Emerging topics and new directions in statistical ecology

Altwegg et al. Supplemental Material

Install and Load Relevant Packages

Here, we install and load the necessary R packages required for data processing, topic modeling, and visualisation.

```
############################################################################ Load
############################################################################ Packages
############################################################################ #############
if (!require("pacman")) install.packages("pacman")
# devtools::install_github('mikajoh/stmprinter')

p_load(quanteda, tm, tidytext, topicmodels, stm, BTM, bibliometrix,
    lexicon, wordcloud, textstem, SnowballC, stringi, furrr,
    LDAvis, stminsights, stmCorrViz, stmprinter, udpipe, igraph,
    ggraph, textplot, tidygraph, hrbrthemes, extrafont, tidyverse,
    readxl, data.table, janitor, ggthemes, DT, gt, gtExtras,
    extrafont, tictoc, scales, ggtext, writexl, ggrepel, ggeasy,
    stmprinter)
"%notin%" <- Negate("%in%")

# extrafont::font_import()
extrafont::loadfonts()
```

Collect Additional Stopwords Used for Data Cleaning and Topic Modelling

Stopwords are frequently occurring words (e.g., "the", "and", "is") that are typically removed during text processing, as they do not contribute significant meaning to the analysis. This step involves expanding the stopword list associated with the text processing and topic modelling packages in R with additional terms specific to the dataset being analysed. The list below is based on Gimenez et. al (2014) which used a subset of the current dataset.

```
############################################################################ Additional
############################################################################ Stopwords
############################################################################ #############


additional_stops <- c("use", "using", "used", "also", "among",
    "will", "always", "one", "first", "though", "may", "often",
    "particular", "account", "additional", "allow", "approach",
    "assess", "available", "based", "can", "case", "combination",
    "combined", "combine", "common", "consider", "considered",
    "considers", "current", "depend", "depends", "describe",
    "describes", "described", "develop", "developped", "develops",
    "differ", "differs", "discuss", "discussed", "discusses",
    "general", "high", "however", "framework", "improves", "improved",
    "improve", "includes", "include", "included", "invetsigate",
    "investigated", "large", "larger", "make", "mean", "number",
    "obtain", "need", "needs", "needed", "new", "order", "perform",
    "performs", "performed", "possible", "potential", "present",
    "problem", "propose", "proposed", "proposes", "provide",
    "provides", "provided", "potential", "related", "relate",
    "relates", "require", "requires", "required", "results",
    "result", "resulting", "set", "several", "similar", "size",
    "study", "studies", "source", "technique", "techniques",
    "three", "type", "two", "understand", "value", "values",
    "well", "within")
```

Load Cleaned Data Set

The raw datasets were sourced as follows:

- Received isec2008-2018 abstracts from Chris Sutherland (file called isecdat.RData)
- Obtained isec2020 abstracts from https://github.com/oliviergimenez/textmining_isec2020
- Obtained isec2022 abstracts from https://github.com/oliviergimenez/topic-isec2022

These were then combined into a single dataset for further processing. The combined dataset of abstracts was then checked for duplicates and extensively cleaned: removing numbers, punctuation, special characters, and stopwords as well as converting all the text in the abstracts to lowercase. *Note:* that the cleaned data is loaded here. The actual data preparation steps and codes are not included in this document. Where applicable, the text document was then used to generate individual word tokens. Each token underwent stemming, which reduces words to their root forms by removing prefixes or suffixes. For example, "analysing," "analysed," and "analysis" would be stemmed to "analys".

```
############################################################################ Load
############################################################################ cleaned
############################################################################ data
############################################################################ #############

isec_dat <- read.csv(here::here("data/isec_datfull_including2018_all_cleaned.csv"))
```

Count of Abstracts over time

Code used to generate plot of count of ISEC abstracts over time.

```
### Count of Abstracts over time

nabstracts <- table(isec_dat$year)

pdf(here::here("Figures/nabstracts.pdf"), 7, 5)
p <- barplot(nabstracts, las = 1, xlab = "Conference Year", ylab = "Number of Abstracts",
    ylim = range(pretty(c(0, nabstracts))))
text(x = p, y = 80, labels = c("St Andrews", "Kent", "Oslo",
    "Montpellier", "Seattle", "St Andrews", "Sydney (online)",
    "Cape Town (hybrid)"), srt = 90)
dev.off()
```

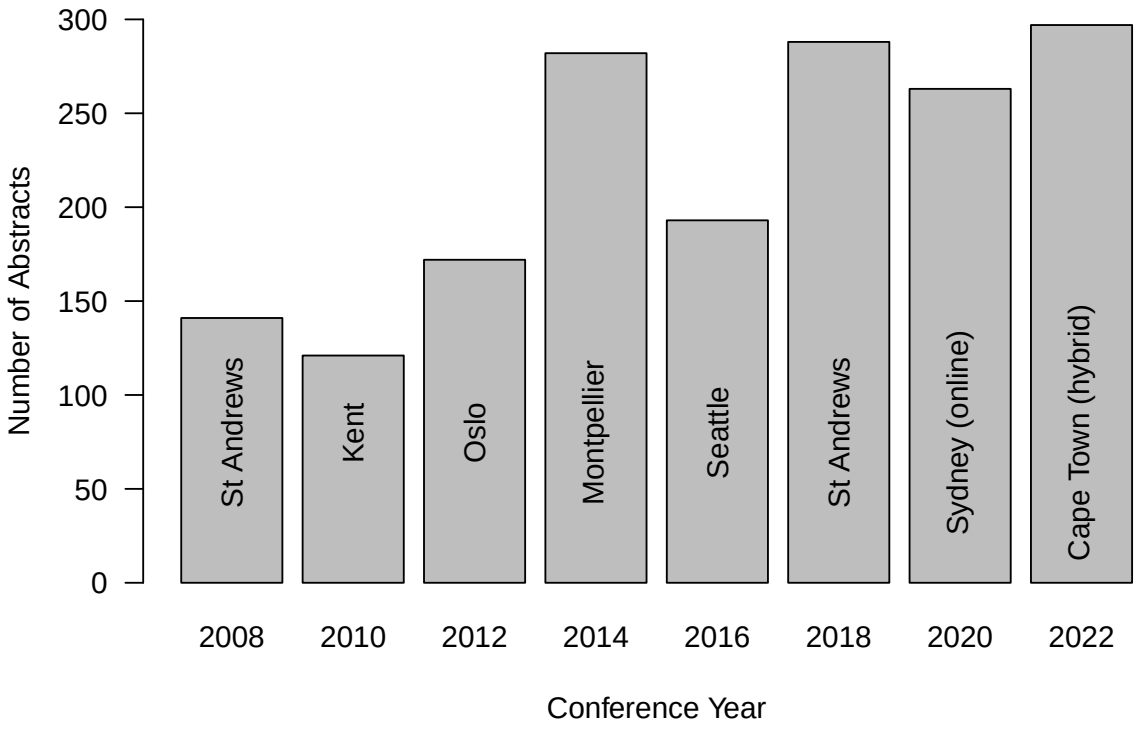


Figure 1: Number of abstracts used in this analysis for each International Statistical Ecology Conference (ISEC). ISEC was held every second year since 2008. The location where each conference took place is written on the bars.

N-gram Analysis

N-gram analysis explores sequences of words or characters in a text to uncover patterns and frequent pairings. It groups words into units of size n, such as unigrams (single words), bigrams (pairs of consecutive words), or trigrams (three-word sequences). In this study, we focused on extracting bigrams to balance simplicity with meaningful contextual insights, enabling the identification of commonly co-occurring words and shedding light on how the context of abstracts has evolved across different conferences.

The code chunck below was used to extract and visualise the bigrams from the ISEC abstracts to help us identify key word pairs that informed both the topic modeling and our review.

```
############################################################################## Abstract
############################################################################## Bigrams
############################################################################## #############


tidy_isec_bigram_data_all_long <- isec_dat %>%
    mutate(line = row_number()) %>%
    unnest_tokens(bigram, abstracts, token = "ngrams", n = 2) %>%
    separate(bigram, into = c("word1", "word2"), sep = " ") %>%
    filter_at(vars(starts_with("word")), all_vars(!is.na(.))) %>%
    filter_at(vars(starts_with("word")), all_vars(. %notin% stop_words$word)) %>%
    filter_at(vars(starts_with("word")), all_vars(. %notin% additional_stops)) %>%
    filter_at(vars(starts_with("word")), all_vars(!str_detect(.,
        "[0-9]+"))) %>%
    mutate_at(vars(starts_with("word")), ~wordStem(.)) %>%
    unite(bigram, c(word1, word2), sep = " ", remove = F) %>%
    group_by(year) %>%
    count(bigram, sort = TRUE) %>%
    ungroup()

### >>> bigram_Wide Data
tidy_isec_bigram_data_all_wide <- tidy_isec_bigram_data_all_long %>%
    pivot_wider(names_from = year, values_from = n) %>%
    adorn_totals("col") %>%
    mutate(data = "bigram") %>%
    select(word_s = bigram, data, `2008`, `2010`, `2012`, `2014`,
        `2016`, `2018`, `2020`, `2022`, Total)

### >>> bigram_Long Data_top10
tidy_isec_bigram_long_top10 <- tidy_isec_bigram_data_all_long %>%
    group_by(year) %>%
    arrange(-n) %>%
    top_n(10) %>%
    ungroup
```

```r
###-------------------------- Bigrams Faceted Bar Graph ----------------##########
tidy_isec_bigram_long_top10 %>%
    ungroup() %>%
    ggplot(aes(x = reorder_within(bigram, n, year), y = n, fill = bigram,
        label = round(n, 0))) + geom_col(show.legend = F) + facet_wrap(~year,
    scales = "free_y") + coord_flip() + scale_x_reordered() +
    scale_y_continuous(expand = c(0, 0)) + geom_text(size = 2.5,
    color = "black", hjust = 1.5) + labs(y = "Count", x = "Bigrams") +
    theme(text = element_text(family = "Times"), axis.text.y = element_markdown(size = 8),
        plot.title.position = "plot", plot.title = element_markdown(size = 14,
            lineheight = 1), plot.margin = margin(rep(15, 4))) +
    theme_bw() + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())


ggsave(here::here("Figures/2. tidy_isec_bigram_long_top10.pdf"),
    device = cairo_pdf, width = 297, height = 210, units = "mm")
```

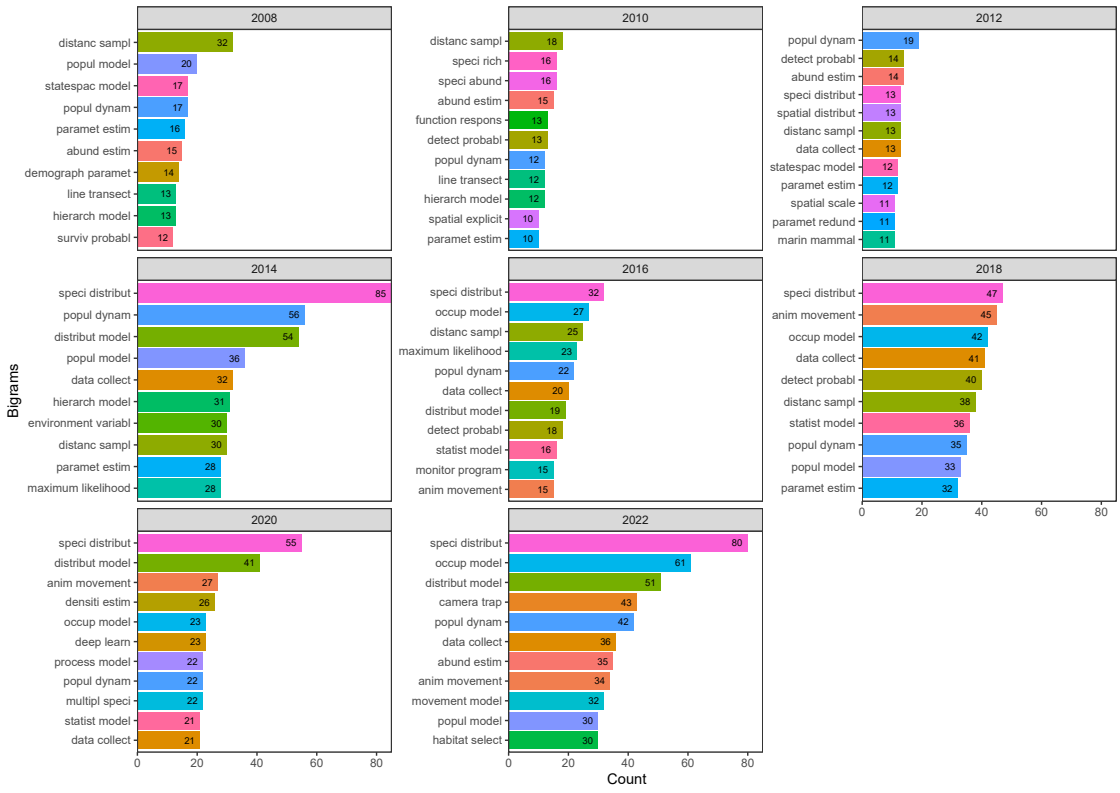


Figure 2: Frequencies of the most common two-word phrases (bigrams) appearing in the abstracts presented at the International Statistical Ecology Conference (ISEC) held every two years from 2008 to 2022. Individual phrases are colour coded to facilitate comparison among years.

Alternative Representation: Bi-gram Top 10 Table

```
###--------------------------- Alternative Representation: Bi-gram Top 10 Table  --###########

bi_tops_rank <- tidy_isec_bigram_long_top10 %>%
    group_by(year) %>%
    mutate(rank = dense_rank(desc(n))) %>%
    ungroup() %>%
    select(-n) %>%
    pivot_wider(names_from = year, values_from = bigram) %>%
    select(rank, `2008`, `2010`, `2012`, `2014`, `2016`, `2018`,
        `2020`, `2022`)

bi_tops_rank %>%
    gt %>%
    gtsave(here::here("Figures/2. tidy_isec_bigra_-table_bi_10.png"))
```

rank	2008	2010	2012	2014	2016	2018	2020	2022
1	distanc sampl	distanc sampl	popul dynam	speci distribut	speci distribut	speci distribut	speci distribut	speci distribut
2	popul model	speci abund, speci rich	abund estim, detect probabl	popul dynam	occup model	anim movement	distribut model	occup mode
3	popul dynam, statespac model	abund estim	data collect, distanc sampl, spatial distribut, speci distribut	distribut model	distanc sampl	occup model	anim movement	distribut model
4	paramet estim	detect probabl, function respons	paramet estim, statespac model	popul model	maximum likelihood	data collect	densiti estim	camera trap
5	abund estim	hierarch model, line transect, popul dynam	marin mammal, paramet redund, spatial scale	data collect	popul dynam	detect probabl	deep learn, occup model	popul dynan
6	demograph paramet	paramet estim, spatial explicit		hierarch model	data collect	distanc sampl	multipl speci, popul dynam, process model	data collect
7	hierarch model, line transect			distanc sampl, environment variabl	distribut model	statist model	data collect, statist model	abund estim
8	surviv probabl			maximum likelihood, paramet estim	detect probabl	popul dynam		anim movement
9					statist model	popul model		movement model
10					anim movement, monitor program	paramet estim		habitat select, popu model

Figure 3: Table of bi-gram word positions over time

Topic Modeling using the STM Package in R

The textProcessor function in the STM package in R was used to clean the text of ISEC abstracts and prepare it for topic modeling (stripping extra white spaces, performing stemming, removing stopwords, dropping sparse words, etc.). The lower.thresh parameter within the textProcessor function was used to remove words that occur below a specified threshold frequency, helping to reduce noise in the dataset. According to the package documentatio, the textProcessor function also performs other corpus manipulations, including renumbering word indices to correct for zero-indexing and/or unused words in the vocabulary vector.

We also obtained plots to determine the number of words and documents that could be removed at different thresholds and to visualise the impact of this removal after setting the threshold (lower.thresh = 50).

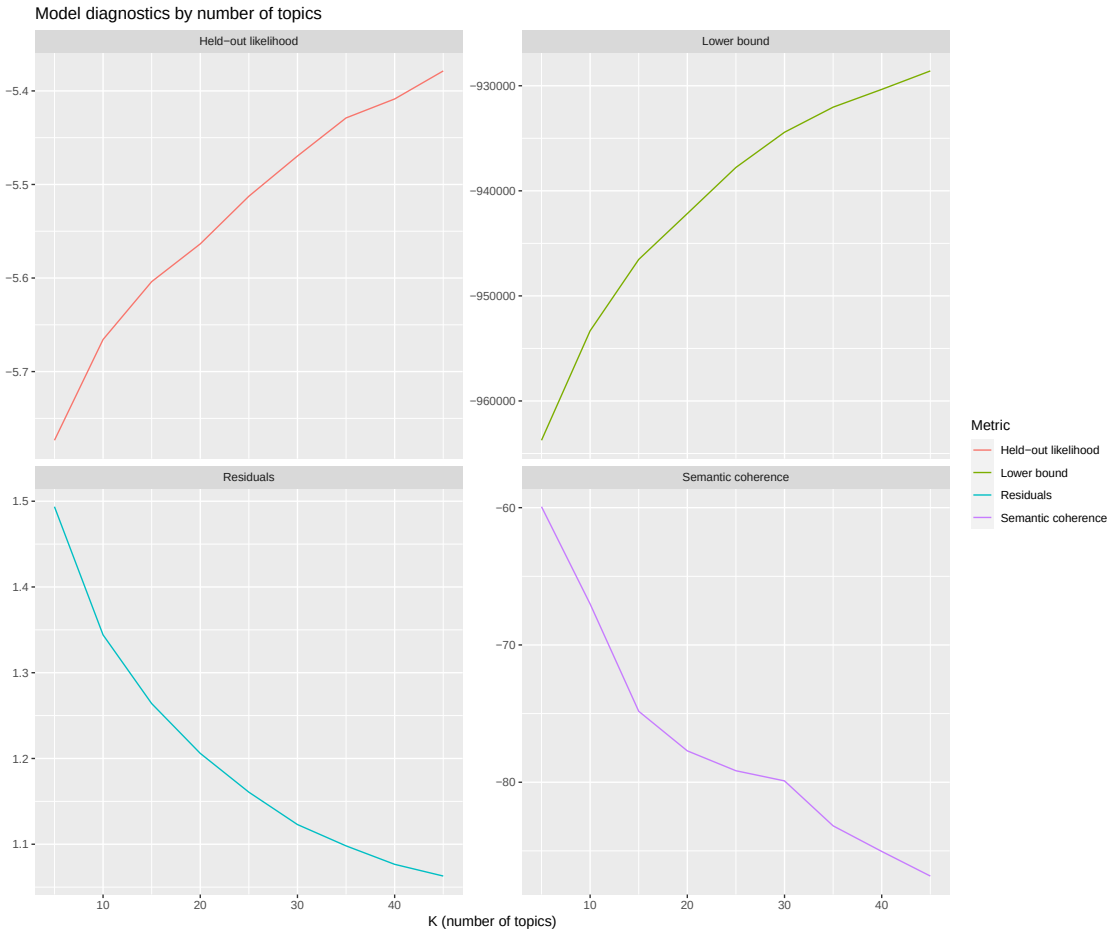
```
############################################################################ Topic
############################################################################ Modeling


### Use Structural Topic Modeling (STM). The str package
### has a bunch of useful tools.

isec_processed_in <- textProcessor(isec_dat$abstracts, metadata = data.frame(yearT = as.numeric(isec_da
    yearF = isec_dat$year, text = isec_dat$abstracts), lowercase = TRUE,
    removestopwords = TRUE, removenumbers = TRUE, removepunctuation = TRUE,
    ucp = TRUE, stem = TRUE, wordLengths = c(3, Inf), verbose = TRUE,
    striphtml = TRUE, onlycharacter = FALSE, custompunctuation = NULL,
    customstopwords = additional_stops)

pdf(here::here("Figures/5. Number of words and documents removeable for different thresholds.pdf"),
    width = 12, height = 7, bg = "white", onefile = T, colormodel = "cmyk",
    paper = "A4")
plotRemoved(isec_processed_in$documents, lower.thresh = seq(1,
    150))
dev.off()

## Set Threshold = 50
isec_processed <- prepDocuments(isec_processed_in$documents,
    isec_processed_in$vocab, isec_processed_in$meta, lower.thresh = 50)


meta <- isec_processed$meta
vocab <- isec_processed$vocab
docs <- isec_processed$documents


pdf(here::here("Figures/5. Number of words and documents removed for different thresholds_post removal
    width = 12, height = 7, bg = "white", onefile = T, colormodel = "cmyk",
    paper = "A4")
plotRemoved(isec_processed$documents, lower.thresh = seq(1, 150))
dev.off()
```

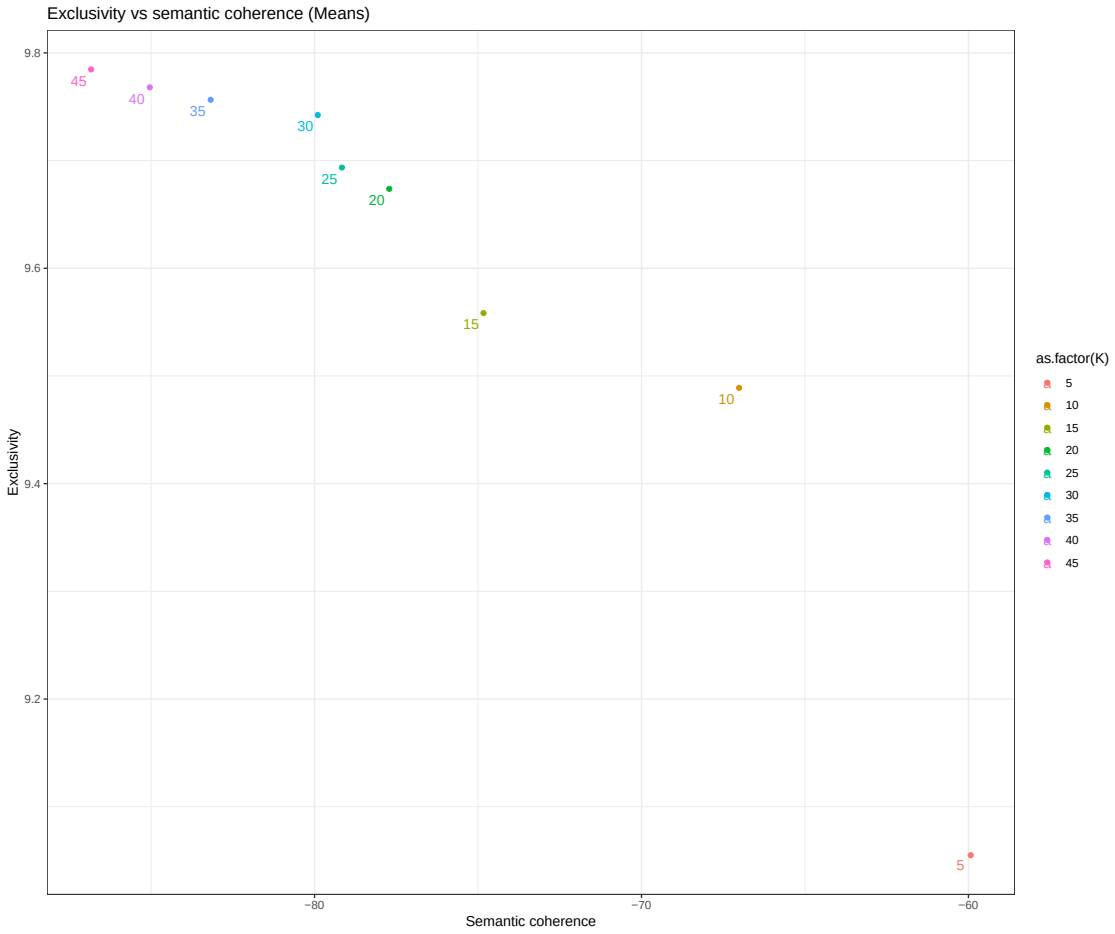
Fit Multiple Models Using plan(multisession) and Get Model Summaries (Exclusivity, Semantic Coherence, etc.)

After preprocessing, we fit multiple topic models with varying parameters (e.g., number of topics). We use plan(multisession) to parallelise the modelling process, allowing us to fit models more efficiently by utilising multiple CPU cores. The models were fitted using the `STM` package in `R`. The number of topics has to be specified by the analyst and there is no single correct way to do so. Specifying too few topics can lead to different themes becoming amalgamated and the identified topics are hard to interpret. We selected the appropriate number of topics based on exclusivity (words unique to specific topics) and semantic coherence (representative words for coherent concepts). With an increasing number of topics, semantic coherence tends to decline, meaning that semantically related words tend to be split across several topics until the same theme appears in several topics, again making the topics hard to interpret.

```
###################### Fit multiple models With time
#################### covariates ######################

plan(multisession)

search_stm_models <- data_frame(K = seq(5, 45, by = 5)) %>%
    mutate(k_model = future_map(K, ~stm(isec_processed$documents,
        isec_processed$vocab, data = isec_processed$meta, prevalence = ~yearT,
        max.em.its = 999, init.type = "Spectral", seed = 2022,
        K = ., verbose = FALSE), .options = furrr_options(seed = T)))

save(search_stm_models, file = here::here("Data/search_stm_models.Rdata"))
load(here::here("Data/search_stm_models.Rdata"))

heldout <- make.heldout(documents = isec_processed$documents,
    vocab = isec_processed$vocab)

k_result <- search_stm_models %>%
    mutate(exclusivity = map(k_model, exclusivity), semantic_coherence = map(k_model,
        semanticCoherence, isec_processed$documents), eval_heldout = map(k_model,
        eval.heldout, heldout$missing), residual = map(k_model,
        checkResiduals, isec_processed$documents), bound = map_dbl(k_model,
        function(x) max(x$convergence$bound)), lfact = map_dbl(k_model,
        function(x) lfactorial(x$settings$dim$K)), lbound = bound +
        lfact, iterations = map_dbl(k_model, function(x) length(x$convergence$bound)))


###################### Summary for Multiple
###################### Models#########################

k_result %>%
    transmute(K, `Lower bound` = lbound, Residuals = map_dbl(residual,
        "dispersion"), `Semantic coherence` = map_dbl(semantic_coherence,
        mean), `Held-out likelihood` = map_dbl(eval_heldout,
        "expected.heldout")) %>%
    gather(Metric, Value, -K) %>%
    ggplot(aes(K, Value, color = Metric)) + geom_line() + facet_wrap(~Metric,
    scales = "free_y") + labs(x = "K (number of topics)", y = NULL,
    title = "Model diagnostics by number of topics")
ggsave(here::here("Figures/2 Model_Diagostics.pdf"), width = 12,
    height = 10)
```

```r
# plot means for better overview
k_result %>%
    select(K, exclusivity, semantic_coherence) %>%
    # filter(K %in% c(20, 35, 30)) %>%
unnest() %>%
    mutate(K = as.factor(K)) %>%
    group_by(K) %>%
    summarize(exclusivity = mean(exclusivity), semantic_coherence = mean(semantic_coherence)) %>%
    ggplot(aes(x = semantic_coherence, y = exclusivity, color = as.factor(K),
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + label
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + =
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + K))
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + +
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + geom_point()
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + +
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + #geom_text(nudge_y
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + =
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + .01,
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + size=2.5)
        label = K)) + geom_point() + #geom_text(nudge_y = .01, size=2.5) + +
geom_text_repel() + theme_bw() + labs(x = "Semantic coherence",
    y = "Exclusivity", title = "Exclusivity vs semantic coherence (Means)")
ggsave(here::here("Figures/2 Mean Exclusivity Vs. Semantic Coherence.pdf"),
    width = 12, height = 10)
```


Model diagnostics by number of topics
Held−out likelihood
Lower bound
Residuals
Semantic coherence
−5.4
−5.5
−5.6
−5.7
−930000
−940000
−950000
−960000
1.5
1.4
1.3
1.2
1.1
−60
−70
−80
10
20
30
40
K (number of topics)
Metric
Held−out likelihood
Lower bound
Residuals
Semantic coherence



After considering a range of models ranging from 1 to 45 topics, we determined a 15-topic model as the most helpful for our analysis of the research covered in the ISEC abstracts.

Profile of the 15 Topics

We used the code chunk below to profile the topics by looking at the words in each topic with the highest likelihood (Probability) and FREX (frequent and exclusive) words - those that are both common in a topic and unique to the topic. We only present the top 10 words for each of the 15 selected topics based on their FREX score to provide insights into the key themes within the dataset.

```
### Profile 15 Topic Model
### ##################################################################


########################## Extract objects for single k
########################## value ###############
########################## ------------------------------------------------------------------------------
k_model_15 <- k_result %>%
    filter(K == 15) %>%
    pull(k_model) %>%
    .[[1]]


sink(here::here("Figures/2. 1 Model_with_15_Topics_STM_Summary.txt"))
summary(k_model_15)
sink()


########## Extract Prob, Frex into a data frames
########## ###############
########## ------------------------------------------------------------------------------


labels <- dput(paste0(c("Topic "), rep(1:15, each = 1)))

prob <- labelTopics(k_model_15, n = 10)$prob
frex <- labelTopics(k_model_15, n = 10)$frex

prob_frex <- data.frame(labels)
names(prob_frex) <- "topic"
prob_frex$Probability <- apply(prob, 1, paste, collapse = ", ")
prob_frex$FREX <- apply(frex, 1, paste, collapse = ", ")

td_beta <- tidy(k_model_15)
td_gamma <- tidy(k_model_15, matrix = "gamma", document_names = rownames(isec_processed))


gamma_terms <- td_gamma %>%
    group_by(topic) %>%
    summarise(gamma = mean(gamma)) %>%
    arrange(desc(gamma)) %>%
    mutate(topic = paste0("Topic ", topic), topic = reorder(topic,
        gamma)) %>%
    mutate(topic_title = as.character(topic))

rm(all_docs)

all_docs <- gamma_terms %>%
```

```r
    # top_n(15, gamma) %>%
left_join(as.data.frame(prob_frex), by = c("topic")) %>%
    as.data.frame() %>%
    mutate(prob_frexs = paste0("Probability: ", Probability,
        "<br>", "FREX: ", FREX)) %>%
    # mutate(prob_frexs = str_replace_all(prob_frexs, '|',
    # '<br>')) %>%
mutate(topic = reorder(topic, gamma))


all_docs %>%
    dplyr::select(1, 2, 6) %>%
    mutate(topic = factor(topic, levels = c("Topic 5", "Topic 4",
        "Topic 6", "Topic 9", "Topic 1", "Topic 3", "Topic 2",
        "Topic 10", "Topic 8", "Topic 13", "Topic 7", "Topic 12",
        "Topic 15", "Topic 11", "Topic 14"), labels = c("Topic 1",
        "Topic 2", "Topic 3", "Topic 4", "Topic 5", "Topic 6",
        "Topic 7", "Topic 8", "Topic 9", "Topic 10", "Topic 11",
        "Topic 12", "Topic 13", "Topic 14", "Topic 15"))) %>%
    gt::gt() %>%
    fmt_markdown(columns = everything()) %>%
    fmt_percent(columns = gamma, decimals = 1) %>%
    tab_options(heading.align = "left", column_labels.font.weight = "bold") %>%
    cols_width(c(1, 2) ~ px(70)) %>%
    cols_align(align = c("right"), columns = 2) %>%
    cols_label(topic = "Topic", gamma = "Percent", prob_frexs = "Highest Probability and Frex Words") %
    gtsave(here::here("Figures/15Topic_Frex_and_prob.pdf"))
```

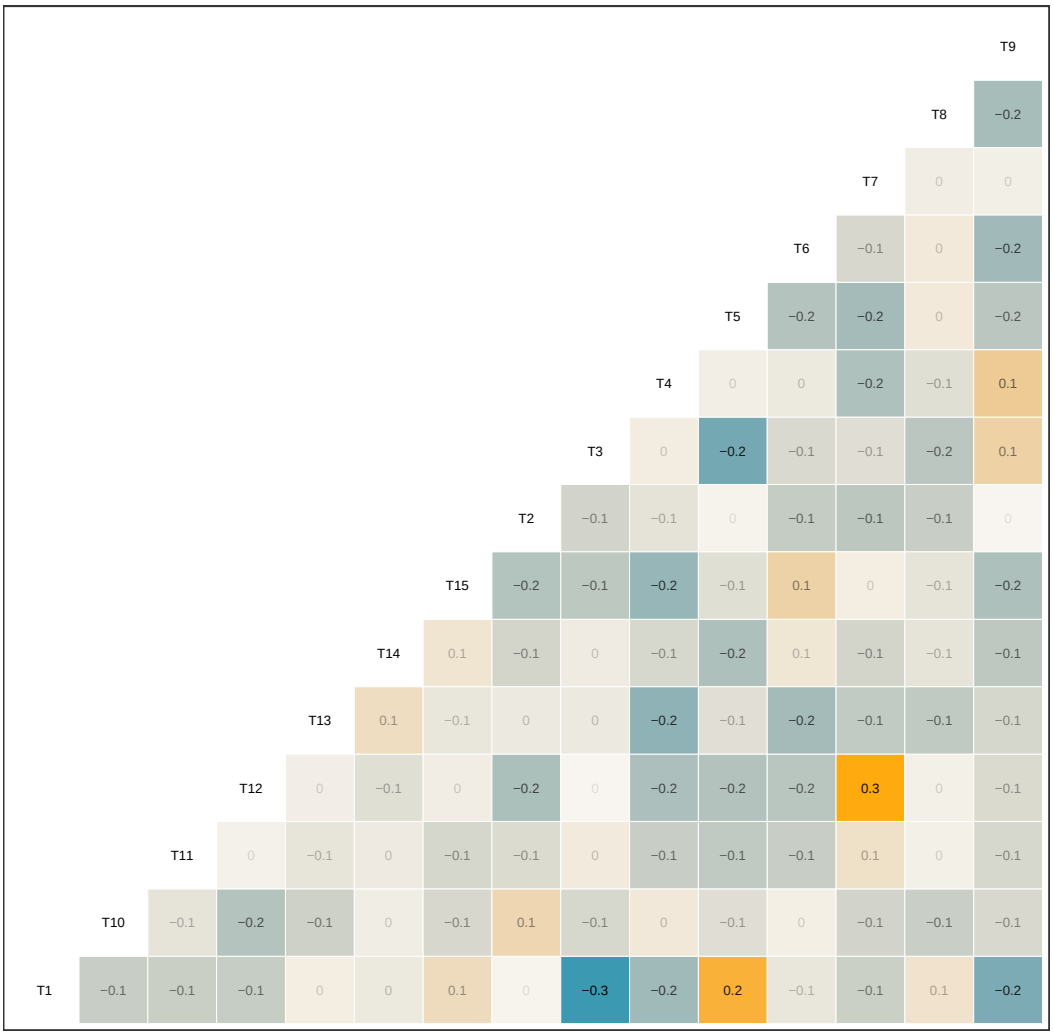
Topic	Percent	Highest Probability and Frex Words
Topic 1	12.2%	Probability: model, fit, distribut, data, covari, process, predict, bayesian, variabl, ecolog FREX: fit, covari, model, latent, flexibl, randomeffect, hierarch, posterior, distribut, linear
Topic 2	8.4%	Probability: speci, communiti, distribut, interact, divers, occup, bird, environment, predict, model FREX: speci, communiti, divers, interact, occup, rich, occurr, multispeci, distribut, trait
Topic 3	7.8%	Probability: sampl, detect, survey, estim, abund, distanc, design, site, probabl, observ FREX: sampl, survey, distanc, detect, transect, design, abund, site, line, effort
Topic 4	7.8%	Probability: spatial, chang, forest, dynam, landscap, scale, process, model, plant, pattern FREX: forest, landscap, plant, tree, dispers, land, chang, diseas, cover, scale
Topic 5	7.7%	Probability: method, data, likelihood, estim, paramet, simul, comput, analysi, exampl, illustr FREX: likelihood, method, comput, approxim, real, illustr, exampl, maximum, mcmc, miss
Topic 6	7.4%	Probability: habitat, variabl, effect, season, environment, site, area, water, year, temperatur FREX: habitat, season, water, temperatur, predat, variabl, prey, river, period, factor
Topic 7	7.3%	Probability: statist, ecolog, manag, network, test, develop, tool, research, packag, analysi FREX: network, statist, learn, ecolog, packag, tool, test, softwar, decis, ecologist
Topic 8	6.9%	Probability: data, inform, collect, monitor, integr, dataset, sourc, differ, process, citizensci FREX: sourc, citizensci, collect, map, project, data, monitor, program, integr, dataset
Topic 9	5.9%	Probability: estim, error, measur, rate, bias, model, uncertainti, growth, paramet, simul FREX: error, interv, measur, bias, uncertainti, negat, binomi, growth, proport, confid
Topic 10	5.8%	Probability: movement, anim, behaviour, state, time, model, behavior, data, forag, observ

Profile the 15 Topics - Topic Correlations

The code chunck below was used to generate the topic correlation plot to visualise the relationships between topics.

Note: To match the following plots to the text, please apply the following mapping: Topic 5 = Topic 1, Topic 4 = Topic 2, Topic 6 = Topic 3, Topic 9 = Topic 4, Topic 1 = Topic 5, Topic 3 = Topic 6, Topic 2 = Topic 7, Topic 10 = Topic 8, Topic 8 = Topic 9, Topic 13 = Topic 10, Topic 7 = Topic 11, Topic 12 = Topic 12, Topic 15 = Topic 13, Topic 11 = Topic 14, Topic 14 = Topic 15.

```
######################################## STM Correlations
######################################## ##############



pdf(here::here("Figures/2. 4 Model_with_15_Topics_Topic_Correlations_2.pdf"))
plot_corr(k_model_15)
dev.off()
```



Link Topic Examples to each Topic

To further understand the context of each topic, the code chunk below was used to link topic examples to each of the identified topics. A representative set of documents or text snippets corresponding to each topic was then retrieved as examples, helping to illustrate the types of content associated with each topic and providing a more meaningful interpretation of the topics.

```
#### Example Text
#### #------------------------------------------------------------------------

# Note: To match the following plots to the text, please
# apply the following mapping: Topic 5 = Topic 1, Topic 4 =
# Topic 2, Topic 6 = Topic 3, Topic 9 = Topic 4, Topic 1 =
# Topic 5, Topic 3 = Topic 6, Topic 2 = Topic 7, Topic 10 =
# Topic 8, Topic 8 = Topic 9, Topic 13 = Topic 10, Topic 7
# = Topic 11, Topic 12 = Topic 12, Topic 15 = Topic 13,
# Topic 11 = Topic 14, Topic 14 = Topic 15.


thoughts <- findThoughts(model = k_model_15, texts = isec_dat$abstracts,
    n = 10, topics = 1:15, thresh = 0.3)$docs
openxlsx::write.xlsx(thoughts, "example_topics.xlsx")
```

Credits

- stm package documentation
- Julia Silge, Training, evaluating, and interpreting topic models, https://juliasilge.com/blog/evaluating-stm/