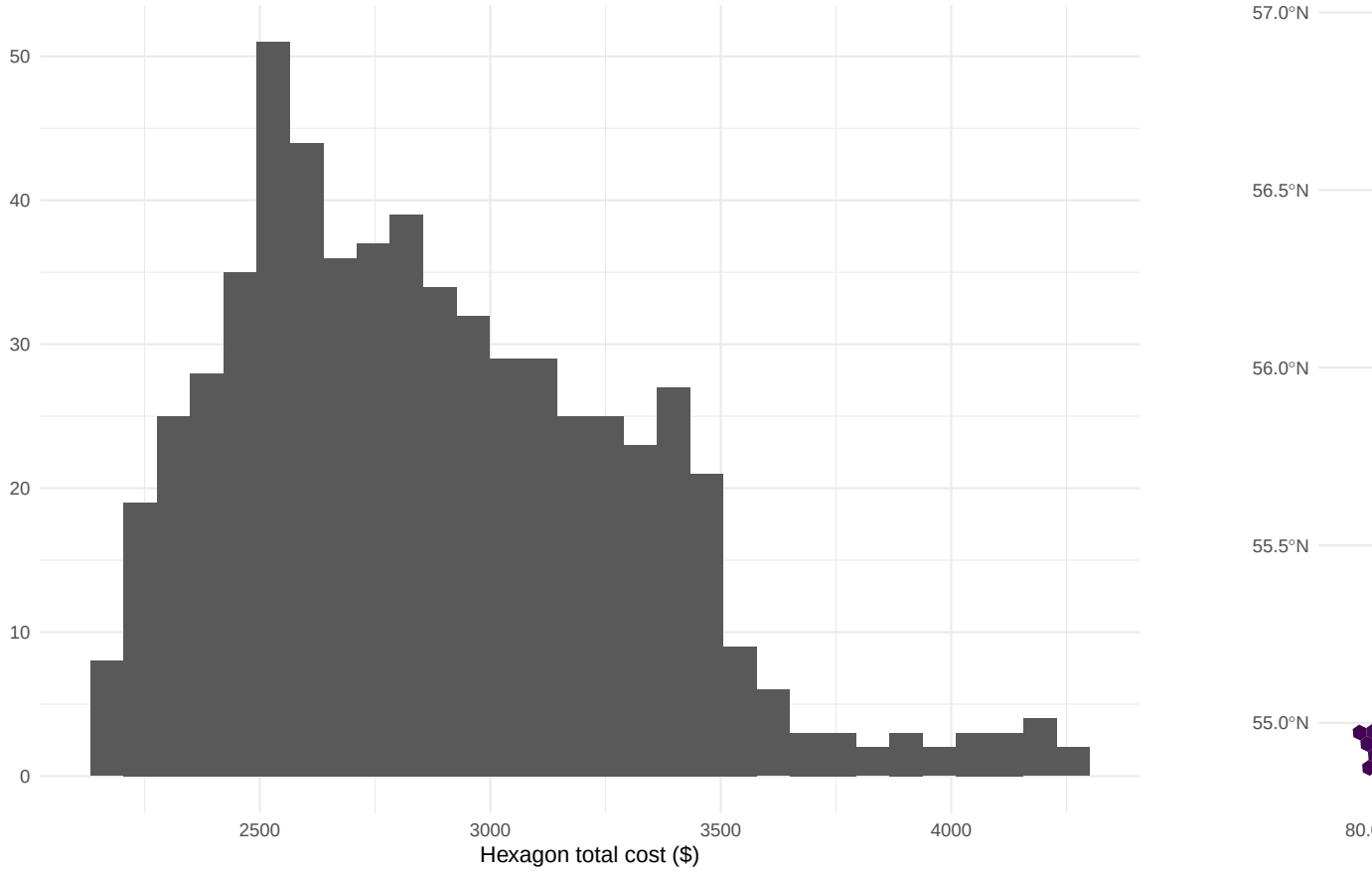
Ecoregion 47



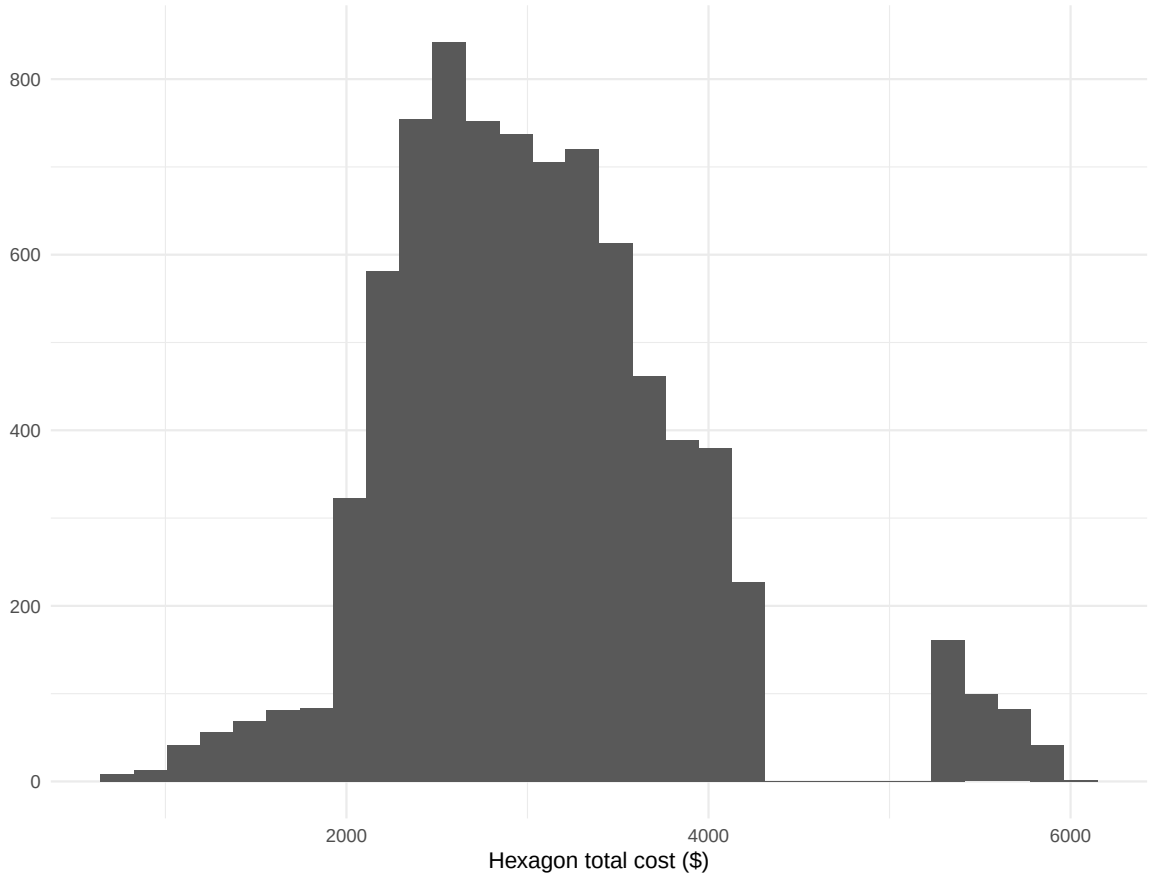
Ecoregion 48



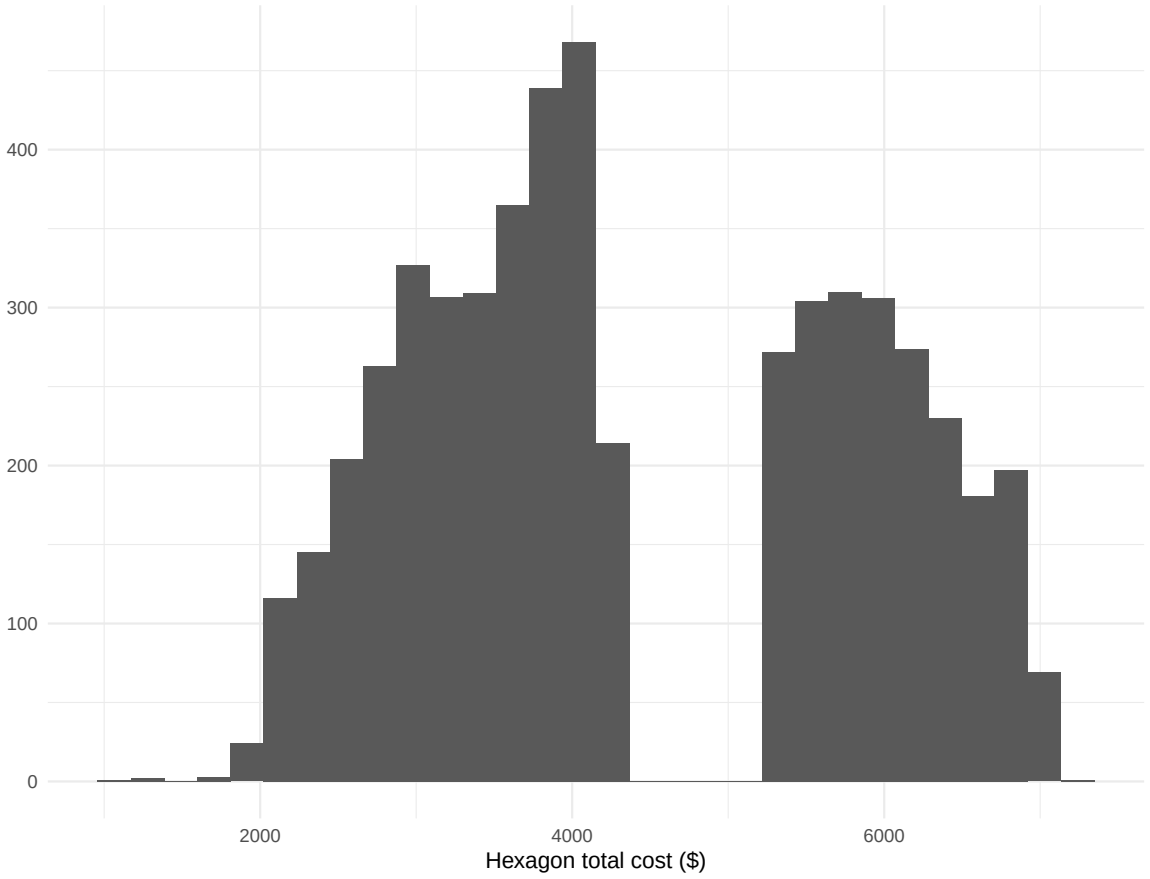
Ecoregion 49



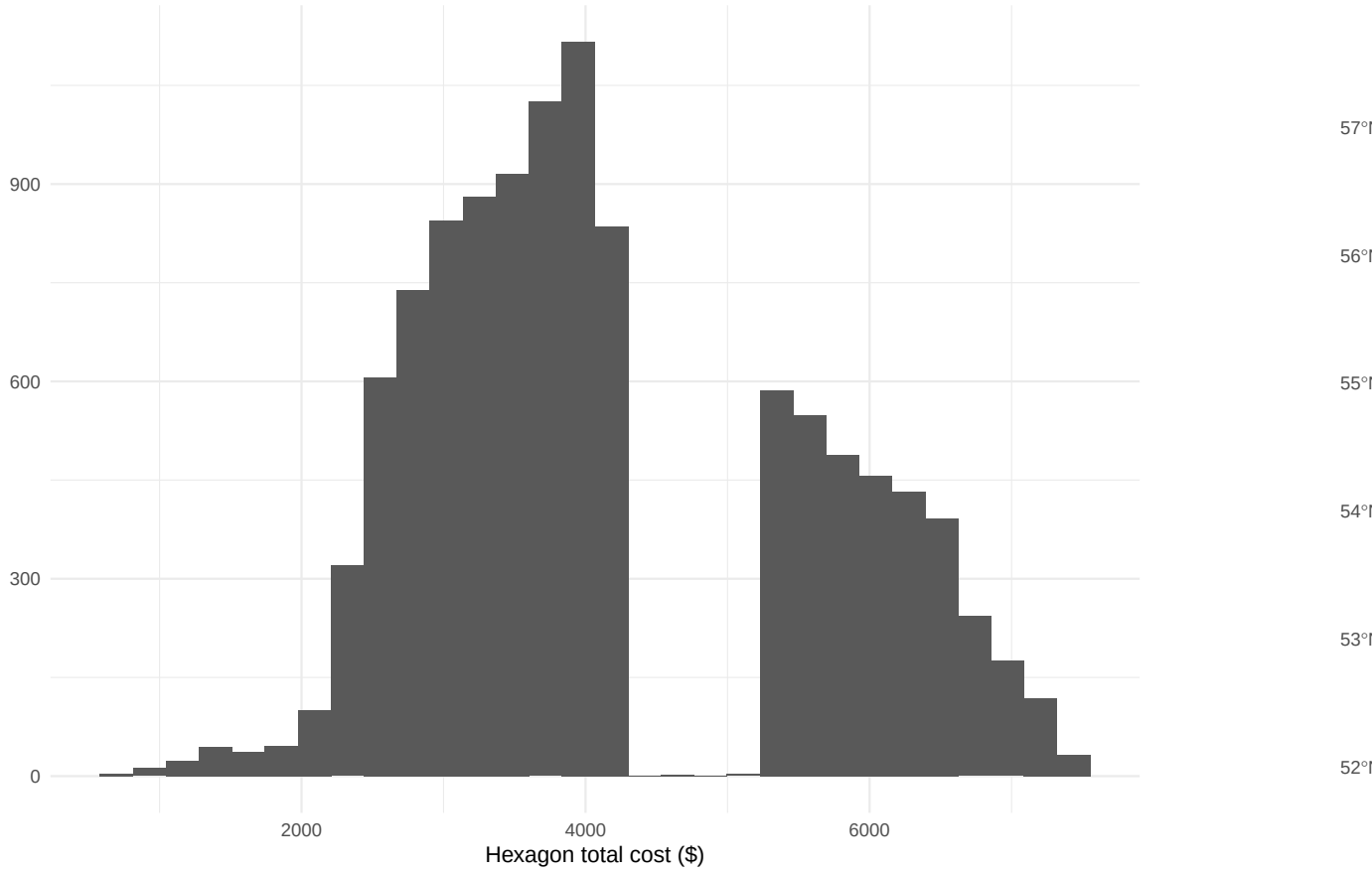
Ecoregion 72



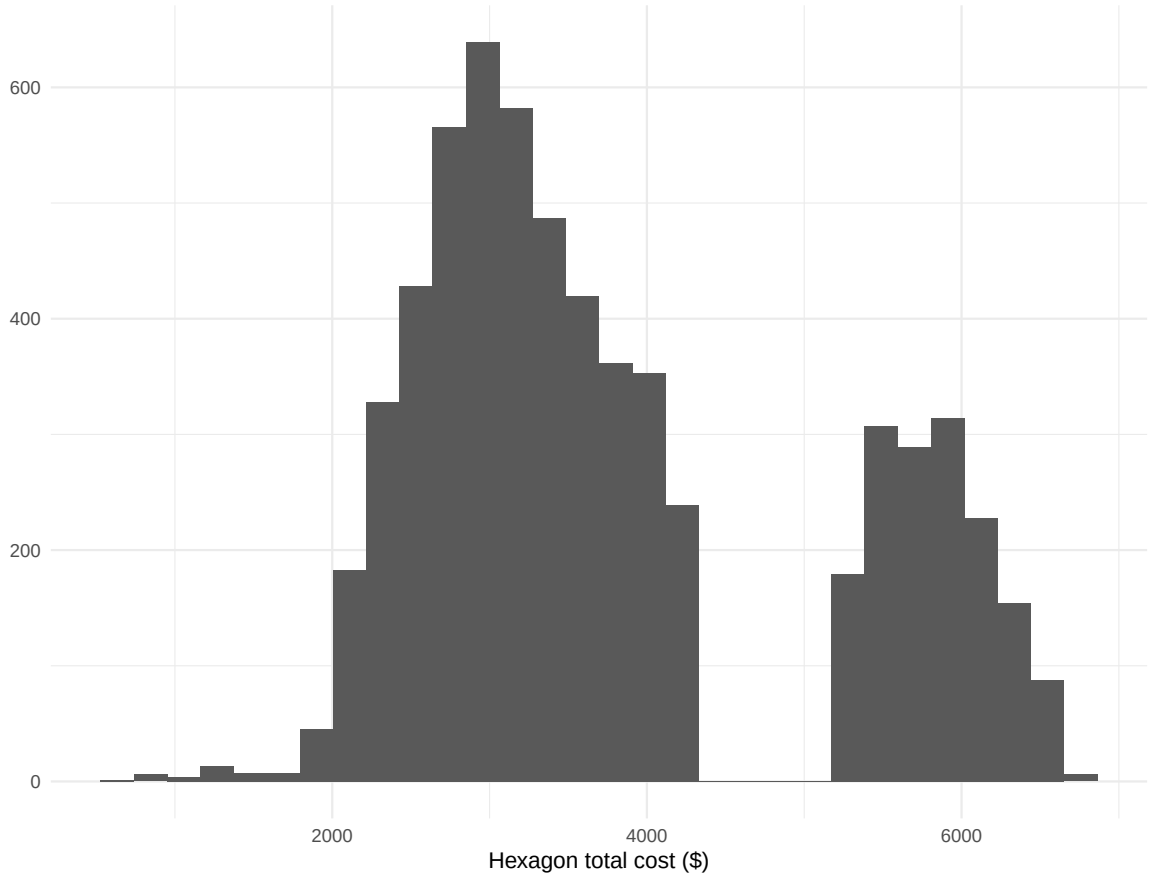
Ecoregion 73



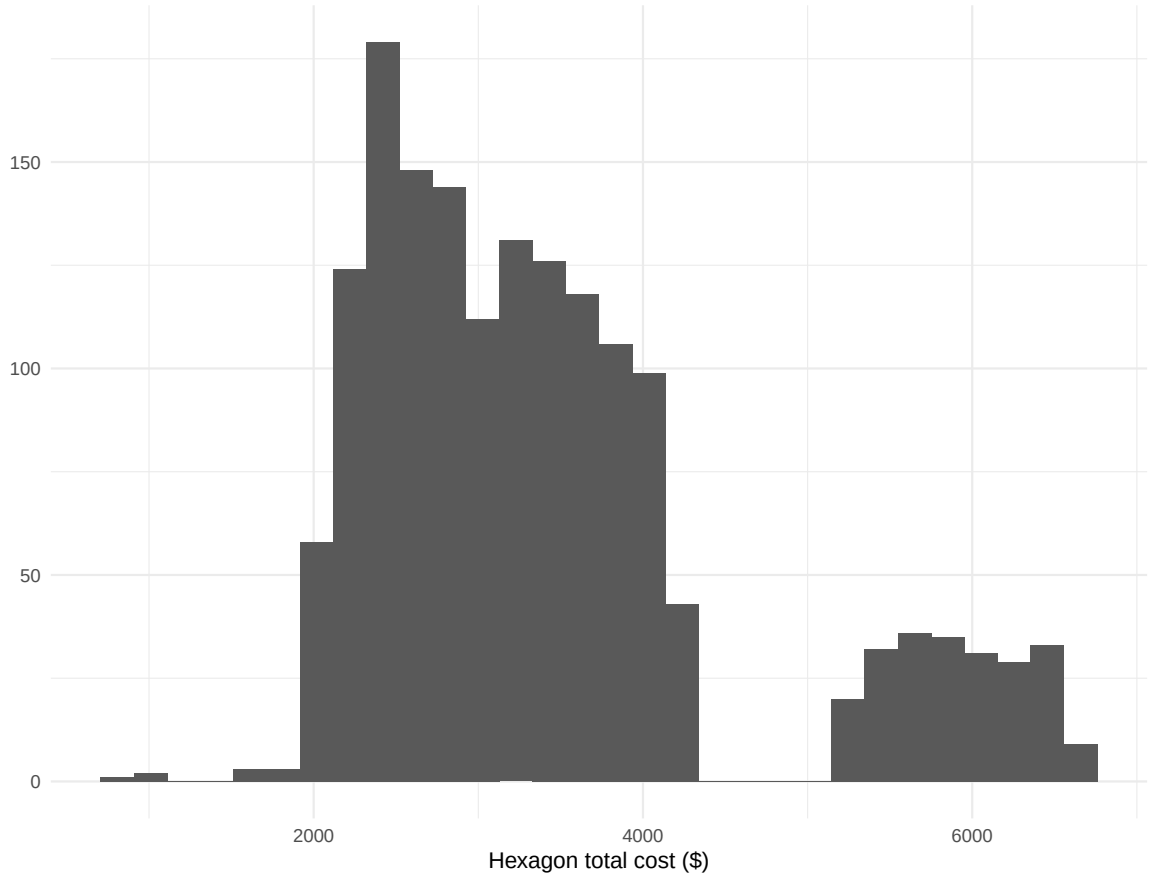
Ecoregion 74



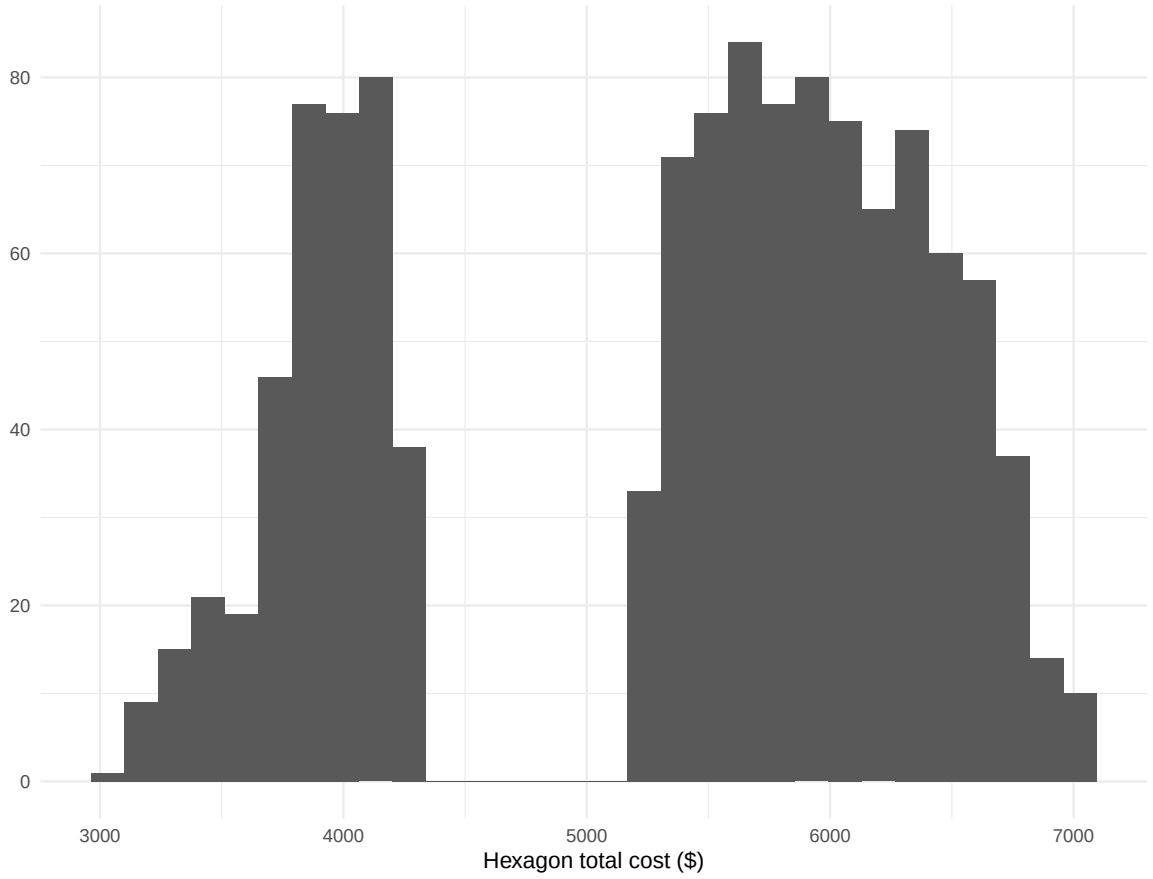
Ecoregion 75



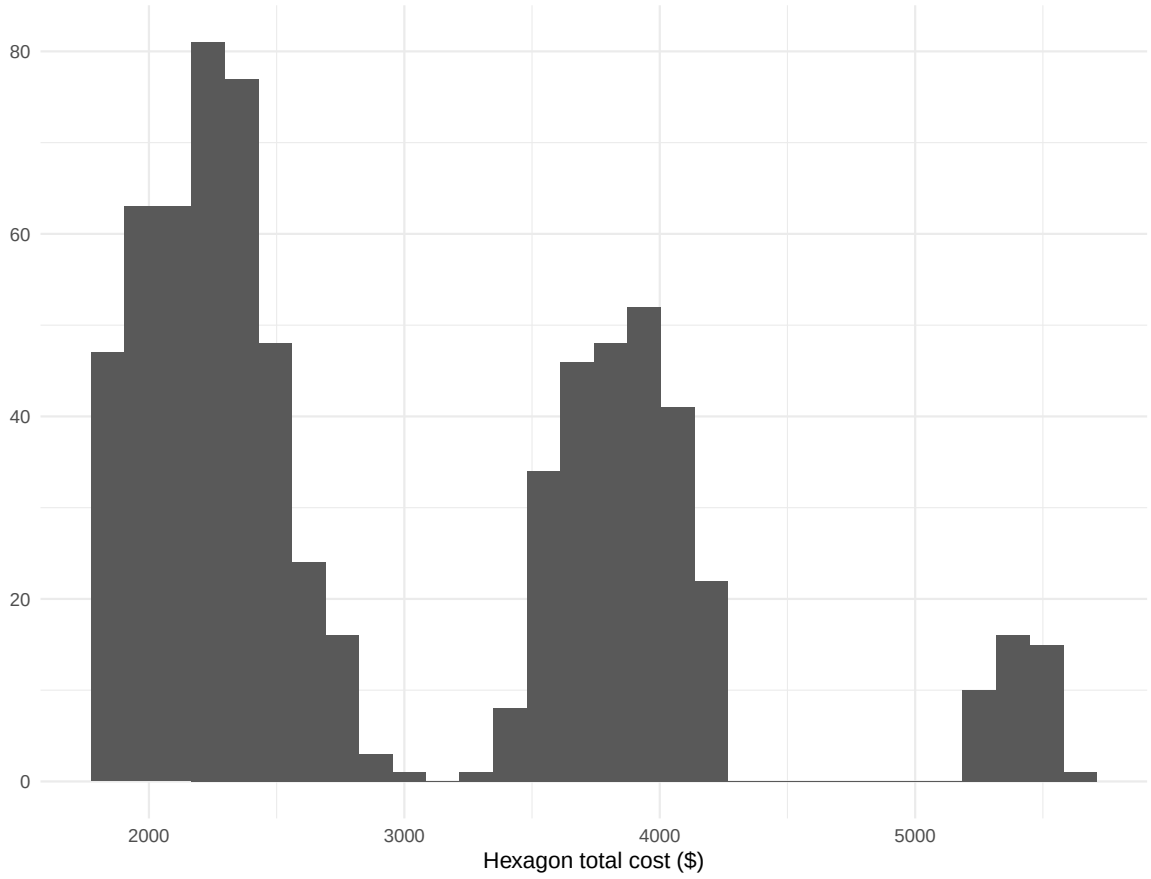
Ecoregion 76



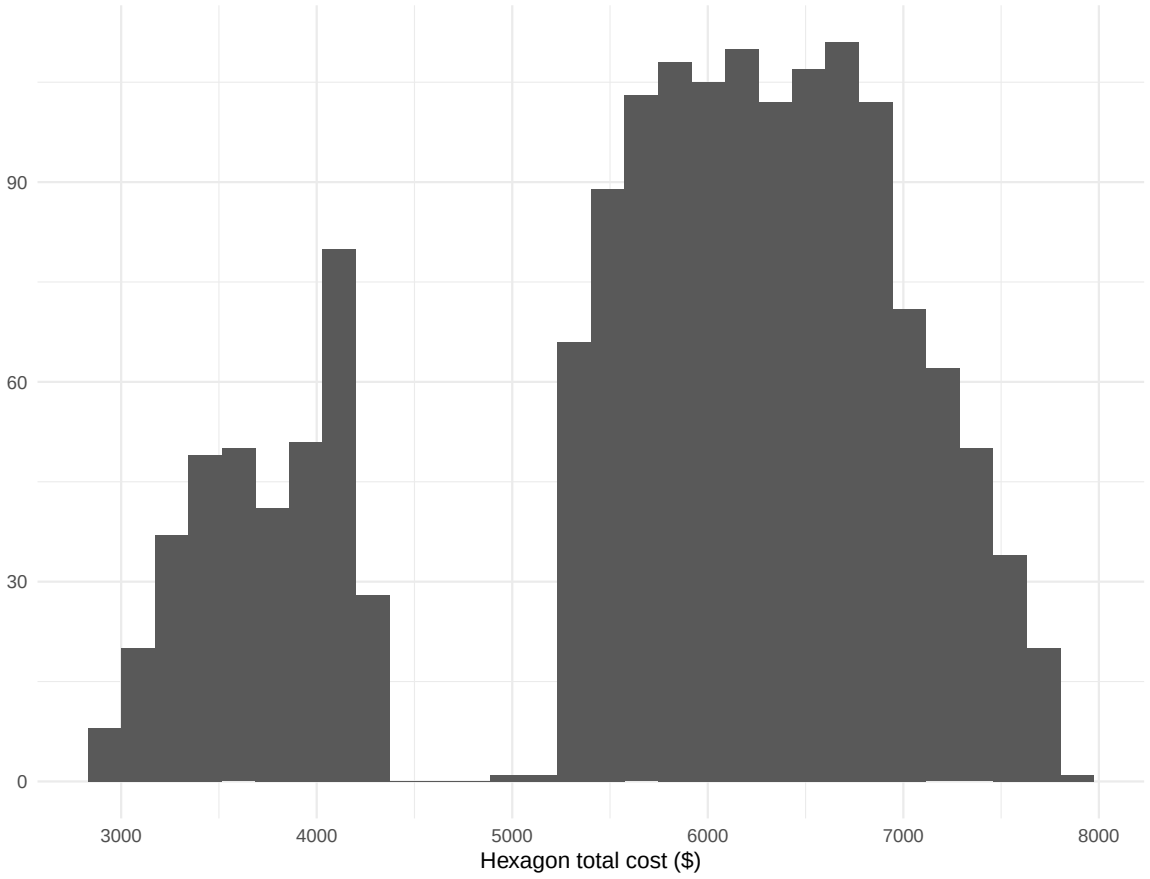
Ecoregion 77



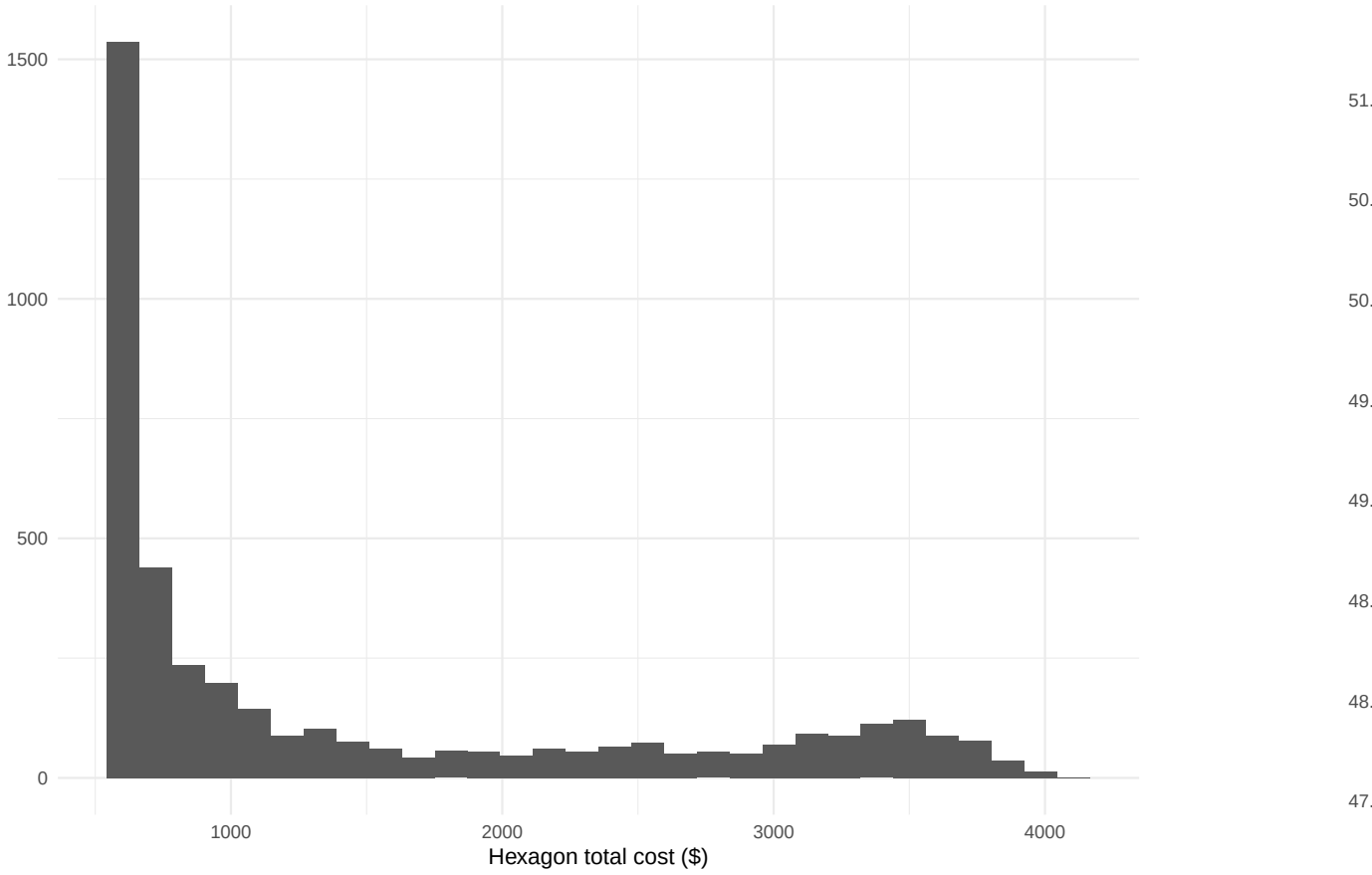
Ecoregion 78



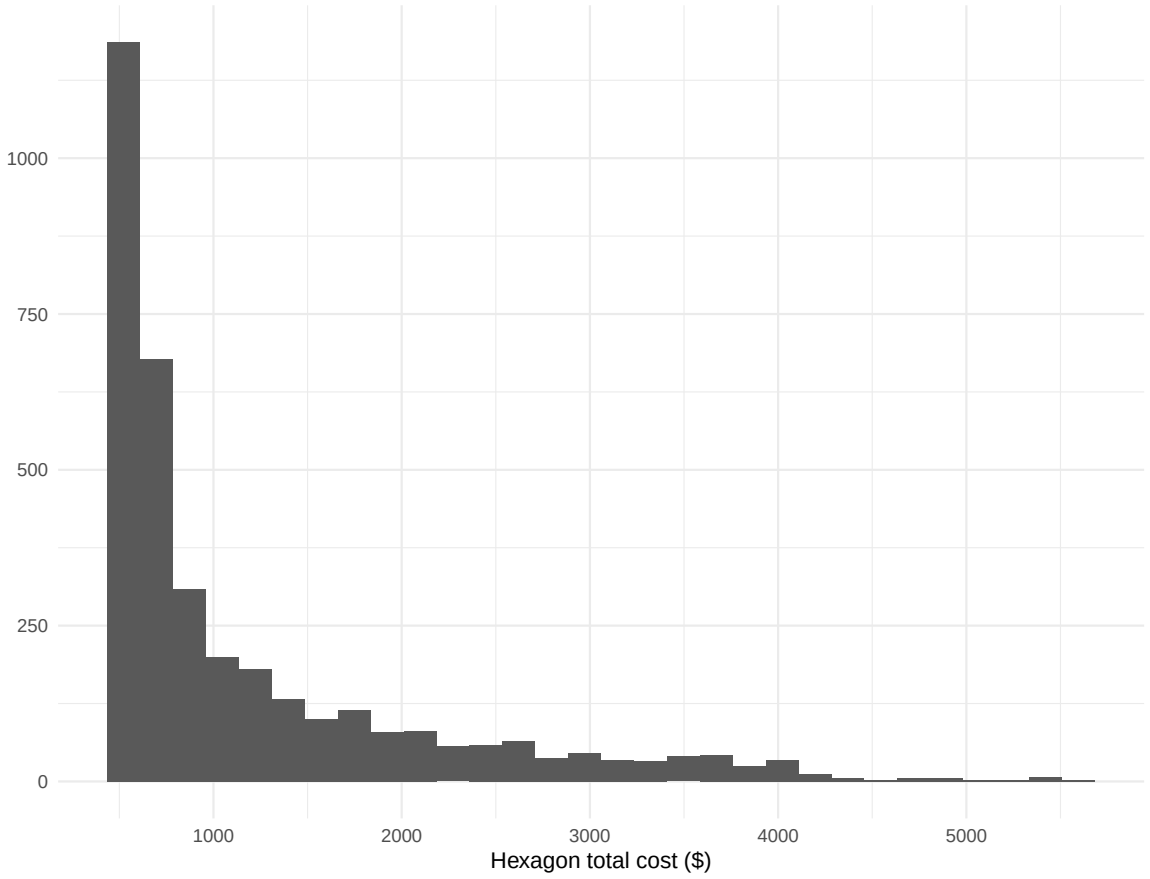
Ecoregion 86



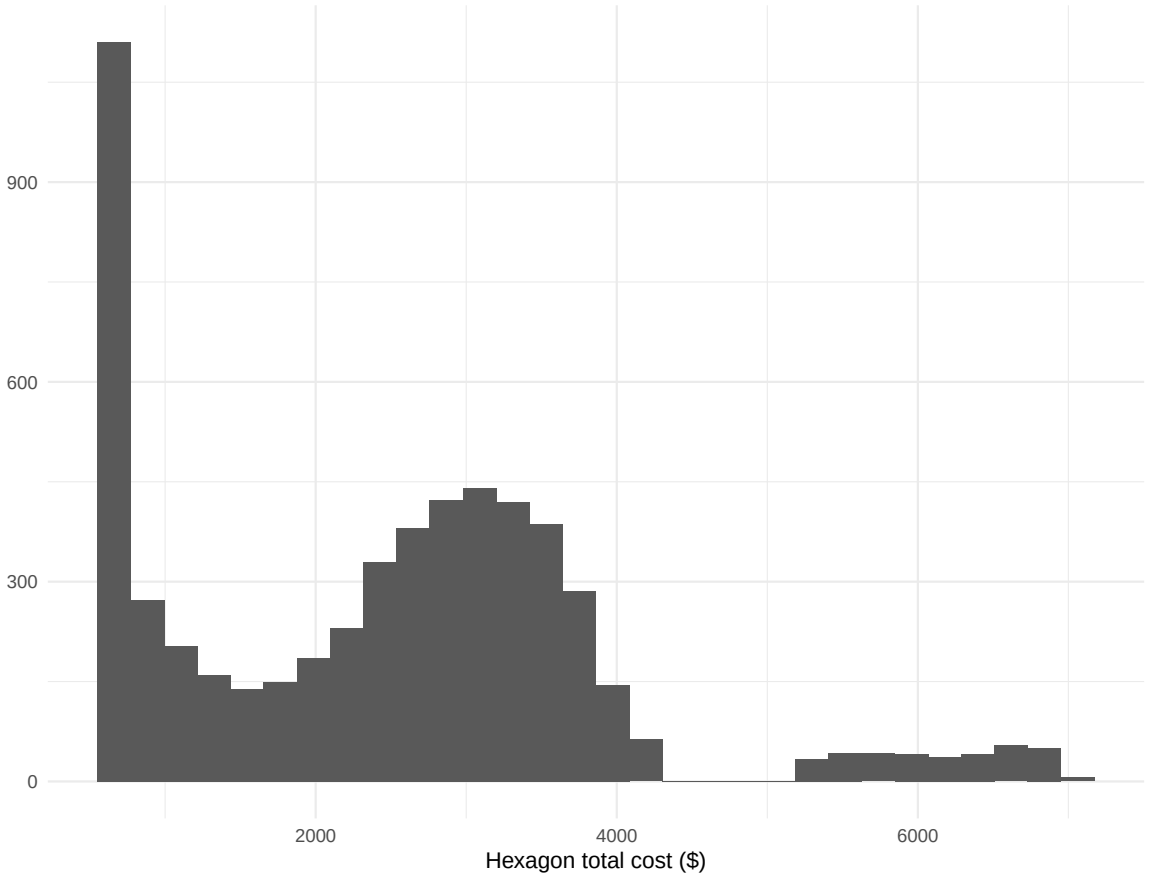
Ecoregion 96



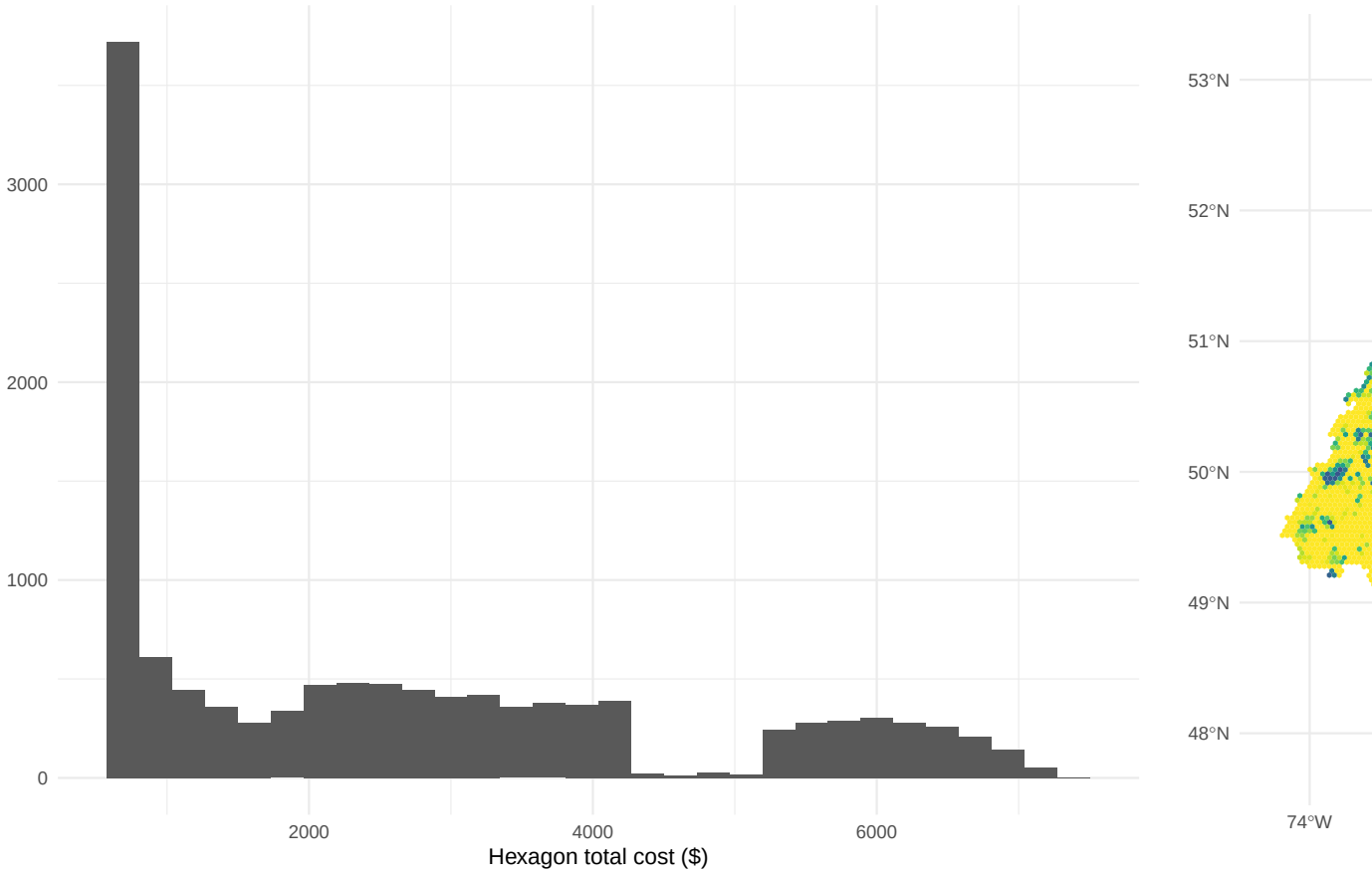
Ecoregion 99



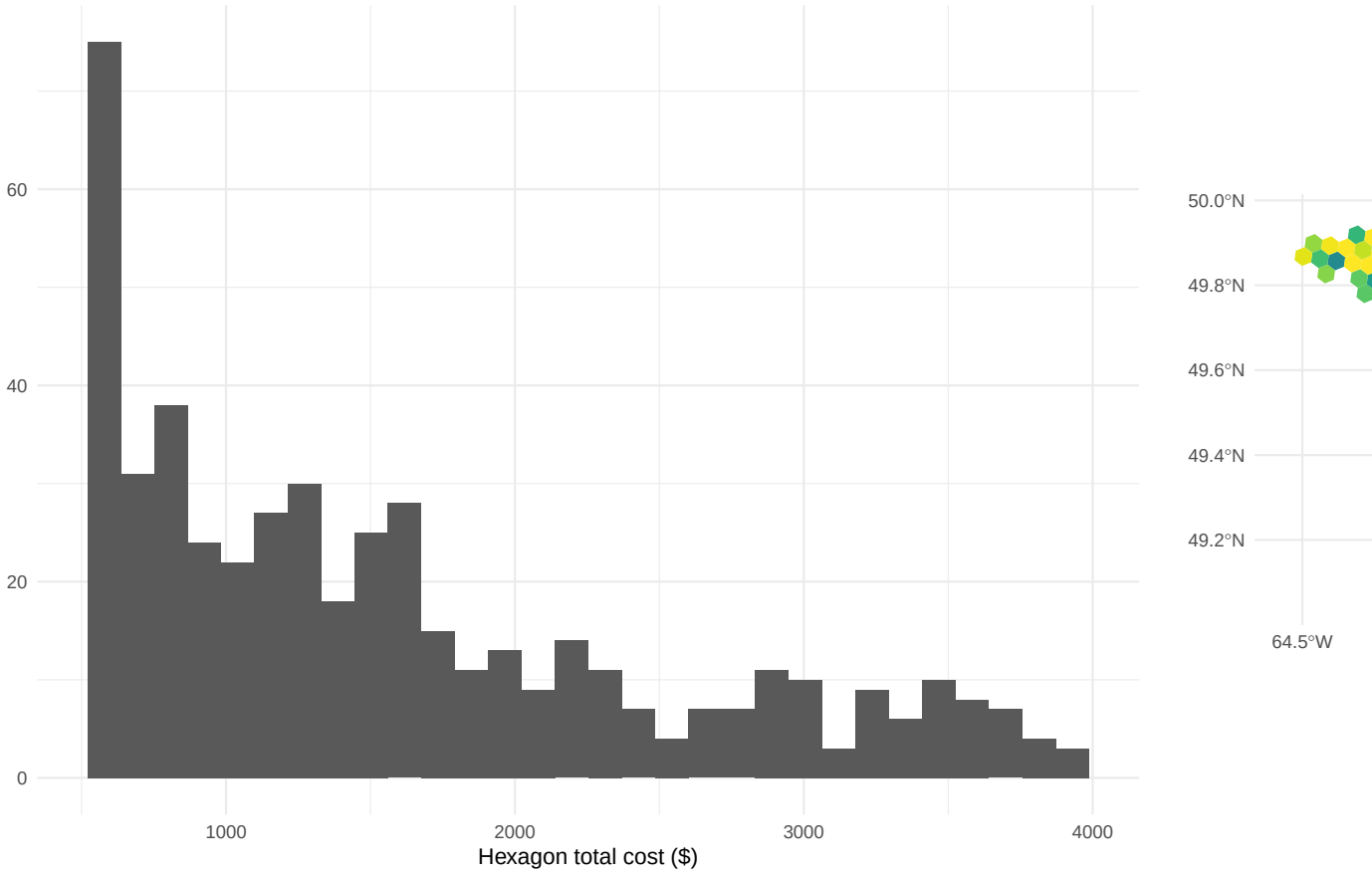
Ecoregion 100



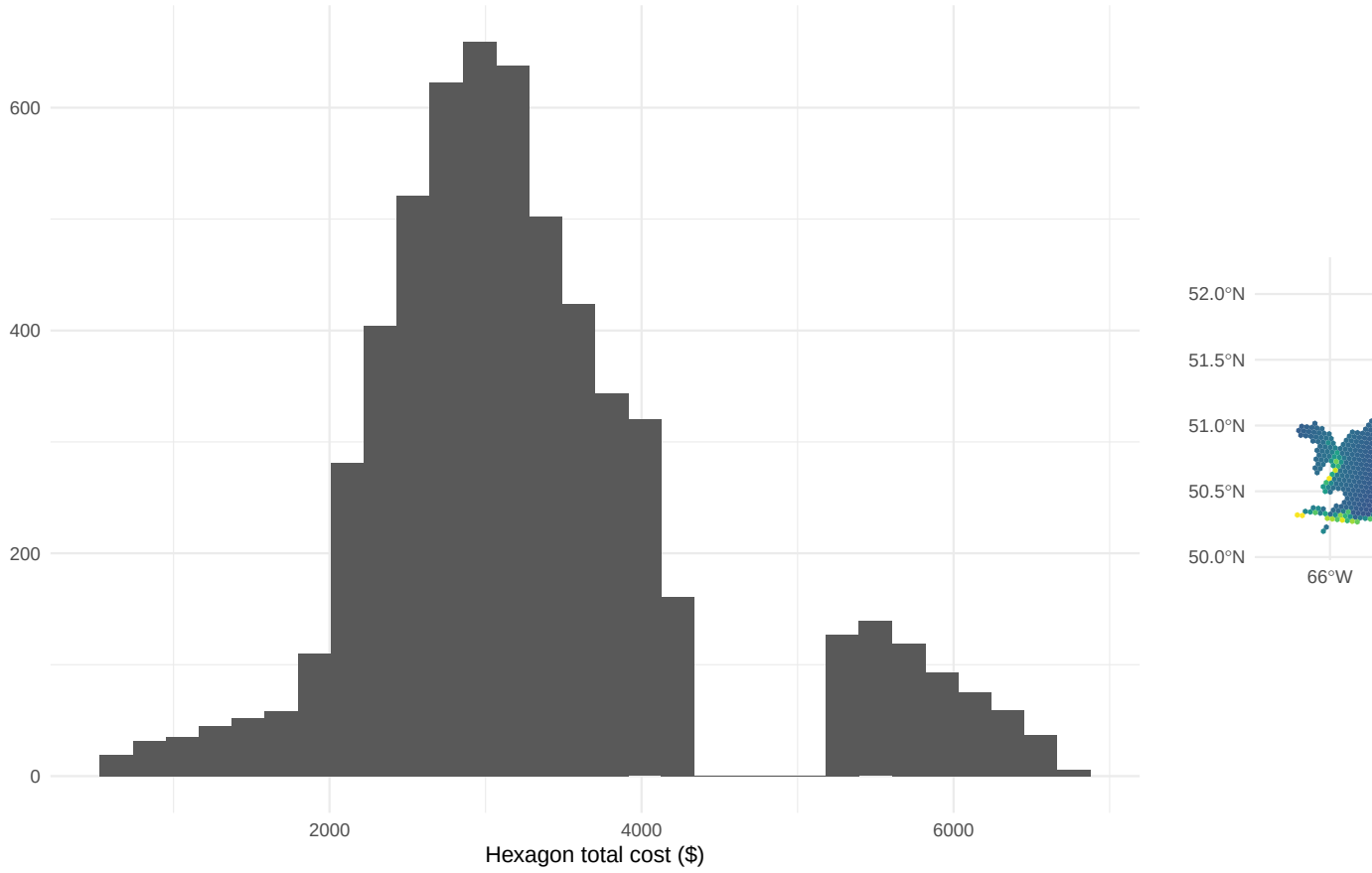
Ecoregion 101



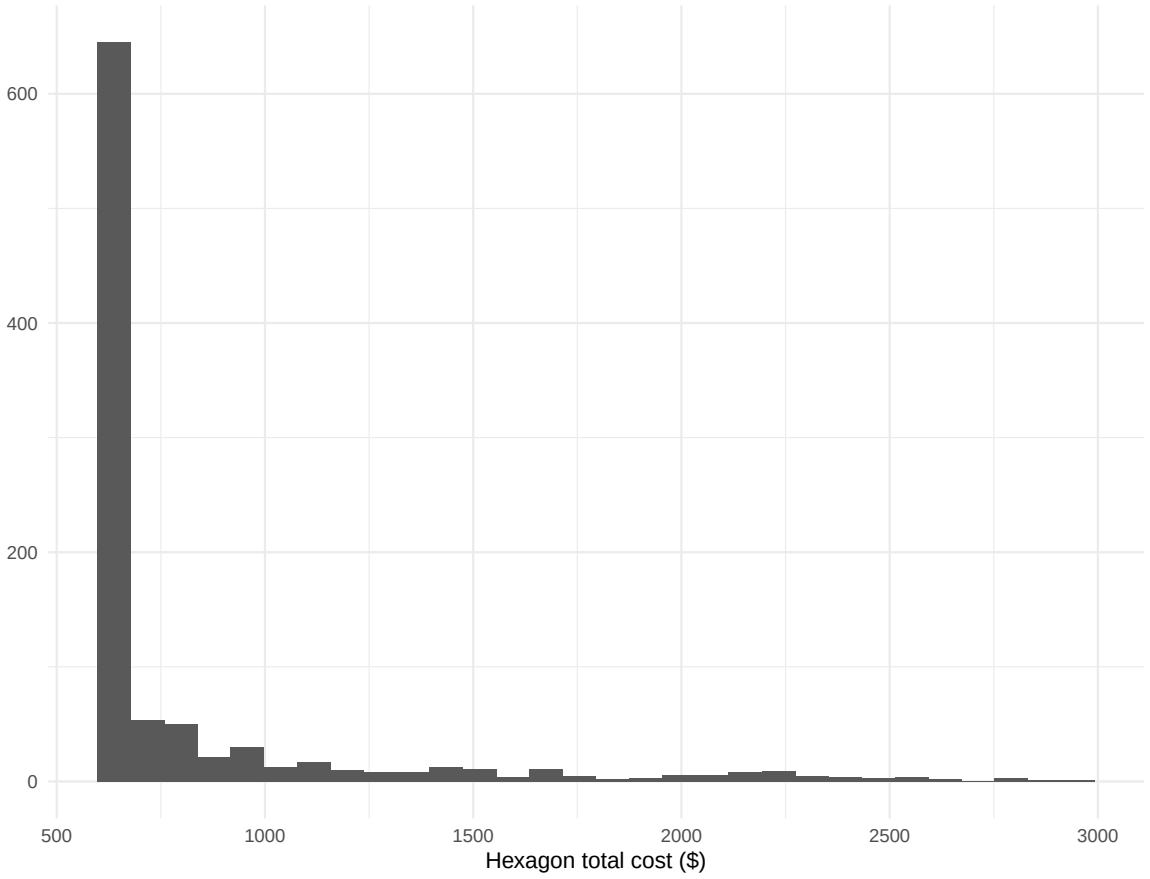
Ecoregion 102



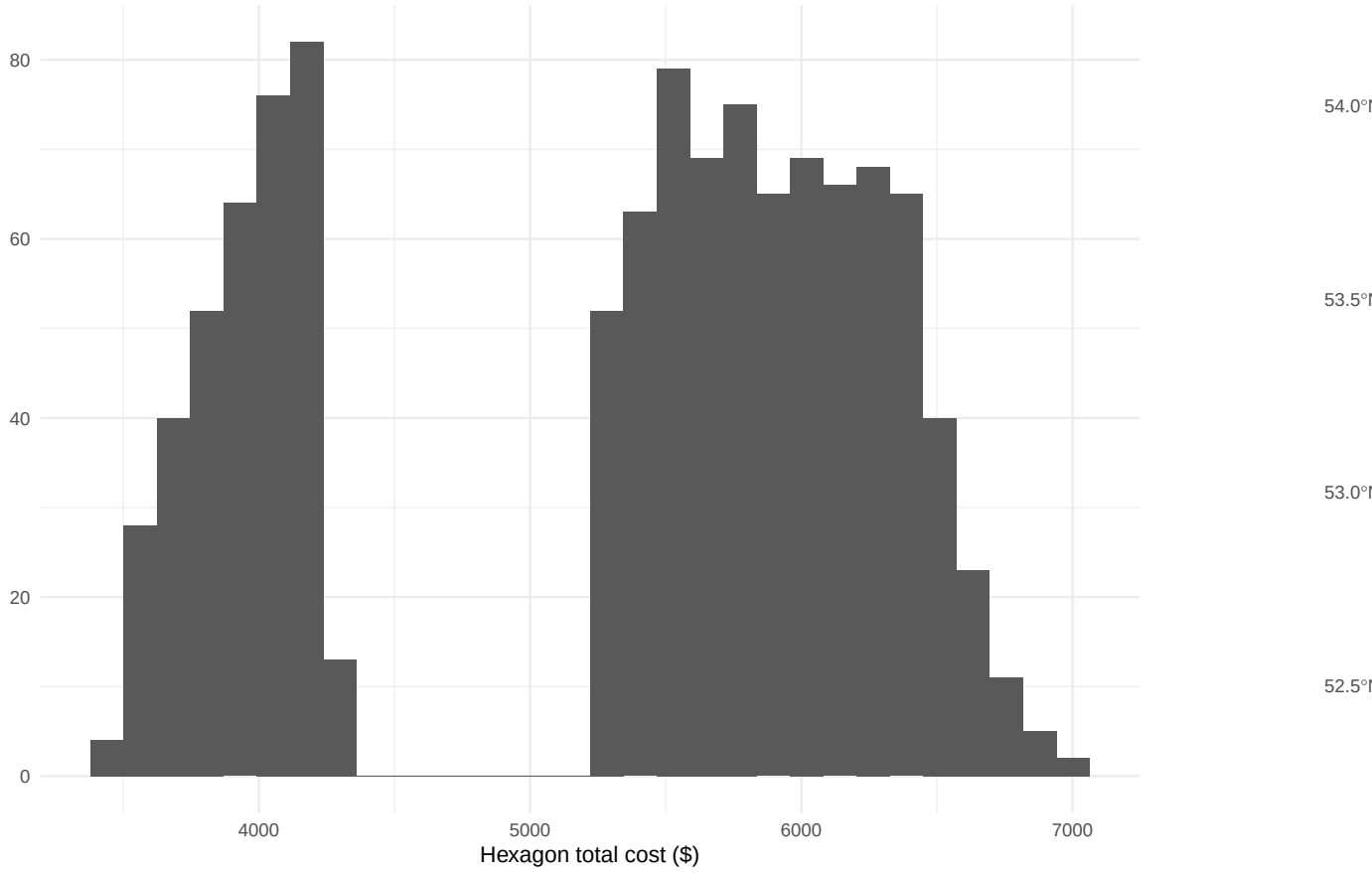
Ecoregion 103



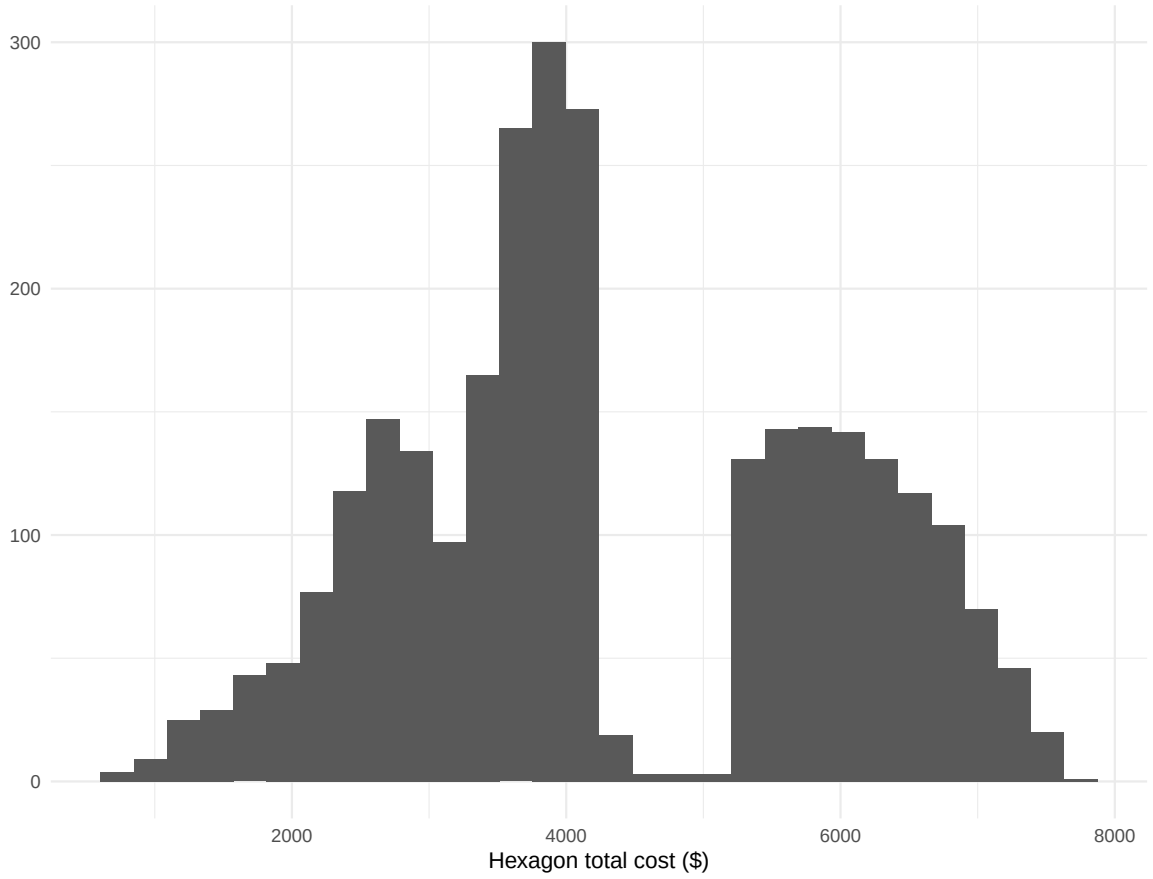
Ecoregion 117



Ecoregion 216



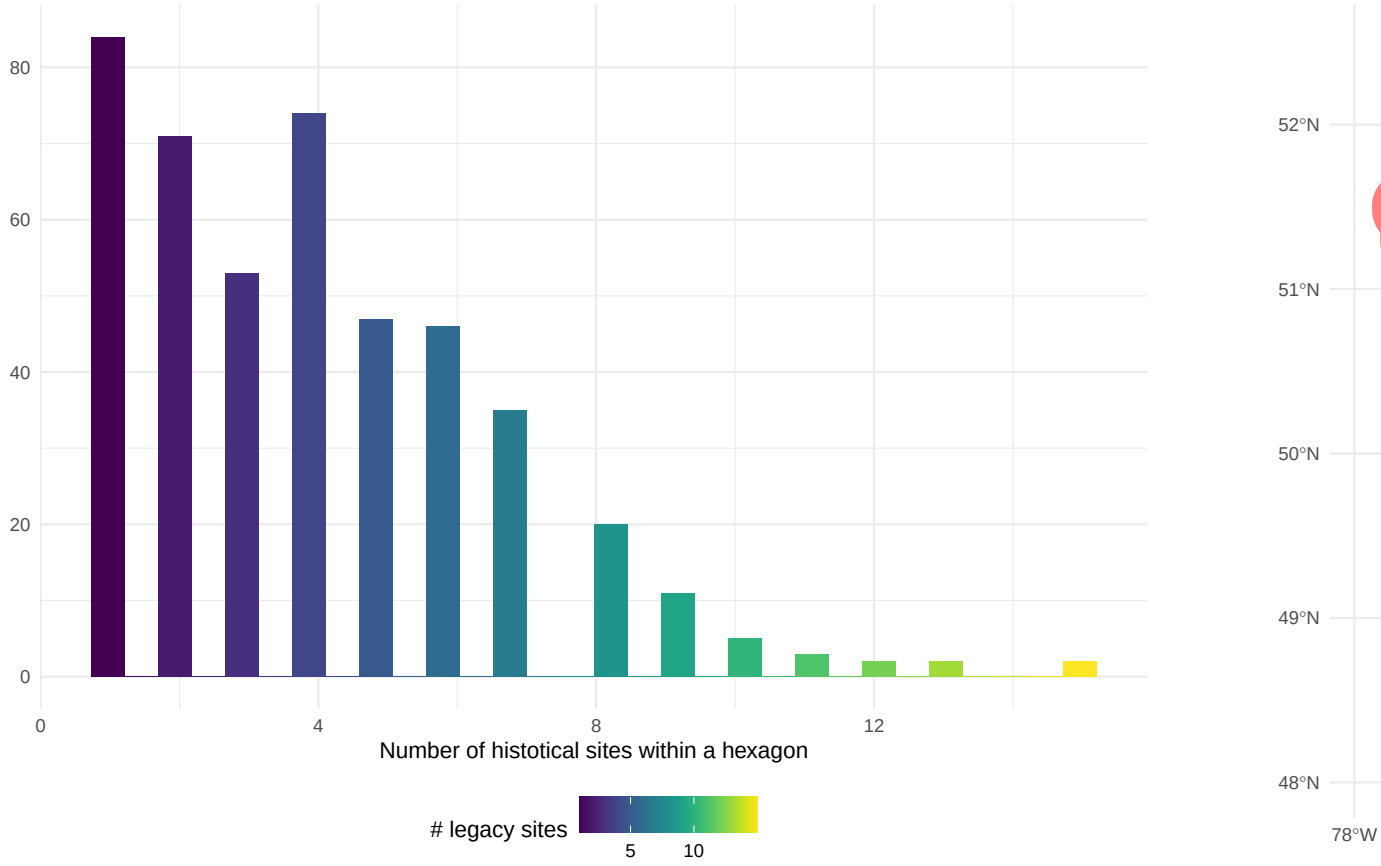
Ecoregion 217


300
200
100
0
2000
4000
6000
8000
Hexagon total cost ($)
53.0°N
52.5°N
52.0°N
51.5°N
51.0°N
50.5°N
50.0°N

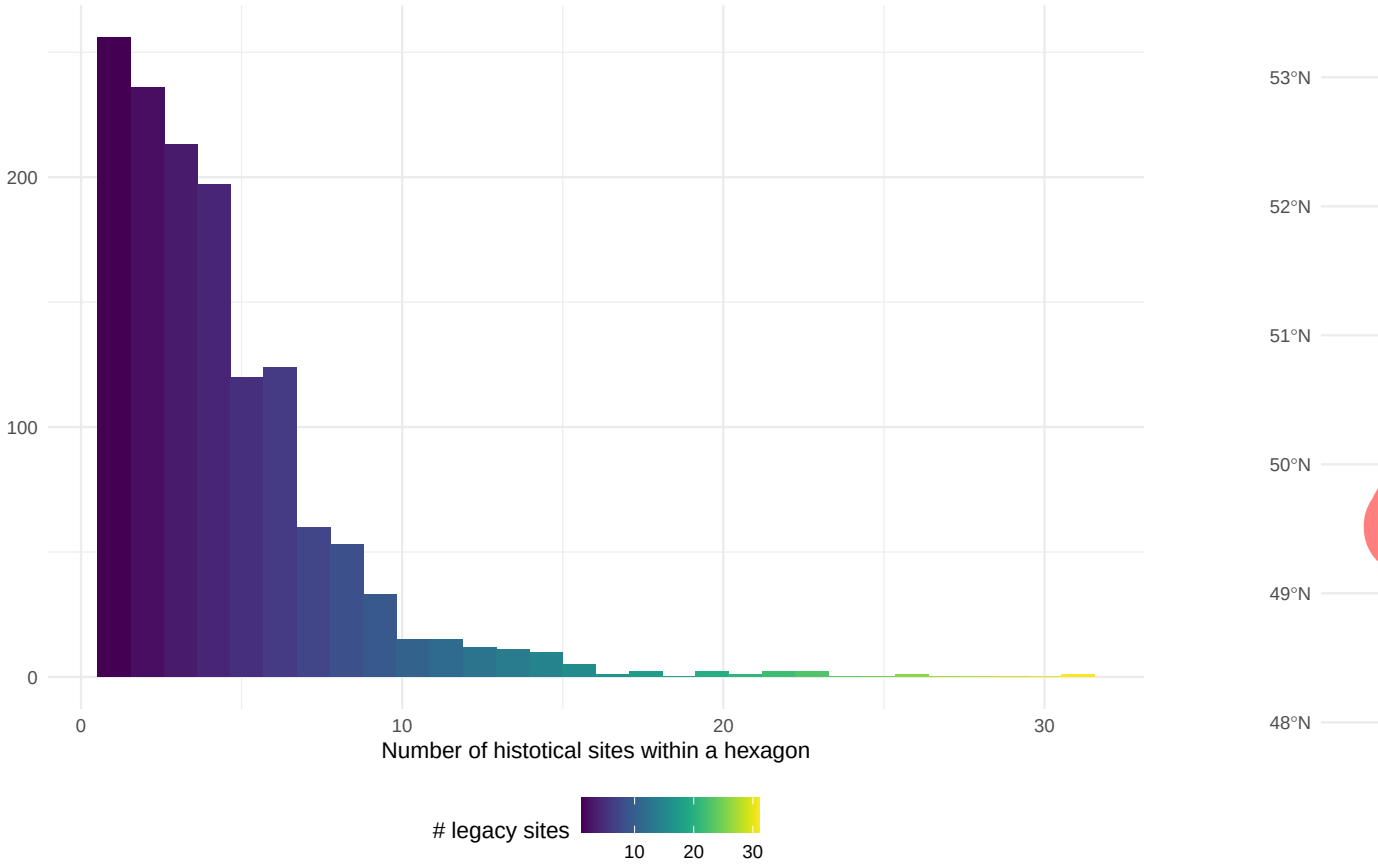
11 Ecoregion summary: historical sites

Historical sites are survey points present in the study area, and a hexagon can contain multiple historical sites. The first figure illustrates the distribution of the number of historical sites within a hexagon throughout the ecoregion. The second image displays the spatial distribution of historical sites in the ecoregion, with each hexagon's color indicating the number of legacy sites present. Around each historical sites, two buffers describe the effect of the historical hexagons on (i) the inclusion probability of neighboring hexagons (dark red) and (ii) the sample size (light red). While the buffer for adjusting the inclusion probability of neighboring hexagons is fixed at 10 kilometers, the buffer for adjusting the sample size varies depending on the ecoregion. For further information, please refer to Chapter 5.

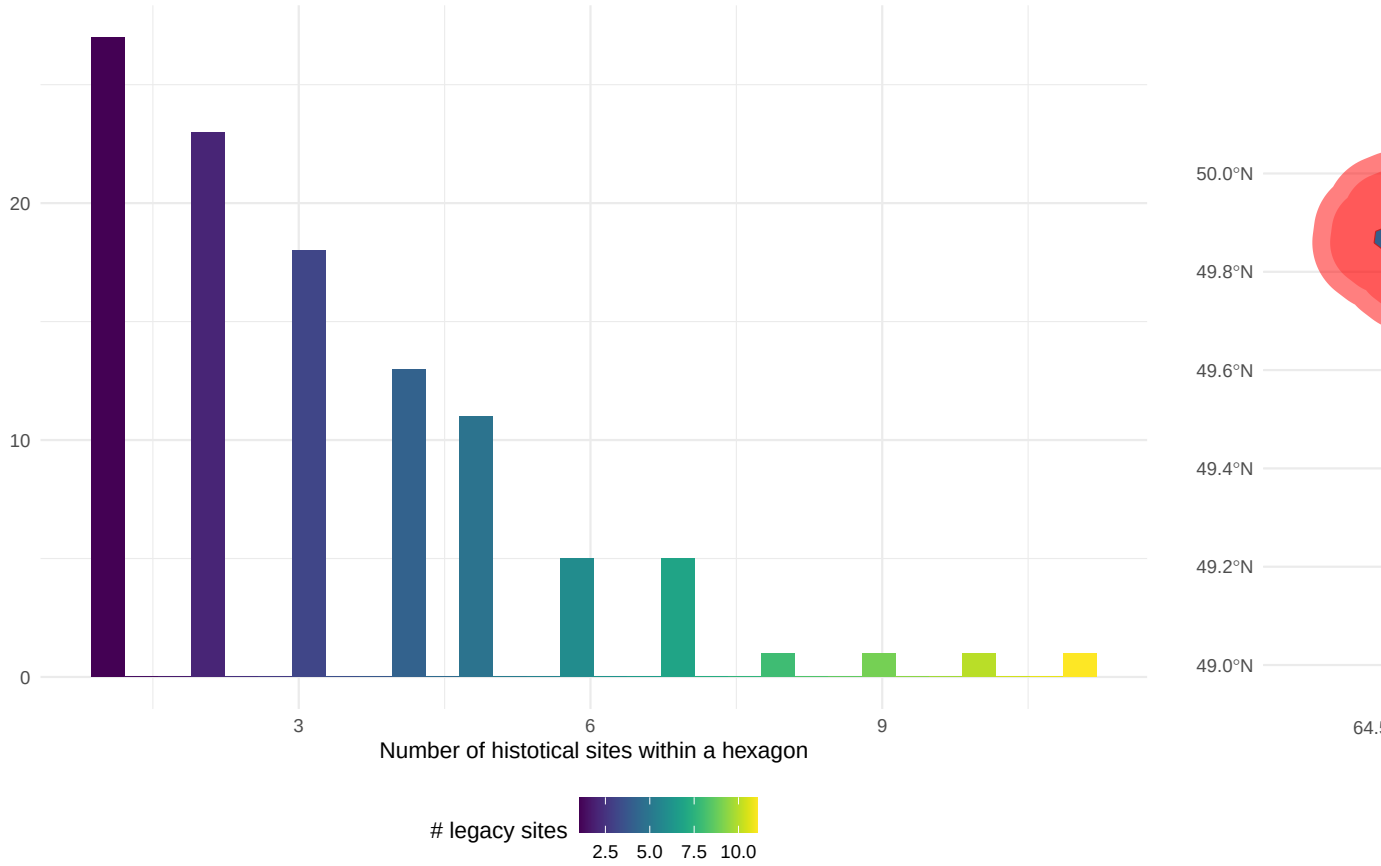
11.1 Ecoregion 100



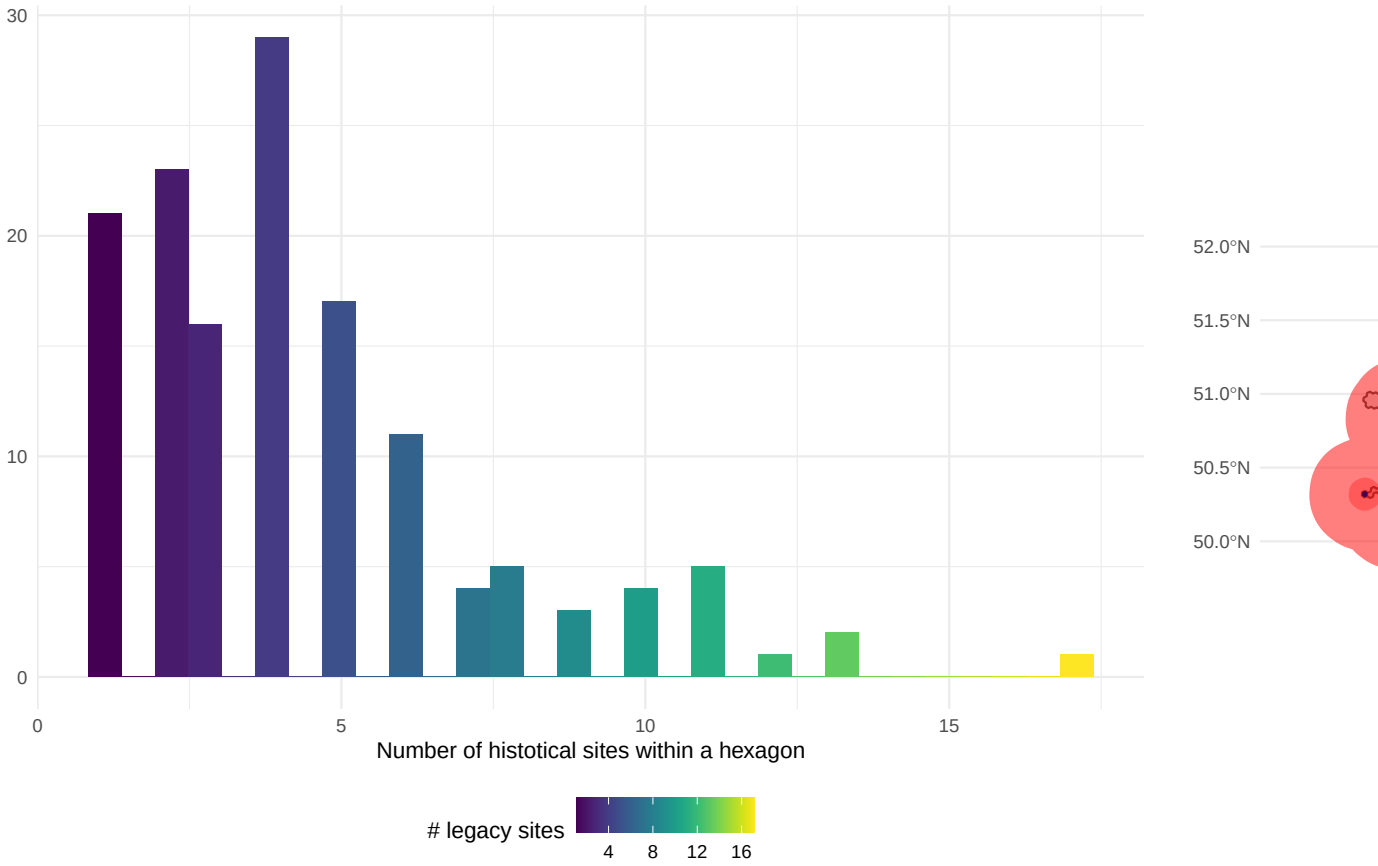
Ecoregion 101



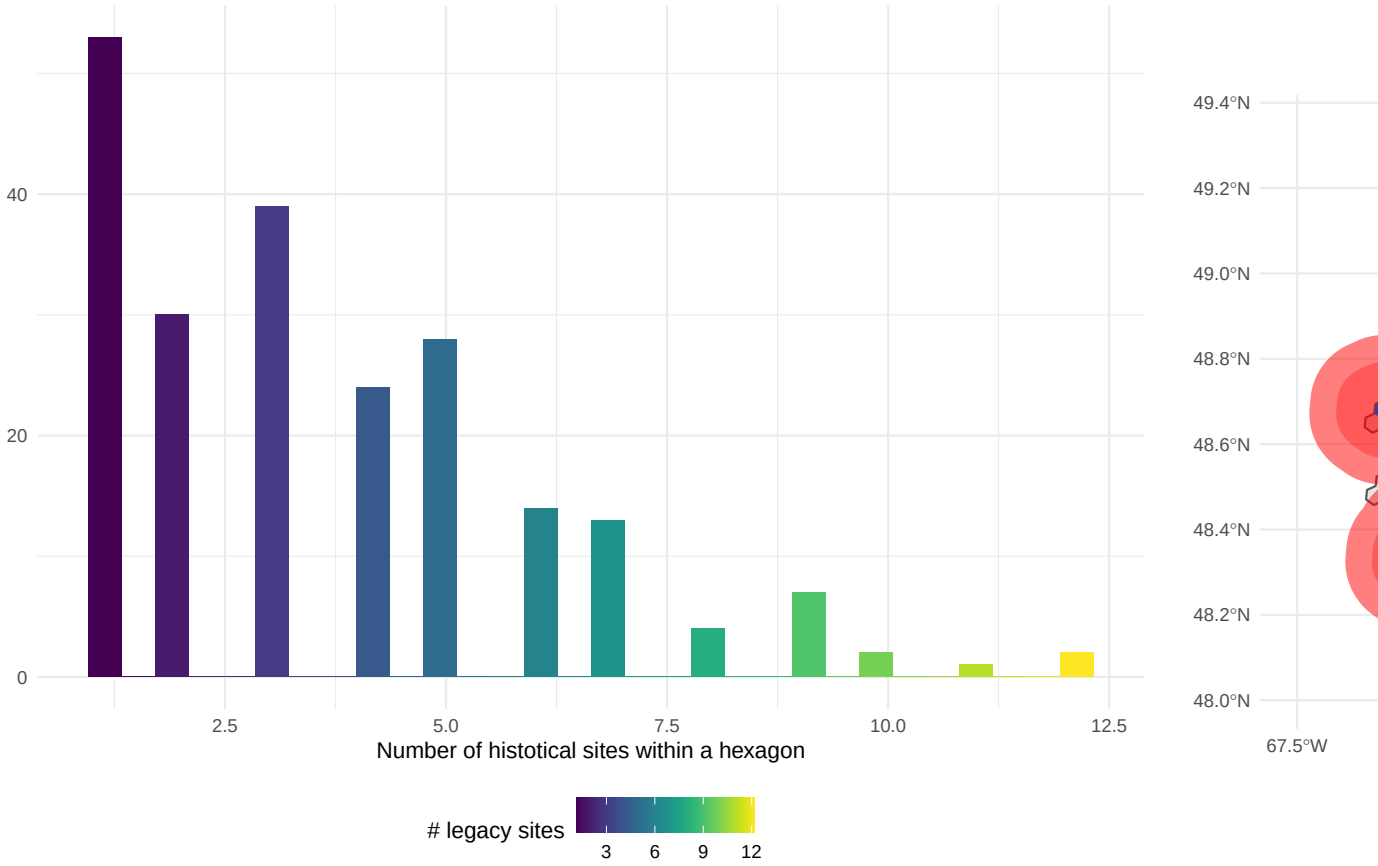
Ecoregion 102



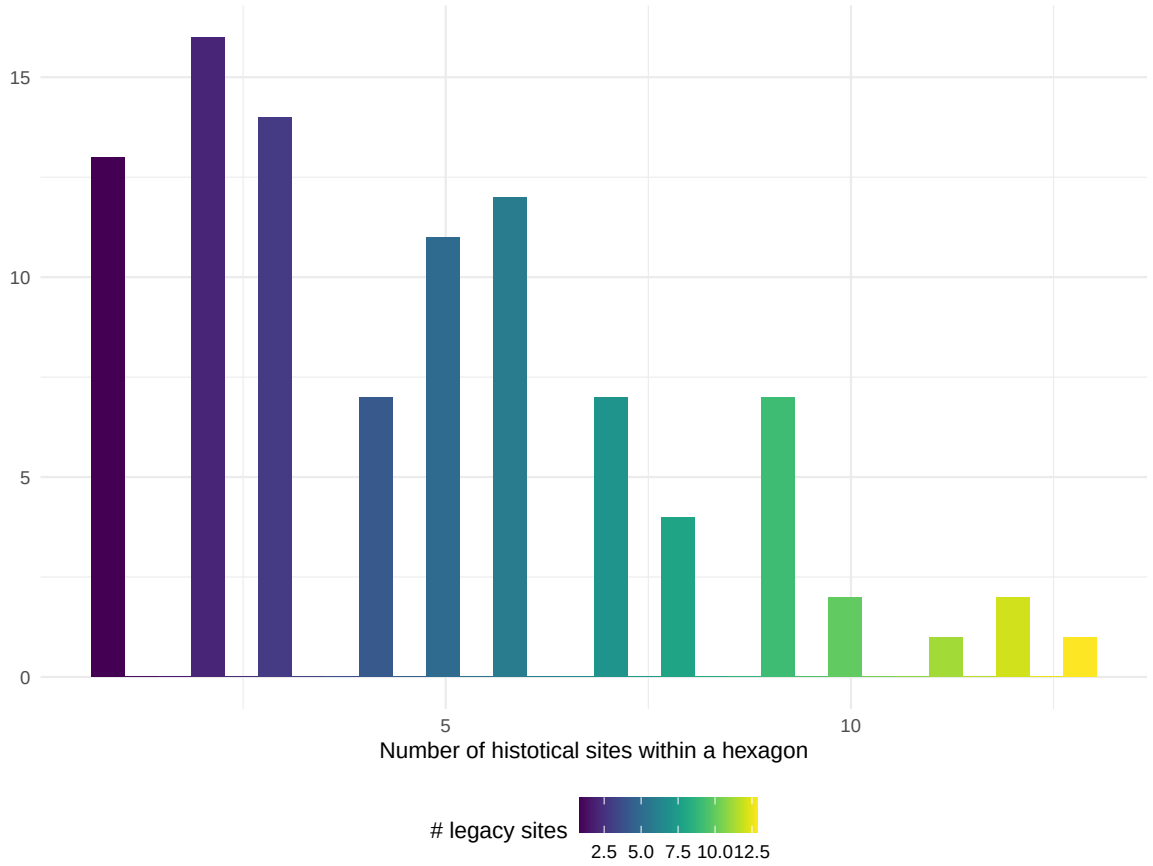
Ecoregion 103



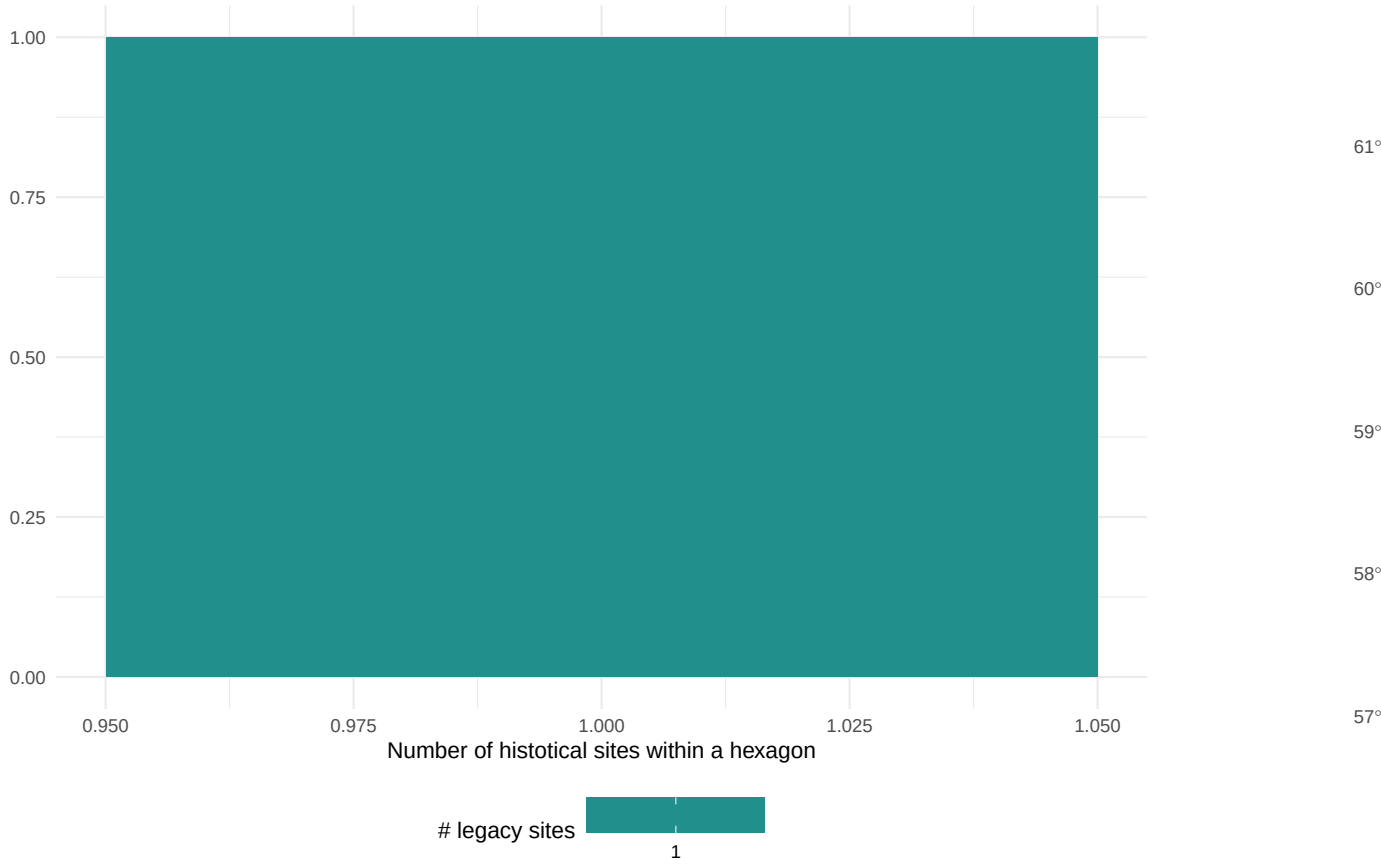
Ecoregion 117



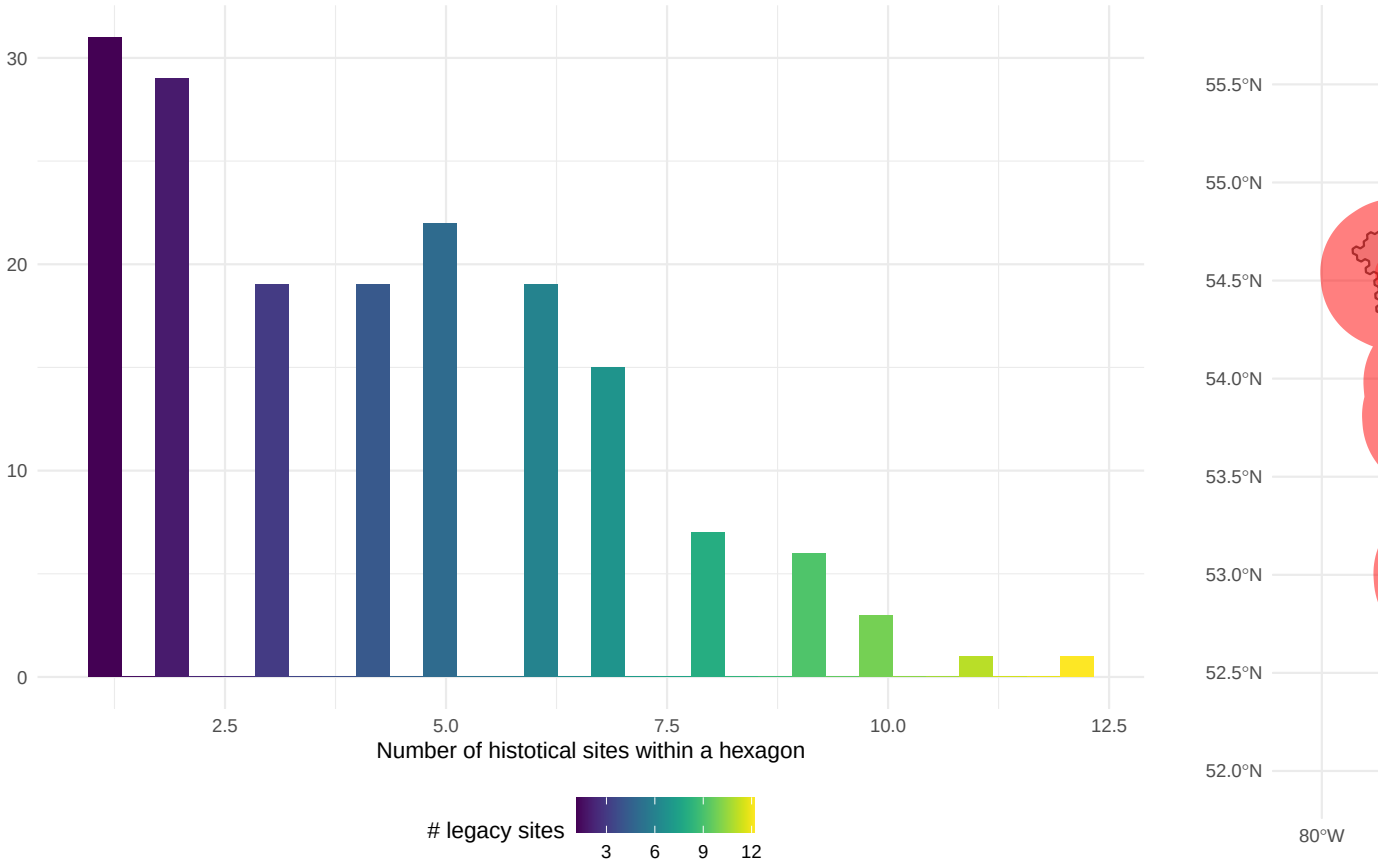
Ecoregion 217



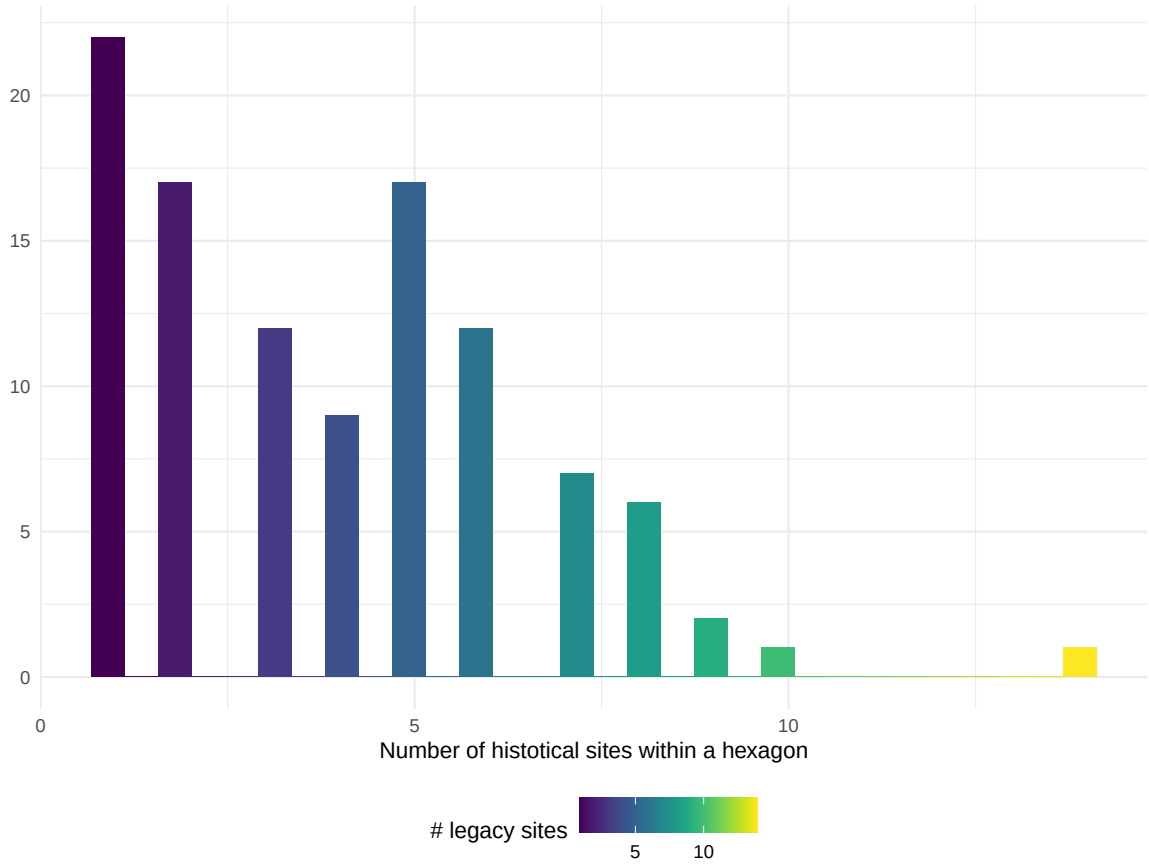
Ecoregion 47



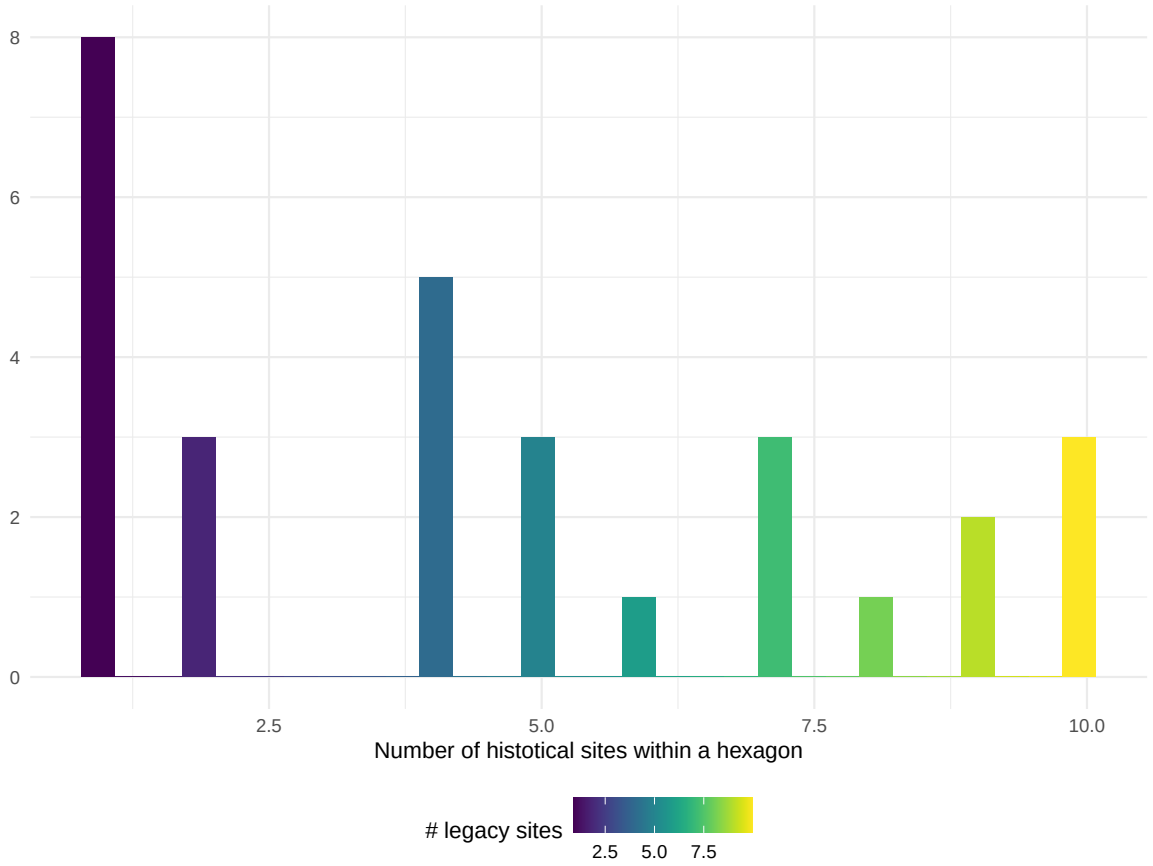
Ecoregion 72



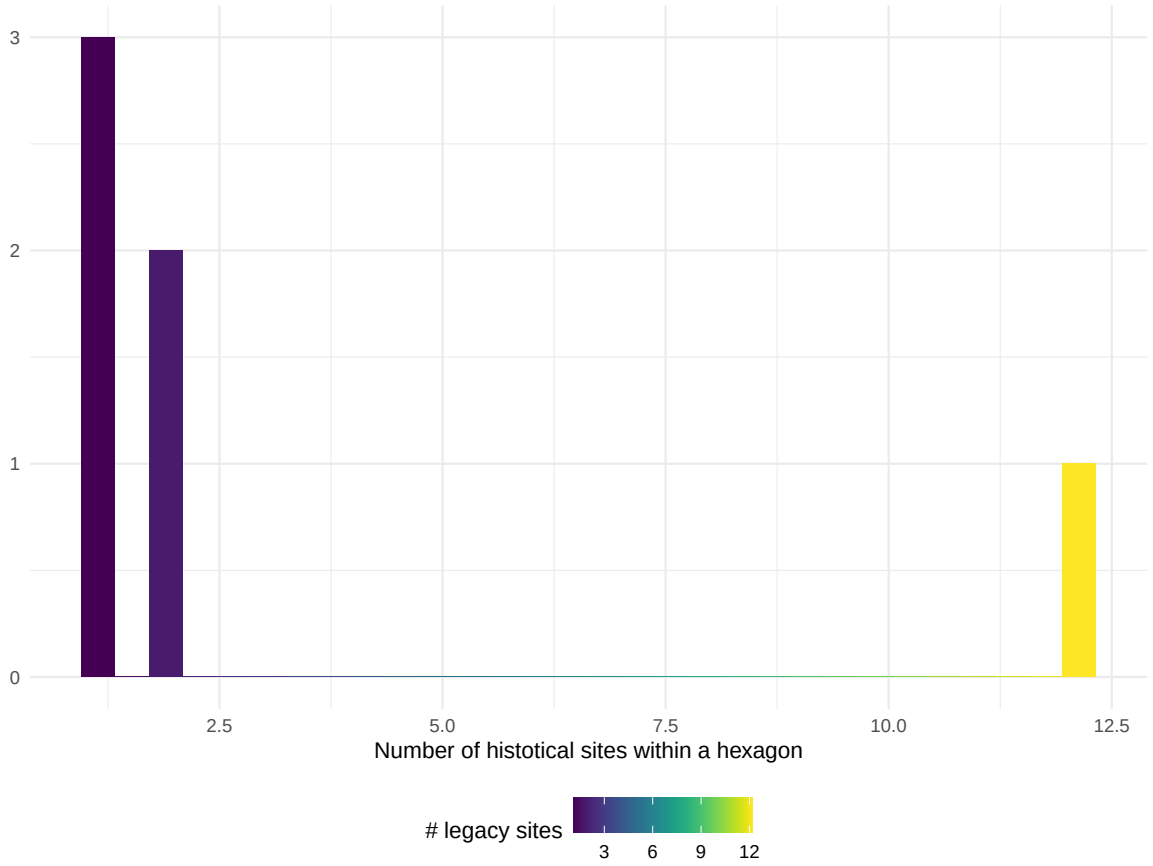
Ecoregion 74



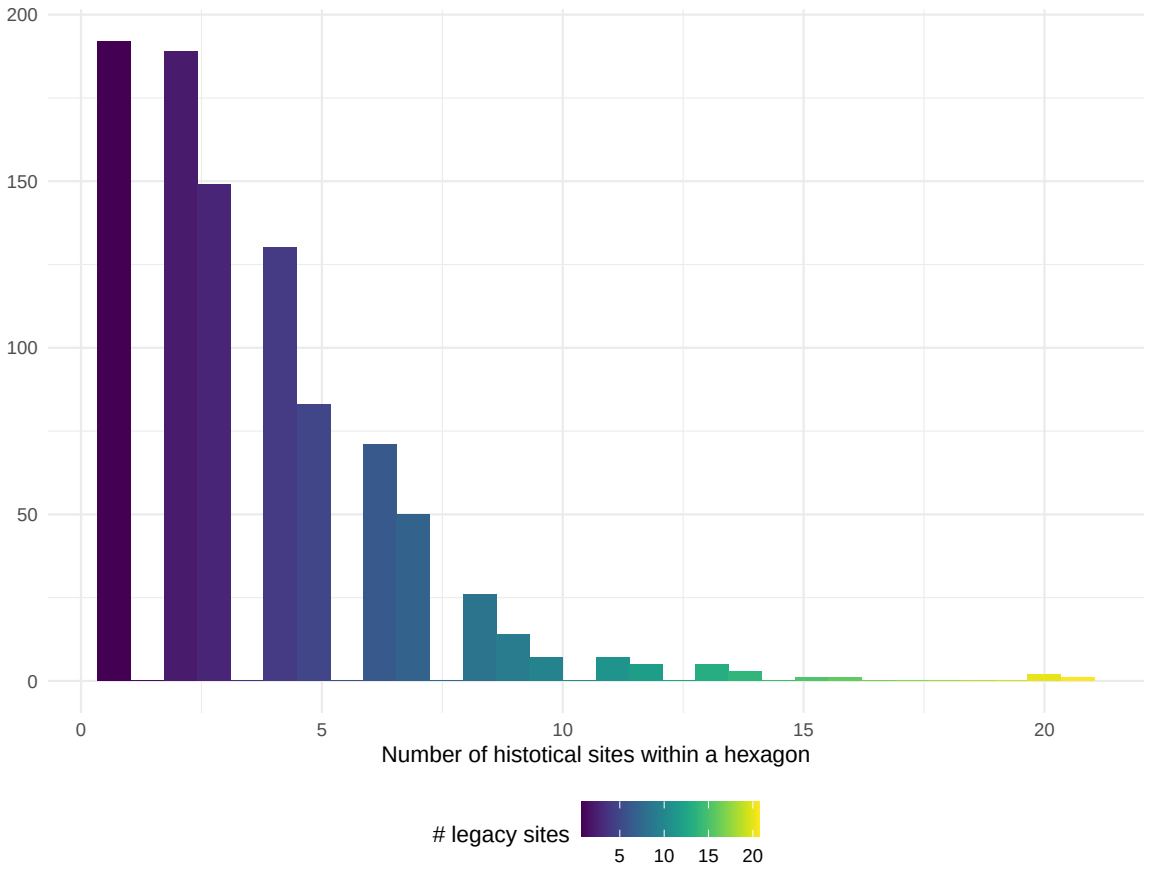
Ecoregion 75



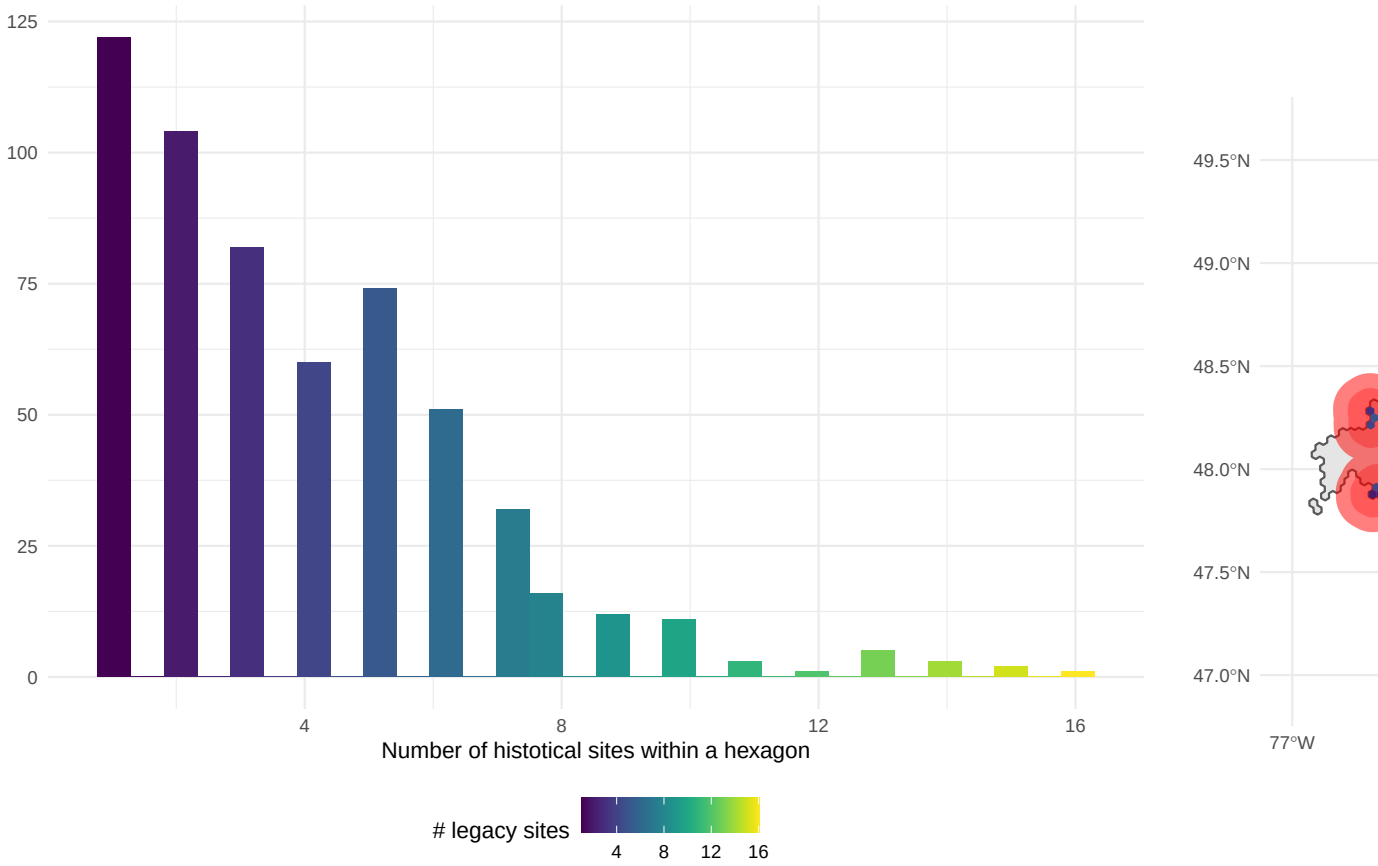
Ecoregion 76



Ecoregion 96



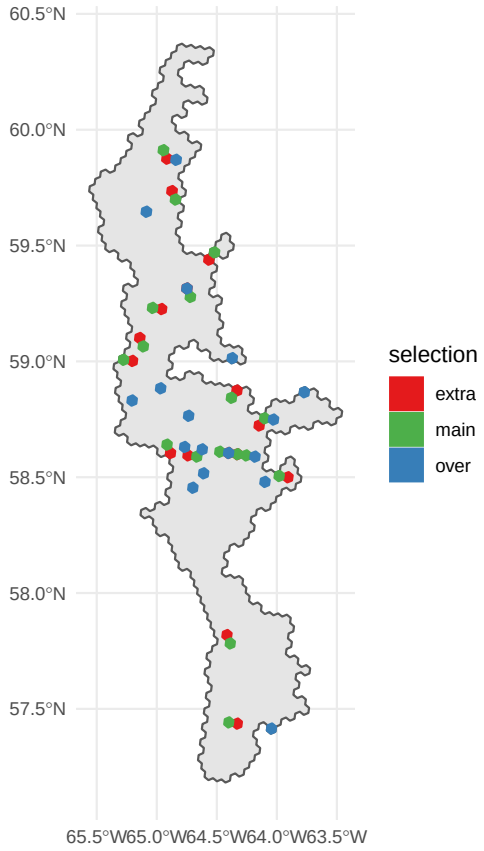
Ecoregion 99


125
100
75
50
25
0
4
8
12
16
Number of histotical sites within a hexagon
legacy sites
4
8
12
16
49.5°N
49.0°N
48.5°N
48.0°N
47.5°N
47.0°N
77°W

12 Ecoregion summary: selected hexagons

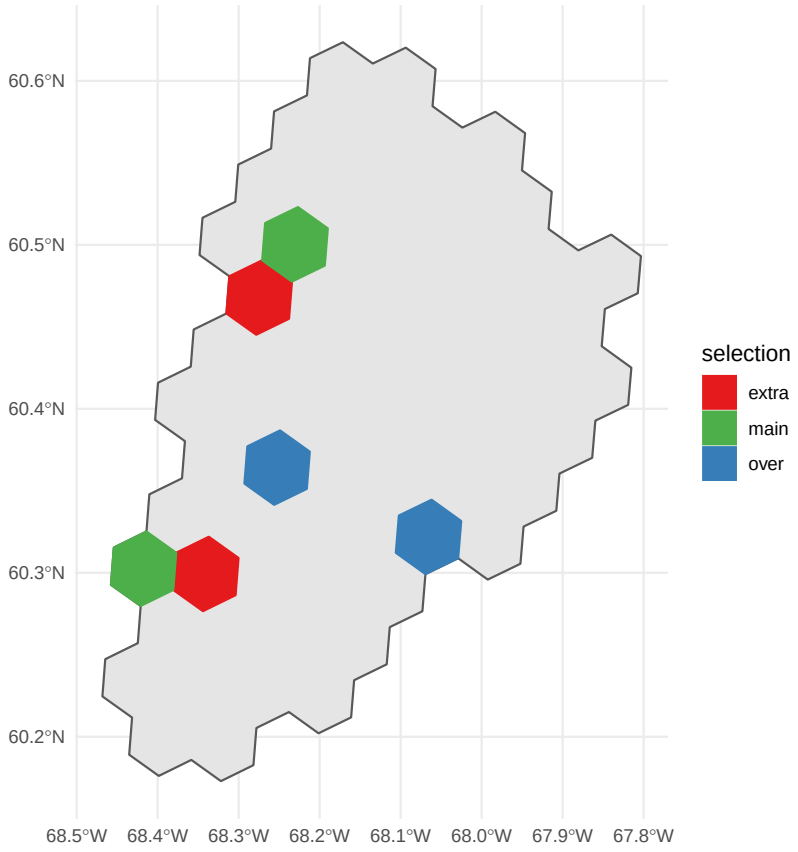
This section displays the selected PSUs (Primary Sampling Units) for each ecoregion in the 2023 sampling design version. The GRTS method selects one main hexagon (marked in green) and one over hexagon (blue) for each ecoregion. As the over hexagon is chosen randomly, we additionally select an extra hexagon (red) next to each main hexagon with the highest inclusion probability among its neighboring hexagons.

12.1 Ecoregion 7



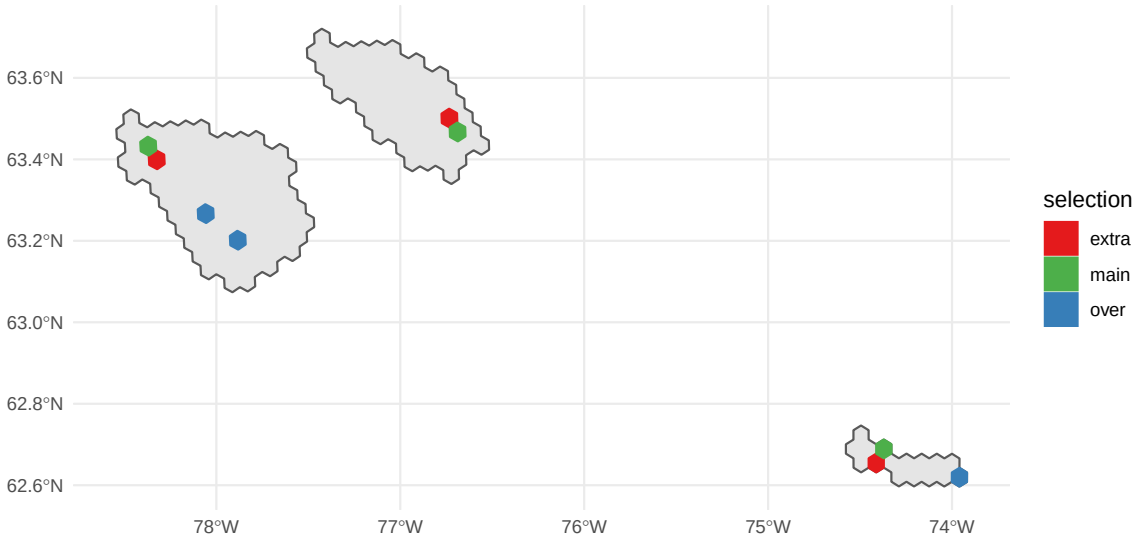
##

Ecoregion 28



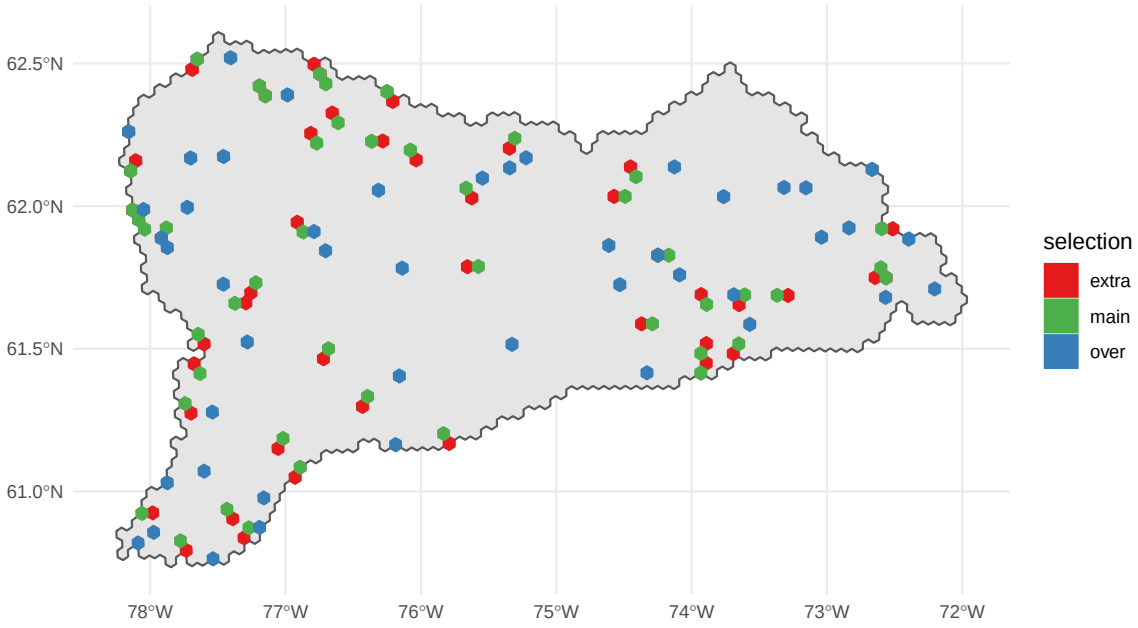
##

Ecoregion 30



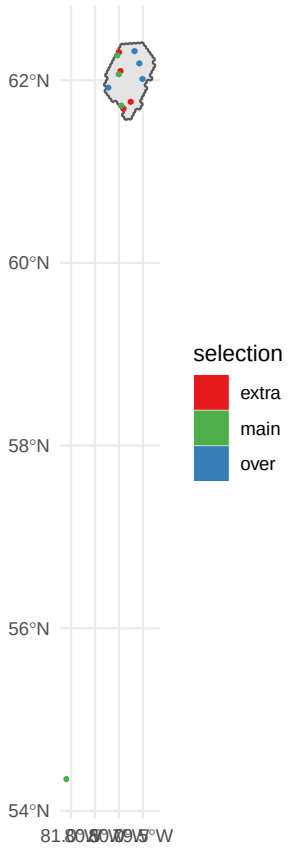
##

Ecoregion 31



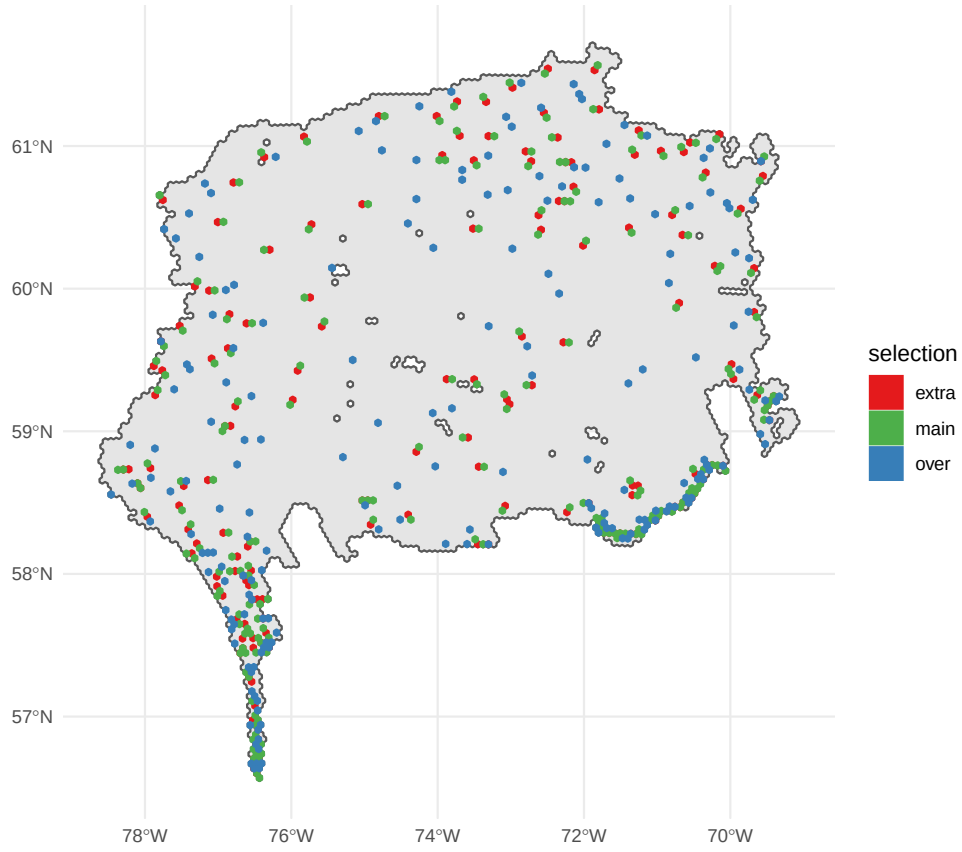
##

Ecoregion 46



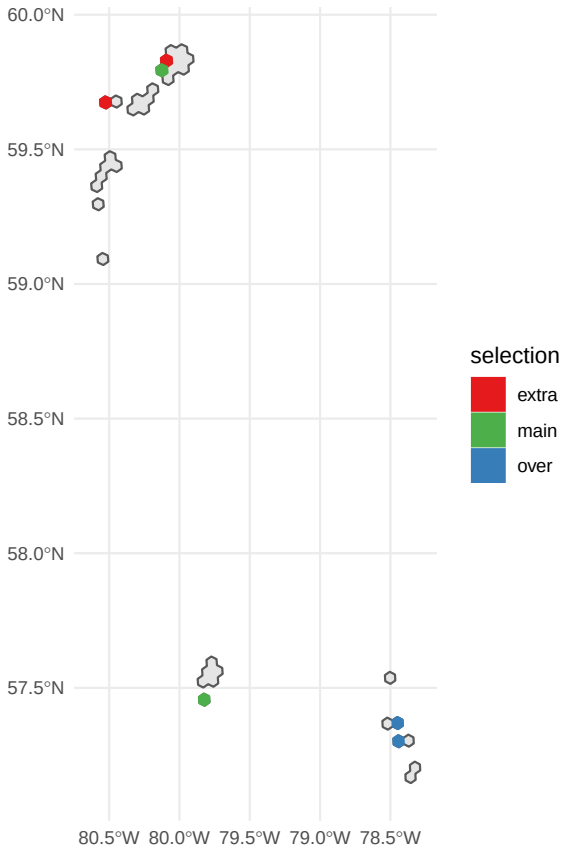
##

Ecoregion 47



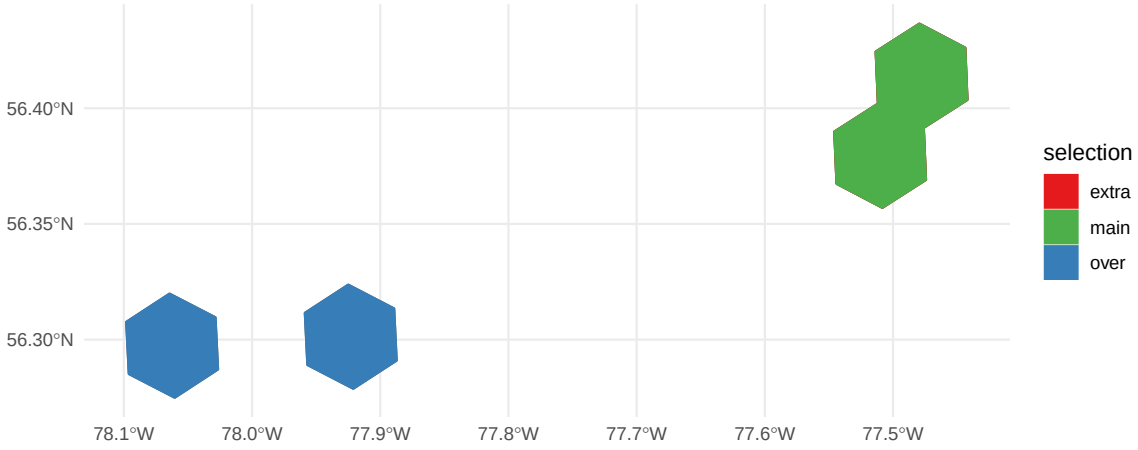
##

Ecoregion 48



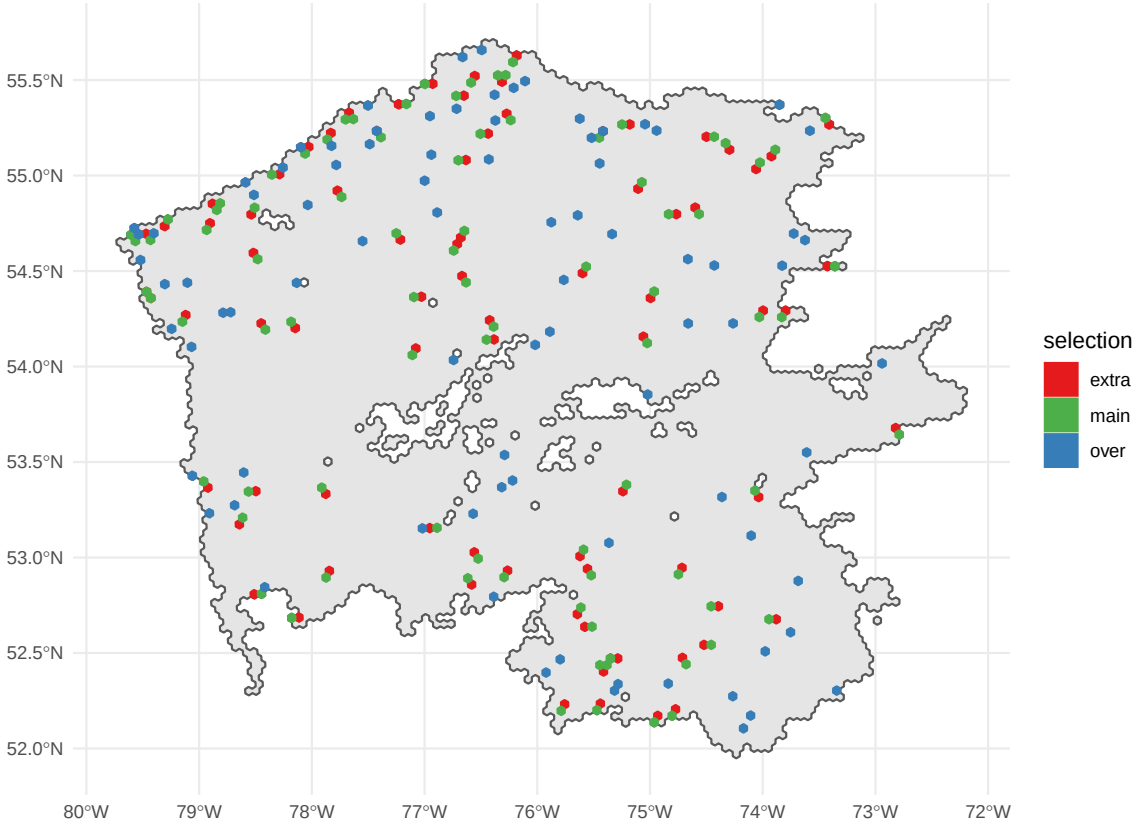
##

Ecoregion 49



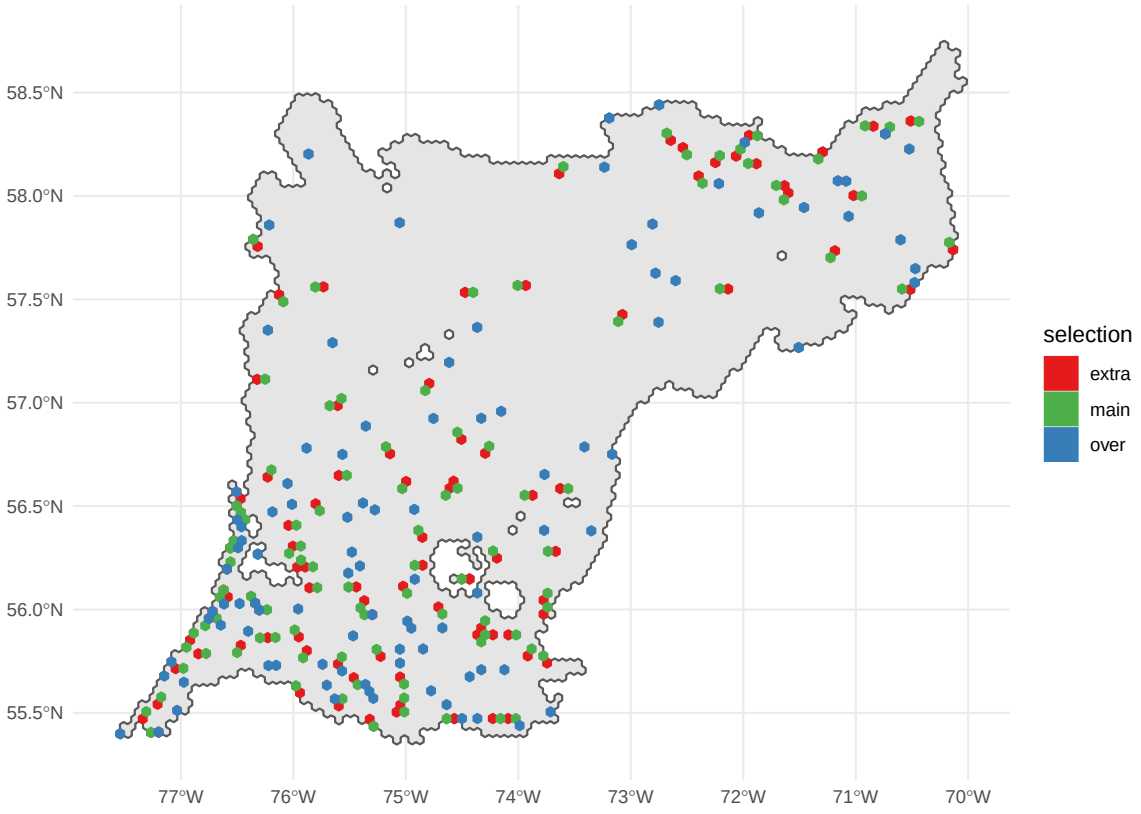
##

Ecoregion 72



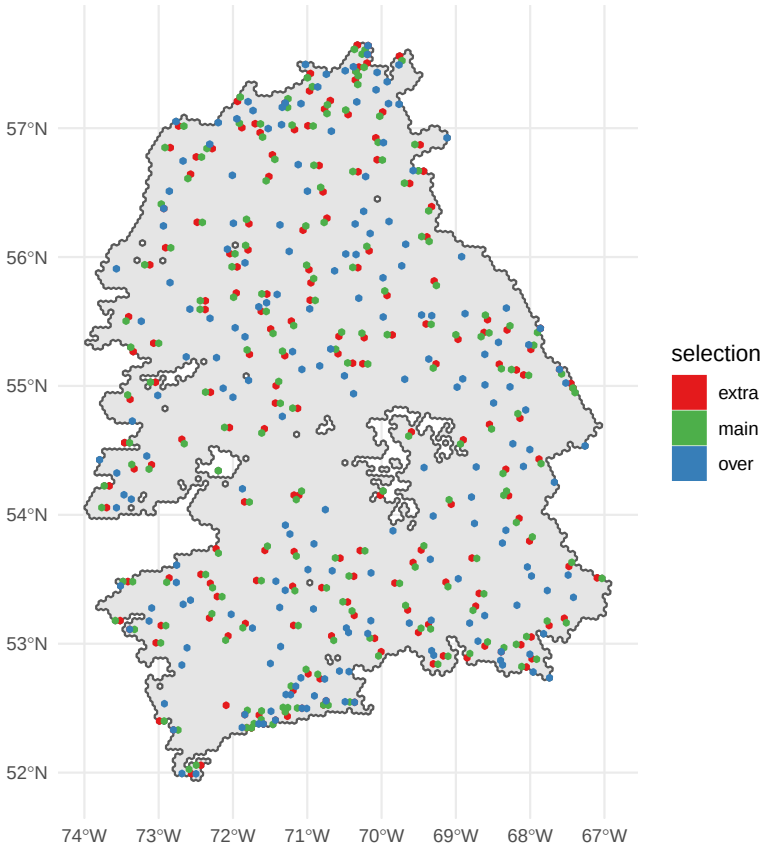
##

Ecoregion 73



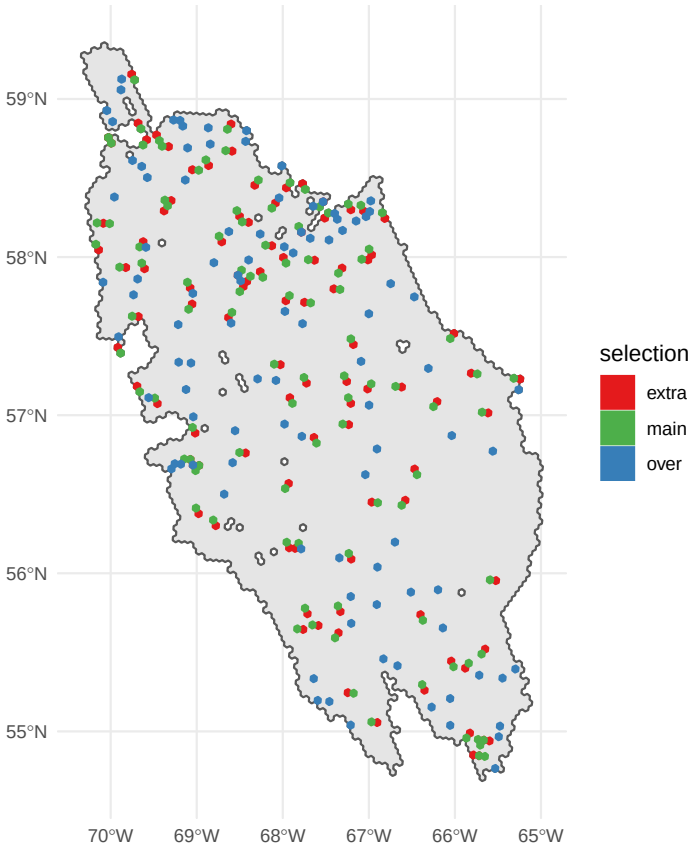
##

Ecoregion 74



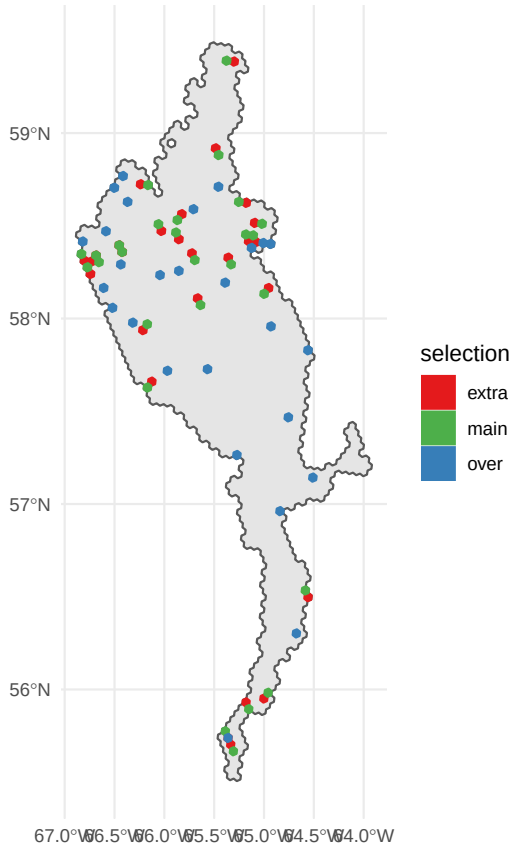
##

Ecoregion 75



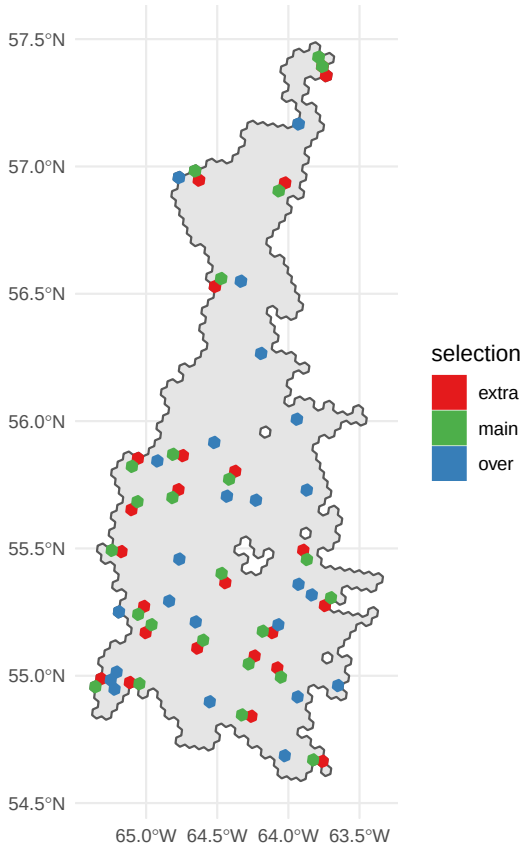
##

Ecoregion 76



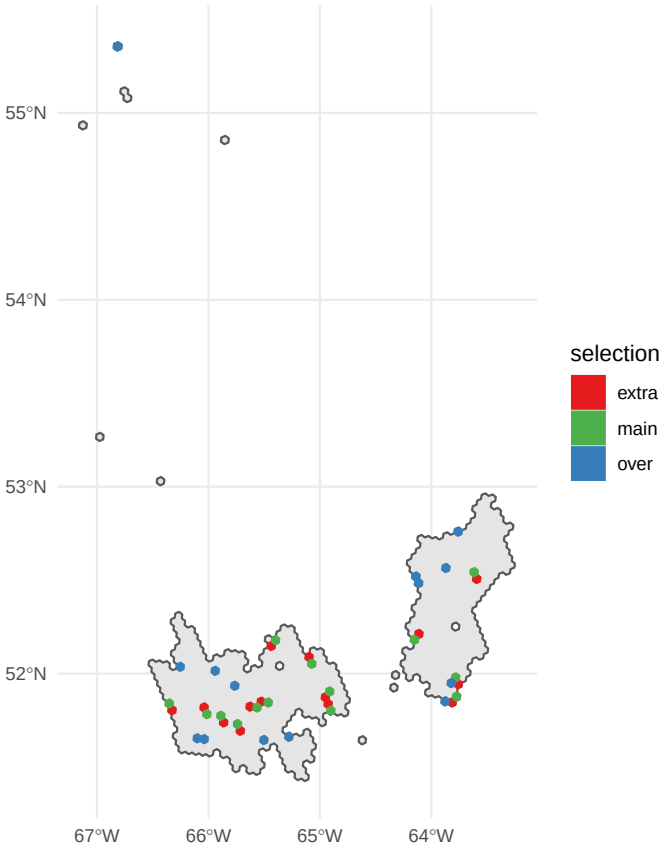
##

Ecoregion 77



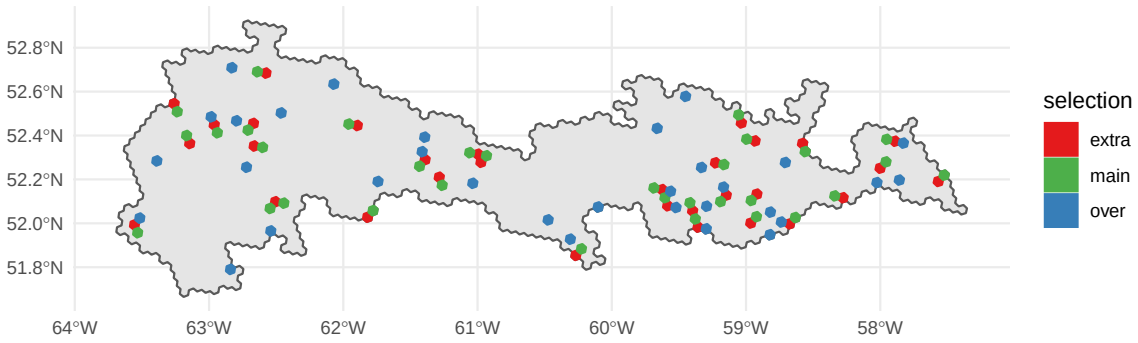
##

Ecoregion 78



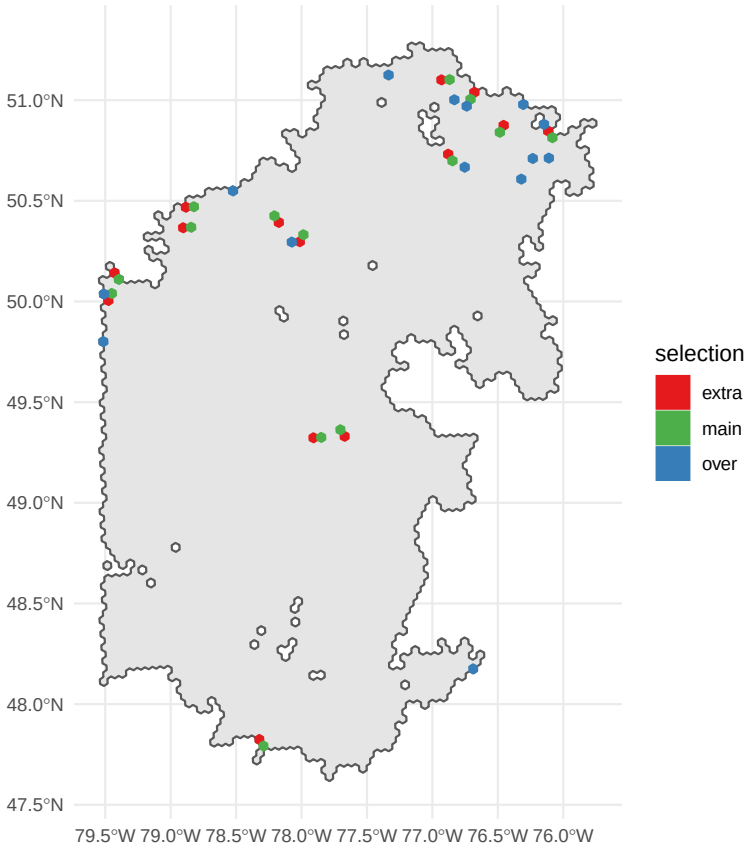
##

Ecoregion 86



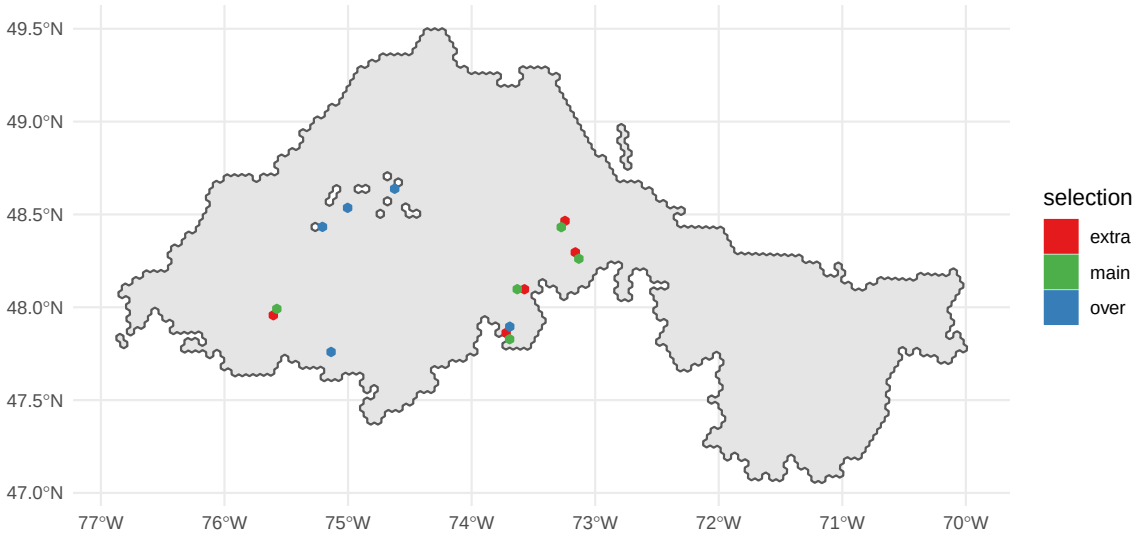
##

Ecoregion 96



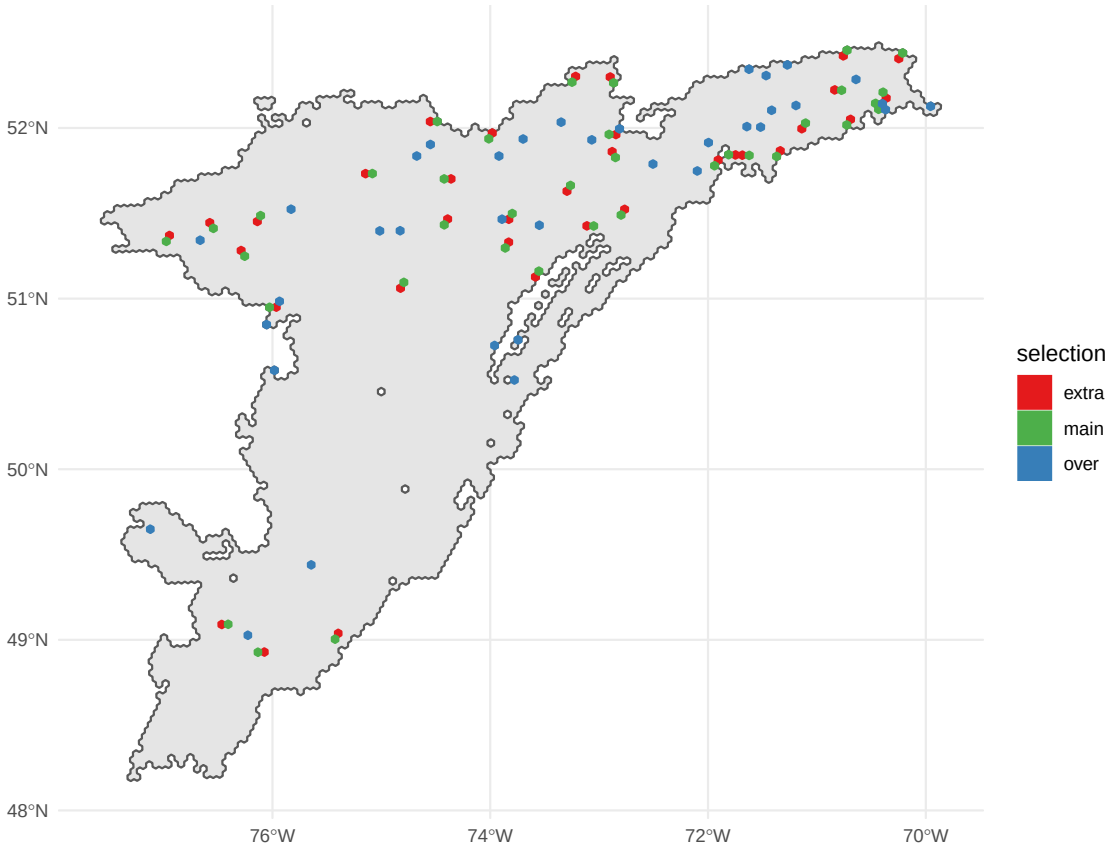
##

Ecoregion 99



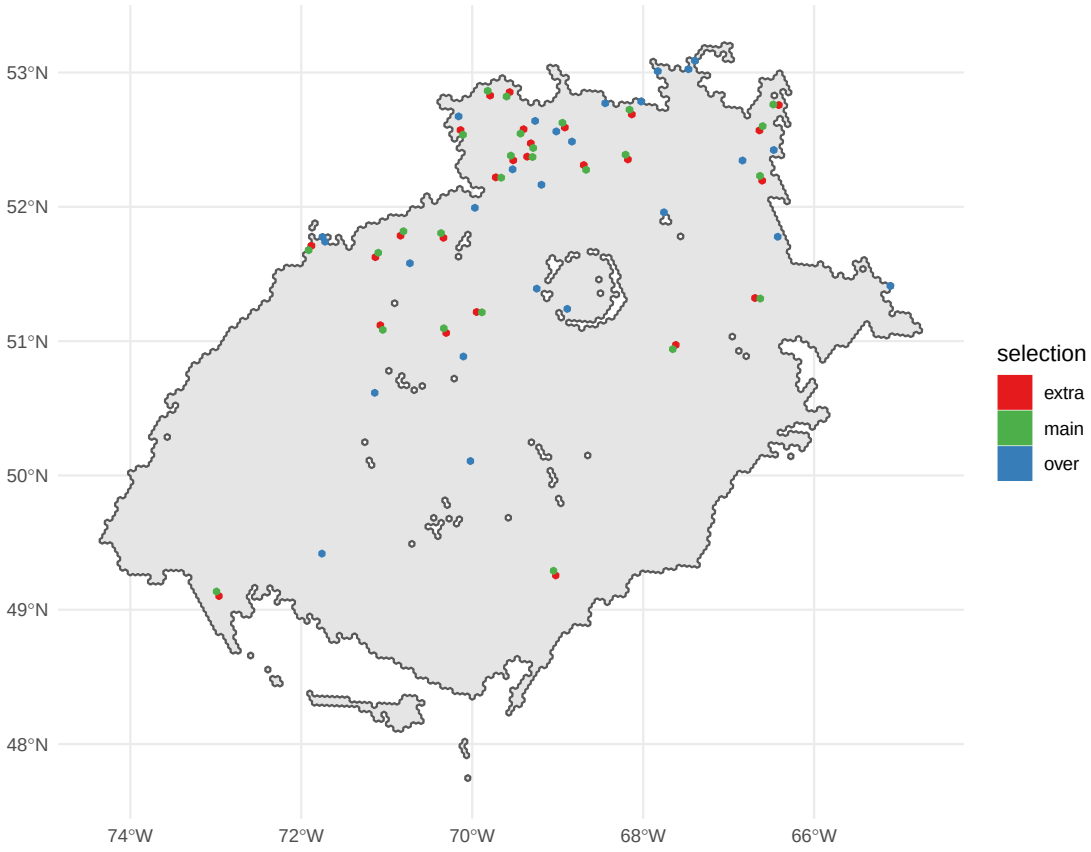
##

Ecoregion 100



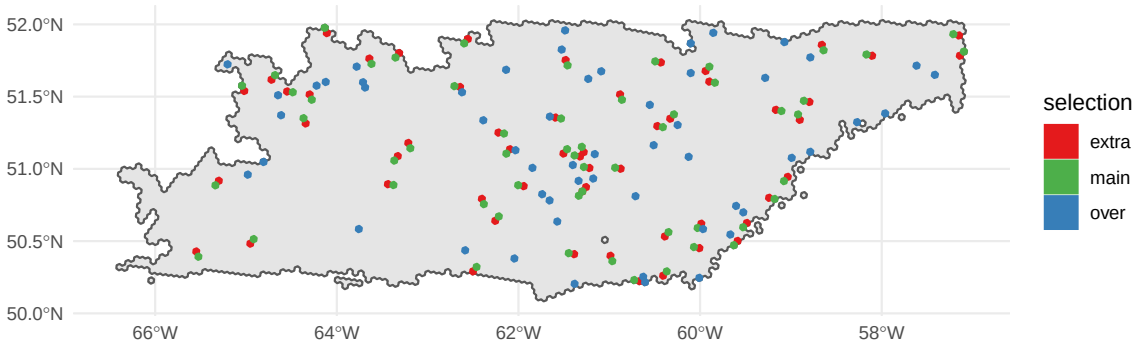
##

Ecoregion 101



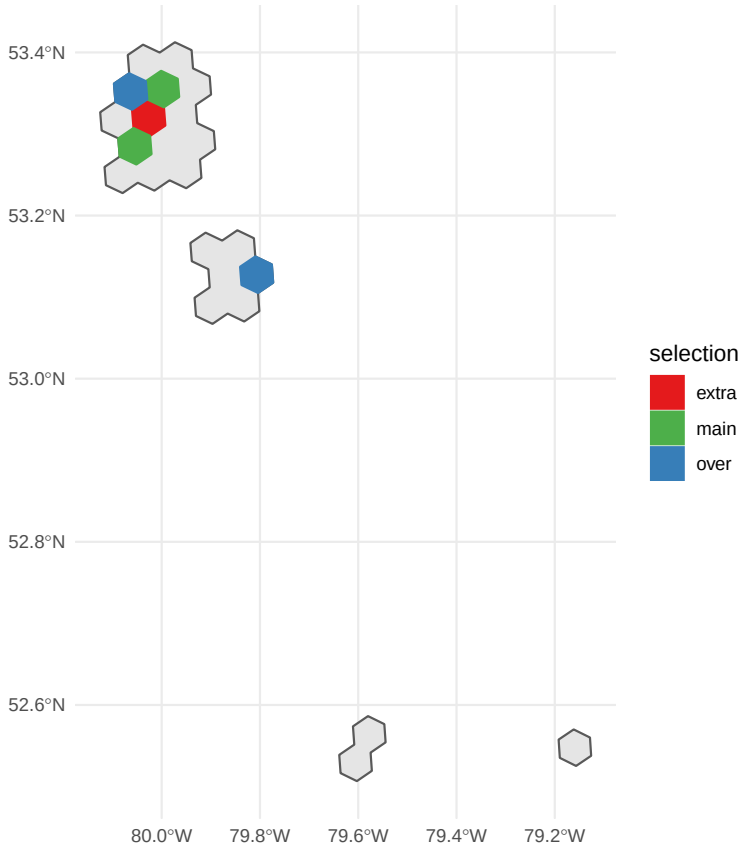
##

Ecoregion 103



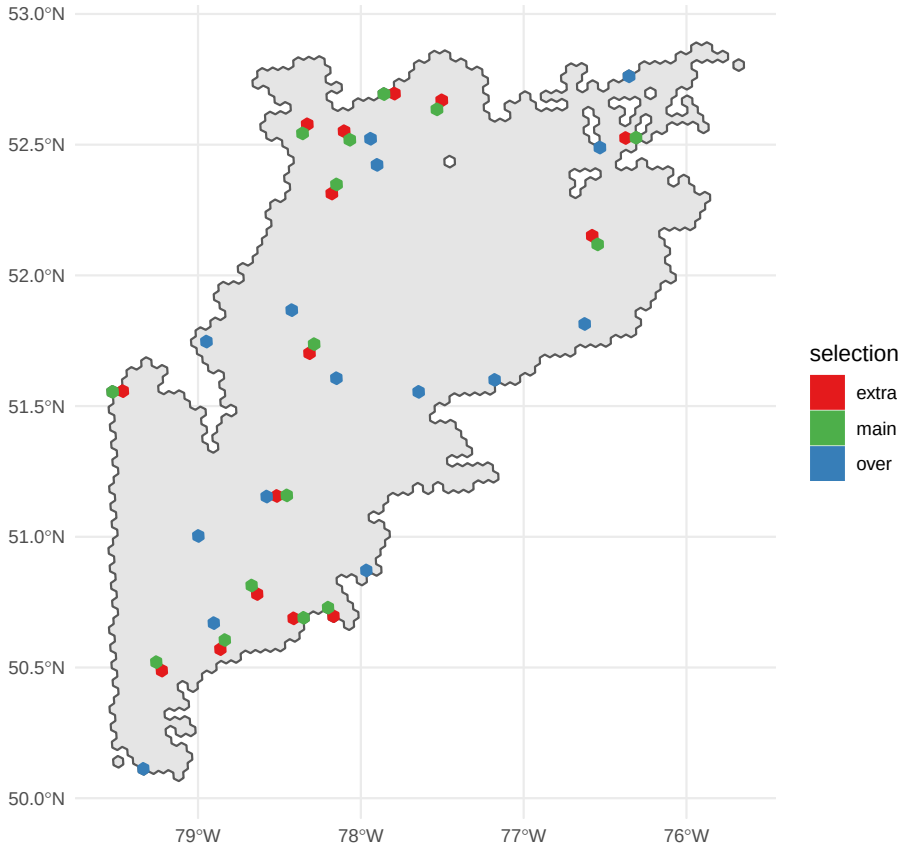
##

Ecoregion 216



##

Ecoregion 217


53.0°N
52.5°N
52.0°N
51.5°N
51.0°N
50.5°N
50.0°N
79°W
78°W
77°W
76°W
selection
extra
main
over

References

Foster, Scott D. 2021. "MBHdesign: An r-Package for Efficient Spatial Survey Designs." *Methods in Ecology and Evolution* 12 (3): 415–20. https://doi.org/https://doi.org/10.1111/2041-210X.13535.

Latifovic, Rasim, Colin Homer, Rainer Ressl, Darren Pouliot, Sheikh Nazmul Hossain, René R Colditz, Ian Olthof, Chandra P Giri, and Arturo Victoria. 2016. "20 North American Land-Change Monitoring System." *Remote Sensing of Land Use and Land Cover*, 303.

Stevens Jr, Don L, and Anthony R Olsen. 2004. "Spatially Balanced Sampling of Natural Resources." *Journal of the American Statistical Association* 99 (465): 262–78.

Van Wilgenburg, C. Lisa AND Campbell, Steven L. AND Mahon. 2020. "A Cost Efficient Spatially Balanced Hierarchical Sampling Design for Monitoring Boreal Birds Incorporating Access Costs and Habitat Stratification." *PLOS ONE* 15 (6): 1–28. https://doi.org/10.1371/journal.pone.0234494.