

visualise

Johnson T.F.

This Rmarkdown contains all code necessary to re-produce plots and key statistics within the paper

Load packages

```
library(dplyr)
library(ggplot2)
library(ggpubr)
library(tidyr)
library(maptools)  ## For wrld_simpl
library(raster)
library(RColorBrewer)
library(ggtree)
library(ape)
library(phytools)
library(INLA)
library(brinla)
```

Descriptives

Overall summary, reporting on the number of populations, species and sites contained within the overall dataset

```
compiled_data = readRDS("../data/derived_data/compiled_data.rds")
length(unique(compiled_data$site_spec))
```

```
## [1] 30730
```

```
length(unique(compiled_data$species))
```

```
## [1] 3141
```

```
length(unique(compiled_data$site))
```

```
## [1] 5991
```

Summaries per dataset Here we report the spatial, taxonomic and temporal extent of each dataset. This information feeds directly into Table S1.

```
analysis_list = readRDS("../data/derived_data/analysis_list.rds")
descriptives_df = NULL
for(a in names(analysis_list)){
  tmp_df = analysis_list[[a]][[1]]
  tmp_df = subset(tmp_df, !is.na(log_abundance))
  tmp_descriptives_df = data.frame(
    Dataset = a,
    Observations = nrow(tmp_df),
    Populations = length(unique(tmp_df$site_spec_code)),
    Species = length(unique(tmp_df$tips_code)),
    Sites = length(unique(tmp_df$site_code)),
    Timeframe = paste0(min(tmp_df$date), " - ", max(tmp_df$date)),
    Latitude = paste0(round(min(tmp_df$latitude),1), " - ", round(max(tmp_df$latitude),1)),
    Longitude = paste0(round(min(tmp_df$longitude),1), " - ", round(max(tmp_df$longitude),1))
  )
  descriptives_df = rbind(descriptives_df, tmp_descriptives_df)
}

print(descriptives_df)
```

##	Dataset	Observations	Populations	Species	Sites	Timeframe
## 1	BioTIME	243993	12065	1233	438	1933 - 2018
## 2	CaPTrends	2670	279	26	165	1880 - 2019
## 3	EA_NFPD	2999	359	16	179	1984 - 2019
## 4	FishGlob	67908	2286	152	229	1977 - 2020
## 5	LPI	77773	3613	1333	1244	1950 - 2020

```
## 6      NAm_BBS      164317      8718      361      584 1966 - 2019
## 7      Pilotto      11353      582      356      67 1974 - 2018
## 8 ReSurvey_Germany      4954      356      93      7 1966 - 2016
## 9      RivFish      40834      2386      191      197 1975 - 2019
## 10     TimeFISH      262      86      52      12 2008 - 2022
##      Latitude      Longitude
## 1  -77.6 - 68.6 -179.8 - 179.2
## 2   -40 - 71.6   -158 - 99.2
## 3   50.4 - 55.4    -3.9 - 0.5
## 4    26 - 62    -178 - 21
## 5  -77.8 - 78.9   -179.6 - 180
## 6   25.9 - 67 -165.3 - -55.4
## 7   40.1 - 67.8    -8.9 - 29.6
## 8   49.8 - 53.1     8.6 - 13.9
## 9  -28.3 - 67.9 -122.4 - 153.4
## 10 -27.7 - -27.1  -48.5 - -48.3
```

Model fit per dataset Here we explore variance partitioning within each model

```
model_sum = read.csv("../outputs/model_summary.csv")
model_compare_df = NULL
for(a in seq(1,30,3)){
  df_tmp = model_sum[c(a:(a+2)),]
  df_tmp2 = data.frame(
    code = df_tmp$code[1],
    phy_h = df_tmp$tip_h[3]/(df_tmp$tip[3] + df_tmp$tip_h[3] + df_tmp$gen[3]), #Contribution of phylogenetic covariance, relative to all t
    spa_h = df_tmp$sit_h[3]/(df_tmp$sit[3] + df_tmp$sit_h[3] + df_tmp$squ[3]), #Contribution of spatial covariance, relative to all spatia
    temp = df_tmp$obv_auto[3]/(df_tmp$obv[3] + df_tmp$obv_auto[3]),
    temp_phi = df_tmp$phi[3], #Rho correlation between abundances
    temp_v = sum(c(df_tmp$obv[3], df_tmp$obv_auto[3]),na.rm = T)/sum(c( #Proportion of variance captured by temporal model components
      df_tmp$fix[3],
      df_tmp$obv[3],
      df_tmp$obv_auto[3],
      df_tmp$tip[3],
      df_tmp$tip_h[3],
      df_tmp$gen[3],
      df_tmp$sit[3],
      df_tmp$sit_h[3],
      df_tmp$squ[3],
```

```

    df_tmp$sig[3]
  ),na.rm = T),
  sit_v = sum(c(df_tmp$sit[3],df_tmp$sit_h[3], df_tmp$squ[3]),na.rm = T)/sum(c( #Proportion of variance captured by spatial model compon
    df_tmp$fix[3],
    df_tmp$obv[3],
    df_tmp$obv_auto[3],
    df_tmp$tip[3],
    df_tmp$tip_h[3],
    df_tmp$gen[3],
    df_tmp$sit[3],
    df_tmp$sit_h[3],
    df_tmp$squ[3],
    df_tmp$sig[3]
  ),na.rm = T),
  phy_v = sum(c(df_tmp$tip[3],df_tmp$tip_h[3], df_tmp$gen[3]),na.rm = T)/sum(c( #Proportion of variance captured by phylogentic model co
    df_tmp$fix[3],
    df_tmp$obv[3],
    df_tmp$obv_auto[3],
    df_tmp$tip[3],
    df_tmp$tip_h[3],
    df_tmp$gen[3],
    df_tmp$sit[3],
    df_tmp$sit_h[3],
    df_tmp$squ[3],
    df_tmp$sig[3]
  ),na.rm = T),
  res_v = sum(c(df_tmp$sig[3]),na.rm = T)/sum(c( #Proportion of variance captured by residuals
    df_tmp$fix[3],
    df_tmp$obv[3],
    df_tmp$obv_auto[3],
    df_tmp$tip[3],
    df_tmp$tip_h[3],
    df_tmp$gen[3],
    df_tmp$sit[3],
    df_tmp$sit_h[3],
    df_tmp$squ[3],
    df_tmp$sig[3]
  ),na.rm = T),

```

```

    fix_v = sum(c(df_tmp$fix[3]),na.rm = T)/sum(c( #Proportion of variance captured by fixed effect
      df_tmp$fix[3],
      df_tmp$obv[3],
      df_tmp$obv_auto[3],
      df_tmp$tip[3],
      df_tmp$tip_h[3],
      df_tmp$gen[3],
      df_tmp$sit[3],
      df_tmp$sit_h[3],
      df_tmp$squ[3],
      df_tmp$sig[3]
    ),na.rm = T)
  )
  model_compare_df = rbind(model_compare_df, df_tmp2)
}

```

Proportion of variance captured by temporal components

```
round(mean(model_compare_df$temp_v),2)
```

```
## [1] 0.37
```

```
round(sd(model_compare_df$temp_v),2)
```

```
## [1] 0.13
```

Proportion of variance captured by spatial components

```
round(mean(model_compare_df$sit_v, na.rm = T),2)
```

```
## [1] 0.08
```

```
round(sd(model_compare_df$sit_v, na.rm = T),2)
```

```
## [1] 0.05
```

Proportion of variance captured by phylogenetic components

```
round(mean(model_compare_df$phy_v),2)
```

```
## [1] 0.05
```

```
round(sd(model_compare_df$phy_v),2)
```

```
## [1] 0.06
```

Proportion of variance captured by residual components

```
round(mean(model_compare_df$res_v),2)
```

```
## [1] 0.24
```

```
round(sd(model_compare_df$res_v),2)
```

```
## [1] 0.24
```

Proportion of variance captured by fixed effect components

```
round(mean(model_compare_df$fix_v),2)
```

```
## [1] 0.25
```

```
round(sd(model_compare_df$fix_v),2)
```

```
## [1] 0.13
```

Correlation between sequential abundance values

```
round(mean(model_compare_df$temp_phi),2)
```

```
## [1] 0.31
```

```
round(sd(model_compare_df$temp_phi),2)
```

```
## [1] 0.42
```

H2 of phylo component i.e. the amount of variance captured by the phylogeny relative to the combined variance captured by the phylogeny and taxonomy

```
round(mean(model_compare_df$phy_h),2)
```

```
## [1] 0.41
```

```
round(sd(model_compare_df$phy_h),2)
```

```
## [1] 0.28
```

H2 of spatial components i.e. the amount of variance captured by space relative to the combined variance captured by the space and the hierarchical space nesting

```
round(mean(model_compare_df$spa_h, na.rm = T),2)
```

```
## [1] 0.34
```

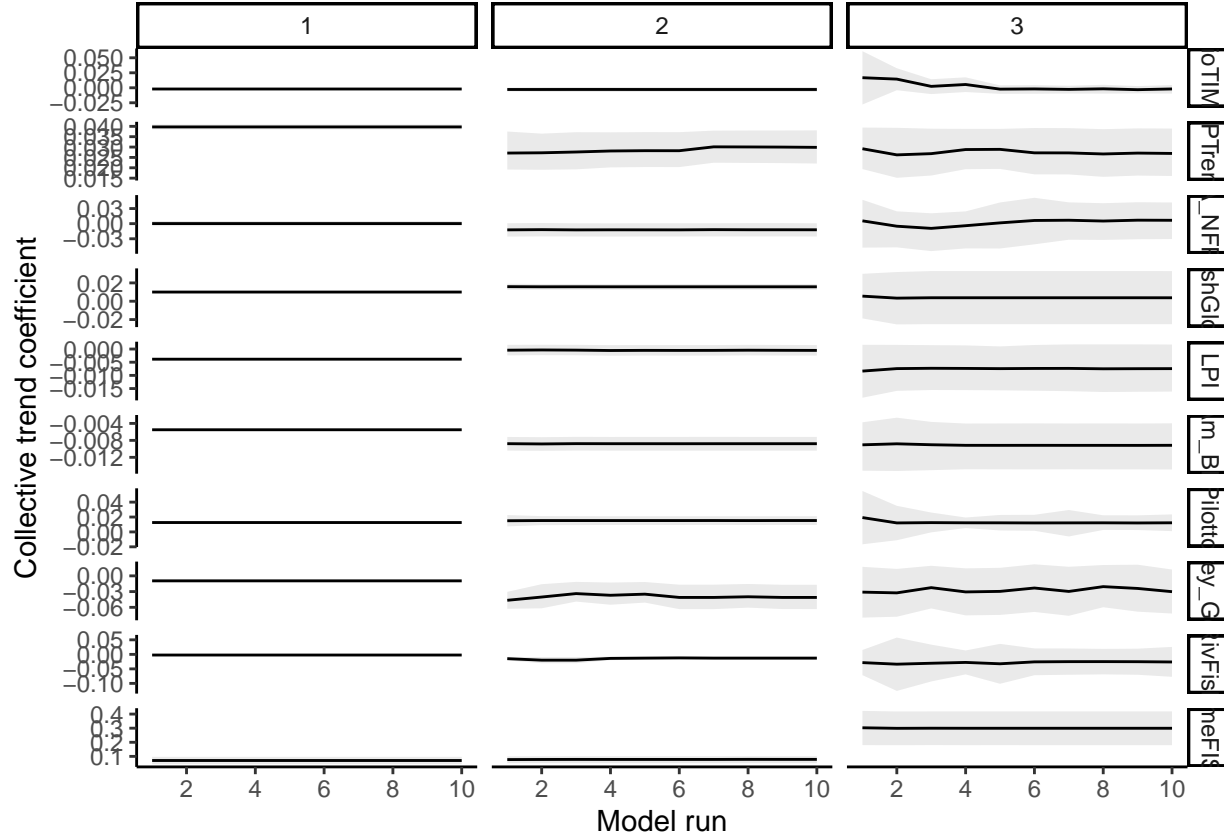
```
round(sd(model_compare_df$spa_h, na.rm = T),2)
```

```
## [1] 0.3
```

Assumptions

We first check the models have converged. Under MCMC Bayesian approaches, convergence can be checked by ensuring independent chains arrive at similar parameter values. This is not possible in INLA where parameters are approximated, but the same principle applies i.e. if the model is re-run multiple times and arrives at similar parameter values, we can be somewhat confident the results are robust. With this in mind, we re-run each model 10 times, using the parameter mode values from the prior run as the initial values in the current run. In the first run, we used the randomly generated initial values. We assess convergence by looking at the main parameter of interest, the coefficient and credible intervals around the collective trend, and it's apparent that after the first 5 runs, parameters begin to stabilise.

```
conv = read.csv("../outputs/model_summary_convergence.csv")
ggplot(conv) +
  geom_ribbon(aes(x = run, ymin = low, ymax = upp), alpha = 0.1) +
  geom_line(aes(x = run, y = coef)) +
  facet_grid(dataset~model, scales = "free") +
  scale_x_continuous(breaks = c(0,2,4,6,8,10)) +
  labs(x = "Model run", y = "Collective trend coefficient") +
  theme_classic()
```

We calculate residuals by subtracting the true abundance values by the median predicted values. We also identify populations with particularly extreme residual values ± 2 standard deviations from the mean residual in each population.

```
analysis_list = readRDS("../data/derived_data/analysis_list.rds")

assumptions_df = NULL
for(a in names(analysis_list)[c(1:10)]) {
  tmp_model = readRDS(paste0("../outputs/model_output_convergence10_", a, ".rds"))
  for(b in 1:3) {
    if(b == 1) {
      pred = tmp_model[[b]]$summary.fitted.values[,4]
    }
  }
}
```

```

    pred = head(pred,-100)
    true = analysis_list[[a]][[1]]$log_abundance
    true = head(true,-100)
    res = (true - pred)
  } else {
    pred = tmp_model[[b]]$summary.fitted.values[,4]
    pred = head(pred,-100)
    true = analysis_list[[a]][[1]]$cent_abundance
    true = head(true,-100)
    res = (true - pred)
  }
  tmp_assumptions_df = data.frame(
    dataset = a,
    model = b,
    site_spec = head(analysis_list[[a]][[1]]$site_spec,-100),
    pred = pred,
    true = true,
    res = res,
    year = head(analysis_list[[a]][[1]]$date,-100))
  res_sum = tmp_assumptions_df %>%
    group_by(site_spec) %>%
    summarise(mean_res = mean(abs(res)), low_cut = mean(res) - (sd(res)*2), high_cut = mean(res) + (sd(res)*2))
  cut = quantile(res_sum$mean_res, probs = 0.5)
  res_sum$hq = ifelse((res_sum$mean_res < cut & res_sum$mean_res > -cut), "Keep", "Remove")
  tmp_assumptions_df = left_join(tmp_assumptions_df, res_sum)
  assumptions_df = rbind(assumptions_df, tmp_assumptions_df)
}
}

dataset_rename = data.frame(
  dataset = c("BioTIME", "CaPTrends", "EA_NFPD", "FishGlob", "LPI", "NA_m_BBS", "Pilotto", "ReSurvey_Germany", "RivFish", "TimeFISH"),
  dataset_code = c("A) BioTIME", "J) Large carnivores", "F) UK riverine fishes", "D) FishGlob", "B) Living Planet", "C) NA Breeding Birds")
)
assumptions_df = left_join(assumptions_df, dataset_rename)
assumptions_df$model = as.character(assumptions_df$model)
assumptions_df$plot_code = paste0(assumptions_df$dataset_code, ": ", assumptions_df$model)

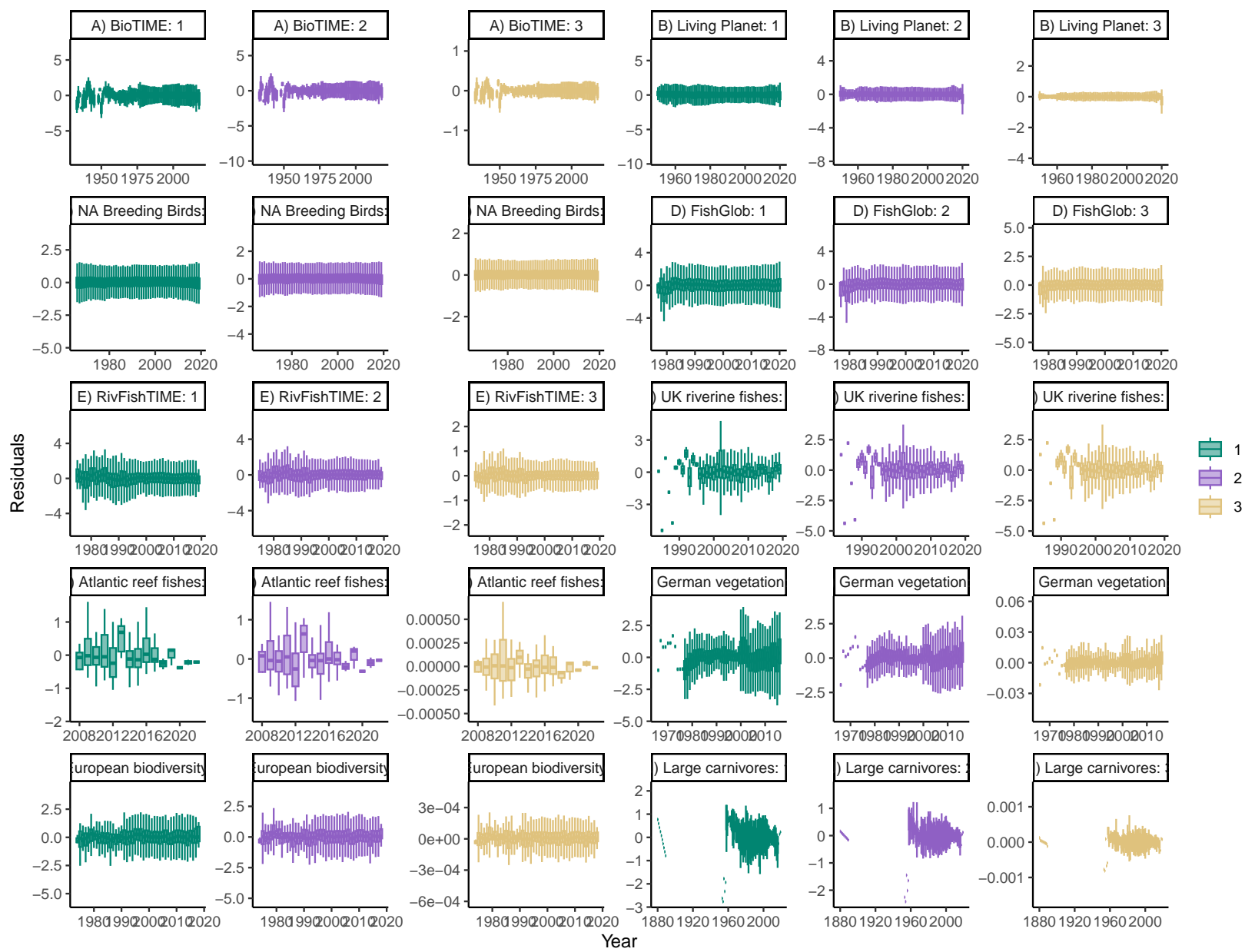
```

Plot median residuals and quantiles against year (the independent variable) to assess heteroscedasticity

```

ggplot(data = assumptions_df) +
  geom_boxplot(aes(x = year, y = res, fill = model, colour = model, group = year), alpha = 0.5, outlier.shape = NA) +
  scale_fill_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = NULL) +
  scale_colour_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = NULL) +
  facet_wrap(plot_code~., scales = "free") +
  labs(x = "Year", y = "Residuals") +
  theme_classic()

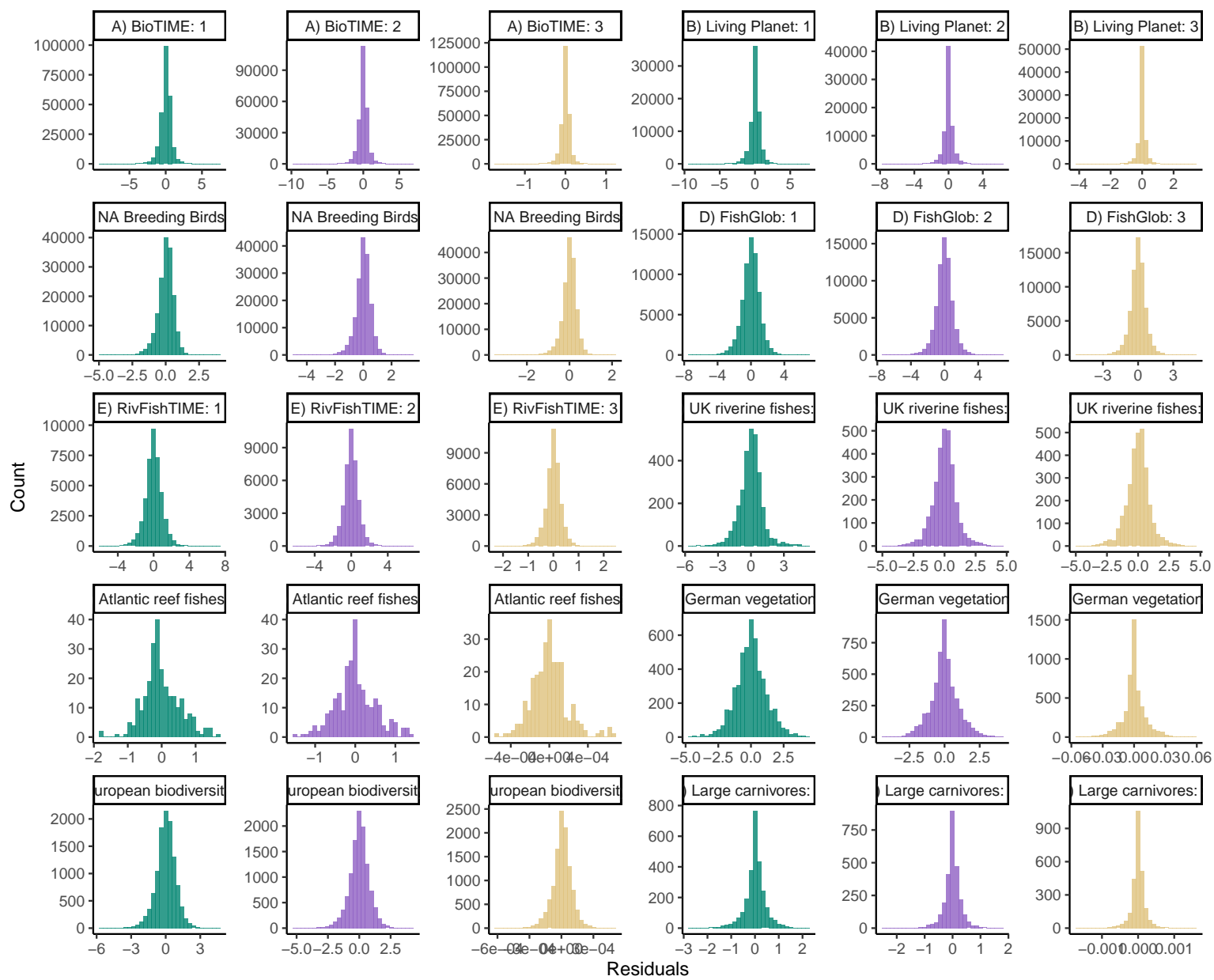
```



```
ggsave("../outputs/figures/assumption_hetero.png",width = 13, height = 7, units = "in", device = "png")
```

Histogram of residuals for each dataset and model combinations to check for normality

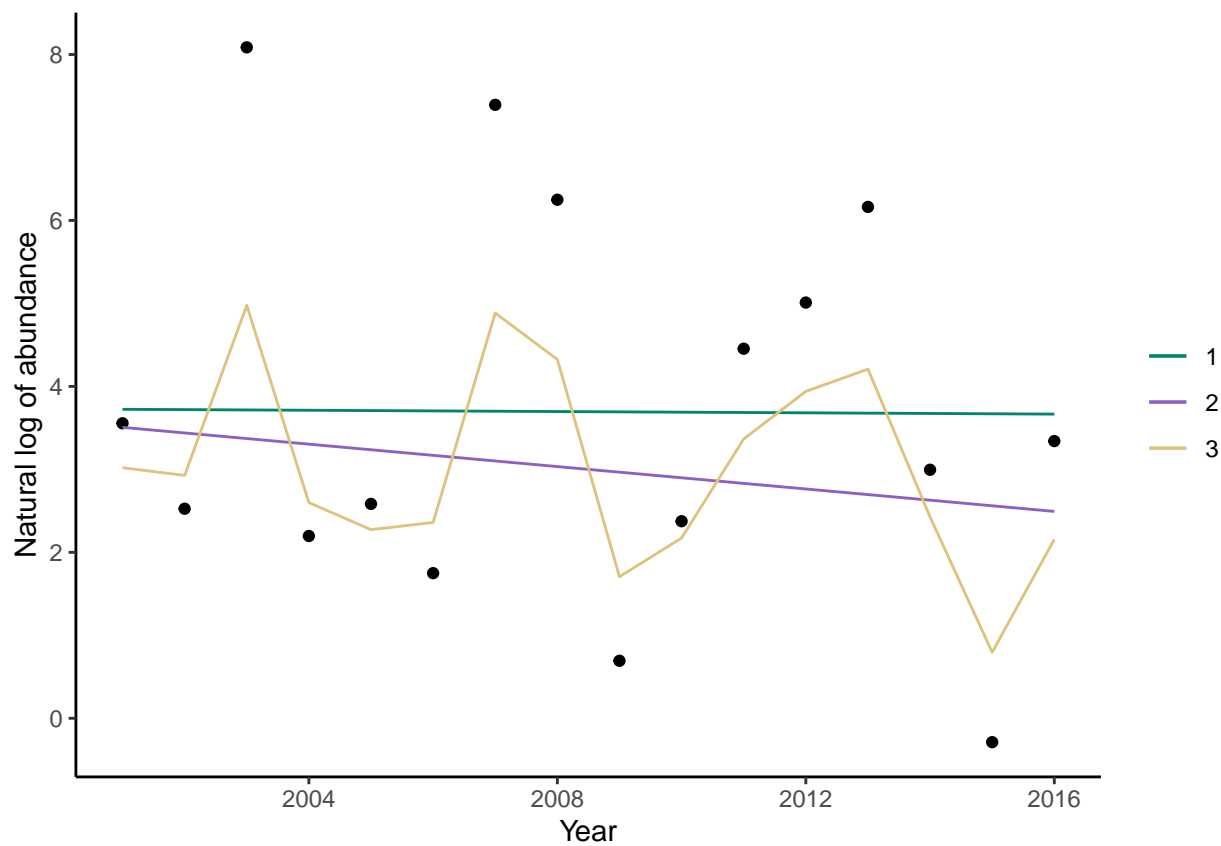
```
ggplot(data = assumptions_df) +  
  geom_histogram(aes(x = res, fill = model), alpha = 0.8, position = "identity") +  
  scale_fill_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = NULL) +  
  facet_wrap(plot_code~., scales = "free") +  
  labs(x = "Residuals", y = "Count") +  
  theme_classic()
```



```
ggsave("../outputs/figures/assumption_normality1.png",width = 13, height = 7, units = "in", device = "png")
```

In some datasets, we see patterns of very heavy tails. These heavy tails tend to happen under two circumstances. 1) Example of population where the model poorly fits due to cyclical and highly variable abundance data

```
ggplot() +  
  geom_point(data = assumptions_df[which(  
    assumptions_df$site_spec == "Chen_2018_Yehliu power plant (site 2a)_25.209615_121.661948Siganus fuscescens_" & assumptions_df$model ==  
  geom_line(data = assumptions_df[which(  
    assumptions_df$site_spec == "Chen_2018_Yehliu power plant (site 2a)_25.209615_121.661948Siganus fuscescens_" & assumptions_df$model ==  
  geom_line(data = assumptions_df[which(  
    assumptions_df$site_spec == "Chen_2018_Yehliu power plant (site 2a)_25.209615_121.661948Siganus fuscescens_" & assumptions_df$model ==  
  geom_line(data = assumptions_df[which(  
    assumptions_df$site_spec == "Chen_2018_Yehliu power plant (site 2a)_25.209615_121.661948Siganus fuscescens_" & assumptions_df$model ==  
  scale_colour_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = NULL) +  
  labs(x = "Year", y = "Natural log of abundance") +  
  theme_classic()
```



```
ggsave("../outputs/figures/assumption_poorfit1.png",width = 4, height = 4, units = "in", device = "png")
```

2) Example of population with generally well fitting data, but just one extreme abundance

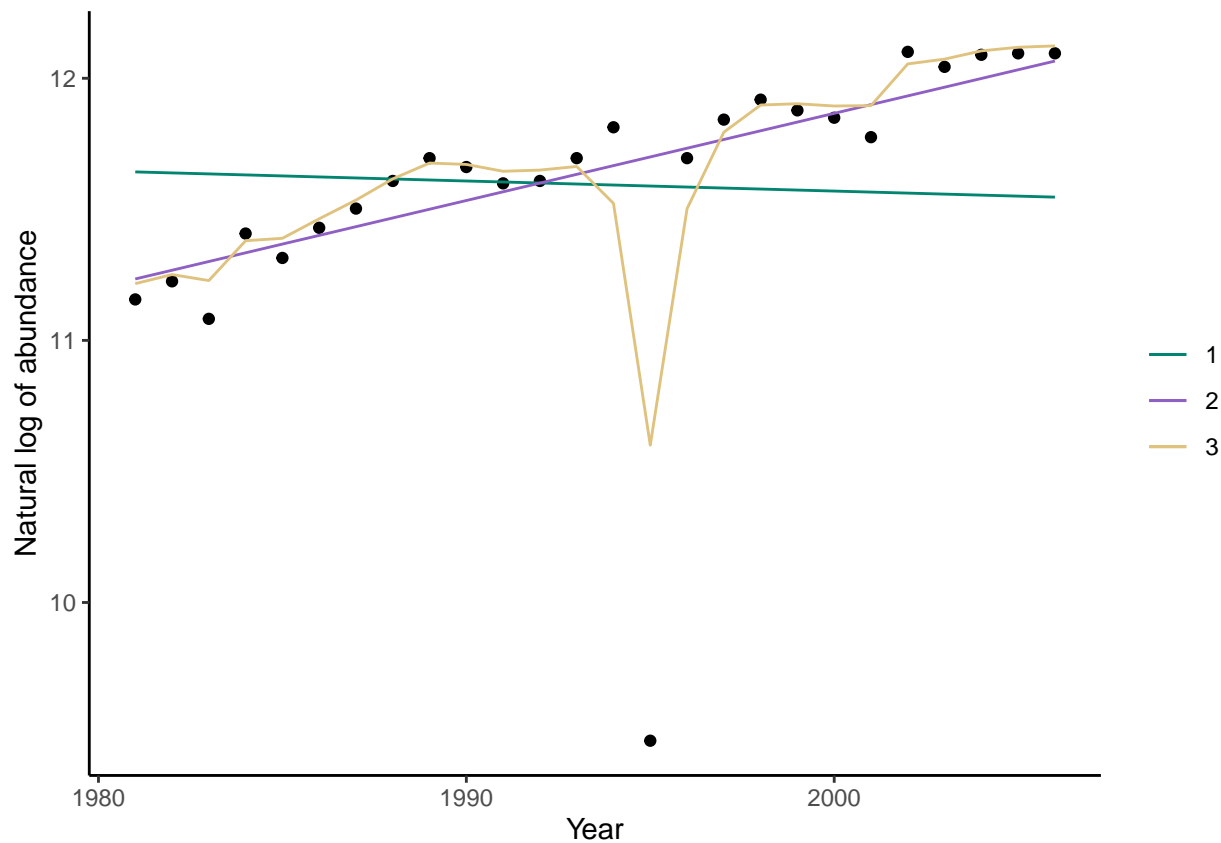
```
ggplot() +
  geom_point(data = assumptions_df[which(
    assumptions_df$site_spec == "Rolley_2006_Western Forest, Wisconsin, USA_44.77778_-91.994720docoileus virginianus_" & assumptions_df$mo
  )]) +
  geom_line(data = assumptions_df[which(
    assumptions_df$site_spec == "Rolley_2006_Western Forest, Wisconsin, USA_44.77778_-91.994720docoileus virginianus_" & assumptions_df$mo
  )]) +
  geom_line(data = assumptions_df[which(
    assumptions_df$site_spec == "Rolley_2006_Western Forest, Wisconsin, USA_44.77778_-91.994720docoileus virginianus_" & assumptions_df$mo
  )])
```



```

assumptions_df$site_spec == "Rolley_2006_Western Forest, Wisconsin, USA_44.77778_-91.994720docoileus virginianus_" & assumptions_df$mo
geom_line(data = assumptions_df[which(
  assumptions_df$site_spec == "Rolley_2006_Western Forest, Wisconsin, USA_44.77778_-91.994720docoileus virginianus_" & assumptions_df$mo
scale_colour_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = NULL) +
labs(x = "Year", y = "Natural log of abundance") +
theme_classic()

```



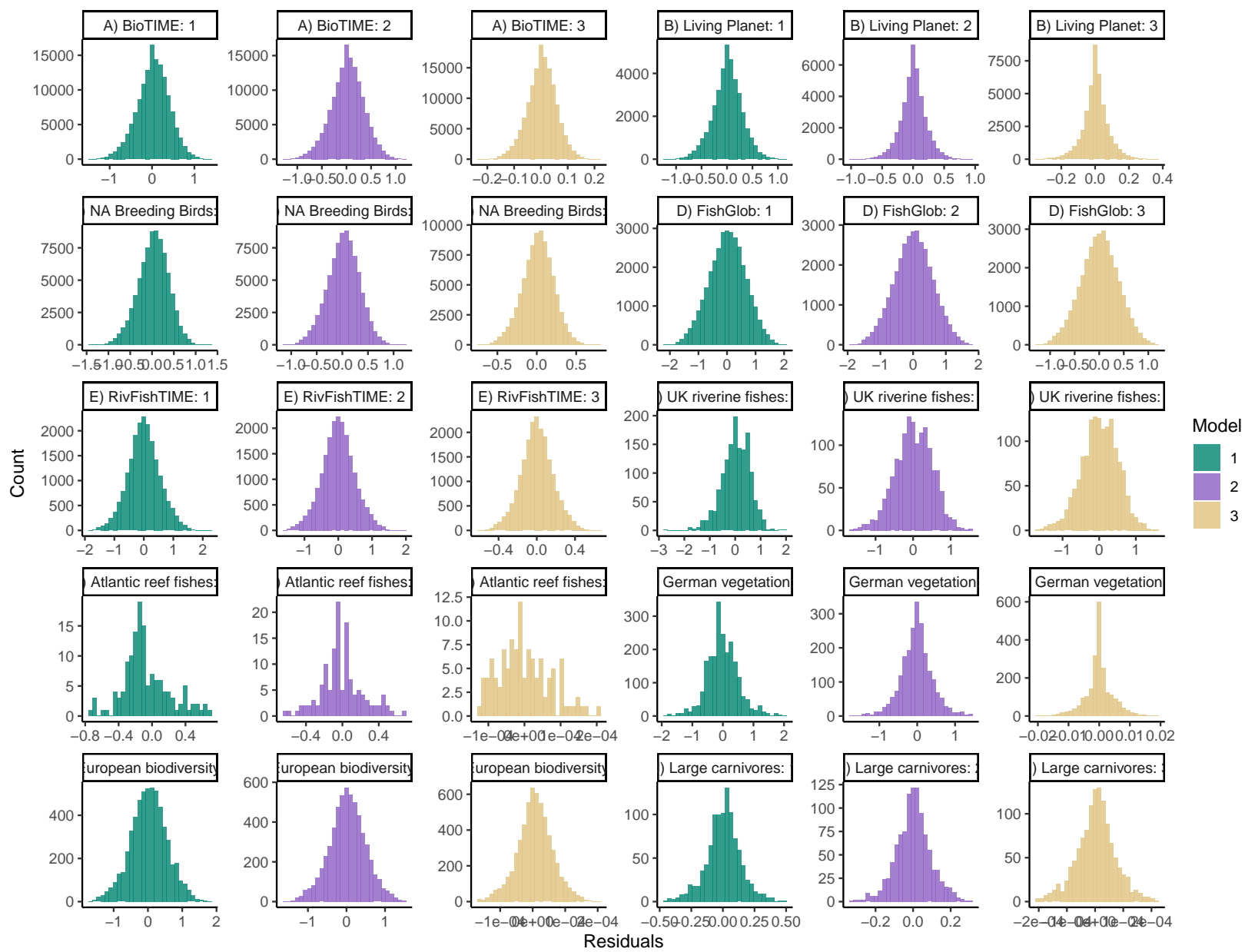
```

ggsave("../outputs/figures/assumption_poorfit2.png",width = 4, height = 4, units = "in", device = "png")

```

When we check the residuals after removing poorly fitting populations and extreme values, the residuals are looking far more normal

```
ggplot(data = assumptions_df[which(assumptions_df$hq == "Keep" & assumptions_df$res > assumptions_df$low_cut & assumptions_df$res < assumpt
  geom_histogram(aes(x = res, fill = model), alpha = 0.8, position = "identity") +
  scale_fill_manual(values = c("#018571", "#8f60c4", "#dfc27d"), name = "Model") +
  facet_wrap(plot_code~., scales = "free") +
  labs(x = "Residuals", y = "Count") +
  theme_classic()
```



```
ggsave("../outputs/figures/assumption_normality2.png", width = 13, height = 7, units = "in", device = "png")
```

To ensure our inference is robust to heavy tails, we remove extreme values from the dataset with the heaviest tails (LPI) and re-model

```
lpi_trim = subset(assumptions_df, dataset == "LPI" & model == "3")
lpi_sub = head(analysis_list[["LPI"]][[1]], -100)
lpi_sub = cbind(lpi_sub, lpi_trim[,c(6,9,10,11)])
lpi_sub$log_abundance = ifelse(lpi_sub$res > lpi_sub$high_cut, NA, lpi_sub$log_abundance)
lpi_sub$log_abundance = ifelse(lpi_sub$res < lpi_sub$low_cut, NA, lpi_sub$log_abundance)
lpi_sub$log_abundance = ifelse(lpi_sub$hq == "Remove", NA, lpi_sub$log_abundance)
lpi_sub$cent_abundance = ifelse(lpi_sub$res > lpi_sub$high_cut, NA, lpi_sub$cent_abundance)
lpi_sub$cent_abundance = ifelse(lpi_sub$res < lpi_sub$low_cut, NA, lpi_sub$cent_abundance)
lpi_sub$cent_abundance = ifelse(lpi_sub$hq == "Remove", NA, lpi_sub$cent_abundance)

analysis_list_lpi = list(lpi_sub, analysis_list[["LPI"]][[2]], analysis_list[["LPI"]][[3]])
#source("modelling_lpi_assumption.R")
```

We detect the same pattern of an increasing standard deviation (from m1 to m2 to m3) around the collective trend. Inference appears robust

```
lpi_models = readRDS("../outputs/model_output_lpi_assumption.rds")
data.frame(
  model = c(1,2,3),
  uncertainty = c(
    lpi_models[[1]]$summary.fixed[2,2],
    lpi_models[[2]]$summary.fixed[2,2],
    lpi_models[[3]]$summary.fixed[2,2]))
```

```
##  model  uncertainty
## 1      1 0.0002001832
## 2      2 0.0047850804
## 3      3 0.0499042094
```

Figure 2

Load data from 'modelling core'

```
model_summary_core = read.csv("../outputs/model_summary.csv")
```

Assess average change in uncertainty and coefficient as you progress through the models

```
dataset_rename = data.frame(
  code = c("BioTIME", "CaPTrends", "EA_NFPD", "FishGlob", "LPI", "NA_m_BBS", "Pilotto", "ReSurvey_Germany", "RivFish", "TimeFISH"),
  dataset_code = c("A) BioTIME", "J) Large carnivores", "F) UK riverine fishes", "D) FishGlob", "B) Living Planet", "C) NA Breeding Birds")
model_summary_core = left_join(model_summary_core, dataset_rename)
```

```
stre = data.frame(
  name = c(rep("2",10), rep("1",10)),
  diff_rate = c((model_summary_core[which(model_summary_core$model == 1),"coef"]) - (model_summary_core[which(model_summary_core$model == 2),"coef"]) - (model_summary_core[which(model_summary_core$model == 3),"coef"])),
  diff_sd = c((model_summary_core[which(model_summary_core$model == 1),"coef_sd"])/(model_summary_core[which(model_summary_core$model == 2),"coef_sd"]) - (model_summary_core[which(model_summary_core$model == 3),"coef_sd"])/(model_summary_core[which(model_summary_core$model == 2),"coef_sd"])),
)
```

```
stre_sum = stre %>%
  dplyr::group_by(name) %>%
  summarise(mn1 = mean(abs(diff_rate)), sd1 = sd(abs(diff_rate)), mn2 = mean(log10(diff_sd)), sd2 = sd(log10(diff_sd)))
```

```
m1_col = "#018571"
m2_col = "#8f60c4"
m3_col = "#dfc27d"
```

```
tmp_mod = lm(log(diff_sd) ~ 0 + name, data = stre)
exp(coef(tmp_mod))
```

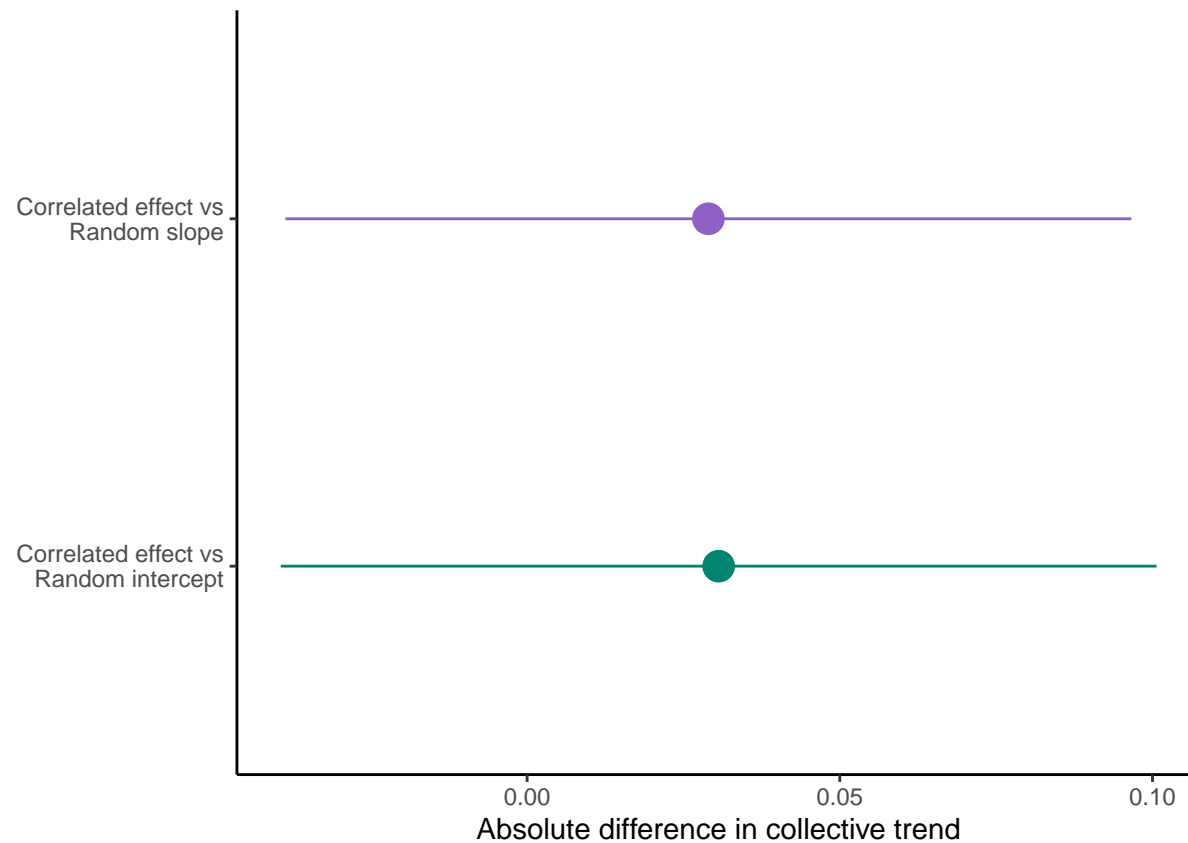
```
##      name1      name2
## 3.38590 25.71641
```

```
exp(confint(tmp_mod))
```

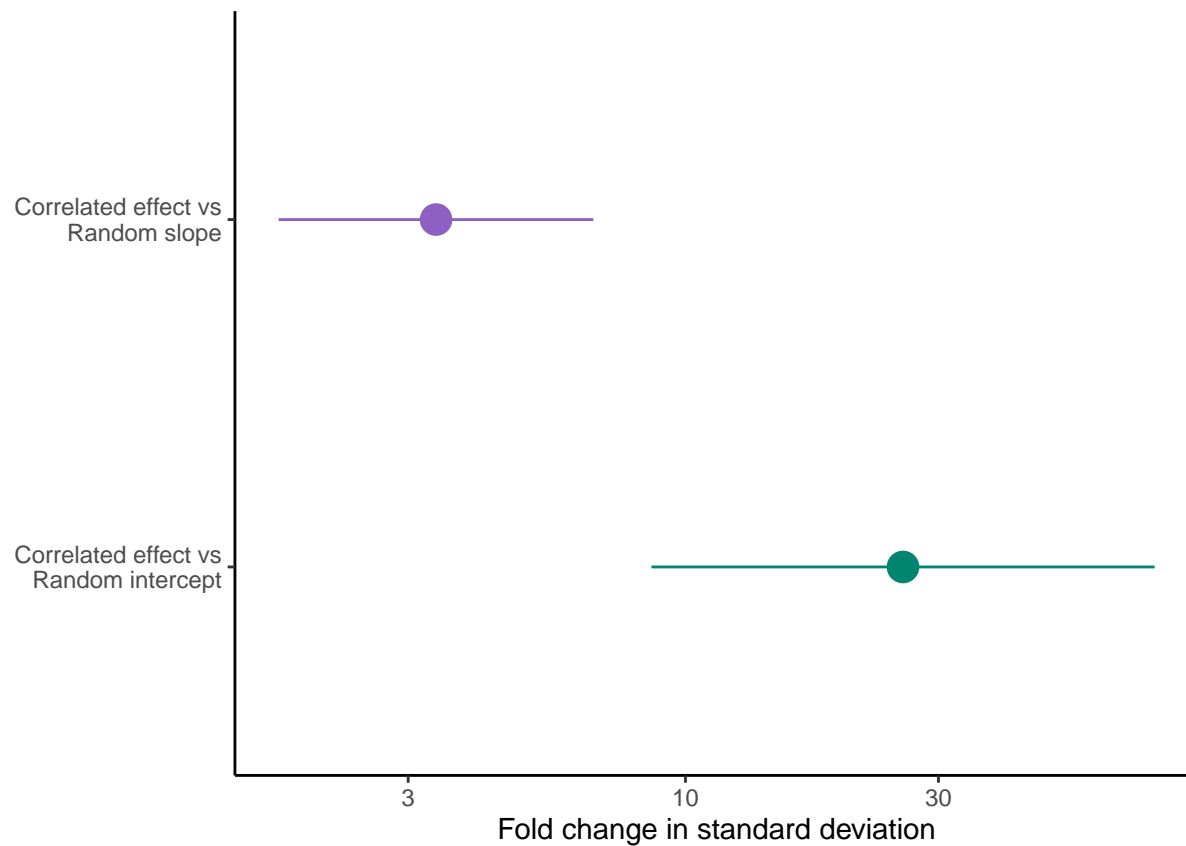
```
##           2.5 %      97.5 %
```

```
## name1 1.848509 6.201928  
## name2 14.039699 47.104571
```

```
ggplot() +  
  geom_pointrange(data = stre_sum, aes(y = rev(name), xmin = mn1 - sd1, x = mn1, xmax = mn1 + sd1, colour = name), size = 1.1) +  
  scale_y_discrete(labels=c("1" = "Correlated effect vs\nRandom intercept",  
                           "2" = "Correlated effect vs\nRandom slope")) +  
  scale_colour_manual(values = c(m2_col, m1_col), guide = "none") +  
  labs(x = "Absolute difference in collective trend", y = "") +  
  theme_classic()
```



```
ggplot() +
  geom_pointrange(data = stre_sum, aes(y = rev(name), xmin = 10^(mn2 - sd2), x = 10^(mn2), xmax = 10^(mn2 + sd2), colour = name), size = 1)
  scale_y_discrete(labels=c("1" = "Correlated effect vs\nRandom intercept",
                           "2" = "Correlated effect vs\nRandom slope")) +
  scale_x_log10() +
  #coord_cartesian(xlim = c(0, 0.025)) +
  scale_colour_manual(values = c(m2_col, m1_col), guide = "none") +
  labs(x = "Fold change in standard deviation", y = "") +
  theme_classic()
```



Next, we use the coefficients and 50% credible intervals from each model, against a baseline abundance of 100, and project abundance patterns for

each abundance dataset.

```
projections_cmb = NULL
for(a in c(1:10)){
  cde = unique(model_summary_core$code)[a]
  df_tmp = subset(model_summary_core, code == cde)
  df_tmp2 = analysis_list[[cde]][[1]]

  run = length(min(df_tmp2$date, na.rm = T):max(df_tmp2$date, na.rm = T))-1
  y1 = c(100)
  y2 = c(100)
  y3 = c(100)
  for(tim in 1:(run)){
    y1 = c(y1,y1[tim]*(1 + df_tmp$coef[1]))
    y2 = c(y2,y2[tim]*(1 + df_tmp$coef[2]))
    y3 = c(y3,y3[tim]*(1 + df_tmp$coef[3]))
  }

  lc1 = c(100)
  lc2 = c(100)
  lc3 = c(100)
  for(tim in 1:(run)){
    lc1 = c(lc1,lc1[tim]*(1 + df_tmp$coef_lc5[1]))
    lc2 = c(lc2,lc2[tim]*(1 + df_tmp$coef_lc5[2]))
    lc3 = c(lc3,lc3[tim]*(1 + df_tmp$coef_lc5[3]))
  }

  uc1 = c(100)
  uc2 = c(100)
  uc3 = c(100)
  for(tim in 1:(run)){
    uc1 = c(uc1,uc1[tim]*(1 + df_tmp$coef_uc5[1]))
    uc2 = c(uc2,uc2[tim]*(1 + df_tmp$coef_uc5[2]))
    uc3 = c(uc3,uc3[tim]*(1 + df_tmp$coef_uc5[3]))
  }

  projections_tmp = data.frame(
```



```

code = df_tmp$dataset_code[1],
year = c(min(df_tmp2$date, na.rm = T):max(df_tmp2$date, na.rm = T)),
y1 = y1, y2 = y2, y3 = y3,
lc1 = lc1, lc2 = lc2, lc3 = lc3,
uc1 = uc1, uc2 = uc2, uc3 = uc3)
projections_cmb = rbind(projections_cmb, projections_tmp)
}

plt1 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[1],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour = "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1950,1980,2010)) +
  scale_y_log10(expand = c(0,0), breaks = c(50,75,100,125)) +
  coord_cartesian(xlim = c(1948,2016), ylim = c(45,130)) +
  labs(x = " ", y = "Abundance") +
  theme_classic() +
  theme(axis.text = element_text(size=12))

plt2 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[5],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour = "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1960,1985,2010)) +
  scale_y_log10(expand = c(0,0), breaks = c(50,75,100,125)) +
  coord_cartesian(ylim = c(45,130)) +
  labs(x = " ", y = "") +

```

```

theme_classic() +
theme(axis.title.y=element_blank(),
      axis.text = element_text(size=12))

plt3 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[6],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1970,1995,2020), limits = c(1966, 2023)) +
  scale_y_log10(expand = c(0,0), breaks = c(50,75,100,125)) +
  coord_cartesian(ylim = c(45,130)) +
  labs(x = " ", y = "") +
  theme_classic() +
  theme(axis.title.y=element_blank(),
        axis.text = element_text(size=12))

plt4 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[4],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1980,1995,2010)) +
  scale_y_log10(expand = c(0,0), breaks = c(50,100,200)) +
  coord_cartesian(ylim = c(40,300)) +
  labs(x = " ", y = "") +
  theme_classic() +
  theme(axis.title.y=element_blank(),
        axis.text = element_text(size=12))

```

```

plt5 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[9],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1985,2000,2015)) +
  scale_y_log10(expand = c(0,0), breaks = c(20,100,500)) +
  coord_cartesian(xlim = c(1980,2018), ylim = c(10,550)) +
  labs(x = " ", y = "") +
  theme_classic() +
  theme(axis.title.y=element_blank(),
        axis.text = element_text(size=12))

plt6 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[3],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
  geom_line(aes(x = year, y2), colour = m2_col) +
  geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
  geom_line(aes(x = year, y3), colour = m3_col) +
  geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
  geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
  scale_x_continuous(expand = c(0,0), breaks = c(1985,2000,2015)) +
  scale_y_log10(expand = c(0,0), breaks = c(50,100,200,400)) +
  coord_cartesian(xlim = c(1984,2020), ylim = c(40,500)) +
  labs(x = " ", y = "Abundance") +
  theme_classic() +
  theme(axis.text = element_text(size=12))

plt7 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[10],]) +
  geom_line(aes(x = year, y1), colour = m1_col) +
  geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +

```

```

geom_line(aes(x = year, y2), colour = m2_col) +
geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
geom_line(aes(x = year, y3), colour = m3_col) +
geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
geom_hline(aes(yintercept = 100), colour = "grey", linetype = "dotted") +
scale_x_continuous(expand = c(0,0), breaks = c(2010,2014,2018)) +
scale_y_log10(expand = c(0,0), breaks = c(100,200,400)) +
coord_cartesian(xlim = c(2008,2020), ylim = c(70,600)) +
labs(x = " ", y = "") +
theme_classic() +
theme(axis.title.y=element_blank(),
      axis.text = element_text(size=12))

plt8 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[8],]) +
geom_line(aes(x = year, y1), colour = m1_col) +
geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
geom_line(aes(x = year, y2), colour = m2_col) +
geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
geom_line(aes(x = year, y3), colour = m3_col) +
geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
geom_hline(aes(yintercept = 100), colour = "grey", linetype = "dotted") +
scale_x_continuous(expand = c(0,0), breaks = c(1980,1995,2010), limits = c(1968, 2018)) +
scale_y_log10(expand = c(0,0), breaks = c(20,100,500)) +
coord_cartesian(ylim = c(10,600)) +
labs(x = " ", y = "") +
theme_classic() +
theme(axis.title.y=element_blank(),
      axis.text = element_text(size=12))

plt9 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[7],]) +
geom_line(aes(x = year, y1), colour = m1_col) +
geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
geom_line(aes(x = year, y2), colour = m2_col) +
geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
geom_line(aes(x = year, y3), colour = m3_col) +
geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +

```

```

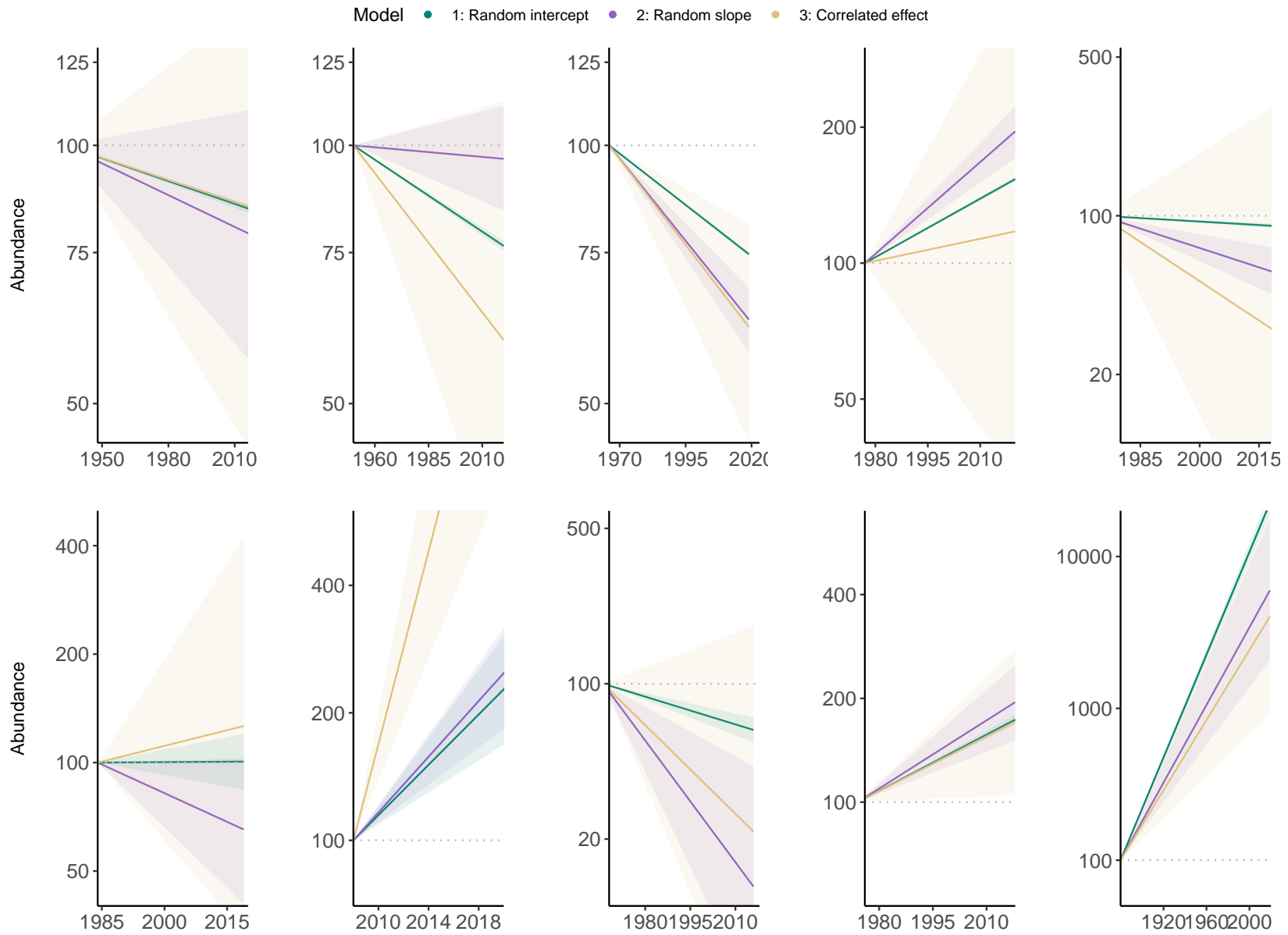
geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
scale_x_continuous(expand = c(0,0), breaks = c(1980,1995,2010), limits = c(1976, 2018)) +
scale_y_log10(expand = c(0,0), breaks = c(100,200,400)) +
coord_cartesian(ylim = c(50,700)) +
labs(x = " ", y = "") +
theme_classic() +
theme(axis.title.y=element_blank(),
      axis.text = element_text(size=12))

legend_creator = data.frame(
  id = c("1","2","3"),
  x = 1000,
  y = 100
)

plt10 = ggplot(data = projections_cmb[projections_cmb$code %in% unique(projections_cmb$code)[2],]) +
geom_line(aes(x = year, y1), colour = m1_col) +
geom_ribbon(aes(x = year, ymin = lc1, ymax = uc1), alpha = 0.1, fill = m1_col) +
geom_line(aes(x = year, y2), colour = m2_col) +
geom_ribbon(aes(x = year, ymin = lc2, ymax = uc2), alpha = 0.1, fill = m2_col) +
geom_line(aes(x = year, y3), colour = m3_col) +
geom_ribbon(aes(x = year, ymin = lc3, ymax = uc3), alpha = 0.1, fill = m3_col) +
geom_hline(aes(yintercept = 100), colour= "grey", linetype = "dotted") +
geom_point(data = legend_creator, aes(x = x, y = y, colour = id)) +
scale_x_continuous(expand = c(0,0), breaks = c(1920,1960,2000), limits = c(1880,2020)) +
scale_y_log10(expand = c(0,0), breaks = c(100,1000,10000)) +
coord_cartesian(ylim = c(50,20000)) +
labs(x = " ", y = "") +
theme_classic() +
scale_colour_manual(name = "Model", labels = c(
  "1: Random intercept",
  "2: Random slope",
  "3: Correlated effect"
), values = c(
  m1_col,
  m2_col,
  m3_col
)) +

```

```
theme_classic() +  
theme(axis.title.y=element_blank(),  
      axis.text = element_text(size=12),  
      legend.position = "bottom")  
  
ggarrange(plt1, plt2, plt3, plt4, plt5, plt6, plt7, plt8, plt9, plt10, ncol = 5, nrow = 2, common.legend = T, legend = "top", align = "hv")
```



```
write.csv(projections_cmb, "../data/derived_data/fig2.csv")
```

#Figure 3

Here we plot the abundance observations and associated predictions for a series of species in a specific site. We plot this data under each model to show how each model fits to the data

```
example_dat = readRDS("../outputs/model_output_convergence10_LPI.rds")
df = readRDS("../data/derived_data/analysis_list.rds")[[5]][[1]]

pop_level = data.frame(id = rownames(example_dat[[1]]$summary.fitted.values))
pop_level$m1_pred = example_dat[[1]]$summary.fitted.values$`0.5quant`
pop_level$m1_pred_lc = example_dat[[1]]$summary.fitted.values$`0.025quant`
pop_level$m1_pred_uc = example_dat[[1]]$summary.fitted.values$`0.975quant`
pop_level$m2_pred = example_dat[[2]]$summary.fitted.values$`0.5quant`
pop_level$m2_pred_lc = example_dat[[2]]$summary.fitted.values$`0.025quant`
pop_level$m2_pred_uc = example_dat[[2]]$summary.fitted.values$`0.975quant`
pop_level$m3_pred = example_dat[[3]]$summary.fitted.values$`0.5quant`
pop_level$m3_pred_lc = example_dat[[3]]$summary.fitted.values$`0.025quant`
pop_level$m3_pred_uc = example_dat[[3]]$summary.fitted.values$`0.975quant`
pop_level = cbind(df, pop_level)

obs_by_site_species = pop_level %>%
  group_by(site_code, species) %>%
  summarise(N = n())

trends_by_site = pop_level %>%
  group_by(site_code, latitude, longitude, species) %>%
  summarise(N = n())

site_freq = as.data.frame(table(unique(pop_level[,c("site_code", "latitude", "longitude", "species")])$site_code))
site_freq = subset(site_freq, Freq == 3)

#Select site 566 as a good example
trends_by_site = trends_by_site[trends_by_site$site_code %in% site_freq$Var1, ]

plt_m1_a = ggplot(data = pop_level[which(pop_level$site_code == 566),]) +
  geom_line(aes(x = date, y = (exp(m1_pred)), colour = species)) +
  geom_ribbon(aes(x = date, ymin = (exp(m1_pred_lc)), ymax = (exp(m1_pred_uc)), fill = species), alpha = 0.1) +
```



```

geom_point(aes(x = date, y = exp(log_abundance), colour = species)) +
scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010), limits = c(1987, 2015)) +
scale_y_continuous(expand = c(0,0)) +
coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
scale_colour_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
scale_fill_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
labs(x = "", y = "Abundance", title = "Random intercept") +
theme_classic() +
theme(plot.title = element_text(size=10, face="italic"))

plt_m2_a = ggplot(data = pop_level[which(pop_level$site_code == 566),]) +
geom_line(aes(x = date, y = (exp(m2_pred + mean_log)), colour = species)) +
geom_ribbon(aes(x = date, ymin = (exp(m2_pred_lc + mean_log)), ymax = (exp(m2_pred_uc + mean_log)), fill = species), alpha = 0.1) +
geom_point(aes(x = date, y = exp(cent_abundance + mean_log), colour = species)) +
scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010), limits = c(1987, 2015)) +
scale_y_continuous(expand = c(0,0)) +
coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
scale_colour_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
scale_fill_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
labs(x = "", y = "Abundance", title = "Random slope") +
theme_classic() +
theme(plot.title = element_text(size=10, face="italic"))

plt_m3_a = ggplot(data = pop_level[which(pop_level$site_code == 566),]) +
geom_line(aes(x = date, y = (exp(m3_pred + mean_log)), colour = species)) +

```

```

geom_ribbon(aes(x = date, ymin = (exp(m3_pred_lc + mean_log)), ymax = (exp(m3_pred_uc + mean_log)), fill = species), alpha = 0.1) +
geom_point(aes(x = date, y = exp(cent_abundance + mean_log), colour = species)) +
scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010), limits = c(1987, 2015)) +
scale_y_continuous(expand = c(0,0)) +
coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
scale_colour_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
scale_fill_manual(
  name = "",
  labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
  values = c("#9fbdbf", "#1885d9", "#1c50d4")) +
labs(x = "", y = "Abundance", title = "Correlated effect") +
theme_classic() +
theme(plot.title = element_text(size=10, face="italic"))

write.csv(pop_level, "../data/derived_data/fig3a.csv")

```

Under the same site described above, we plot the predicted site level trend. In this trend we only show uncertainty in the rate of change, the primary paramter of interest, not the intercept. This allows direct comparisons in uncertainty between the model types

```

site_level = example_dat[[1]]$summary.random$site_code[,c(1,4,5,6)]
site_level = cbind(site_level, example_dat[[2]]$summary.random$site_code[,c(4,5,6)])
site_level = cbind(site_level, example_dat[[3]]$summary.random$site_code2[,c(4,5,6)] + example_dat[[3]]$summary.random$site_code2[,c(4,5,6)])
colnames(site_level) = c("site_code", "m1_lc_a", "m1_med_a", "m1_uc_a", "m2_lc_a", "m2_med_a", "m2_uc_a", "m3_lc_a", "m3_med_a", "m3_uc_a")

region_level = example_dat[[1]]$summary.random$region_code[,c(1,5)]
region_level = cbind(region_level, example_dat[[2]]$summary.random$region_code[,c(5)])
region_level = cbind(region_level, example_dat[[3]]$summary.random$region_code[,c(5)])
colnames(region_level) = c("region_code", "m1_med_b", "m2_med_b", "m3_med_b")

region_site_link = unique(df[,c("region_code", "site_code")])
region_site_link = left_join(region_site_link, site_level)
region_site_link = left_join(region_site_link, region_level, by = "region_code")

region_site_link$m1_lc = region_site_link$m1_lc_a + region_site_link$m1_med_b
region_site_link$m1_med = region_site_link$m1_med_a + region_site_link$m1_med_b

```

```

region_site_link$m1_uc = region_site_link$m1_uc_a + region_site_link$m1_med_b
region_site_link$m2_lc = region_site_link$m2_lc_a + region_site_link$m2_med_b
region_site_link$m2_med = region_site_link$m2_med_a + region_site_link$m2_med_b
region_site_link$m2_uc = region_site_link$m2_uc_a + region_site_link$m2_med_b
region_site_link$m3_lc = region_site_link$m3_lc_a + region_site_link$m3_med_b
region_site_link$m3_med = region_site_link$m3_med_a + region_site_link$m3_med_b
region_site_link$m3_uc = region_site_link$m3_uc_a + region_site_link$m3_med_b

region_site_link_trim = subset(region_site_link, region_site_link$site_code == 566)

sit_level = data.frame(
  year = df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  mn1 = 4.72 + region_site_link_trim$m1_med + example_dat[[1]]$summary.fixed[2,4]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.v
  lc1 = 4.72 + region_site_link_trim$m1_med + example_dat[[1]]$summary.fixed[2,3]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.v
  uc1 = 4.72 + region_site_link_trim$m1_med + example_dat[[1]]$summary.fixed[2,5]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.v
  mn2 = 4.72 + (region_site_link_trim$m2_med + example_dat[[2]]$summary.fixed[2,4])*df$year_centre[c((nrow(example_dat[[2]]$summary.fitted.
  lc2 = 4.72 + (region_site_link_trim$m2_lc + example_dat[[2]]$summary.fixed[2,3])*df$year_centre[c((nrow(example_dat[[2]]$summary.fitted.
  uc2 = 4.72 + (region_site_link_trim$m2_uc + example_dat[[2]]$summary.fixed[2,5])*df$year_centre[c((nrow(example_dat[[2]]$summary.fitted.
  mn3 = 4.72 + (region_site_link_trim$m3_med + example_dat[[3]]$summary.fixed[2,4])*df$year_centre[c((nrow(example_dat[[3]]$summary.fitted.
  lc3 = 4.72 + (region_site_link_trim$m3_lc + example_dat[[3]]$summary.fixed[2,3])*df$year_centre[c((nrow(example_dat[[3]]$summary.fitted.
  uc3 = 4.72 + (region_site_link_trim$m3_uc + example_dat[[3]]$summary.fixed[2,5])*df$year_centre[c((nrow(example_dat[[3]]$summary.fitted.
)

plt_m1_b = ggplot() +
  geom_line(data = pop_level[which(pop_level$site_code == 566),], aes(x = date, y = exp(m1_pred)), colour = species), alpha = 0.5) +
  geom_line(data = sit_level, aes(x = year+2001, y = exp(mn1)), colour = "black") +
  geom_ribbon(data = sit_level, aes(x = year+2001, ymin = exp(lc1), ymax = exp(uc1)), alpha = 0.2, fill = "black") +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  scale_y_continuous(expand = c(0,0)) +
  coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
  scale_colour_manual(
    name = "",
    labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
    values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
  scale_fill_manual(
    name = "",
    labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),

```

```

    values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
  labs(x = "", y = "", title = "") +
  theme_classic() +
  theme(plot.title = element_text(size=10, face="italic"))

plt_m2_b = ggplot(data = pop_level[which(pop_level$site_code == 566),]) +
  geom_line(aes(x = date, y = (exp(m2_pred + mean_log)), colour = species), alpha = 0.5) +
  geom_line(data = sit_level, aes(x = year+2001, y = exp(mn2)), colour = "black") +
  geom_ribbon(data = sit_level, aes(x = year+2001, ymin = exp(lc2), ymax = exp(uc2)), alpha = 0.2, fill = "black") +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  scale_y_continuous(expand = c(0,0)) +
  coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
  scale_colour_manual(
    name = "",
    labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
    values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
  scale_fill_manual(
    name = "",
    labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
    values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
  labs(x = "", y = "", title = "") +
  theme_classic() +
  theme(plot.title = element_text(size=10, face="italic"))

plt_m3_b = ggplot() +
  geom_line(data = pop_level[which(pop_level$site_code == 566),], aes(x = date, y = (exp(m3_pred + mean_log)), colour = species), alpha = 0.5) +
  geom_line(data = sit_level, aes(x = year+2001, y = exp(mn3)), colour = "black") +
  geom_ribbon(data = sit_level, aes(x = year+2001, ymin = exp(lc3), ymax = exp(uc3)), alpha = 0.2, fill = "black") +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  scale_y_continuous(expand = c(0,0)) +
  coord_cartesian(ylim = c(1,600), xlim = c(1987, 2015)) +
  scale_colour_manual(
    name = "",
    labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
    values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
  scale_fill_manual(

```

```

name = "",
labels = c("M. daubentoni", "M. emarginatus", "M. nattereri"),
values = c("#9fbd9f", "#1885d9", "#1c50d4")) +
labs(x = "", y = "", title = "") +
labs(x = "", y = "", title = "") +
theme_classic() +
theme(plot.title = element_text(size=10, face="italic"))

write.csv(sit_level, "../data/derived_data/fig3b.csv")

```

Next we plot all site level trends to depict heterogeneity in spatial trend variation, and then display the global trend over this site level variability, indicating that random intercept and random slope models are a poor representation of this heterogeneity. In this trend we only show uncertainty in the rate of change, the primary parameter of interest, not the intercept. This allows direct comparisons in uncertainty between the model types.

```

expand_reg_sit_link = unique(df[,c("site_code", "mean_log", "mean_year")])
cmb_df = NULL
for(a in 1:nrow(region_site_link)){
  mn_lg = mean(subset(expand_reg_sit_link, site_code == region_site_link$site_code[a])$mean_log)[1]
  tmp_df = data.frame(
    site_code = subset(expand_reg_sit_link, site_code == region_site_link$site_code[a])$site_code[1],
    year_centre = seq(-20,20,length.out = 30),
    year_adj = seq(-20,20,length.out = 30) + subset(expand_reg_sit_link, site_code == region_site_link$site_code[a])$mean_year[1],
    mn_lg = mn_lg,
    y1 = 4.72 + -0.004*c(seq(-20,20,length.out = 30)) + region_site_link$m1_med[a], #2.801 -0.002
    y2 = 4.72 + (-0.001 + region_site_link$m2_med[a])*c(seq(-20,20,length.out = 30)), #-0.002
    y3 = 4.72 + (-0.008 + region_site_link$m3_med[a])*c(seq(-20,20,length.out = 30)) #-0.002
  )
  cmb_df = rbind(cmb_df, tmp_df)
}

```

```

glob_level = data.frame(
  year = df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  mn1 = example_dat[[1]]$summary.fixed[2,4]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  lc1 = example_dat[[1]]$summary.fixed[2,3]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  uc1 = example_dat[[1]]$summary.fixed[2,5]*df$year_centre[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  mn2 = example_dat[[2]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],
  lc2 = example_dat[[2]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))],

```

```

uc2 = example_dat[[2]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))]
mn3 = example_dat[[3]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))]
lc3 = example_dat[[3]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))]
uc3 = example_dat[[3]]$summary.fitted.values[c((nrow(example_dat[[1]]$summary.fitted.values)-99):nrow(example_dat[[1]]$summary.fitted.values))]
)

m1_col = "#018571"
m2_col = "#8f60c4"
m3_col = "#dfc27d"

plt_m1_c = ggplot() +
  geom_line(data = cmb_df,
            aes(x = year_centre+2001, y = exp(y1), group = site_code), alpha = 0.02, colour = "grey") +
  geom_line(data = glob_level, aes(x = year+2001, y = exp(mn1+4.72)), colour = "#018571", size = 1.5) +
  geom_ribbon(data = glob_level, aes(x = year+2001, ymin = exp(lc1+4.72), ymax = exp(uc1+4.72)), alpha = 0.4, fill = "#018571") +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  scale_y_continuous(expand = c(0,0)) +
  coord_cartesian(ylim = c(0,200), xlim = c(1987, 2015)) +
  labs(x = "", y = "", title = "") +
  theme_classic()

plt_m2_c = ggplot() +
  geom_line(data = cmb_df,
            aes(x = year_centre+2001, y = exp(y3), group = site_code), alpha = 0.02, colour = "grey") +
  geom_line(data = glob_level, aes(x = year+2001, y = exp(mn2+4.72)), colour = "#8f60c4", size = 1.5) +
  geom_ribbon(data = glob_level, aes(x = year+2001, ymin = exp(lc2+4.72), ymax = exp(uc2+4.72)), alpha = 0.4, fill = "#8f60c4") +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  scale_y_continuous(expand = c(0,0)) +
  coord_cartesian(ylim = c(50,200), xlim = c(1987, 2015)) +
  labs(x = "", y = "", title = "") +
  theme_classic()

plt_m3_c = ggplot() +
  geom_line(data = cmb_df,

```

```

      aes(x = year_centre+2001, y = exp(y3), group = site_code), alpha = 0.02, colour = "grey") +
    geom_line(data = glob_level, aes(x = year+2001, y = exp(mn3+4.72)), colour = "#dfc27d", size = 1.5) +
    geom_ribbon(data = glob_level, aes(x = year+2001, ymin = exp(lc3+4.72), ymax = exp(uc3+4.72)), alpha = 0.4, fill = "#dfc27d") +
    scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
    scale_y_continuous(expand = c(0,0)) +
    coord_cartesian(ylim = c(50,200), xlim = c(1987, 2015)) +
    labs(x = "", y = "", title = "") +
    theme_classic()

write.csv(glob_level, "../data/derived_data/fig3c.csv")

```

We then take global level trend coefficients and 50% credible intervals to project abundance trends over time

```

ab_lc = c(100)
ab_mn = c(100)
ab_uc = c(100)
for(a in c(1:28)){
  ab_lc = c(ab_lc, ab_lc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[1]]$marginals.fixed[[2]])[1]))
  ab_mn = c(ab_mn, ab_mn[a]*(1 + example_dat[[1]]$summary.fixed$`0.5quant`[2]))
  ab_uc = c(ab_uc, ab_uc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[1]]$marginals.fixed[[2]])[2]))
}
example_dat_proj_m1 = data.frame(
  year = c(1987:2015),
  ab_lc = ab_lc,
  ab_mn = ab_mn,
  ab_uc = ab_uc
)

plt_m1_d = ggplot(example_dat_proj_m1) +
  geom_line(aes(x = year, y = ab_mn), colour = "#018571", size = 2) +
  geom_ribbon(aes(x = year, ymin = ab_lc, ymax = ab_uc), alpha = 0.3, fill = "#018571") +
  geom_hline(aes(yintercept = 100), linetype = "dashed") +
  scale_y_continuous(expand = c(0,0)) +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  coord_cartesian(ylim = c(75,125)) +
  theme_classic() +

```

```

labs(x = "Year", y = "Projected\\nabundance", title = "")

ab_lc = c(100)
ab_mn = c(100)
ab_uc = c(100)
for(a in c(1:28)){
  ab_lc = c(ab_lc, ab_lc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[2]]$marginals.fixed[[2]])[1]))
  ab_mn = c(ab_mn, ab_mn[a]*(1 + example_dat[[2]]$summary.fixed$`0.5quant`[2]))
  ab_uc = c(ab_uc, ab_uc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[2]]$marginals.fixed[[2]])[2]))
}
example_dat_proj_m2 = data.frame(
  year = c(1987:2015),
  ab_lc = ab_lc,
  ab_mn = ab_mn,
  ab_uc = ab_uc
)

plt_m2_d = ggplot(example_dat_proj_m2) +
  geom_line(aes(x = year, y = ab_mn), colour = "#8f60c4", size = 2) +
  geom_ribbon(aes(x = year, ymin = ab_lc, ymax = ab_uc), alpha = 0.3, fill = "#8f60c4") +
  geom_hline(aes(yintercept = 100), linetype = "dashed") +
  scale_y_continuous(expand = c(0,0)) +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  coord_cartesian(ylim = c(75,125)) +
  theme_classic() +
  labs(x = "Year", y = "Projected\\nabundance", title = "")

ab_lc = c(100)
ab_mn = c(100)
ab_uc = c(100)
for(a in c(1:28)){
  ab_lc = c(ab_lc, ab_lc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[3]]$marginals.fixed[[2]])[1]))
  ab_mn = c(ab_mn, ab_mn[a]*(1 + example_dat[[3]]$summary.fixed$`0.5quant`[2]))
  ab_uc = c(ab_uc, ab_uc[a]*(1 + inla.hpdmarginal(0.5, example_dat[[3]]$marginals.fixed[[2]])[2]))
}

```



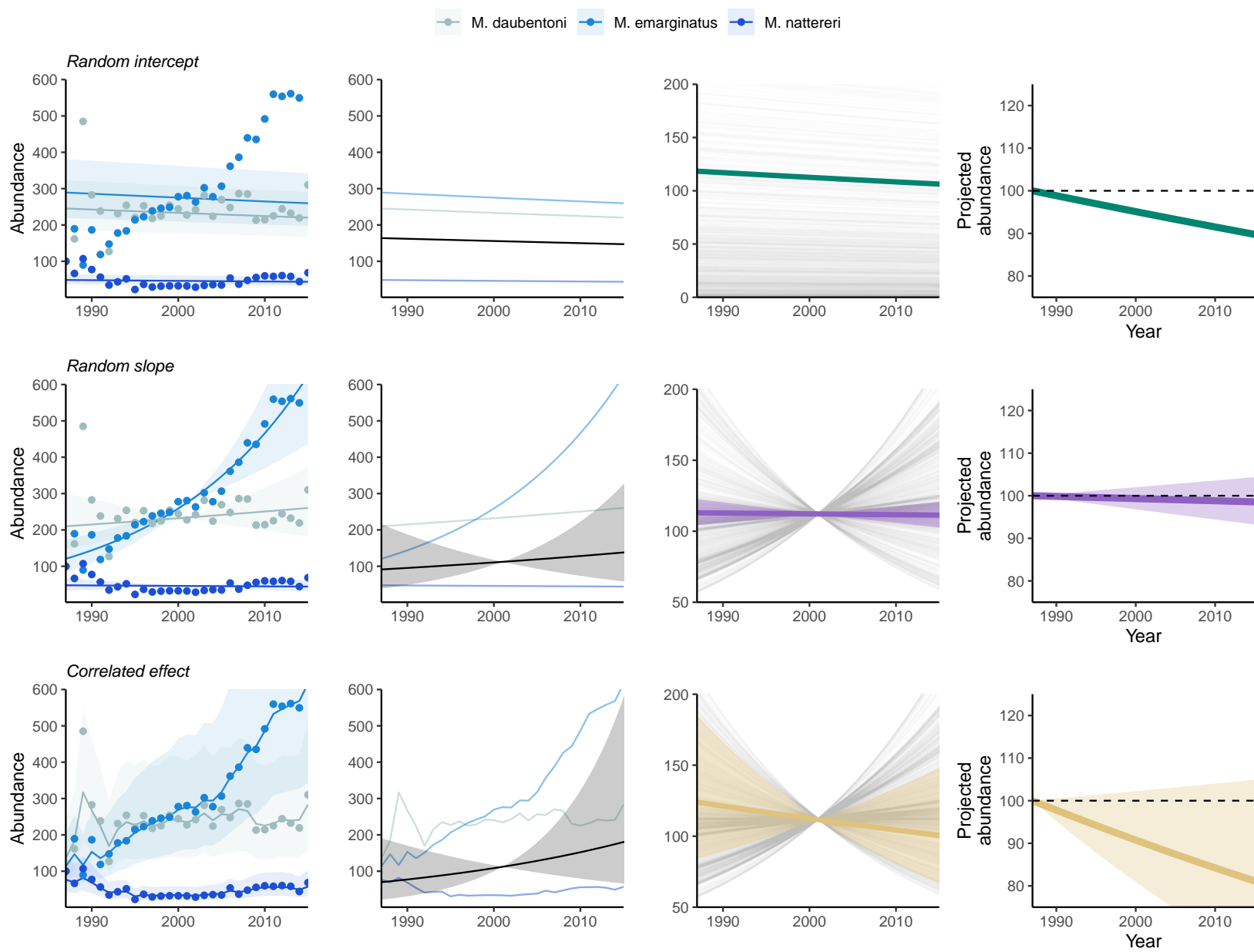
```

}
example_dat_proj_m3 = data.frame(
  year = c(1987:2015),
  ab_lc = ab_lc,
  ab_mn = ab_mn,
  ab_uc = ab_uc
)

plt_m3_d = ggplot(example_dat_proj_m3) +
  geom_line(aes(x = year, y = ab_mn), colour = "#dfc27d", size = 2) +
  geom_ribbon(aes(x = year, ymin = ab_lc, ymax = ab_uc), alpha = 0.3, fill = "#dfc27d") +
  geom_hline(aes(yintercept = 100), linetype = "dashed") +
  scale_y_continuous(expand = c(0,0)) +
  scale_x_continuous(expand = c(0,0), breaks = c(1990, 2000, 2010)) +
  coord_cartesian(ylim = c(75,125)) +
  theme_classic() +
  labs(x = "Year", y = "Projected\nabundance", title = "")

ggarrange(
  plt_m1_a,
  plt_m1_b,
  plt_m1_c,
  plt_m1_d,
  plt_m2_a,
  plt_m2_b,
  plt_m2_c,
  plt_m2_d,
  plt_m3_a,
  plt_m3_b,
  plt_m3_c,
  plt_m3_d,
  ncol = 4, nrow = 3, common.legend = T)

```



```
write.csv(rbind(example_dat_proj_m1, example_dat_proj_m2, example_dat_proj_m3), "../data/derived_data/fig3d.csv")
```

Figure 4

Here we plot estimates of abundance change over space for one high profile species - American Robin. To calculate this abundance change in a given location, we sum: population trend + species trend + genus trend + phylogeny trend + site trend + region trend + spatial trend + overall trend. We then take uncertainty around spatial trends to estimate whether a trend would be significant at a selection of thresholds. This model is based on BioTIME data.

```
m3 = readRDS("../outputs/model_output_predict2_BioTIME.rds")
m3 = m3[[3]]
tmp_df = readRDS("../data/derived_data/analysis_list_predict2.rds")
tmp_df = tmp_df[[1]][[1]]
link_df = unique(tmp_df[,c("site_spec", "site_spec_code", "site_code", "region_code", "tips_code", "genus_code", "lat_round", "lon_round", "species_code")])
link_df = link_df[c(12066:14976),]

tips_code_med = m3$summary.random$tips_code[,c(1,2)]
colnames(tips_code_med) = c("ID", "val_tips_code_mn")
link_df = left_join(link_df, tips_code_med, by = c("tips_code" = "ID"))

tips_code2_med = m3$summary.random$tips_code2[,c(1,2)]
colnames(tips_code2_med) = c("ID", "val_tips_code2_mn")
link_df = left_join(link_df, tips_code2_med, by = c("tips_code" = "ID"))

genus_code_med = m3$summary.random$genus_code[,c(1,2)]
colnames(genus_code_med) = c("ID", "val_genus_code_mn")
link_df = left_join(link_df, genus_code_med, by = c("genus_code" = "ID"))

site_code_med = m3$summary.random$site_code[,c(1,2)]
colnames(site_code_med) = c("ID", "val_site_code_mn")
link_df = left_join(link_df, site_code_med, by = c("site_code" = "ID"))

site_code2_med = m3$summary.random$site_code2[,c(1,2)]
colnames(site_code2_med) = c("ID", "val_site_code2_mn")
link_df = left_join(link_df, site_code2_med, by = c("site_code" = "ID"))
```

```

site_spec_code_med = m3$summary.random$site_spec_code[,c(1,2,3)]
colnames(site_spec_code_med) = c("ID", "val_site_spec_code_mn", "val_site_spec_code_sd")
site_spec_code_med$site_spec_qt400 = site_spec_code_med$val_site_spec_code_mn - (0.25*site_spec_code_med$val_site_spec_code_sd) #20% 0.25
site_spec_code_med$site_spec_qt600 = site_spec_code_med$val_site_spec_code_mn + (0.25*site_spec_code_med$val_site_spec_code_sd)
site_spec_code_med$site_spec_qt300 = site_spec_code_med$val_site_spec_code_mn - (0.52*site_spec_code_med$val_site_spec_code_sd) #40%
site_spec_code_med$site_spec_qt700 = site_spec_code_med$val_site_spec_code_mn + (0.52*site_spec_code_med$val_site_spec_code_sd)
site_spec_code_med$site_spec_qt200 = site_spec_code_med$val_site_spec_code_mn - (0.84*site_spec_code_med$val_site_spec_code_sd) #60%
site_spec_code_med$site_spec_qt800 = site_spec_code_med$val_site_spec_code_mn + (0.84*site_spec_code_med$val_site_spec_code_sd)
site_spec_code_med$site_spec_qt100 = site_spec_code_med$val_site_spec_code_mn - (1.28*site_spec_code_med$val_site_spec_code_sd) #80%
site_spec_code_med$site_spec_qt900 = site_spec_code_med$val_site_spec_code_mn + (1.28*site_spec_code_med$val_site_spec_code_sd)
link_df = left_join(link_df, site_spec_code_med, by = c("site_spec_code" = "ID"))

region_code_med = m3$summary.random$region_code[,c(1,2)]
colnames(region_code_med) = c("ID", "val_region_code_mn")
link_df = left_join(link_df, region_code_med, by = c("region_code" = "ID"))

link_df$pop_trend_sit_mn =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +
  link_df$val_genus_code_mn +
  link_df$val_site_code_mn +
  link_df$val_site_code2_mn +
  link_df$val_region_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_sit_100 =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +
  link_df$val_genus_code_mn +
  link_df$val_site_code_mn +
  link_df$val_site_code2_mn +
  link_df$site_spec_qt100 +
  link_df$val_region_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_sit_200 =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +

```

```

link_df$val_genus_code_mn +
link_df$val_site_code_mn +
link_df$val_site_code2_mn +
link_df$site_spec_qt200 +
link_df$val_region_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_sit_300 =
link_df$val_tips_code_mn +
link_df$val_tips_code2_mn +
link_df$val_genus_code_mn +
link_df$val_site_code_mn +
link_df$val_site_code2_mn +
link_df$site_spec_qt300 +
link_df$val_region_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_sit_400 =
link_df$val_tips_code_mn +
link_df$val_tips_code2_mn +
link_df$val_genus_code_mn +
link_df$val_site_code_mn +
link_df$val_site_code2_mn +
link_df$site_spec_qt400 +
link_df$val_region_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_sit_600 =
link_df$val_tips_code_mn +
link_df$val_tips_code2_mn +
link_df$val_genus_code_mn +
link_df$val_site_code_mn +
link_df$val_site_code2_mn +
link_df$site_spec_qt600 +
link_df$val_region_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_sit_700 =

```

```

link_df$val_tips_code_mn +
link_df$val_tips_code2_mn +
link_df$val_genus_code_mn +
link_df$val_site_code_mn +
link_df$val_site_code2_mn +
link_df$site_spec_qt700 +
link_df$val_region_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_sit_800 =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +
  link_df$val_genus_code_mn +
  link_df$val_site_code_mn +
  link_df$val_site_code2_mn +
  link_df$site_spec_qt800 +
  link_df$val_region_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_sit_900 =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +
  link_df$val_genus_code_mn +
  link_df$val_site_code_mn +
  link_df$val_site_code2_mn +
  link_df$site_spec_qt900 +
  link_df$val_region_code_mn +
  m3$summary.fixed$mean[2]

link_df_sum = link_df
link_df_sum$cat = NA
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_400 > 0, "+20%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_300 > 0, "+40%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_200 > 0, "+60%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_100 > 0, "+80%", link_df_sum$cat)

```

```

link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_600 < 0, "-20%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_700 < 0, "-40%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_800 < 0, "-60%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_sit_900 < 0, "-80%", link_df_sum$cat)
link_df_sum$cat = ifelse(is.na(link_df_sum$cat), "0%", link_df_sum$cat)
link_df_sum$cat = factor(link_df_sum$cat, levels = c("-80%", "-60%", "-40%", "-20%", "0%", "+20%", "+40%", "+60%", "+80%"))
link_df_sum$cat_num = as.numeric(link_df_sum$cat)

## Example SpatialPolygonsDataFrame
data(wrld_simpl)
SPDF <- subset(wrld_simpl, ISO3=="USA" | ISO3=="CAN" | ISO3=="MEX" )

## Example RasterLayer
rst <- rasterFromXYZ(link_df_sum[,c("lon_round", "lat_round", "cat_num")], crs=proj4string(SPDF))
## crop and mask
rst <- crop(rst, extent(SPDF))
rst <- mask(rst, SPDF)

rst_df <- as(rst, "SpatialPixelsDataFrame")
rst_df <- as.data.frame(rst_df)
colnames(rst_df) <- c("value", "x", "y")

world <- map_data("world")

plt_map = ggplot() +
  geom_tile(data=rst_df, aes(x=x, y=y, fill=as.factor(value)), alpha=0.8) +
  geom_map(data = world, map = world, aes(long, lat, map_id = region), fill = NA, colour = "black", alpha = 0.4) +
  #geom_polygon(data=SPDF, aes(x=long, y=lat, group=group),
  #            fill=NA, color="black", size=1) +
  coord_sf(xlim = c(-130,-60), ylim = c(20,60)) +
  scale_x_continuous(expand = c(0,0)) +
  scale_y_continuous(expand = c(0,0)) +
  scale_fill_brewer(palette = "RdBu", labels = c("-80", "-60", "-40", "-20", "0", "+20", "+40", "+60", "+80"), name = "Confidence interval")
  labs(x = "Longitude", y = "Latitude") +
  theme_classic()

```

```
write.csv(rst_df, "../data/derived_data/fig4a.csv")
```

Here we plot estimates of abundance change over a phylogeny by taking the sum of: species trend + genus trend + phylogeny trend + overall trend. We then take uncertainty around phylogeny trends to estimate whether a trend would be significant at a selection of thresholds. This model is based on BioTIME data

```
m3 = readRDS("../outputs/model_output_convergence10_BioTIME.rds")
m3 = m3[[3]]
tmp_df = readRDS("../data/derived_data/analysis_list.rds")
tmp_df = tmp_df[[1]][[1]]
link_df = unique(tmp_df[,c("tips_code", "genus_code", "species")])

tips_code_med = m3$summary.random$tips_code[,c(1,2)]
colnames(tips_code_med) = c("ID", "val_tips_code_mn")
link_df = left_join(link_df, tips_code_med, by = c("tips_code" = "ID"))

tips_code2_med = m3$summary.random$tips_code2[,c(1,2,3,4,6)]
colnames(tips_code2_med) = c("ID", "val_tips_code2_mn", "val_tips_code2_sd", "lc", "uc")
tips_code2_med$tip_qt400 = tips_code2_med$val_tips_code2_mn - (0.25*tips_code2_med$val_tips_code2_sd) #20%
tips_code2_med$tip_qt600 = tips_code2_med$val_tips_code2_mn + (0.25*tips_code2_med$val_tips_code2_sd)
tips_code2_med$tip_qt300 = tips_code2_med$val_tips_code2_mn - (0.52*tips_code2_med$val_tips_code2_sd) #40%
tips_code2_med$tip_qt700 = tips_code2_med$val_tips_code2_mn + (0.52*tips_code2_med$val_tips_code2_sd)
tips_code2_med$tip_qt200 = tips_code2_med$val_tips_code2_mn - (0.84*tips_code2_med$val_tips_code2_sd) #60%
tips_code2_med$tip_qt800 = tips_code2_med$val_tips_code2_mn + (0.84*tips_code2_med$val_tips_code2_sd)
tips_code2_med$tip_qt100 = tips_code2_med$val_tips_code2_mn - (1.28*tips_code2_med$val_tips_code2_sd) #80%
tips_code2_med$tip_qt900 = tips_code2_med$val_tips_code2_mn + (1.28*tips_code2_med$val_tips_code2_sd)
link_df = left_join(link_df, tips_code2_med, by = c("tips_code" = "ID"))

genus_code_med = m3$summary.random$genus_code[,c(1,2)]
colnames(genus_code_med) = c("ID", "val_genus_code_mn")
link_df = left_join(link_df, genus_code_med, by = c("genus_code" = "ID"))

link_df$pop_trend_tip_mn =
  link_df$val_tips_code_mn +
  link_df$val_tips_code2_mn +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]
```



```

link_df$pop_trend_tip_100 =
  link_df$val_tips_code_mn +
  link_df$tip_qt100 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_200 =
  link_df$val_tips_code_mn +
  link_df$tip_qt200 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_300 =
  link_df$val_tips_code_mn +
  link_df$tip_qt300 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_400 =
  link_df$val_tips_code_mn +
  link_df$tip_qt400 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_600 =
  link_df$val_tips_code_mn +
  link_df$tip_qt600 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_700 =
  link_df$val_tips_code_mn +
  link_df$tip_qt700 +
  link_df$val_genus_code_mn +
  m3$summary.fixed$mean[2]

link_df$pop_trend_tip_800 =
  link_df$val_tips_code_mn +

```

```

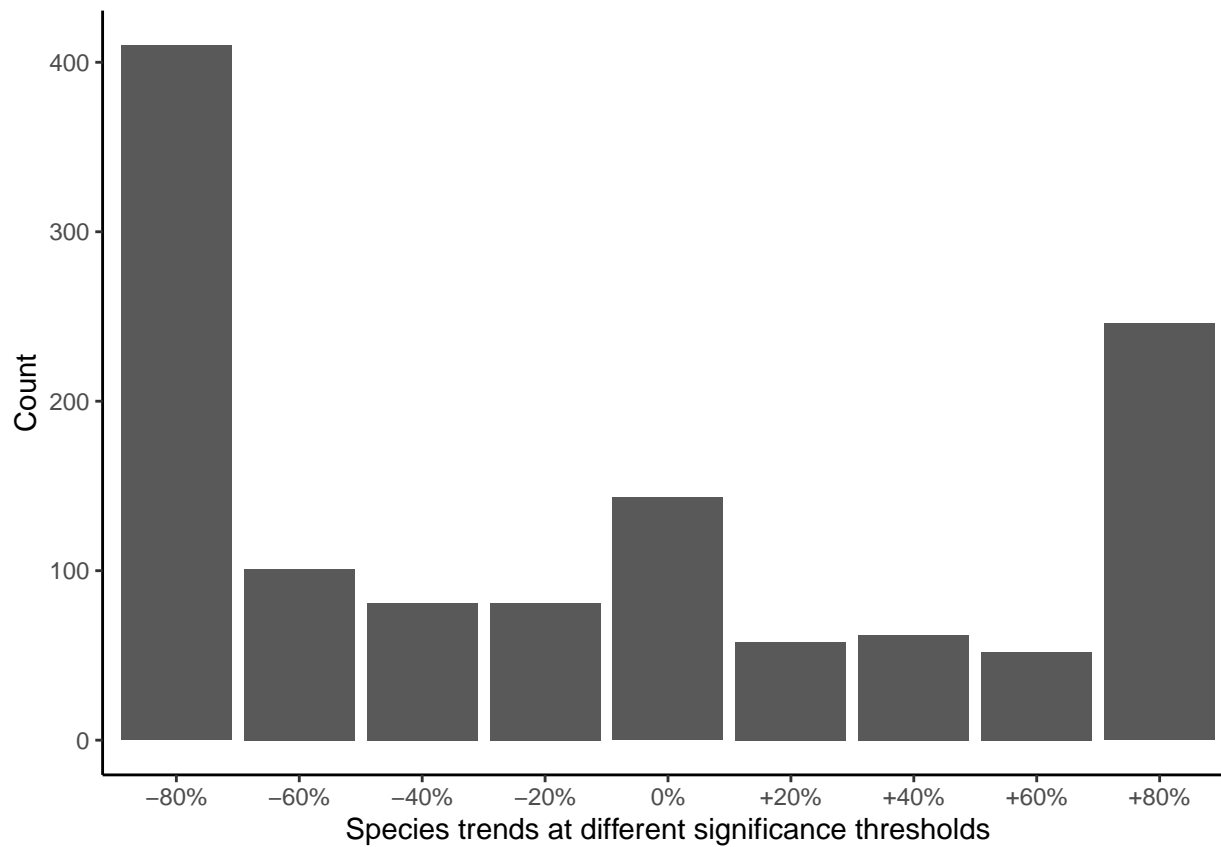
link_df$tip_qt800 +
link_df$val_genus_code_mn +
m3$summary.fixed$mean[2]

link_df$pop_trend_tip_900 =
link_df$val_tips_code_mn +
link_df$tip_qt900 +
link_df$val_genus_code_mn +
m3$summary.fixed$mean[2]

link_df_sum = link_df
link_df_sum$cat = NA
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_400 > 0, "+20%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_300 > 0, "+40%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_200 > 0, "+60%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_100 > 0, "+80%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_600 < 0, "-20%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_700 < 0, "-40%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_800 < 0, "-60%", link_df_sum$cat)
link_df_sum$cat = ifelse(link_df_sum$pop_trend_tip_900 < 0, "-80%", link_df_sum$cat)
link_df_sum$cat = ifelse(is.na(link_df_sum$cat), "0%", link_df_sum$cat)
link_df_sum$cat = factor(link_df_sum$cat, levels = c("-80%", "-60%", "-40%", "-20%", "0%", "+20%", "+40%", "+60%", "+80%"))

ggplot(link_df_sum) +
  geom_bar(aes(cat)) +
  labs(x = "Species trends at different significance thresholds", y = "Count") +
  theme_classic()

```



```
spec_link = unique(tmp_df[,c("tips_code", "tips_chr")])
spec_link = subset(spec_link, !is.na(tips_chr))
link_df_sum2 = left_join(link_df_sum[,c("tips_code", "cat")], spec_link, by = "tips_code")
link_df_sum2 = subset(link_df_sum2, !is.na(link_df_sum2$tips_chr))
keep = unique(link_df_sum2$tips_chr)

tr = readRDS("../data/derived_data/trees.rds")[[1]][[3]]
tr2 = keep.tip(tr, keep)
```

```
tr2$edge.length = log(tr2$edge.length) - min(log(tr2$edge.length)) + 0.01
tr2 = force.ultrametric(tr2)
```

```
## *****
## *                               Note:                               *
## *   force.ultrametric does not include a formal method to         *
## *   ultrametricize a tree & should only be used to coerce         *
## *   a phylogeny that fails is.ultrametric due to rounding --      *
## *   not as a substitute for formal rate-smoothing methods.       *
## *****
```

```
p = ggtree(tr2, layout = "circular")
```

```
plt_trends = data.frame(
  cat = link_df_sum2$cat
)
```

```
plt_trends$cat = as.factor(plt_trends$cat)
```

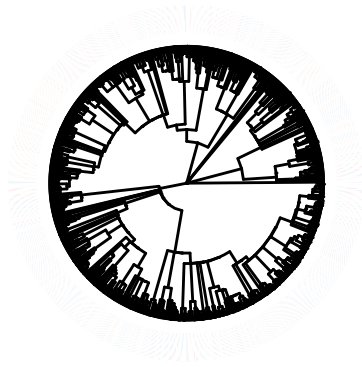
```
rownames(plt_trends) = link_df_sum2$tips_chr
```

```
plt_clad = gheatmap(p, plt_trends, width = 0.2, colnames = F) +
```

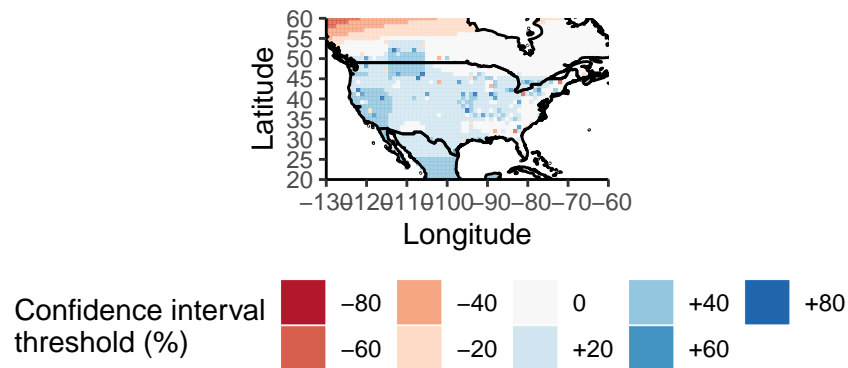
```
  scale_fill_brewer(palette = "RdBu", labels = c("-80", "-60", "-40", "-20", "0", "+20", "+40", "+60", "+80"), name = "Confidence interval")
```

```
ggarrange(plt_clad, plt_map, ncol = 1, labels = c("A", "B"), common.legend = T, heights = c(1.2, 0.65), legend = "bottom")
```

A



B



```
write.csv(plt_trends, "../data/derived_data/fig4b.csv")
```

Phylogeny - sensitivity analysis

Load data and develop a dataset summarising the three phylogeny models (A: Full - OTL, B: Restricted - OTL, C: Restricted - TimeTree). The OTL models use a phylogeny lacking branch lengths from the Open Tree of Life. The TimeTree phylogeny contains branch lengths, but has a 'restricted' taxonomic extent. In this restricted dataset, we only kept species occurring in both the Open Tree of Life and TimeTree.

```

model_summary_core = read.csv("../outputs/model_summary.csv")
model_summary_phylo = read.csv("../outputs/model_summary_phylo.csv")

model_summary_core = subset(model_summary_core, model == 3)
df_tmp = rbind(model_summary_core, model_summary_phylo)

df_tmp$phy_h = df_tmp$tip_h/(df_tmp$tip + df_tmp$tip_h + df_tmp$gen)
df_tmp$spa_h = df_tmp$sit_h/(df_tmp$sit + df_tmp$sit_h + df_tmp$squ)
df_tmp$temp = df_tmp$obv_auto/(df_tmp$obv + df_tmp$obv_auto)
df_tmp$temp_v = (df_tmp$obv + df_tmp$obv_auto)/(
  df_tmp$fix +
  df_tmp$obv +
  df_tmp$obv_auto +
  df_tmp$gen +
  df_tmp$sit +
  df_tmp$sit_h +
  df_tmp$squ +
  df_tmp$sig)
df_tmp$sit_v = (df_tmp$sit + df_tmp$sit_h + df_tmp$squ)/(
  df_tmp$fix +
  df_tmp$obv +
  df_tmp$obv_auto +
  df_tmp$gen +
  df_tmp$sit +
  df_tmp$sit_h +
  df_tmp$squ +
  df_tmp$sig)
df_tmp$phy_v = (df_tmp$tip + df_tmp$tip_h + df_tmp$gen)/(
  df_tmp$fix +
  df_tmp$obv +
  df_tmp$obv_auto +
  df_tmp$gen +
  df_tmp$sit +
  df_tmp$sit_h +
  df_tmp$squ +
  df_tmp$sig)
df_tmp$res_v = (df_tmp$sig)/(

```

```

df_tmp$fix +
df_tmp$obv +
df_tmp$obv_auto +
df_tmp$tip +
df_tmp$tip_h +
df_tmp$gen +
df_tmp$sit +
df_tmp$sit_h +
df_tmp$squ +
df_tmp$sig)
df_tmp$fix_v = (df_tmp$fix)/(
df_tmp$fix +
df_tmp$obv +
df_tmp$obv_auto +
df_tmp$gen +
df_tmp$sit +
df_tmp$sit_h +
df_tmp$squ +
df_tmp$sig)

dataset_rename = data.frame(
  code = c("BioTIME", "CaPTrends", "EA_NFPD", "FishGlob", "LPI", "NA_m_BBS", "Pilotto", "ReSurvey_Germany", "RivFish", "TimeFISH"),
  dataset_code = c("A) BioTIME", "J) Large carnivores", "F) UK riverine fishes", "D) FishGlob", "B) Living Planet", "C) NA Breeding Birds")
df_tmp = left_join(df_tmp, dataset_rename)

```

Collective trends and estimates of uncertainty for each of the model and dataset combinations

```

ggplot(data = df_tmp) +
  geom_pointrange(aes(x = model, ymin = coef_lc, y = coef, ymax = coef_uc, colour = model)) +
  scale_colour_manual(values = c("grey10", "grey50", "grey80"), labels = c("Full: OTL", "Restricted: OTL", "Restricted: TimeTree"), name = NULL) +
  scale_x_discrete(labels = c("A", "B", "C")) +
  labs(x = "", y = expression(beta)) +
  #coord_flip() +
  facet_wrap(dataset_code~., scales = "free") +
  theme_classic() +

```

```
theme(legend.position = "bottom")
```

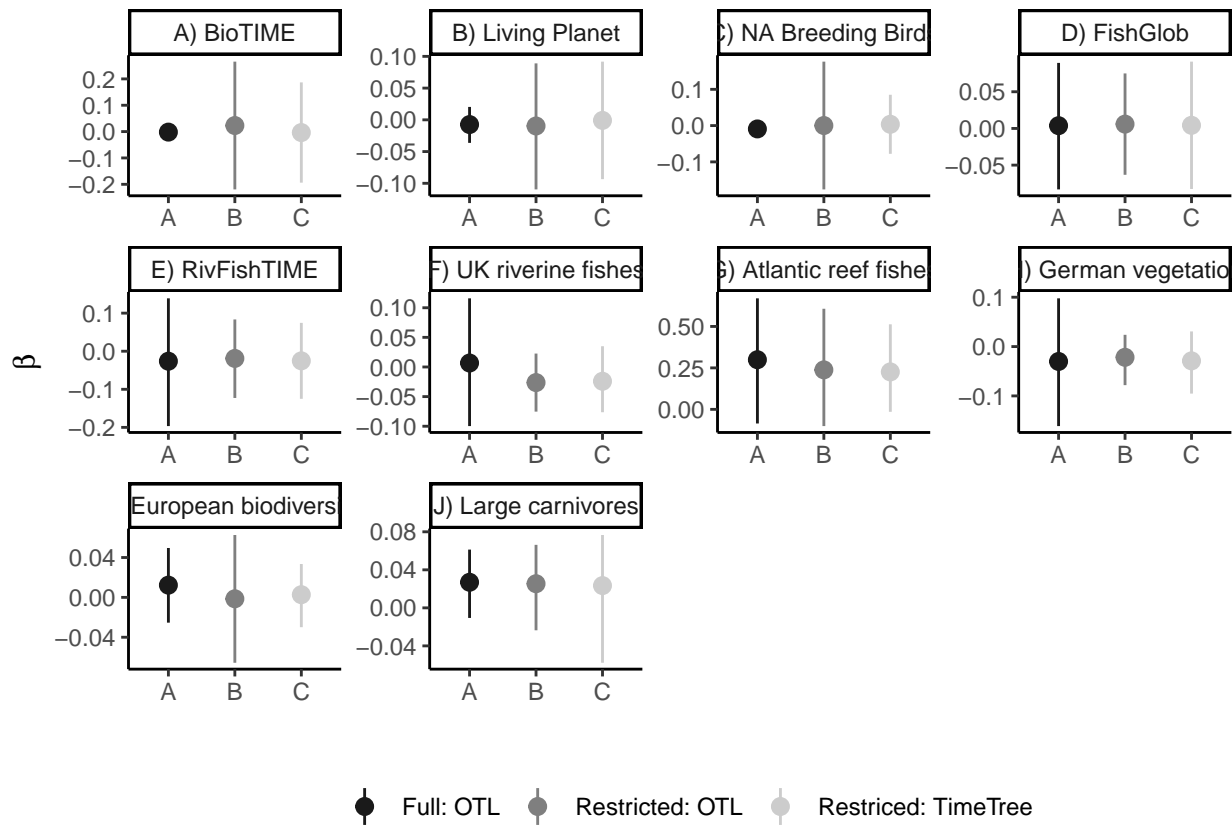


Figure describing how the proportion of variance captured by space and time are the driving forces behind the degree of uncertainty in the collective trend.

```
df_rotl_trim = subset(df_tmp,
                      model == "3_rotl_trim"
                      )
df_timetree_trim = subset(df_tmp,
                          model == "3_timetree"
```

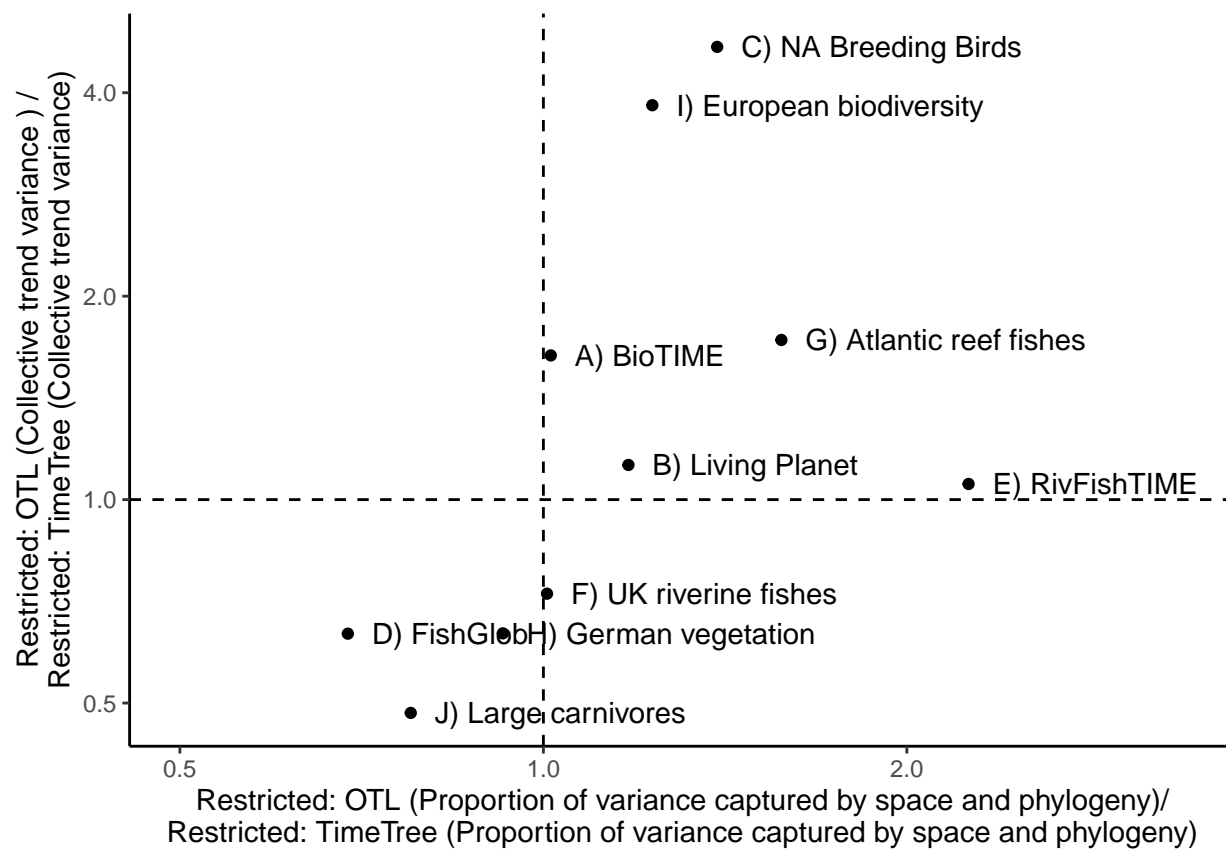


```

)
df_compare = data.frame(
  name = df_rotl_trim$dataset_code,
  sd_div = df_rotl_trim$coef_sd^2/df_timetree_trim$coef_sd^2,
  var = (df_rotl_trim$sit_v + df_rotl_trim$phy_v)/
    (df_timetree_trim$sit_v + df_timetree_trim$phy_v))

ggplot(data = df_compare) +
  geom_point(aes(x = var, y = sd_div)) +
  geom_text(aes(x = var, y = sd_div, label = name), hjust = 0, nudge_x = 0.02) +
  scale_y_log10(breaks = c(0.5,1,2,4,8)) +
  scale_x_log10(breaks = c(0.5,1,2,4,8), limits = c(0.5,3.4)) +
  geom_hline(aes(yintercept = 1), linetype = "dashed") +
  geom_vline(aes(xintercept = 1), linetype = "dashed") +
  labs(x = "Restricted: OTL (Proportion of variance captured by space and phylogeny)/\nRestricted: TimeTree (Proportion of variance captured by space and phylogeny)",
       y = "Restricted: OTL (Proportion of variance captured by space and phylogeny)/\nRestricted: TimeTree (Proportion of variance captured by space and phylogeny)") +
  theme_classic()

```



Structure - sensitivity analysis

Load data

```
model_summary_core = read.csv("../outputs/model_summary.csv")
model_summary_struc = read.csv("../outputs/model_summary_structure.csv")
```

Manipulate data. Renaming columns. Calculating the collective trend variance difference between the random slope model and all other models

```

model_summary_core = subset(model_summary_core, model != 1, select = c(
  "X", "code", "model", "coef", "coef_sd", "coef_lc", "coef_uc"
))
colnames(model_summary_core)[3] = "model_type"

df_tmp = rbind(model_summary_core, model_summary_struc)

df_tmp = df_tmp %>% mutate(model_type=recode(model_type,
  '3'='1) Space + Time + Phylogeny',
  'Time absent'='2) Space + Phylogeny',
  'Space absent'='3) Time + Phylogeny',
  'Phylogeny absent'='4) Space + Time',
  'Time and space absent'='5) Phylogeny',
  'Time and phylogeny absent'='6) Space',
  'Space and phylogeny absent'='7) Time',
  '2'='8) Random slope model'
))

df_tmp2 = spread(df_tmp[,c(2,3,5)], model_type, coef_sd)
df_tmp2$`1) Space + Time + Phylogeny` = df_tmp2$`1) Space + Time + Phylogeny` / df_tmp2$`8) Random slope model`
df_tmp2$`2) Space + Phylogeny` = df_tmp2$`2) Space + Phylogeny` / df_tmp2$`8) Random slope model`
df_tmp2$`3) Time + Phylogeny` = df_tmp2$`3) Time + Phylogeny` / df_tmp2$`8) Random slope model`
df_tmp2$`4) Space + Time` = df_tmp2$`4) Space + Time` / df_tmp2$`8) Random slope model`
df_tmp2$`5) Phylogeny` = df_tmp2$`5) Phylogeny` / df_tmp2$`8) Random slope model`
df_tmp2$`6) Space` = df_tmp2$`6) Space` / df_tmp2$`8) Random slope model`
df_tmp2$`7) Time` = df_tmp2$`7) Time` / df_tmp2$`8) Random slope model`

df_tmp3 = gather(df_tmp2, model, sd_div, `1) Space + Time + Phylogeny`:`7) Time`, factor_key=TRUE)

```

Calculate average variance difference and standard deviation

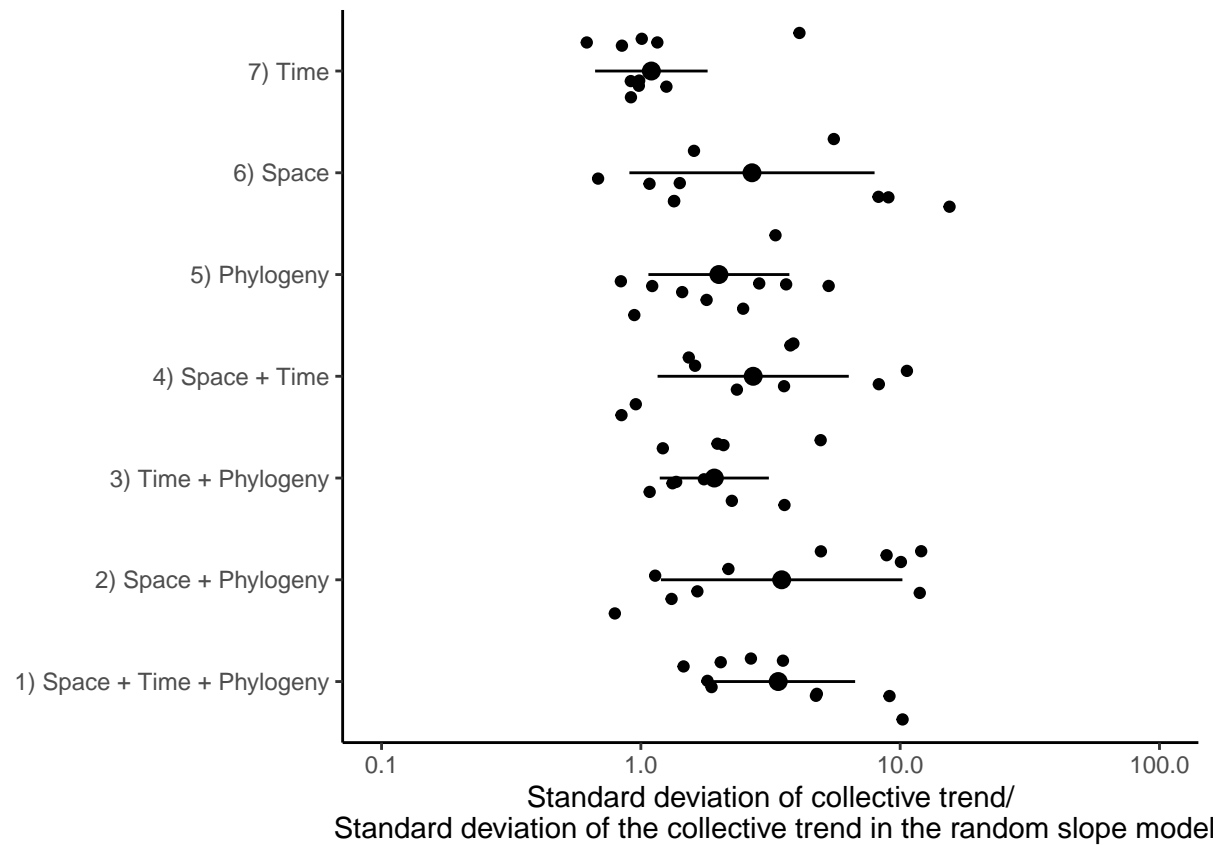
```

df_tmp4 = df_tmp3 %>%
  group_by(model) %>%
  summarise(mn = mean(log(sd_div)), sd = sd(log(sd_div)))

```

Plot variance differences

```
ggplot() +
  geom_jitter(data = df_tmp3, aes(y = model, x = sd_div)) +
  geom_pointrange(data = df_tmp4, aes(y = model, x = exp(mn), xmin = exp(mn-sd), xmax = exp(mn+sd))) +
  coord_cartesian(xlim = c(0.1,100)) +
  labs(x = "Standard deviation of collective trend/\n Standard deviation of the collective trend in the random slope model", y = "") +
  scale_x_log10() +
  theme_classic()
```



```
ggsave("../outputs/figures/extended_data_fig2.tiff", width = 9, height = 7, device='tiff', dpi=300)
```

Prediction - trends

Here we take the predictions of missing trends in each dataset and estimate the median absolute error in prediction. This predictive error is then summarised across datasets, and then compared across models

```
miss_t = read.csv("../outputs/model_summary_miss_trends.csv")
mae_full = NULL
for(a in names(analysis_list)[c(1:10)]){
  tmp_df = subset(miss_t, code == a)
  tmp_mae_full = data.frame(
    code = a,
    mae2 = median(abs(abs(tmp_df$tr - tmp_df$m2)/tmp_df$m2)*100),
    mae3 = median(abs(abs(tmp_df$tr - tmp_df$m3)/tmp_df$m3)*100)
  )
  mae_full = rbind(mae_full, tmp_mae_full)
}

mean(mae_full$mae2)
```

```
## [1] 86.68759
```

```
sd(mae_full$mae2)
```

```
## [1] 213.8192
```

```
median(mae_full$mae2)
```

```
## [1] 14.16661
```

```
mean(mae_full$mae3)
```

```
## [1] 18.33057
```

```
sd(mae_full$mae3)
```

```
## [1] 11.55232
```

```
median(mae_full$mae3)
```

```
## [1] 12.71837
```

```
mean(mae_full$mae2)/mean(mae_full$mae3)
```

```
## [1] 4.729126
```

```
median(mae_full$mae2)/median(mae_full$mae3)
```

```
## [1] 1.11387
```

Prediction - abundances

Here we take the predictions of missing abundance in each dataset and estimate the median absolute error in prediction. This predictive error is then summarised across datasets, and then compared across models

```
cmb_preds = NULL
for(a in names(analysis_list)[c(1:10)]){
  print(a)
  tmp_df_predict = readRDS(paste0("../outputs/model_output_miss_abundance_", a, ".rds"))

  tmp_preds = data.frame(
    code = a,
    log_abundance = tmp_df_predict[[4]]$log_abundance,
    mean_log = tmp_df_predict[[4]]$mean_log,
    miss = ifelse(is.na(tmp_df_predict[[4]]$log_abundance_trim), "Missing", "Complete"),
    pred1 = tmp_df_predict[[1]]$summary.fitted.values$mean,
    pred2 = tmp_df_predict[[2]]$summary.fitted.values$mean,
    pred3 = tmp_df_predict[[3]]$summary.fitted.values$mean
  )
  cmb_preds = rbind(cmb_preds, tmp_preds)
}
```

```
## [1] "BioTIME"
## [1] "CaPTrends"
## [1] "EA_NFPD"
## [1] "FishGlob"
## [1] "LPI"
## [1] "NA_m_BBS"
## [1] "Pilotto"
## [1] "ReSurvey_Germany"
## [1] "RivFish"
## [1] "TimeFISH"
```

```
cmb_preds = subset(cmb_preds, miss == "Missing")
cmb_preds$pred2 = cmb_preds$pred2 + cmb_preds$mean_log
cmb_preds$pred3 = cmb_preds$pred3 + cmb_preds$mean_log
cmb_preds$diff1 = abs(abs(cmb_preds$log_abundance - cmb_preds$pred1)/cmb_preds$pred1)*100
cmb_preds$diff2 = abs(abs(cmb_preds$log_abundance - cmb_preds$pred2)/cmb_preds$pred2)*100
cmb_preds$diff3 = abs(abs(cmb_preds$log_abundance - cmb_preds$pred3)/cmb_preds$pred3)*100
```

```
sum_preds = cmb_preds %>%
  group_by(code) %>%
  summarise(mn1 = median(diff1), mn2 = median(diff2), mn3 = median(diff3))
#sum_preds$mn1 = signif(sum_preds$mn1, 3)
#sum_preds$mn2 = signif(sum_preds$mn2, 3)
#sum_preds$mn3 = signif(sum_preds$mn3, 3)
mean(sum_preds$mn1)
```

```
## [1] 29.09903
```

```
sd(sum_preds$mn1)
```

```
## [1] 26.47071
```

```
median(sum_preds$mn1)
```

```
## [1] 20.01756
```

```
mean(sum_preds$mn2)
```

```
## [1] 20.39236
```

```
sd(sum_preds$mn2)
```

```
## [1] 14.28271
```

```
median(sum_preds$mn2)
```

```
## [1] 17.77691
```

```
mean(sum_preds$mn3)
```

```
## [1] 17.51973
```

```
sd(sum_preds$mn3)
```

```
## [1] 9.247632
```

```
median(sum_preds$mn3)
```

```
## [1] 17.51286
```

```
mean(sum_preds$mn1)/mean(sum_preds$mn3)
```

```
## [1] 1.660929
```

```
mean(sum_preds$mn2)/mean(sum_preds$mn3)
```

```
## [1] 1.163965
```


Driver of uncertainty

We now explore how the variance captured by spatial and phylogenetic covariance terms (relative to combined variance of covariance + hierarchical terms) impacts our inference

```
model_summary_core = read.csv("../outputs/model_summary.csv")

stre2 = data.frame(
  diff_rate = c((model_summary_core[which(model_summary_core$model == 2), "coef"])) - (model_summary_core[which(model_summary_core$model == 2), "coef"]),
  diff_sd = c((model_summary_core[which(model_summary_core$model == 3), "coef_sd"])) / (model_summary_core[which(model_summary_core$model == 2), "coef_sd"]),
  var_tip = rowSums((model_summary_core[which(model_summary_core$model == 3), c("tip", "gen", "tip_h")])),
  var_phy = (model_summary_core[which(model_summary_core$model == 3), c("tip_h")]),
  var_loc = rowSums((model_summary_core[which(model_summary_core$model == 3), c("sit", "squ", "sit_h")])),
  var_spa = (model_summary_core[which(model_summary_core$model == 3), c("sit_h")])
)
stre2$phy_h = stre2$var_phy / stre2$var_tip
stre2$spa_h = stre2$var_spa / stre2$var_loc
stre2$mean_h = rowMeans(stre2[, c("phy_h", "spa_h")])

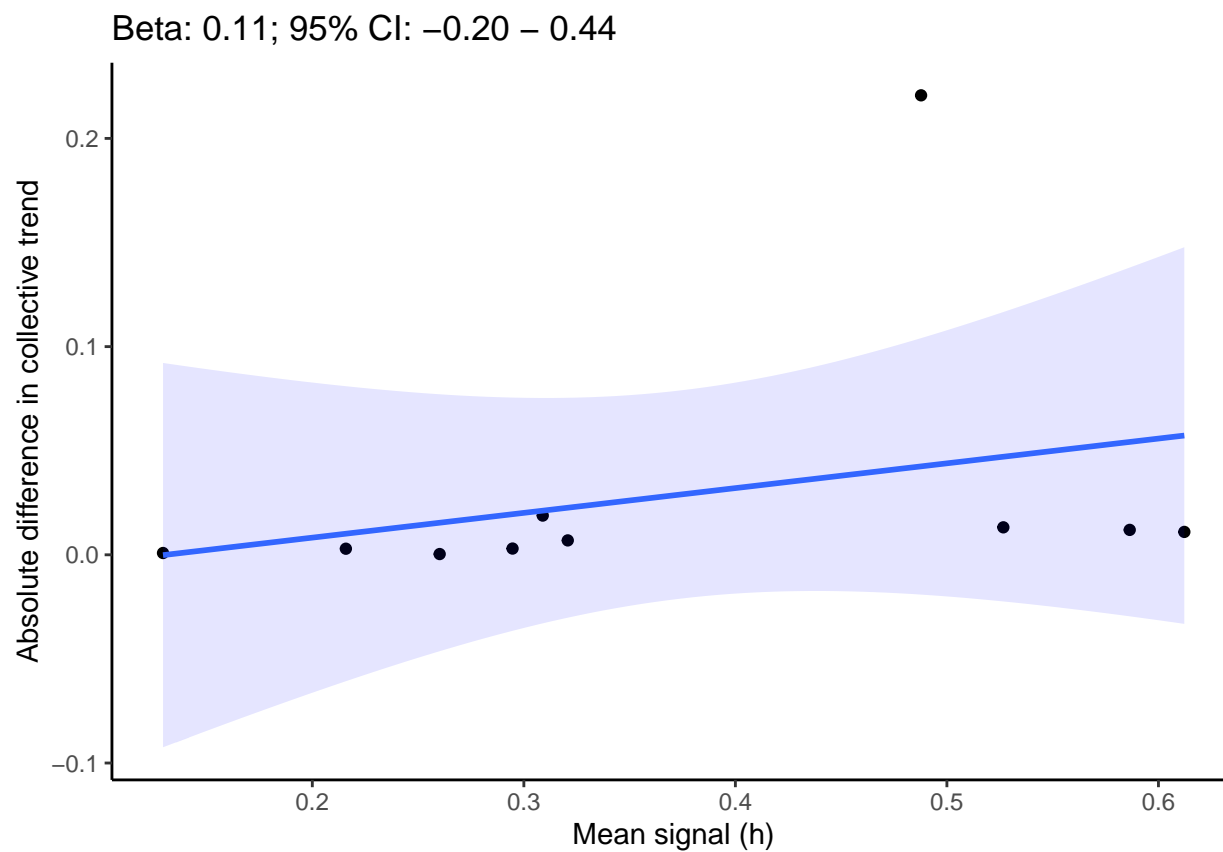
summary(lm(abs(diff_rate) ~ mean_h, data = stre2))

##
## Call:
## lm(formula = abs(diff_rate) ~ mean_h, data = stre2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.046289 -0.029550 -0.015364 -0.003568  0.178205
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.01554    0.05581  -0.278   0.788
## mean_h       0.11895    0.13738   0.866   0.412
##
## Residual standard error: 0.06859 on 8 degrees of freedom
## Multiple R-squared:  0.08569,    Adjusted R-squared:  -0.0286
## F-statistic: 0.7498 on 1 and 8 DF,  p-value: 0.4118
```

```
confint(lm(abs(diff_rate) ~ mean_h, data = stre2))
```

```
##              2.5 %    97.5 %  
## (Intercept) -0.1442316 0.1131511  
## mean_h      -0.1978410 0.4357477
```

```
ggplot(stre2) +  
  geom_point(aes(x = mean_h, y = abs(diff_rate))) +  
  geom_smooth(aes(x = mean_h, y = abs(diff_rate)), method = "lm", fill = "blue", alpha = 0.1) +  
  labs(x = "Mean signal (h)", y = "Absolute difference in collective trend", title = "Beta: 0.11; 95% CI: -0.20 - 0.44") +  
  theme_classic()
```



```
summary(lm(log10(diff_sd) ~ mean_h, data = stre2))
```

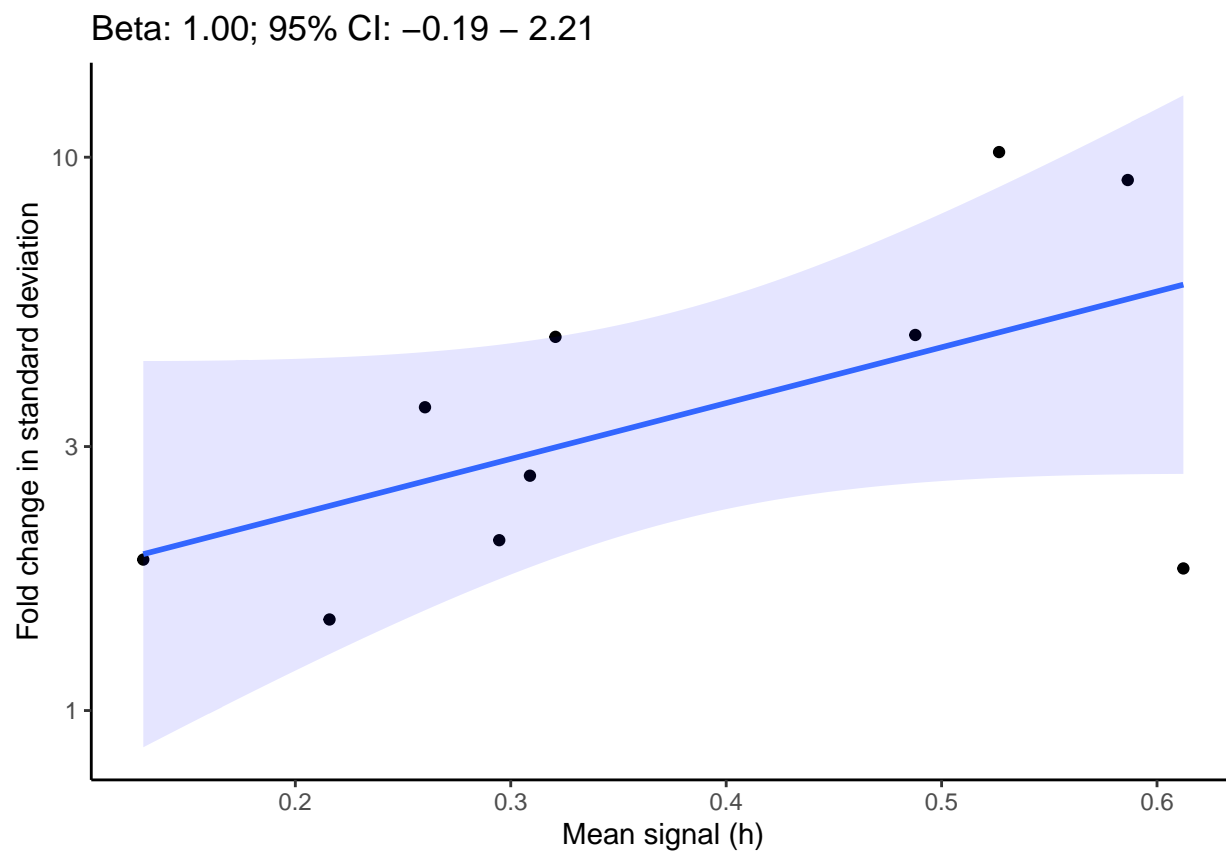
```
##
## Call:
## lm(formula = log10(diff_sd) ~ mean_h, data = stre2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51293 -0.11592  0.01238  0.18315  0.32588
##
```

```
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.1521     0.2110   0.721  0.4917
## mean_h       1.0088     0.5195   1.942  0.0881 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2594 on 8 degrees of freedom
## Multiple R-squared:  0.3203, Adjusted R-squared:  0.2354
## F-statistic: 3.771 on 1 and 8 DF,  p-value: 0.0881
```

```
confint(lm(log10(diff_sd) ~ mean_h, data = stre2))
```

```
##           2.5 %    97.5 %
## (Intercept) -0.3345974 0.6387605
## mean_h      -0.1892213 2.2068549
```

```
ggplot(stre2) +
  geom_point(aes(x = mean_h, y = (diff_sd))) +
  geom_smooth(aes(x = mean_h, y = (diff_sd)), method = "lm", fill = "blue", alpha = 0.1) +
  scale_y_log10() +
  labs(x = "Mean signal (h)", y = "Fold change in standard deviation", title = "Beta: 1.00; 95% CI: -0.19 - 2.21") +
  theme_classic()
```



```
ggsave("../outputs/figures/extended_data_fig1.tiff",width = 4, height = 4, device='tiff', dpi=300)
```

Reproducibility

Date rendered

```
## [1] "2024-02-07 19:27:16 GMT"
```

Session info

```

## R version 4.2.3 (2023-03-15)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Monterey 12.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats      graphics  grDevices  utils      datasets  methods
## [8] base
##
## other attached packages:
## [1] brinla_0.1.0      INLA_23.04.24      foreach_1.5.2      Matrix_1.5-3
## [5] phytools_1.5-1    maps_3.4.1         ape_5.7-1          ggtree_3.6.2
## [9] RColorBrewer_1.1-3 raster_3.6-23      maptools_1.1-8     sp_2.1-1
## [13] tidyr_1.3.0       ggpubr_0.6.0       ggplot2_3.4.4      dplyr_1.1.2
##
## loaded via a namespace (and not attached):
## [1] nlme_3.1-162      sf_1.0-13          lubridate_1.9.2
## [4] doParallel_1.0.17 numDeriv_2016.8-1.1 tools_4.2.3
## [7] backports_1.4.1   utf8_1.2.4         R6_2.5.1
## [10] KernSmooth_2.23-20 mgcv_1.8-42        DBI_1.1.3
## [13] lazyeval_0.2.2    colorspace_2.1-0   withr_2.5.2
## [16] gridExtra_2.3     tidyselect_1.2.0   mnormt_2.1.1
## [19] phangorn_2.11.1   compiler_4.2.3     textshaping_0.3.6
## [22] cli_3.6.1         expm_0.999-7       labeling_0.4.3
## [25] scales_1.2.1      classInt_0.4-9     quadprog_1.5-8
## [28] proxy_0.4-27      systemfonts_1.0.4  digest_0.6.33
## [31] yulab.utils_0.0.9 foreign_0.8-84     rmarkdown_2.23
## [34] pkgconfig_2.0.3   htmltools_0.5.5    plotrix_3.8-2
## [37] fastmap_1.1.1     highr_0.10         rlang_1.1.2
## [40] rstudioapi_0.15.0 optimParallel_1.0-2 gridGraphics_0.5-1
## [43] generics_0.1.3    farver_2.1.1       combinat_0.0-8
## [46] jsonlite_1.8.7    car_3.1-2          magrittr_2.0.3
## [49] ggplotify_0.1.2   patchwork_1.1.3    Rcpp_1.0.11

```

## [52] munsell_0.5.0	fansi_1.0.5	abind_1.4-5
## [55] lifecycle_1.0.4	terra_1.7-39	scatterplot3d_0.3-44
## [58] yaml_2.3.7	carData_3.0-5	clusterGeneration_1.3.7
## [61] MASS_7.3-58.2	grid_4.2.3	lattice_0.20-45
## [64] cowplot_1.1.1	splines_4.2.3	knitr_1.43
## [67] pillar_1.9.0	igraph_1.5.0	ggsignif_0.6.4
## [70] codetools_0.2-19	fastmatch_1.1-3	glue_1.6.2
## [73] evaluate_0.21	ggfun_0.1.2	vctrs_0.6.4
## [76] treeio_1.22.0	gtable_0.3.4	purrr_1.0.1
## [79] cachem_1.0.8	xfun_0.39	broom_1.0.5
## [82] e1071_1.7-13	tidytree_0.4.5	coda_0.19-4
## [85] rstatix_0.7.2	class_7.3-21	ragg_1.2.5
## [88] tibble_3.2.1	iterators_1.0.14	aplot_0.2.0
## [91] memoise_2.0.1	units_0.8-2	timechange_0.2.0