GBIF_DB post original study temporal: Filter redlining hexes to original paper cities (HOLC + GBIF timeseries)

Diego Ellis Soto

2026-01-15

Here we extend our original HOLC–GBIF sampling-density analysis, but beyond the time period of our original study.

It uses cloud-hosted, hex-indexed (H3) versions of GBIF and HOLC datasets to (i) subset HOLC hexes to the original study cities, (ii) join GBIF records to HOLC grades via shared H3 indices, (iii) estimate polygon area from the number of associated hex cells (resolution 10; ~0.015 km² per cell), and (iv) compute annual sampling density by HOLC grade for 2000–2023, including post-2020 exploratory summaries and plots.

Next steps look at code: g1c

```
knitr::opts_chunk$set(
  echo = TRUE, message = FALSE, warning = FALSE
)
suppressPackageStartupMessages({
library(dplyr)
library(readr)
library(duckdbfs)
library(ggplot2)
library(glue)
})
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```
duckdb_secrets("", "", "s3-west.nrp-nautilus.io")
```

```
## [1] 1
```

```
# edit these filters as you like
gbif <- open_dataset("s3://public-gbif/2025-06/hex") |>
  filter(
    `class` == "Aves",
#    institutioncode %in% c("CLO", "iNaturalist")
  )

# -------------------------
# 0) Cities used in original analysis (local)
# -------------------------
comp <- read_csv("../../indir/Biodiv_Greeness_Social/main_combined_2022-05-27.csv")
```

```
## Rows: 9851 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): id, state, city, holc_id, holc_grade, city_state, msa_NAME
## dbl (25): area_holc_km2, holc_tot_pop, msa_GEOID, msa_M, msa_p, msa_H, msa_e...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
cities_used <- comp %>%
  distinct(city, state) %>%
  mutate(city_state = paste(city, state, sep = ", "))


# -------------------------
# 2) Lazy Redlining table (DuckDB)
# -------------------------
redlining_lazy <- open_dataset("s3://public-redlining/hex") %>%
  mutate(h10 = sql("('0x' || h10)::UBIGINT")) %>%   # your hotfix
  # rename(holc_grade = grade) %>%
  mutate(city_state = paste(city, state, sep = ", "))

# -------------------------
# 3) Filter redlining to ONLY your cities (no join, no copy_to)
# -------------------------
redlining <- redlining_lazy %>%
  filter(city_state %in% cities_used$city_state)

# -------------------------
# 4) Areas (still lazy)
# -------------------------
hex_area <- redlining %>%
  group_by(area_id) %>%
  summarise(area = n() * 0.015047502, .groups = "drop")

# -------------------------
# 5) Join to GBIF (both lazy → no auto_copy)
# -------------------------
dt <- redlining %>%
  left_join(gbif, by = c("h0", "h10"))


trend = dt |>
  dplyr::count(area_id, grade, year, institutioncode) |>
  dplyr::inner_join(hex_area) |>
  dplyr::mutate(count_density = n / area)
```

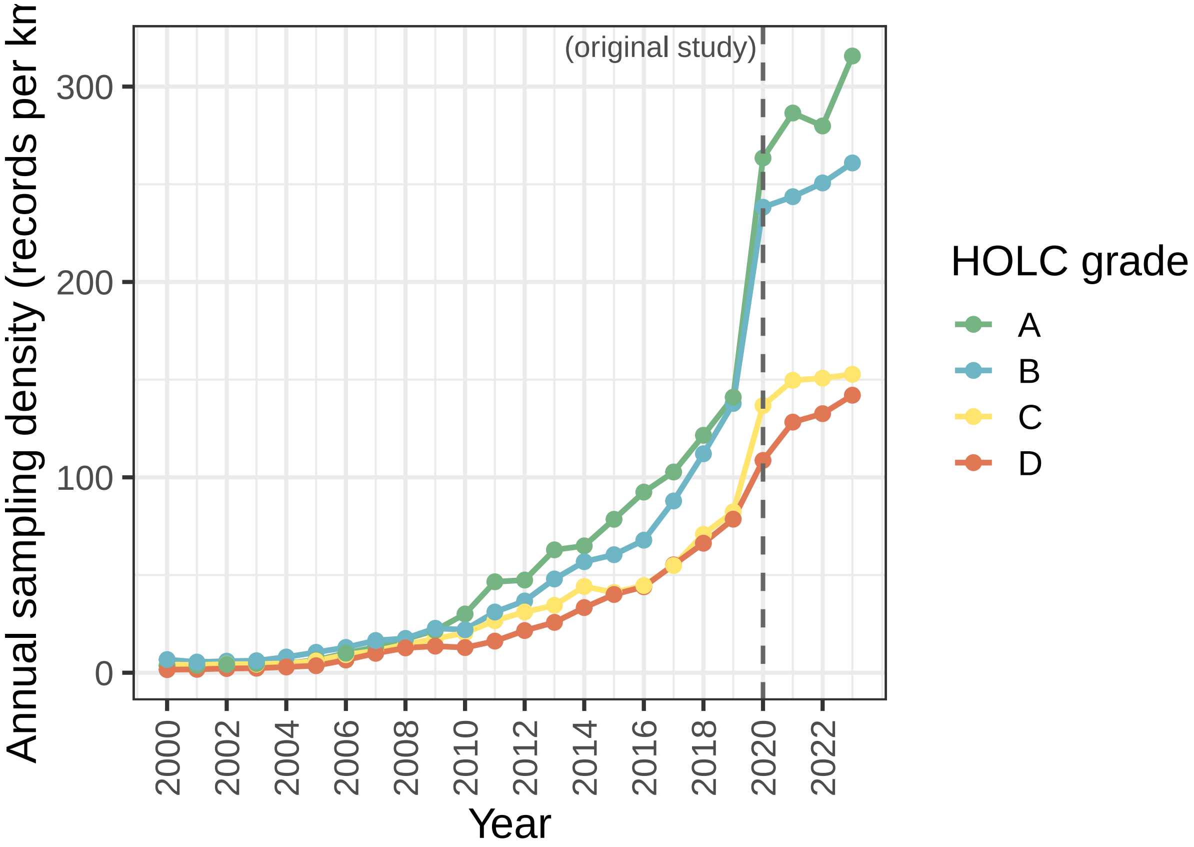
```
## Joining with `by = join_by(area_id)`
```

```r
# --- Areas ---
grade_area <- redlining %>%
  distinct(area_id, grade) %>%
  left_join(hex_area, by = "area_id") %>%
  group_by(grade) %>%
  dplyr::summarise(total_area = sum(area), .groups = "drop")

# --- Annual counts ---
trend_year <- dt %>%
  dplyr::filter(!is.na(year),
         grade %in% c("A","B","C","D"),
         year >= 2000, year <= 2023) %>%
  dplyr::count(area_id, grade, year) %>%
  group_by(grade, year) %>%
  dplyr::summarise(total_n = sum(n), .groups = "drop") %>%
  left_join(grade_area, by = "grade") %>%
  dplyr::mutate(density = total_n / total_area)

# --- Cumulative version (matches your 2nd plot concept) ---
trend_cum <- trend_year %>%
  dplyr::group_by(grade) %>%
  dplyr::arrange(year) %>%
  dplyr::mutate(density_cum = cumsum(total_n) / total_area) %>%
  ungroup()

# Colors (use yours)
holc_cols <- c(
  "A" = "#76b583",
  "B" = "#6eb6c5",
  "C" = "#ffe56d",
  "D" = "#e07856"
)

holc_cols <- c(
  "A" = "#76b583",
  "B" = "#6eb6c5",
  "C" = "#ffe56d",
  "D" = "#e07856"
)


# p_annual <- ggplot(trend_year, aes(x = year, y = density, color = grade)) +
#   geom_line(linewidth = 1) +
#     geom_point(size = 2) +
#   theme_bw() +
#   coord_cartesian(xlim = c(2000, 2023)) +
#   labs(
#     x = "Year",
#     y = "Sampling density (records per km²) - annual",
#     color = "HOLC grade"
#   ) +
#   scale_colour_manual(values = holc_cols)+
#     theme_bw(16)
# # +  coord_cartesian(ylim = c(0, 300)) + scale_y_continuous(breaks = seq(0, 300, by = 100))

# p_annual
#
```

```r
# ggsave('../../outdir/accu_time_post_2020_original_cities.png')


trend_year_plot <- trend_year %>%
  filter(year >= 2000, year <= 2023)

# p_annual_v2 <- ggplot(trend_year_plot, aes(x = year, y = density, color = grade)) +
#   geom_line(linewidth = 1) +
#   geom_point(size = 2) +
#   theme_bw(base_size = 16) +
#   scale_colour_manual(values = holc_cols) +
#   scale_x_continuous(limits = c(2000, 2023), breaks = seq(2000, 2023, 2)) +
#   labs(
#     x = "Year",
#     y = "Sampling density (records per km²) - annual",
#     color = "HOLC grade"
#   )
#
# p_annual_v2

# ggsave('../../outdir/accu_time_post_2020_original_cities_until_2023.png')

p_annual_v3 <- ggplot(trend_year_plot, aes(x = year, y = density, color = grade)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +

  # vertical line at 2020
  geom_vline(
    xintercept = 2020,
    linetype = "dashed",
    linewidth = 0.8,
    colour = "grey40"
  ) +

  # annotation
  annotate(
    "text",
    x = 2019.8,
    y = Inf,
    # label = "Ellis-Soto et al. 2023\n(original study)",
    label = "(original study)",
    vjust = 1.5,
    # hjust = -0.05, # right side
       hjust = 1, # left side
    size = 4,
    colour = "grey30"
  ) +

  theme_bw(base_size = 16) +
  scale_colour_manual(values = holc_cols) +
  scale_x_continuous(
    limits = c(2000, 2023),
    breaks = seq(2000, 2023, 2)
  ) +
  labs(
    x = "Year",
    y = "Annual sampling density (records per km²)",
    color = "HOLC grade"
  ) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
```

```
## Warning: Missing values are always removed in SQL aggregation functions.
## Use ‘na.rm = TRUE‘ to silence this warning
## This warning is displayed once every 8 hours.
```

```
p_annual_v3
```



```
ggsave('../../outdir/accu_time_post_2020_original_cities_until_2023_vline.png')
```

```
## Saving 6.5 x 4.5 in image
```

```
sessionInfo()
```

```
## R version 4.4.1 (2024-06-14)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sonoma 14.6
```

```
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] glue_1.8.0        ggplot2_4.0.1     duckdbfs_0.1.0.99 readr_2.1.5
## [5] dplyr_1.1.4
##
## loaded via a namespace (and not attached):
##  [1] bit_4.6.0          gtable_0.3.6       crayon_1.5.3       compiler_4.4.1
##  [5] tidyselect_1.2.1   blob_1.2.4         parallel_4.4.1     dichromat_2.0-0.1
##  [9] textshaping_1.0.1  systemfonts_1.2.3  scales_1.4.0       yaml_2.3.10
## [13] fastmap_1.2.0      R6_2.6.1           labeling_0.4.3     generics_0.1.4
## [17] knitr_1.50         tibble_3.3.0       DBI_1.2.3          pillar_1.11.1
## [21] RColorBrewer_1.1-3 tzdb_0.5.0         rlang_1.1.6        xfun_0.52
## [25] fs_1.6.6           S7_0.2.1           bit64_4.6.0-1      cli_3.6.5
## [29] withr_3.0.2        magrittr_2.0.4     digest_0.6.37      grid_4.4.1
## [33] vroom_1.6.5        rstudioapi_0.17.1  dbplyr_2.5.0       hms_1.1.3
## [37] lifecycle_1.0.4    vctrs_0.6.5        evaluate_1.0.3     farver_2.1.2
## [41] duckdb_1.2.1       ragg_1.4.0         rmarkdown_2.29     purrr_1.2.0
## [45] tools_4.4.1        pkgconfig_2.0.3    htmltools_0.5.8.1
```