🥹 Kapitola 1: Kdy a jak použít nepřímou ordinaci

Kryštof & Klára: Analýza dat v ekologii společenstev, kuchařka.

Kapitola 1: Kdy a jak použít nepřímou ordinaci

Kryštof Chytrý & Klára Klinkovská

2024-10-30: verze 1

Když máme nějaké vzorky charakterizované více proměnnými (tj. druhy, charakteristikami prostředí, nebo vzácněji morfometrickými měřeními) a chceme vědět, jakým způsobem se liší, nebo jaké jsou nejsilnější gradienty podle kterých se vzorky obměňují, můžeme použít některou z nepřímých ordinací (PCA, CA, PCoA, NMDS). Do nepřímých ordinací nevkládáme žádné naše představy o faktorech, které naše vzorky ovlivňují. Pokud bychom takové představy měli a chtěli otestovat, zda-li jsou správné, například kdybychom chtěli vědět, jak společenstvo reaguje na gradient vlhkosti nebo různé typy zásahů, nebo jestli se naše společenstvo mění v čase, pokud máme opakovaná pozorování, měli bychom použít přímé ordinace jako jsou RDA, CCA nebo dbRDA.

Teorie: jak si vybrat mezi PCA, CA, PCoA a NMDS

Snad platí obecně, že když si pro proměnné prostředí zvolíte PCA a pro druhovou matici PCoA, tak nic nepokazíte. Pokud tedy máte dojem, že vaše data nijak nevybočují z běžných dat, se kterými v ekologii společenstev pracujeme (kontinuální proměnné prostředí pro PCA, fytocenologické snímky, vzorky společenstev rozsivek nebo hmyzu pro PCoA), tak se následujícím odstavcem nemusíte moc trápit. 🙂

To ovšem neznamená, že by ostatní metody byly nějakým způsobem špatné, jsou jen podstatně méně univerzální. Hlavní a v podstatě jediný rozdíl mezi PCA, CA a PCoA je metoda výpočtu nepodobností. Zbytek algoritmu, který rozkládá variabilitu na jednotlivé ordinační osy je stejný. PCA je založená na euklidovské vzdálenosti, jinými slovy tedy k výpočtu používá přímo hodnoty z druhové matice nebo matice proměnných prostředí. Tím vzniká tzv. problém dvojitých nul, tedy situace, kdy absence stejného druhu dělá dva vzorky podobnější. Kdybychom použili třeba Bray-Curtisovu matici nepodobností, absence stejného druhu podobnost dvou vzorků neovlivňuje. Nelze jednoznačně říct, co z toho je více správně. Můžeme si ale popsat, jaký vliv to bude mít na naše data. Pokud máme krátký gradient a silný kontrast mezi dvěma typy stanovišť, nebo typy zásahů, skutečnost, že sdílená absence stejného druhu zvyšuje podobnost mezi vzorky nám nemusí vadit a někdy

nám může dokonce pomoct. U proměnných prostředí tento problém nenastává, protože tam obvykle míváme nenulové hodnoty pro všechny lokality/vzorky. Když ale v druhové matici máme větší počet stanovišť a obecně více heterogenní datový soubor, tak nám dvojité absence můžou výsledek modelu zásadně ovlivnit a vygenerovat výsledek, který nedává úplně dobrý smysl. V takovém případě je CA, PCoA nebo NMDS lepší metoda.

Kryštof ani Klárka nevědí, kdy by použili CA nebo DCA, až to jednou zjistí, tak to sem doplní.

Když se rozhodneme použít metodu založenou na nepodobnostní matici, máme na výběr mezi PCoA a NMDS. Zatímco PCoA je více konvenční ordinace založená na rozkladu variability po eigenvektorech, tedy jednotlivé osy mají význam a lze je interpretovat nezávisle na sobě, NMDS funguje úplně jinak. Pro NMDS nejprve vymezíme ring, ve kterém se může realizovat, ve formě počtu ordiačních os. NMDS se poté snaží iterativním algoritmem do toho prostoru vměstnat naše vzorky tak, aby minimalizovala rozdíl mezi vzdálenostmi mezi lokalitami v tomhle ringu a jejich skutečnými nepodobnostmi. Tento rozdíl pak nějakým způsobem sečte a vyjádří ho hodnotou tzv. stresu. Když je hodnota stresu příliš velká, tedy nad 0.2, v nejhorším případě 0.3 (arbitrární threshold), hodí se dát NMDS více prostoru ve formě další osy, a tím její stres snížit. Většinou zobrazujeme pouze 2 a maximálně 3 osy, protože ve více osách se špatně orientuje. Někdy se ale dělá to, že modelu dáme k dispozici o jednu osu navíc než chceme zobrazit, prostě abychom mu umožnili někam tu zbytkovou variabilitu uklidit.

Tohle je moko skalní. Moko je bratranec morčat a kapybar, žije v jižní Americe a o ordinacích neví vůbec nic. Nebuď jako moko a pokračuj ve čtení.

Praxe: jak na ordinační analýzy v Rku

Abychom mohli data o druhovém složení společenstev modelovat pomocí ordinací, musíme je uložit do tabulky (které říkáme druhová matice) v tzv. wide formátu:

- Druhy po sloupcích
- Vzorky/lokality po řádcích
- Hodnoty presence a absence (1 a 0) nebo abundance (obvykle logaritmicky transformované, protože jinak mají méně nebo i běžně časté druhy v modelu zanedbatelnou váhu)

Druhy →

Vzorky/lokality ↓

	Acer campestre_7	Acer pseudoplatanus_7	Achillea millefolium agg._6	Acinos arvensis_6	Adonis vernalis_6	Aegopodium podagraria_6	Agrimonia eupatoria_6
1	0	0	2	0	2	0	0
2	0	0	2	0	0	0	0
3	0	0	2	0	2	0	0
4	0	0	2	0	2	0	0
5	0	0	2	0	3	0	0
6	0	0	2	0	2	0	0
7	0	0	0	0	0	0	0
8	1	0	2	0	0	0	0
9	0	0	1	0	0	0	0
10	0	0	2	0	0	0	0

Při nahrávání druhové matice do R musíme vzít v potaz, jakého typu je soubor, ve kterém máme druhovou matici (jakou má koncovku). Pokud je to csv soubor (textový soubor s oddělovači), použijeme funkce `read_csv()` nebo `read.csv()`. Tyto funkce jsou téměř identické, ale ta první vám data uloží do R jako "tibble", ta druhá jako "data.frame". Můžete si zkusit porovnat rozdíly a pokračovat s formátem, který vám více vyhovuje, je to jen otázka vkusu. Funkce `read_csv()` je součástí `tidyverse`. Pokud máme data oddělená středníky a ne čárkou, což může někdy generovat český excel, můžeme použít funkci `read_csv2()`. Pokud se jedná o excelovský file, který má obvykle koncovku xlsx, můžeme použít funkci `read_excel()` nebo `read_xlsx()`, které jsou v knihovně `readxl`. U těchto funkcí specifikujeme argument `sheet`. Můžeme použít jméno listu (sheetu) nebo jeho pořadí.

```
spe <- read_excel('data/LimestoneQuarries_Aucheno/Aucheno_sum.xlsx', sheet = 1)
```

Teď máme v `spe` uloženou druhovou matici. Před fitováním ordinace můžeme, ale nemusíme, provést několik mezikroků, které záleží na typu dat, se kterými pracujeme. Například tato křísí data mají velké množství druhů, které jsou jen na jedné lokalitě, což by neúměrně zvětšovalo nepodobnost mezi vzorky a vznikl by tím trochu chaos. Proto tyto druhy odstraníme. To můžeme udělat takto:

```
spe <- spe[, colSums(spe > 0) > 1]
```

Funkce `colSums(spe > 0)` sečte počet nenulových hodnot ve sloupcích matice, tedy udělá vektor frekvencí druhů (počet vzorků, ve kterých se druh vyskytuje). Potom se zeptáme, kde je frekvence větší než jedna a dostaneme logický vektor obsahující TRUE a FALSE podle toho, jestli jednotlivé druhy tuto podmínku splňují. Na základě toho vybereme pouze ty sloupce z druhové tabulky, pro které je podmínka TRUE, tedy se vyskytují na více než jedné lokalitě. To stejné se dá udělat i v `dplyr` chainu, možná trošku elegantněji. Používejte prosím to, co se vám líbí víc.

```
spe <- spe |>
  select(where(~sum(.x > 0) > 1))
```

V druhové matici máme data o abundancích, které bude lepší logaritmicky transformovat. Protože data obsahují nuly, použijeme funkci `log1p()`, která ke každé hodnotě před logaritmováním přičte 1. Tento krok můžeme udělat buď samostatně před vlastní analýzou:

```
spe.log <- log1p(spe)
```

Anebo jej rovnou integrujeme do funkce, která nám počítá ordinaci:

```
capscale(log1p(spe) ~ 1,...)
```

V této fázi můžeme začít se samotnou ordinací. PCoA počítáme funkcí `capscale()`, která má tyto argumenty. Jako vždy je ale pro hlubší pochopení skvělé si přečíst dokumentaci funkce: [capscale function - RDocumentation].

- `formula`: před vlnovku zadáváme druhovou matici. Když počítáme PCoA, tak za ni umístíme jedničku, formula tedy vypadá takhle (`spe ~ 1`). Pokud bychom počítali

přímou ordinaci dbRDA, použijeme stejnou funkci, ale místo jedničky vkládáme prediktory.

- `method`: pro metodu použitou k výpočtu matice nepodobností. Pokud bychom použili `"euclidean"` získali bychom PCA místo PCoA. Methoda `"bray"` počítá Bray-Curtisovu nepodobnost, případně Sorensenovu, pokud máme prezenční/absenční data. Tady si můžeme připomenout, že Bray-Curtis a Sorensen jsou úplně stejné algoritmy, lišící se jen tím, jestli počítají s abundancemi nebo jen jedničkami a nulami pro prezenci/absenci.
- `sqrt.dist`: odmocnění nepodobností, je to jedna z možností, jak zabránit vzniku imaginárních os s negativními eigenvalues. Nejprve tento argument ignorujte, když vám ale model bude říkat, že máte nějakou imaginary variabilitu, tak to tam prostě přidejte a většinou se to tím spraví.
- `binary`: zatím nevím, co to přesně dělá, ale prostě to tam dejte. :-)

```
pco <- capscale(spe.log ~ 1, method = 'bray', sqrt.dist = T, binary = F)
```

V objektu `pco` nyní máme výsledek ordinace. Tím, že ho pustíme do konzole si ho můžeme prohlédnout. Moc od toho ale nečekejte. :-) Vlastně tam uvidíte akorát eigenhodnoty pro jednotlivé osy a jejich součet. Tato čísla sama o sobě nemají žádný velký význam (respektive nějaký mají, ale nám to není moc platné). Když ale eigenhodnoty jednotlivých os vydělíme celkovou variabilitou (tedy součtem všech eigenvalues), dozvíme se, jaký podíl variability vysvětlují jednotlivé osy. Pro převedení na procenta vynásobíme 100. To lze udělat takhle elegantně:

```
eigenvals(pco) / sum(eigenvals(pco)) * 100
```

Co s těmito hodnotami? V prvé řadě by nám měly posloužit k výběru počtu os, které chceme zobrazit. Ten vybíráme tak, že se díváme, kde vysvětlená variabilita náhle rychle klesne. Většinou ale nezobrazujeme více než tři osy, takže nám to akorát tak napoví, jestli zobrazit dvě nebo tři osy. I když někdy téměř všechnu vysvětlenou variabilitu vysvětlí jedna osa, vždycky zobrazujeme minimálně dvě. Pak je také dobrým zvykem informaci o vysvětlené variabilitě přidat do popisků os, abychom ukázali, jak dobře tyto osy společenstvo reprezentují. Jaké hodnoty vysvětlené variability jsou dobré a které špatné? To se těžko popisuje. 🙂 Záleží na počtu vzorků a celkové variabilitě. Když máte málo vzorků (třeba 10), čekejte i poměrně vysoké procento vysvětlené variability dosahující až 50% a možná i víc. S větším množstvím vzorků (50 až 100) i druhů ale tohle číslo málokdy přesahuje 20%. Další

trošku složitější literatura na tohle téma: On the variation explained by ordination and constrained ordination axes

Ordinace jako obrázek (s pomocí ggplotu)

Když teď máme model spočítaný a víme, kolik os chceme zobrazit, je na čase z modelu udělat obrázek. V dnešní době je standardním postupem používat pro vizualizaci ggplot, který je součástí `tidyverse` package. Bohužel v roce 2024 zatím není na Ústavu botaniky a zoologie žádný předmět, kde byste se ggplot naučili slušně používat. Mrzí nás to a pevně doufáme, že se to v brzké budoucnosti změní. Nezoufejte ale, naštěstí existuje nepřeberné množství velmi kvalitních zdrojů, kterým kraluje tato online učebnice (odkaz přímo na kapitolu o ggplotu): 1 Data visualization – R for Data Science (2e). Ordinační graf totiž není vůbec nic složitějšího než normální scatterplot (bodový graf). Nejprve si teda musíme z modelu extrahovat souřadnice pro naše vzorky na osách, které chceme v grafu zobrazit, v našem případě na první a druhé ose. A pak s nimi můžeme zacházet jako s jakýmikoli jinými hodnotami, které chceme zobrazit v grafu. K tomu slouží funkce `scores()`, která má následující argumenty:

- `choices`: pro které ordinační osy chci vypsat souřadnice v ordinačním prostoru, default je první dvě osy, pokud chci i další, musím nadefinovat, např. `choices = 1:3` pro první tři osy
- `display`: možnosti `"sites"` nebo `"species"`, podle toho, jestli chci vytáhnout souřadnice pro vzorky nebo pro druhy
- `scaling`: na co zaměříme pozornost, pokud je škálování 1 (`"sites"`), lze interpretovat vzdálenost mezi vzorky, při možnosti 2 (`"species"`) zůstávají zachované úhly vyjadřující korelace mezi druhy (proměnnými), při škálování 3 (`"symmetric"`) nelze přesně interpretovat ani jedno

- `tidy`: pokud je `TRUE`, převede vzniklou matici na data.frame, kde jsou společně druhy i lokality, nefunguje pak výběr zobrazení vzorků pomocí argumentu `display`. Klidně to zkuste, prohlédněte si výstup a podle toho se rozmyslete, který se vám líbí víc.

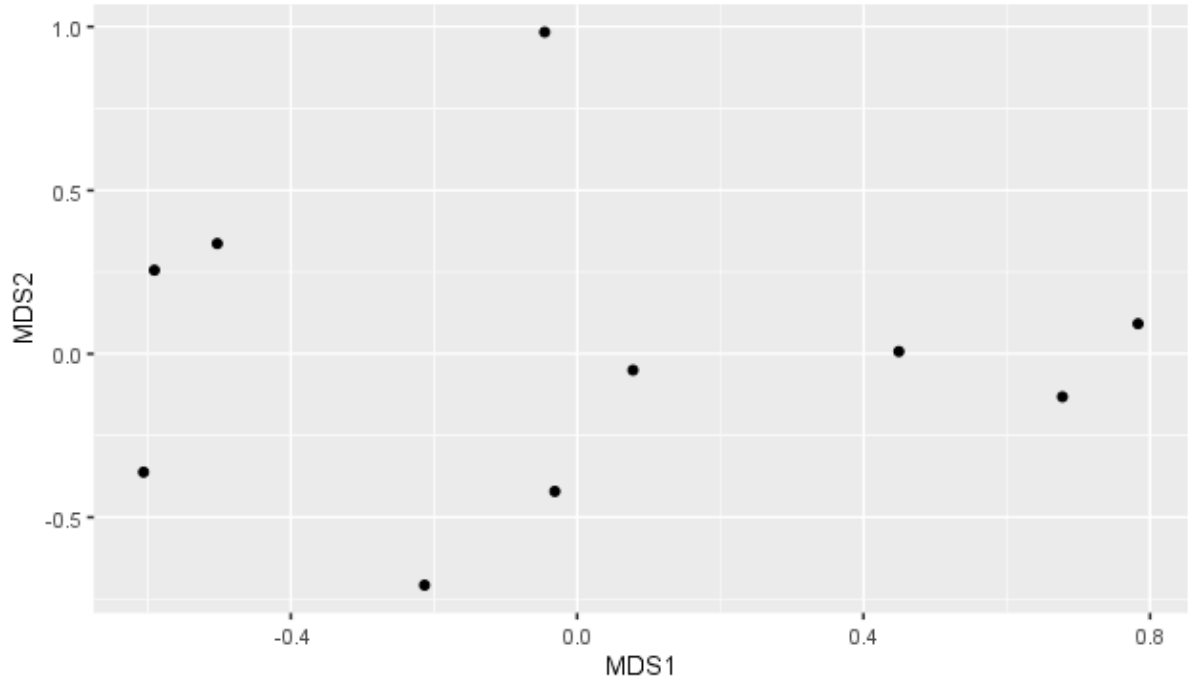
```
site.scores <- scores(pco, display = 'sites', scaling = 'symm') |>
 as_tibble()
```

Pro `tidy = T` by výběr skórů pro vzorky vypadal takto:

```
site.scores <- scores(pco, scaling = 'symm', tidy = T) |>
  filter(score == 'sites')
```

Stejnou funkci můžeme použít i pro extrahování skóre pro jednotlivé druhy, jen změníme argument `display` z hodnoty "`sites`" na "`species`".
Pro úplně nejjednodušší zobrazení ordinačního modelu můžeme použít následující call:

```
site.scores |>
 ggplot(aes(MDS1, MDS2)) +
 geom_point()
```



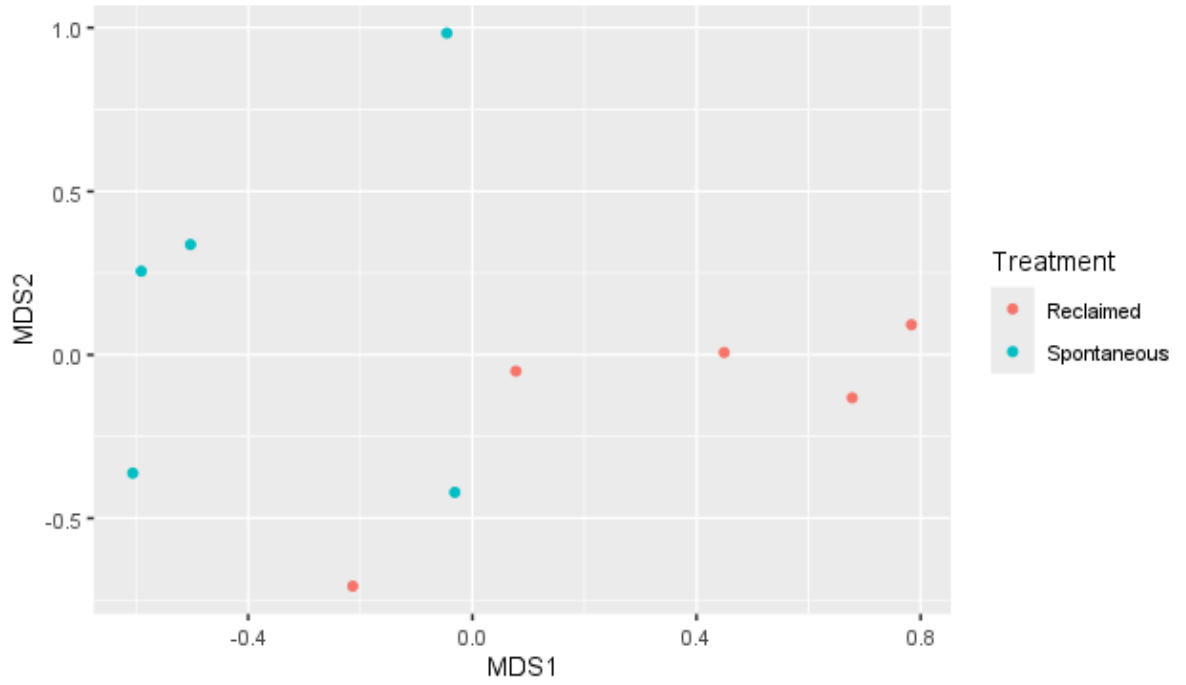
Ten vám vygeneruje tento velmi jednoduchý graf. Co z něj můžeme vyčíst? Zatím nic. 🙂 Abychom pochopili, které lokality za našimi symboly stojí, musíme si na skóre lokalit napojit hlavičková data. Ty si nejprve nahrajeme do R stejně jako druhovou matici a pak použijeme příkaz `bind_cols()`, kterým prostě spojíme, bindneme, obě tabulky k sobě. Vycházíme při tom z předpokladu, že vaše druhová matice byla seřazená úplně stejně jako vaše tabulka proměnných prostředí. Pokud to tak nebylo, máte velký problém. Radši to teda ještě jednou zkontrolujte.

```
env <- read_excel('data/LimestoneQuarries_Aucheno/Aucheno_sum.xlsx',
            sheet = 2)

bind_cols(env, site.scores)
```

```
# A tibble: 10 x 6
   site      Treatment   SpRich `Celkem N`    MDS1     MDS2
   <chr>     <chr>        <dbl>      <dbl>   <dbl>    <dbl>
 1 Cikanka   Spontaneous     24         86 -0.0453  0.984
 2 Cikanka   Reclaimed       32        127  0.784   0.0916
 3 Hvizdalka Spontaneous     28        190 -0.0311 -0.421
 4 Hvizdalka Reclaimed       28        412  0.449   0.00660
 5 Koneprusy Spontaneous     29        179 -0.503   0.337
 6 Koneprusy Reclaimed       31        275  0.0779 -0.0503
 7 Menany    Spontaneous     17        157 -0.606  -0.362
 8 Menany    Reclaimed       26        124 -0.213  -0.708
 9 Paraple   Reclaimed       26        193  0.678  -0.132
10 Paraple   Spontaneous     21         78 -0.591   0.255
```

Když příkaz `bind_cols()` pustíme pouze do konzole, akorát nám ukáže výstup. Abychom mohli pokračovat, buď ho musíme uložit do nového objektu, nebo jej prostě přiložíme k chainu na generování obrázku. A rovnou navíc přidáme k řádku, který generuje puntíky `aes(colour = Treatment)`, abychom puntíky obarvili podle proměnné Treatment, která byla uložená v datech o proměnných prostředí.

```
bind_cols(env, site.scores) |>
 ggplot(aes(MDS1, MDS2)) +
 geom_point(aes(colour = Treatment))
```



No a super! Začíná to vypadat jako skutečná ordinace.

Nevyžádaný career advice. Tenhle graf svoji práci udělá a možná by v nejhorším případě stačil i do diplomky. Přiznejme si ale, že to prostě není ono. Vždyť grafy dělané v R mohou být velmi informativní a zároveň elegantní (Top R Graph Examples: A Curated Collection). Může v nich být nejen poznání, ale i kousek umění. Existují i grafy interaktivní (klaraklink.shinyapps.io/pladias_occ_models/), kde můžeme v hotovém grafu vybírat, co přesně zobrazit. Udělat ten nejlepší graf přesně podle vlastních představ se ale bohužel budete muset naučit sami. **Jestli na to ale máte jen trochu času, fakt to udělejte**. Nemyslím, tím, že to tak moc potřebujete pro dokončení tohoto předmětu, pokud se ale chcete věnovat ekologii nebo biologii obecně (včetně ochrany přírody), zpracování dat je prostě strašně důležité, protože tam ta pravá ekologie a pochopení přírody teprve začíná. Navíc to posouvá vaši pozici na pracovním trhu o kategorii úplně někam jinam. Přitom ta cesta není tak složitá a fakt stačí zkouknout jen pár videí na YT a hlavně zkoušet, zkoušet, odpočívat, dát si kafe (třeba od Casino Mocca | Kávépörkölő) a zase zkoušet. 🙂 Co všechno můžeme s ggplotem dělat, jak měnit barvy, tvary a velikosti symbolů, jaké různé geometrie tam můžete vkládat si prosím přečtěte v již výše sdílené knize R4DS (online!): 1 Data visualization – R for Data Science (2e), hodně se toho také můžete dozvědět v tomhle cheasheetu: Data visualization with ggplot2 :: Cheat Sheet. Existuje taky asi tisíc videí, které vám můžou strašně moc pomoct, stačí v YT vyhledávat ggplot2. Skvělé video vysvětlující základní princip ggplotu je třeba tohle: ggplot2 explained in 5 minutes!, stejně jako ostatní videa z kanálu Online Course on the Basics of R for Ecologists!

Tohle je vombat severní. Vombat severní má stejně jako vombat obecný hranatá hovínka a o ordinacích neví vůbec nic. No a teď je na pokraji vyhynutí. Naštěstí se jej ale daří chovat v zoologických zahradách. Nebuď jako vombat severní a pokračuj ve čtení.

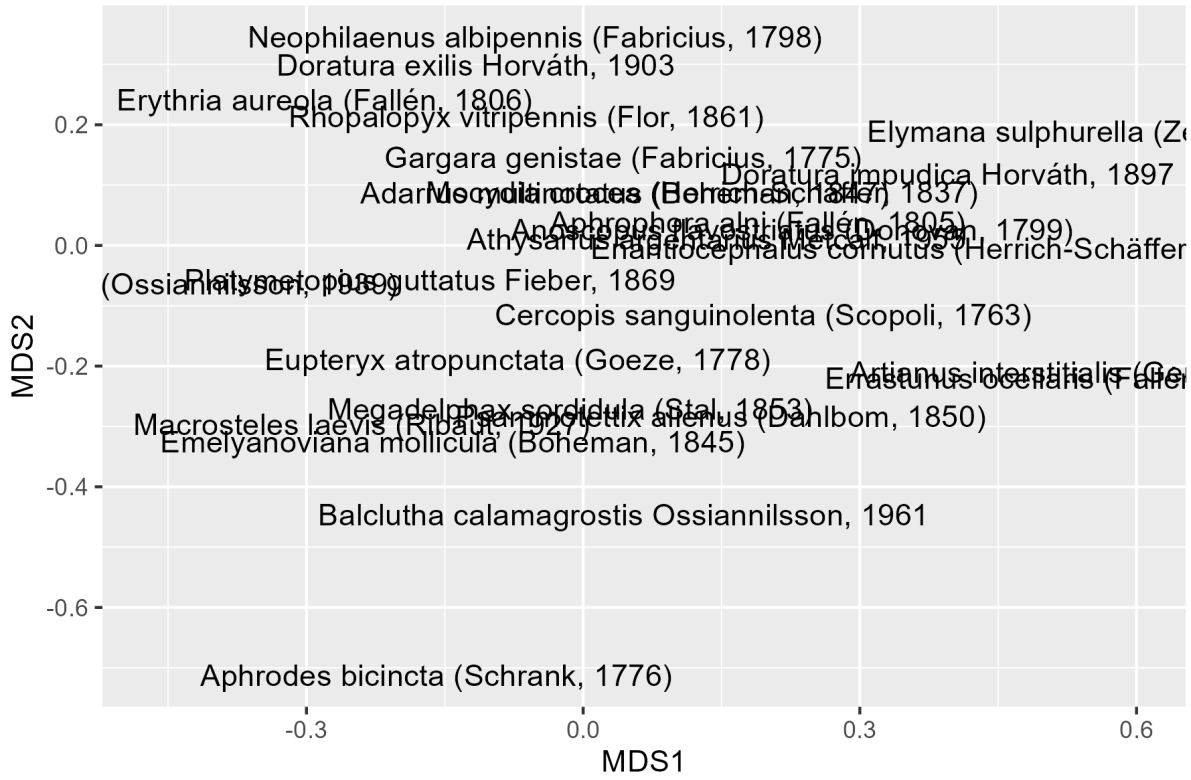
Druhy v ordinaci

Stejně jako vzorky můžeme v ordinačním prostoru zobrazit také druhy. Abychom dostali souřadnice jednotlivých druhů v ordinačním prostoru, použijeme opět příkaz `scores()`, jen změníme argument `display` na `"species"`. Zároveň do příkazu `as_tibble()`, kterým převádíme matici na tibble přidáme argument `rownames`, který říká, že se mají jména druhů uložená v rownames uložit do nového sloupečku nazvaného `"species_name"`:

```
species.scores <- scores(pco, display = 'species', scaling = 'symm') |>
 as_tibble(rownames = 'species_name')
```

Už teď můžeme promítnout druhy do ordinačního diagramu, ale nebude to moc pěkné:

```
species.scores |>
 ggplot(aes(MDS1, MDS2)) +
 geom_text(aes(label = species_name))
```



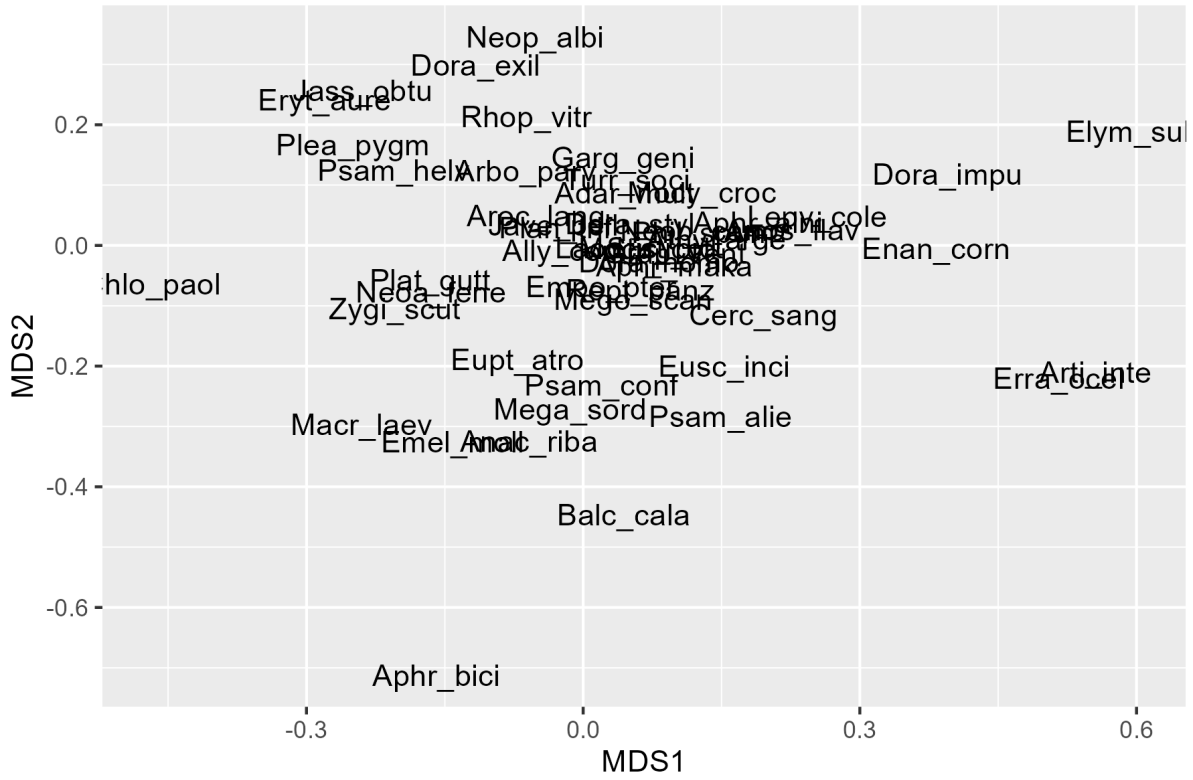
Až by se dalo říct, že výsledek je celkem škaredý. Abychom se v grafu lépe vyznali, je dobré zkrátit jména druhů např. na první tři nebo čtyři písmena druhového a rodového jména. Pomocí funkce `mutate()` přidáme do tabulky s druhovými skóry sloupeček se zkratkami

druhových jmen `short_name`. Ze sloupce `species_name` nejdříve pomocí funkce `word()` vybereme první slovo (rodové jméno), pak z něj funkcí `str_sub()` vybereme první čtyři písmena. To stejné uděláme pro druhé slovo, tedy druhové jméno, a následně obě části spojíme funkcí `str_c()`, které navíc řekneme, že mezi ty dvě čtyřpísmenné části má vložit podtržítko:

```
species.scores <- scores(pco, display = 'species', scaling = 'symm') |>
 as_tibble(rownames = 'species_name') |>
 mutate(short_name = str_c(species_name |> word(1) |> str_sub(1, 4),
                           species_name |> word(2) |> str_sub(1, 4),
                           sep = '_'))
```

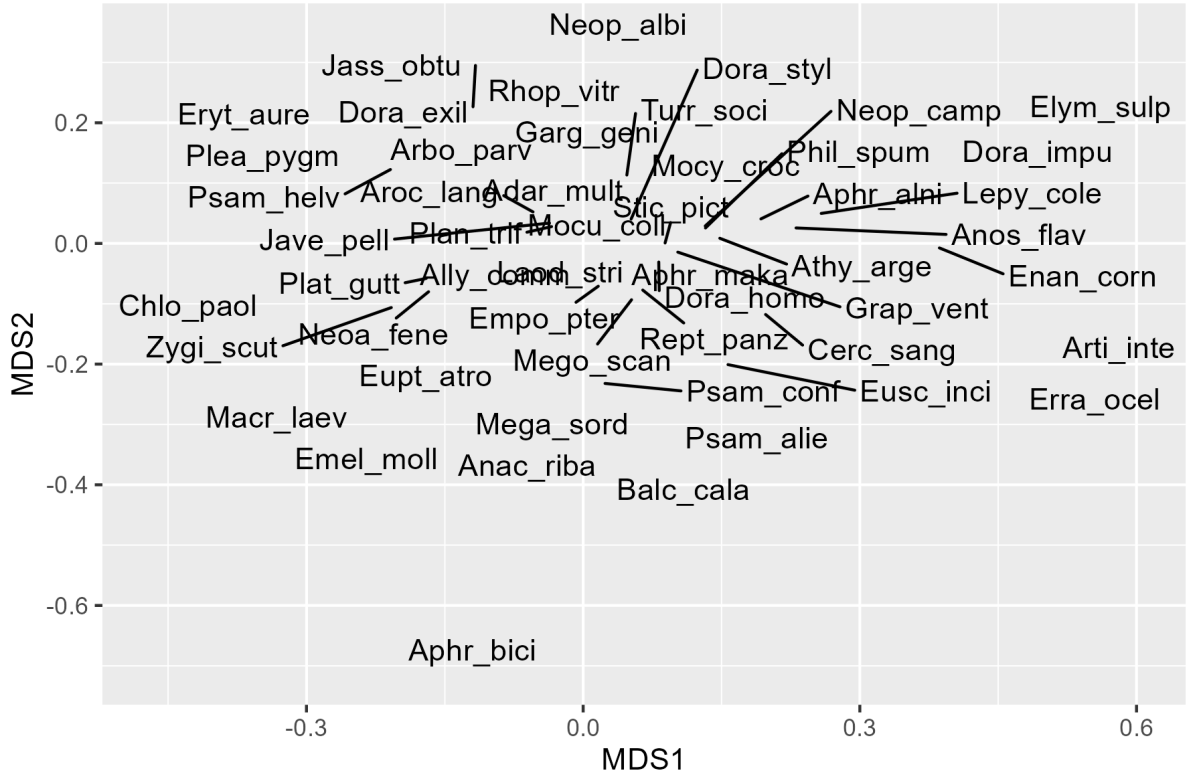
Do ordinačního diagramu teď promítneme zkratky druhů a dostaneme o něco přehlednější graf. Stále to ale není úplně ono.

```
species.scores |>
 ggplot(aes(MDS1, MDS2)) +
 geom_text(aes(label = short_name))
```



Velmi užitečná pro přehlednější zobrazení většího množství popisků v grafu je funkce `geom_text_repel()` z balíčku `ggrepel`, která posune popisky tak, aby se navzájem co nejméně překrývaly (tedy aby se vzájemně odpuzovaly, spojitost s repelentem není náhodná):

```
species.scores |>
 ggplot(aes(MDS1, MDS2)) +
 geom_text_repel(aes(label = short_name), max.overlaps = Inf)
```



Pořád je v tom ale docela zmatek i když se snažíme promítnout jen asi 50 druhů. Největší chumel popisků uprostřed ordinačního diagramu nám navíc nic moc o gradientech v datech neříká (druhy uprostřed nejsou nijak výrazně vázané ani na jeden z hlavních gradientů, které představují zobrazené osy). Dost pravděpodobně se dostanete do situace, kdy máte v druhové matici druhů mnohem víc, takže se bude hodit naučit se vybrat ty, které nejlépe korelují se zobrazenými ordinačními osami. V PCA a CA je možné výběr nejlépe fitujících druhů nebo proměnných provést pomocí funkce `goodness()`. Pro PCoA to fungovat nebude, protože počítá s maticí nepodobností a ne s druhovou maticí přímo. Je potřeba provést srovnání post-hoc pomocí funkce `envfit()`, která nám poví, jak dobře jednotlivé druhy korelují se zobrazenými ordinačními osami. Kromě samotného ordinačního modelu do ní zadáváme několik argumentů:

- `env`: matici proměnných, pro které chceme zjistit korelaci s osami, pojmenování `env` od toho, že se funkce `envfit()` běžně používá pro pasivní promítání proměnných prostředí do nepřímé ordinace, úplně stejně ji ale můžeme použít pro druhy. V takovém případě sem přidáme druhovou matici.
- `display`: necháme default `"sites"`
- `choices`: pro které osy chceme zjistit korelaci, default první a druhá
- `permutations`: defaultně implementovaný permutační test s 999 permutacemi, který vrátí p-hodnotu, rozhodně nenahrazuje test vlivu dané proměnné na druhové složení, k tomu je potřeba přímá ordinace, pro zrychlení výpočtu můžeme permutace vypnout tím, že zadáme 0

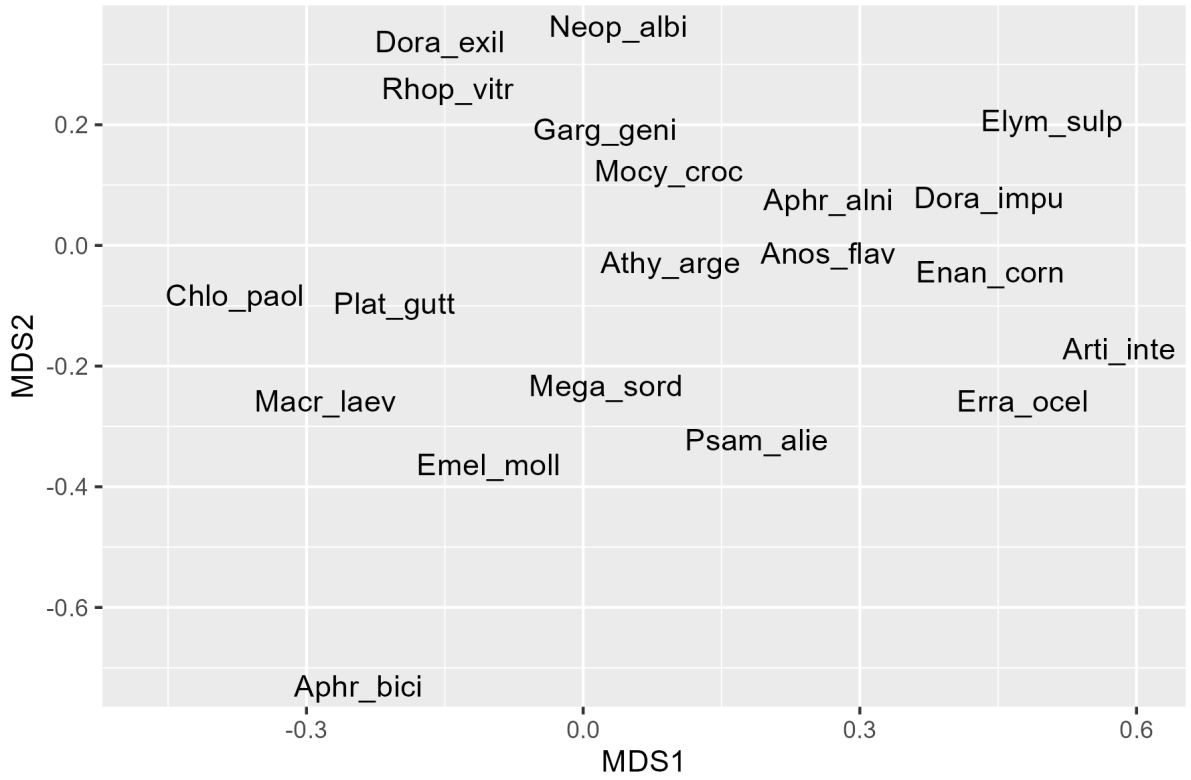
```
ef <- envfit(pco, env = spe.log, display = 'si', choices = 1:2, permutations =
0)
```

Síla korelace s ordinačními osami je teď schovaná v objektu `ef`, můžeme ji z něj vytáhnout jako vektor pomocí `ef$vectors$r` a nejlepší bude, když hodnoty přímo přidáme do tabulky s druhovými skóry, abychom podle nich mohli tabulku filtrovat. V `mutate()` přidáme další sloupeček, do kterého tento vektor nahrajeme:

```
species.scores <- scores(pco, display = 'species', scaling = 'symm') |>
 as_tibble(rownames = 'species_name') |>
 mutate(short_name = str_c(species_name |> word(1) |> str_sub(1, 4),
                           species_name |> word(2) |> str_sub(1, 4),
                           sep = '_'),
        r = ef$vectors$r)
```

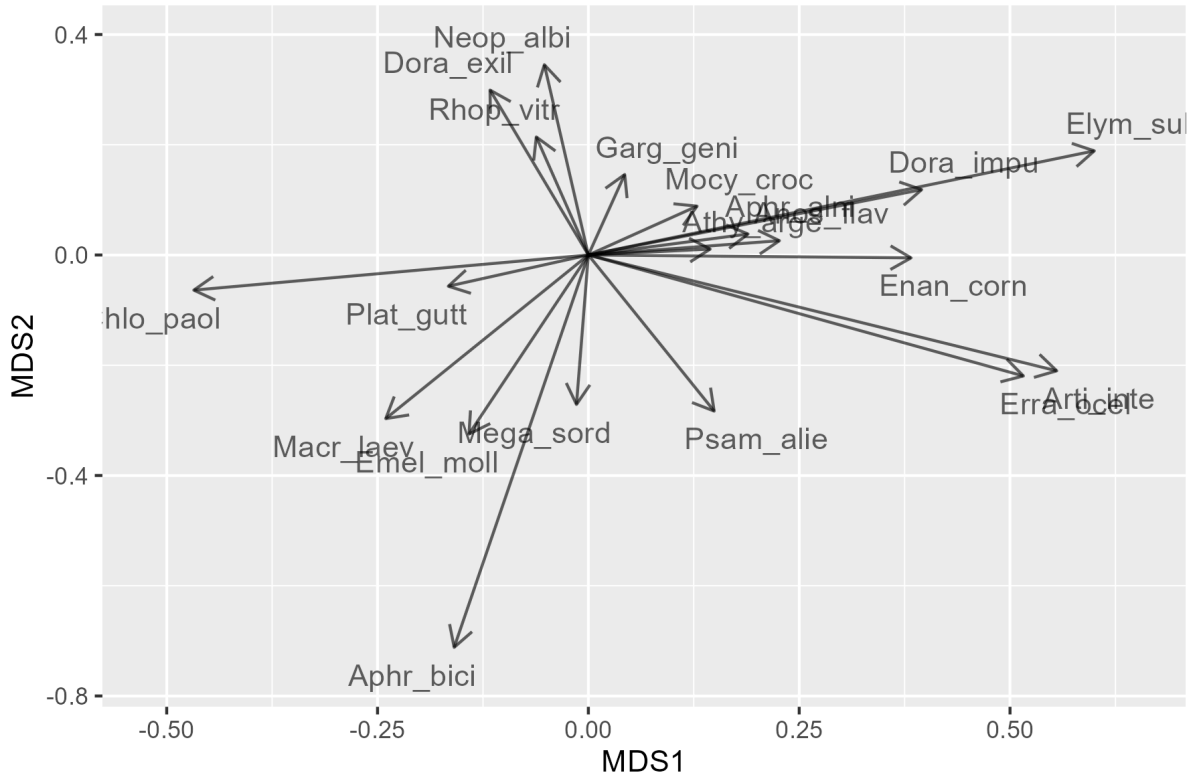
Než promítneme druhy do ordinačního diagramu, můžeme teď jednoduše z tabulky vybrat nejlépe fitující druhy např. tím, že vyfiltrujeme jen druhy s hodnotou `r` větší než arbitrárně zvolený threshold, např. `filter(r > 0.25)`. (Malá poznámka: r je koeficient determinace, čím je větší, tím je fit druhu lepší.) Další možností je použít funkci `slice_max()`, která seřadí tabulku podle dané proměnné (v našem případě `r`) a pak vybere zadaný počet řádků s nejvyšší hodnotou, takže sem můžeme zadat, kolik druhů se má v ordinačním diagramu zobrazit.

```
species.scores |>
 slice_max(r, n = 20) |>
 ggplot(aes(MDS1, MDS2)) +
 geom_text_repel(aes(label = short_name), max.overlaps = Inf)
```



Skóre druhů v PCoA vyjadřuje korelaci s ordinačními osami, správně bychom měli druhy v tomto ordinačním diagramu zobrazit šipkami pro směr a sílu této korelace. Pro šipky přidáme do ggplotu další vrstvu `geom_segment()`. Této funkci zadáme, že všechny šipky mají začínat v bodě (0, 0) a končit v bodě definovaném skóry daného druhu. Aby se nám šipky nepřekrývaly s jejich popisky, posuneme popisky druhů po osách o kousek dál tím, že k jejich souřadnicím přičteme malou hodnotu, kterou vynásobíme znaménkem dané souřadnice pro posun dál od nuly. Funkce `geom_segment()` defaultně kreslí čáry, my ale chceme, aby se z čar staly šipky, je proto potřeba zadat ještě argument `arrow`. Definice, jak má vypadat šipka je lehce složitější, ale v zásadě říká, že na konci čáry má být šipka, jejíž konce jsou dlouhé 0,3 cm. Nastavením argumentu `alpha` šipky i popisky trošku zprůhledníme, aby byly čitelné i překrývající se objekty.

```
species.scores |>
 slice_max(r, n = 20) |>
 ggplot(aes(MDS1, MDS2)) +
 geom_segment(aes(x = 0, y = 0, xend = MDS1, yend = MDS2),
              arrow = arrow(length = unit(0.3, 'cm')), alpha = 0.6) +
 geom_text(aes(MDS1 + sign(MDS1) * 0.05, MDS2 + sign(MDS2) * 0.05, label =
short_name), alpha = 0.6)
```



No není to nádhera přátelé? Takovýto graf by už v diplomce v pohodě obstál. Byť teda ještě je, co zlepšovat. Jak jej vylepšit se dozvíte třeba na výše zmíněných odkazech, nebo se prostě můžete zeptat ChatGPT, nebo přímo nás. Poslední věc, kterou se tady prozatím naučíme, je graf uložit do souboru. K tomu nám poslouží funkce `ggsave()`.

```
ggsave('plot.png', height = 7, width = 7)
```

Pomocí argumentů `height` a `width` můžeme ovlivnit poměr stran, ale také rozlišení a tedy i velikost textu a symbolů. Čím menší height a width, tak bude text větší a písmena se budou více překrývat.

A je to. Pokud budete chtít nějaké téma rozpracovat víc do hloubky, nebo něčemu nerozumíte, prostě do textu vložte komentář a fakt se vůbec nestyďte to dělat. My se na to pokusíme obratem reagovat a téma upřesníme.

Na závěr tady máme ještě nějaký tip na hudbu k práci. Japonský Ambient Playlist.

🥺 Kapitola 2: Kdy a jak použít přímou ordinaci

Kryštof & Klára: Analýza dat v ekologii společenstev, kuchařka.

Kapitola 2: Kdy a jak použít přímou ordinaci

Kryštof Chytrý & Klára Klinkovská

Stále v procesu!!

Zatímco u nepřímých ordinací jsme se soustředili na metody a o modelovaných společenstvech jsme skoro skoro nic nepsali. V případě přímých ordinací to vezmeme úplně naopak. Přímé ordinace (RDA, CCA a dbRDA) totiž používáme tehdy, když nás zajímá, jak naše data reagují na nějaký gradient proměnných prostředí. Nemusí to být jen takové ty klasické proměnné prostředí, jako jsou například nadmořská výška, pH, biomasa atd. Může to být v podstatě jakákoliv proměnná, od které máme informaci pro každý náš vzorek. Hned v prvním příkladu si ukážeme, že to může být třeba i čas. Myslíme si proto, že pro pochopení přímých ordinací bude nejlepší je ukázat přímo na konkrétních případech. Zvolili jsme pro tenhle účel dva příklady, kterými jsme se sami v poslední době zabývali.

Nejprve bude Klárka analyzovat svoje vlastní data o změnách vegetace, která sbírala pro case study do svého doktorátu. Klárku zajímalo, jak se ve středomoravských Karpatech změnila vegetace širokolistých suchých trávníků za téměř 40 let od doby, kdy ji tam snímkoval Bohumil Trávníček. Navštívila ty stejné lokality a stejnou metodikou zapsala snímky vegetace. Následně pak analyzovala stará i nová data dohromady, aby zjistila, jak se vegetace změnila, a jestli je tato změna signifikantní. To je úplně modelový příklad, kdy je vhodné použít přímou ordinaci. Kdybychom použili nepřímou ordinaci, tak bychom asi snad dokázali odhadnout, jakým způsobem se vegetace mění, ale jen za předpokladu, že by byla data velmi homogenní a změny velmi výrazné a konzistentní. I tak bychom ale nedokázali říct, jestli je ta změna signifikantní. Na téma proč je důležité se tím zabývat a používat přímé ordinace na místo těch nepřímých máme jednu hezkou historku ze života:

Kryštof nedávno recenzoval článek o změnách vegetace v maďarských písečných stepích. Jeho autoři nejprve použili nepřímou ordinaci a zjistili, že se jejich vegetace v podstatě vůbec nezměnila. Kryštof jim to v recenzi vytknul a navrhl jim použít místo toho přímou ordinaci. Autoři dali na jeho radu, no a ejhle, najednou analýza místo více méně náhodného shluku ukázala jasně interpretovatelnou, byť celkem malou, změnu. No a chudáci autoři museli změnit závěry článku z “vegetace se vůbec nezměnila” na “vegetace se přeci jen celkem změnila”. 🙂 Jak je možné, že změnu vegetace nedokázali odhalit pomocí nepřímé

ordinace, ale až za pomocí té přímé? Je to tím, že přímé ordinace mají dvě velmi užitečné vlastnosti, o kterých si za chvíli povíme.

Maďarské písčité stepi (homokpuszty) jsou domovem slepce malého. Slepec vypadá a chová se jako krtek, jen na rozdíl od krtka hmyzožravce, je ale hlodavec. Naši kuchařku by si bohužel přečíst nemohl, je totiž jediným úplně slepým hlodavcem.

Nejprve si ale něco povíme k druhému příkladu. Jde o experiment, ve kterém Jakub, svého času ještě působící v Českých Budějovicích, zkoumal možnosti využití poloparazitického kokrhele (*Rhinanthus alectorolophus*) na potlačování expanzní třtiny křovištní (*Calamagrostis epigejos*). Třtinu křovištní určitě zná velká většina z vás. Je to celkem pěkná vysoká tráva, která se někdy dává do suchých vazeb. Je to ovšem taky pořádná mrcha, která vytlačuje z trávníků vzácné rostliny. Abychom tomuto procesu zabránili a udrželi biodiverzitu luk, musíme je dostatečně často sekat. Sečí se třtina pomalu potlačuje, je to ale pořád běh na dlouhou trať. Před časem ale někoho napadlo, že by se mohly pro tento účel použít poloparazitické rostliny, které se většinou orientují na konkurenčně silnější druhy, a tím, že je oslabují, dávají prostor druhům konkurenčně slabším. Jakub na tohle téma připravil experiment, ve kterém chtěl srovnat vliv kokrhele s pouhým samotným kosením.

Design experimentu je celá věda sama o sobě. Především je u nich potřeba nad daty přemýšlet ještě před tím, než je začneme sbírat. Pokud plánujete nějaký experiment, je dobré si nejprve rozmyslet, na jaké nejjednodušší srovnání jsme naši otázku schopní zjednodušit. V jednoduchosti je síla a ve statistice to platí obzvlášť. Tohle jednoduché srovnání, třeba ve formě několika různých zásahů pak replikujeme na několika různých

lokalitách, nebo v několika různých blocích na jedné lokalitě. Množstvím opakování získává analýza na síle, takže platí, že čím víc, tím líp. Zároveň ale existuje teorie, která vám může pomoci, jak určit nezbytný minimální počet opakování, pokud alespoň trochu tušíte, jak heterogenní vaše data budou. Pokud vás zajímá design experimentů, přečtěte si prosím odpovídající kapitolu skript Davida Zeleného.

Nyní se ale vraťme k těm dvěma užitečným vlastnostem přímých ordinací.

Užitečná vlastnost 1

Přímé ordinace dokážou studovanou proměnnou (nebo i více proměnných) použít přímo při výpočtu modelu.

Proměnným, které tímto způsobem používáme, říkáme prediktory (jako u lineárních modelů). Konkrétně v Klárčině příkladu máme jako prediktor čas. Protože ale máme jen dva časové úseky, použijeme k jeho vyjádření faktor se dvěmi úrovněmi (0: staré, 1: nové). Když ordinaci takový prediktor dáme, ordinace se napřed podívá, jak druhy reagují na tento gradient a vytvoří pro něj jeho vlastní ordinační osu (constrained axis). V případě, kdy máme jako prediktor pouze čas, budeme mít jedinou constrained ordinační osu. Pokud bychom proměnných měli více, budeme mít i více constrained ordinačních os, ale pozor, není to vztah 1:1 (jeden prediktor ≠ jedna ordinační osa), záleží, jak jsou si jednotlivé prediktory podobné a nepodobné. Constrained osy se jinak chovají jako běžné ordinační osy u nepřímých analýz, čímž mám na mysli, že mají na sebe vázané nějaké eigenhodnoty a tedy je s nimi spojená nějaká vysvětlená variabilita.

Možná vás teď napadlo, že když ty plochy “násilím” rozdělíme na staré a nové, tak je přece jasné, že budou staré a nové plochy tvořit vlastní shluky. To je pravda a při interpretaci bychom na to měli pamatovat. Přímé ordinace totiž interpretujeme jinak než nepřímé. U nepřímých ordinací nás zajímá, jak jsou vzorky (lokality, zápisy nebo snímky) od sebe odlišné. Má proto smysl se dívat na to, jestli se naše vzorky nebo druhy nějak shlukují, nebo naopak jestli se řadí podél nějakého implicitního gradientu. U přímých ordinací jsme si ten gradient vytvořili sami a skutečně tedy nemá smysl se dívat, jak se na něm naše vzorky rozmisťují. Má ale smysl studovat, jak na tento gradient reagují jednotlivé druhy, a proto je také správné v přímých ordinacích zobrazovat jen druhové složení. Zobrazit si v modelu jen druhy by ovšem byla informace bez kontextu, proto navíc ještě zobrazujeme prediktory formou šipek. Když se ponoříte do ekologické literatury, najdete ale spoustu přímých

ordinací, kde autoři zobrazují i vzorky, a Kryštof se tímto přiznává, že to někdy udělal i on. Je to zbytečné, protože jejich zobrazení nemůžeme nijak interpretovat.

Užitečná vlastnost 2

Přímé ordinace dokážou odfiltrovat variabilitu, která nás nezajímá.

Jeden z důvodů, proč Maďaři nakonec v datech nějaký trend našli, bylo to, že si odfiltrovali variabilitu, která je nezajímá, a soustředili se jen na tu, která zbyla. To se dělá podobným způsobem, jako fitování prediktorů. Nejprve ordinaci dáme prediktory, jejichž působení chceme odfiltrovat, a ona pro ně stejným způsobem udělá ordinační osy (conditional axes) a spočítá eigenhodnoty. To vše jen proto, aby tyto osy následně vzala a prostě vyhodila. Takovýmto ordinacím říkáme částečné (partial) ordinace, např. partial-PCA, partial-PCoA, partial-RDA, partial-dbRDA atd. Jestli vás zarazilo PCA a PCoA v kapitole o přímých ordinacích, tak věžte, že stejná metoda jde použít i pro nepřímé ordinace.

Kdy je vhodné odfiltrovat vliv nějaké proměnné? Například když máme experiment s větším množstvím opakování. V takovém případě nás zajímá efekt našeho treatmentu, nikoliv efekt toho, že jsou si jednotlivá opakování mírně odlišná. Nejprve tedy ordinace odfiltruje vliv jednotlivých opakování, a poté se může soustředit na vliv našeho treatmentu. Podobně tuto vlastnost můžeme použít i v případě opakovaných zápisů. Klárka ve svém případě například odfiltrovala vliv plochy, protože ji zajímá, jak se mění vegetace v rámci jednotlivých ploch a heterogenita dat by tyto změny zbytečně překrývala. No a pokud máme experiment, který pozorujeme v průběhu času, můžeme udělat obojí, tedy zároveň odfiltrovat vliv opakování i plochy. Je to trošku podobný princip, jako když místo jednoduchého lineárního modelu použijeme lineární model se smíšenými efekty (mixed-effect model). Když analyzujete zároveň mnohorozměrná a jednorozměrná data, co mají podobnou strukturu, hodí se částečnou přímou ordinaci doplnit právě mixed-effect modelem, např. když chcete na stejných datech kromě vlivu zásahu na druhové složení otestovat také jeho vliv na druhovou bohatost.

Myslím, že je čas si dát kafe. Kryštofa v poslední době nejvíc zaujalo anglické Darkhouse Coffee svojí nevtíravou a nekomplikovanou chutí, což z něj dělá kafe, které se jen tak neomrzí. Jaké kafe máte rádi vy? 😉

Příklad 1: Změny jihomoravských suchých trávníků za posledních 40 let

V tomto příkladu se budeme snažit najít odpověď na dvě otázky: 1) Jak se změnila společenstva suchých trávníků v průběhu posledních desetiletí a je tahle změna statisticky průkazná? 2) Jsou změny stejné v chráněných územích i mimo ně?

Jednou z možností, jak vyhodnotit změny vegetace v průběhu času jsou tzv. resurvey studie. Pro vyhodnocení změn rostlinných společenstev za posledních téměř 40 let máme k dispozici historické fytocenologické snímky (záznam druhového složení vegetace na vymezené ploše) z 80. let a pro porovnání k nim snímky, které se Klárka snažila zapsat na stejných místech v roce 2022. Zároveň se zhruba polovina snímků nachází v chráněných územích a polovina mimo ně, a tak můžeme otestovat, jestli je nastavený ochranářský management efektivní pro zachování příznivého stavu jihomoravských suchých trávníků.

Tak pojďme na to. Chceme otestovat změny druhového složení, takže jako první krok musíme načíst druhovou matici. Rovnou už z ní vyhodíme druhy, co se vyskytují jen v jednom snímku, abychom zmírnili šum v datech. Další, s čím už dál nechceme počítat je číslo snímku, tento sloupec z tabulky vyhodíme. Na závěr přípravy dat ještě odmocníme abundance druhů v procentech. To proto, aby měly druhy s nižšími abundancemi v analýze větší váhu, stejně tak bychom pokryvnosti mohli i logaritmovat.

```
spe <-
read_csv("data/S_Moravia_basiphilous_grass_KK/basiphilous_grasslands_S_Moravia_species.csv") |>
   select(where(~sum(. > 0) > 1), -Releve_number) |>
   sqrt()
```

Další, co budeme pro analýzu potřebovat jsou hlavičková data. Využijeme je pro identifikaci (1) nových a starých snímků, (2) dvojic pozorování té stejné plochy, a (3) toho, jestli se plocha nachází v chráněném území, nebo mimo něj. Velmi důležité upozornění: je potřeba mít snímky (nebo lokality, vzorky….) v hlavičkové tabulce seřazené stejně jako v druhové matici! Jinak se nám může stát, že přiřadíme historickému snímku identifikaci nového, v případě experimentu experimentální zásah kontrole nebo něco podobného. Raději si to proto ještě jednou zkontrolujte třeba tak, že si v excelu zkopírujete kódy snímků z druhových dat a kody snímků z hlavičkových dat vedle sebe. 🙂

V rámci přípravy dat ještě převedeme na faktor dvě proměnné, které máme v hlavičkové tabulce uložené jako číselné (numeric), ale ve skutečnosti by měly být kategorické. Jednou z nich je informace o čase snímkování, kterou máme uloženou ve sloupci `Rs_observ`. Tato proměnná je číselná s údaji 1980 a 2022, v tomto případě, kdy máme pro čas jen dvě

úrovně “historický” a “současný” je lepší ji převést na kategoriální proměnnou. Stejně tak na faktor převedeme i proměnnou `Rs_plot`, která obsahuje číselná id pro každou plochu, ale ve skutečnosti to není kontinuální proměnná. Použijeme příkaz `mutate()`, kterým modifikujeme už existující sloupce v tabulce. Abychom mohli naráz změnit více sloupců, použijeme příkaz `across()`, kam zadáme, které sloupce chceme modifikovat a poté funkci pro modifikaci sloupců, v našem případě funkci `factor()`.

```
head <-
read_csv("data/S_Moravia_basiphilous_grass_KK/basiphilous_grasslands_S_Moravia_head.csv") |>
mutate(across(c(Rs_observ, Rs_plot), ~factor(.x)))
```

A už se můžeme pustit do vlastní analýzy. Pro vyhodnocení změn v druhovém složení v průběhu času použijeme přímou ordinaci dbRDA (distance-based redundancy analysis), což je přímá ordinace ekvivalentní k PCoA. Použijeme tedy opět funkci `capscale()`, jen do ní navíc oproti PCoA přidáme za vlnovku prediktory, jejichž vliv na druhové složení chceme testovat. Jako prediktor zadáme časové období, jestli byl snímek zapsán v 80. letech nebo v roce 2022 (`Rs_observ`). Změna v druhovém složení probíhá vždy v rámci jedné plochy, proto chceme v analýze odfiltrovat variabilitu mezi plochami a testovat čistě změnu v rámci jedné plochy. V modelu to ošetříme tak, že sem zadáme příslušnost k ploše uloženou v poli `Rs_plot` jako kovariátu příkazem `Condition()`. Oproti argumentům, které už známe z PCoA zadáme ještě argument `data`, kterým funkci říkáme, odkud si má model vzít prediktory.

```
ord <- capscale(spe ~ Rs_observ + Condition(Rs_plot),
                data = head, distance = "bray", sqrt.dist = T)
```

Výsledek ordinace teď máme uložený v objektu `ord`. Když ho pustíme do konzole, zobrazí se nám jeho summary, tedy celková variabilita, kolik variability vysvětlují kovariáty (Conditional), prediktory (Constrained) a další neomezené osy, kam model uklidil zbylou variabilitu (Unconstrained). Eigenvalues můžeme převést na procenta vysvětlené variability tak, že je vydělíme celkovou variabilitou a vynásobíme 100.

```
eigenvals(ord) / ord$tot.chi * 100
```

Z výsledku zjistíme, že jediná přímá osa představující čas vysvětluje necelá 3 % variability. Vypadá to jako velmi malé číslo, je to ale dané větším množstvím snímků i druhů v našem souboru. Hlavně nás teď bude zajímat, jestli je změna v druhovém složení v průběhu času signifikantní. To zjistíme tak, že model otestujeme permutačním testem. Do příkazu

`anova()` zadáme výsledek přímé ordinace a protože chceme testovat jen změnu v rámci jedné plochy, musíme definovat, že se mají mezi sebou permutovat jen pozorování příslušející k dané ploše. Uděláme to tak, že v příkazu `how()` definujeme bloky v rámci kterých se pozorování permutují. Navíc zadáváme ještě počet permutací.

```
anova(ord, permutations = how(blocks = head$Rs_plot, nperm = 999))
```

A máme první signifikantní výsledek! Můžeme tedy říct, že druhové složení vegetace jihomoravských suchých trávníků se od 80. let signifikantně změnilo. No jo, jenomže pořád nevíme, jak. Abychom se dozvěděli víc, otestujeme korelaci jednotlivých druhů s omezenou ordinační osou, tedy s časem. Do funkce `envfit()` tentokrát zadáme, že chceme korelaci jen s první (omezenou) ordinační osou a abychom korelaci otestovali, definujeme permutační test, stejně jako jsme to dělali před chvílí.

```
ef <- envfit(ord, spe, choices = 1, permutations = how(blocks = head$Rs_plot),
nperm = 999, display = "lc")
```

V objektu `ef` teď máme uložené hodnoty pro sílu a signifikanci korelace jednotlivých druhů s první ordinační osou (časem). Zajímají nás hlavně druhy se signifikantním vztahem k tomuto prediktoru, protože to jsou druhy, které od 80. let signifikantně přibyly anebo naopak ubyly. Vytáhneme si tedy z objektu `ef` jen druhy signifikantně korelované s první osou. Směr korelace (tzn. jestli druh přibývá nebo ubývá) je schovaný v `ef$vectors$arrows`, tento vektor převedeme na tibble a přidáme argument `rownames` abychom do dalšího sloupečku `SpecName_Layer` dostali jména druhů. Připojíme další dva sloupečky `p` pro p hodnotu vyjadřující signifikanci korelace a `r` pro její sílu. Podle p hodnoty vyfiltrujeme jen signifikantně korelované druhy. Na závěr tabulku ještě seřadíme, aby v ní byly nejdřív všechny ubývající druhy a pak všechny přibývající a navíc ještě seřazené podle p hodnoty od těch nejvíc ubývajících/přibývajících.

```
sig.spe <- as_tibble(ef$vectors$arrows, rownames = "SpecName_Layer") |>
   mutate(p = ef$vectors$pvals, r = ef$vectors$r) |>
   filter(p < 0.05) |>
   arrange(CAP1, p)
```

A je to. Takovou tabulku už můžeme s minimem dalších úprav prezentovat v diplomce. Jednoduše ji vyexportujeme příkazem `write_csv()`.

```
write_csv(sig.spe, 'results/increasing_decreasing_species.csv')
```

Navíc můžeme druhy taky zobrazit v ordinačním diagramu. Příkazem `scores()` si vytáhneme z ordinačního objektu souřadnice jednotlivých druhů na prvních dvou osách, tedy jediné přímé, omezené ose a první nepřímé ose. Zajímají nás jen korelace druhů s ordinačními osami, nebudeme v ordinačním prostoru zobrazovat snímky, to se u přímých ordinací nedělá, proto argumentem `display = 'sp'` zobrazíme jen druhy a nastavíme scaling na `"species"`. Argumentem `correlation = T` specifikujeme, že pozice druhů v ordinačním prostoru mají představovat korelaci s osami, ne kovarianci. V ordinačním diagramu budeme zobrazovat kromě druhů ještě šipku pro prediktor. Aby bylo obojí, prediktor i druhy v grafu dobře vidět, šipky pro druhy argumentem const prodloužíme. Nepotřebujeme mezi sebou porovnávat délku šipky pro prediktor a druhy, jejich poměr je arbitrární a tak ho můžeme upravit tak, aby nebyla šipka pro prediktor zbytečně dlouhá.

Matici vytvořenou příkazem `scores()` převedeme na tibble a rownames uložíme do sloupečku `SpecName_Layer`. Pro výběr jen signifikantně přibývajících nebo ubývajících druhů si ukážeme novou funkci `semi_join()`. Patří mezi tzv. filtering joiny, které filtrují řádky v jedné tabulce podle druhé tabulky, tedy zachová v tabulce `species.scores` jen ty řádky, které mají svůj identifikátor v tabulce `sig.spe`. Můžete si to předtavit i tak, že na tyto řádky by se druhá tabulka napojila při použití funkce `left_join()`. Příkazem `join_by()` funkci řekneme, v jakém sloupci má hledat společný identifikátor dvou použitých tabulek. Defaultně by vzala všechny sloupce se stejným názvem, takové sloupce jsou teď ale dva `SpecName_Layer` a `CAP1`. `CAP1` ale znamená v každé tabulce něco jiného a tak musíme sami zadat, že společný identifikátor je jen ve sloupci `SpecName_Layer`. Víc se o join funkcích můžete dočíst tady: [19 Joins – R for Data Science (2e)]. Pro přehlednější zobrazení druhů v ordinačním diagramu ještě vytvoříme zkratky druhových jmen.

```
species.scores <- scores(ord, scaling = 'sp', correlation = T, display = 'sp',
const = 30) |>
   as_tibble(rownames ='SpecName_Layer') |>
   semi_join(sig.spe, join_by(SpecName_Layer)) |>
   mutate(short_name = str_c(SpecName_Layer |> word(1) |> str_sub(1, 4),
                             SpecName_Layer |> word(2) |> str_sub(1, 4),
                             sep = '_'))
```

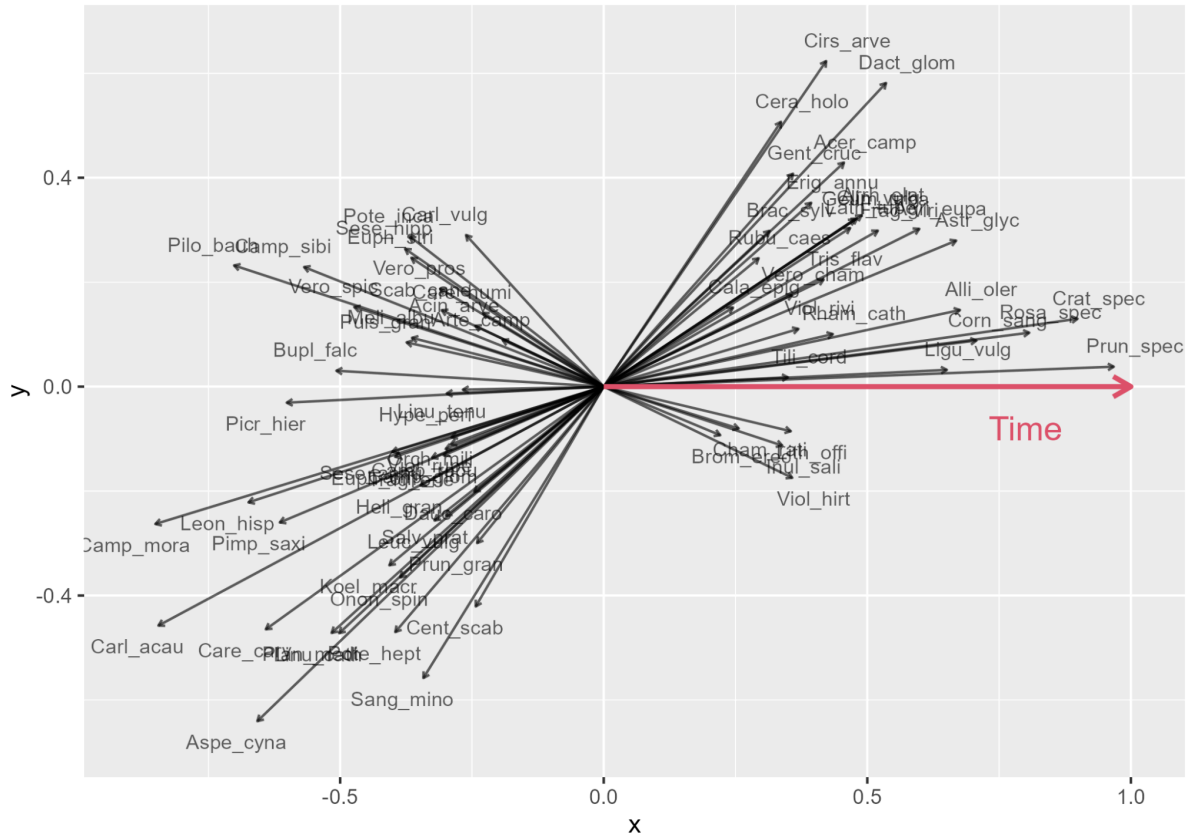
Kromě druhů chceme v ordinačním diagramu zobrazit ještě šipku pro směr prediktoru. Skóre pro prediktory vytahujeme z ordinačního objektu opět funkcí `scores()` a prediktory zobrazujeme pomocí `display = "bp"`.

```
pred <- scores(ord, scaling = 'sp', display = 'bp') |>
```

```
   as_tibble(rownames = 'predictor')
```

No a už můžeme nakreslit graf. Primárně budeme kreslit šipky, takže do globálního `aes()` ggplotu zadáme jejich začátek v bodě (0, 0) a konec v souřadnicích na přímé a první nepřímé ose, které máme v tabulce. Šipky kreslíme příkazem `geom_segment()`, popisky příkazem `geom_text()`. Kromě šipek pro druhy chceme zobrazit i šipku pro prediktor. Jeho skóre na ordinačních osách máme v jiné tabulce, než z které začínáme kreslit ggplot. Abychom mohli šipku nakreslit, zadáme do příkazu `geom_segment()` argument data, kterým funkci řekneme, odkud si má souřadnice vytáhnout. Protože se sloupečky v tabulce pred jmenují stejně jako v tabulce species.scores, nemusíme definovat `aes()` zvlášť. Přidáme popisek pro šipku znázorňující směr prediktoru a graf je hotový.

```
species.scores |>
   ggplot(aes(x = 0, y = 0, xend = CAP1, yend = MDS1)) +
   geom_segment(arrow = arrow(length = unit(0.1, 'cm')), alpha = 0.6) +
   geom_text(aes(x = CAP1 + 0.04 * sign(CAP1), y = MDS1 + 0.04 * sign(MDS1),
label = short_name), size = 3, alpha = 0.6) +
   geom_segment(data = pred, arrow = arrow(length = unit(0.3, 'cm')), color = 2,
linewidth = 1) +
   geom_text(data = pred, aes(x = CAP1 - 0.2, y = MDS1 - 0.08), label = 'Time',
color = 2, size = 5)
```



Samozřejmě je tu ještě dost prostoru pro zlepšení, to ale necháme na vás a pro tuto chvíli se zaměříme na interpretaci výsledků. Šipka pro prediktor čas směřuje doprava, druhy se šipkami směřujícími stejným směrem jsou ve snímcích z roku 2022 signifikantně více zastoupené než v historických snímcích. Můžeme říct, že v průběhu času signifikantně přibyly. Signifikantně pozitivně s časem korelují např. konkurenčně silné druhy trav *Arrhenatherum elatius, Calamagrostis epigejos, Dactylis glomerata* a také dřeviny rodů *Crataegus, Prunus, Rosa, Ligustrum vulgare, Cornus mas*. Na druhou stranu podél první ordinační osy směřují šipky pro spoustu typických druhů suchých trávníků, např. *Asperula cynanchica, Bupleurum falcatum, Carlina acaulis, Sanguisorba minor*. Z toho můžeme vyčíst, že z jihomoravských suchých trávníků od 80. let ubylo specilalizovaných druhů a začaly více zarůstat ovsíkem, třtinou nebo dřevinami, což může souviset s upuštěním od tradičního hospodaření.

Zhruba polovina ploch v našem datasetu se ale nachází v maloplošných chráněných územích, kde probíhá ochranářský management zaměřený na zachování diverzity suchých trávníků. Zatím jsme testovali změny v průběhu času pro celý dataset dohromady, ale třeba to v chráněných územích nebude tak zlé? Jestli je rozdíl v tom, jak se mění druhové složení na lokalitách v chráněných územích a mimo ně, zjistíme pomocí další přímé ordinace. Budou do ní vstupovat stejná data, jen změníme prediktory. Místo času tentokrát jako prediktory použijeme čas, příslušnost k chráněnému území (proměnná `protected`) a jejich interakci, která nám říká, jestli je změna v čase probíhá v chráněných územích a mimo ně rozdílně. A protože chceme testovat jen a pouze tuto interakci, jako kovariáty do `Condition()` přidáme oba prediktory ze kterých je interakce složená, abychom odfiltrovali jejich vliv.

```
ord.prot <- capscale(spe ~ protected * Rs_observ + Condition(protected +
Rs_observ), distance = "bray", sqrt.dist = T, data = head)
```

Spočítáme kolik variability omezená osa vysvětluje:

```
eigenvals(ord.prot) / ord.prot$tot.chi * 100
```

Tentokrát prediktor vysvětluje ani ne jedno procento z celkové variability. Důležitější ale je, jestli je jeho vliv na druhové složení společenstva signifikantní. Spočítáme tedy permutační test. Pro otestování významnosti interakce tentokrát nebudeme permutovat pozorování v rámci jedné plochy, ale plochy mezi sebou.

```
anova(ord.prot, permutations = how(plots = Plots(head$Rs_plot, type = "free"),
```

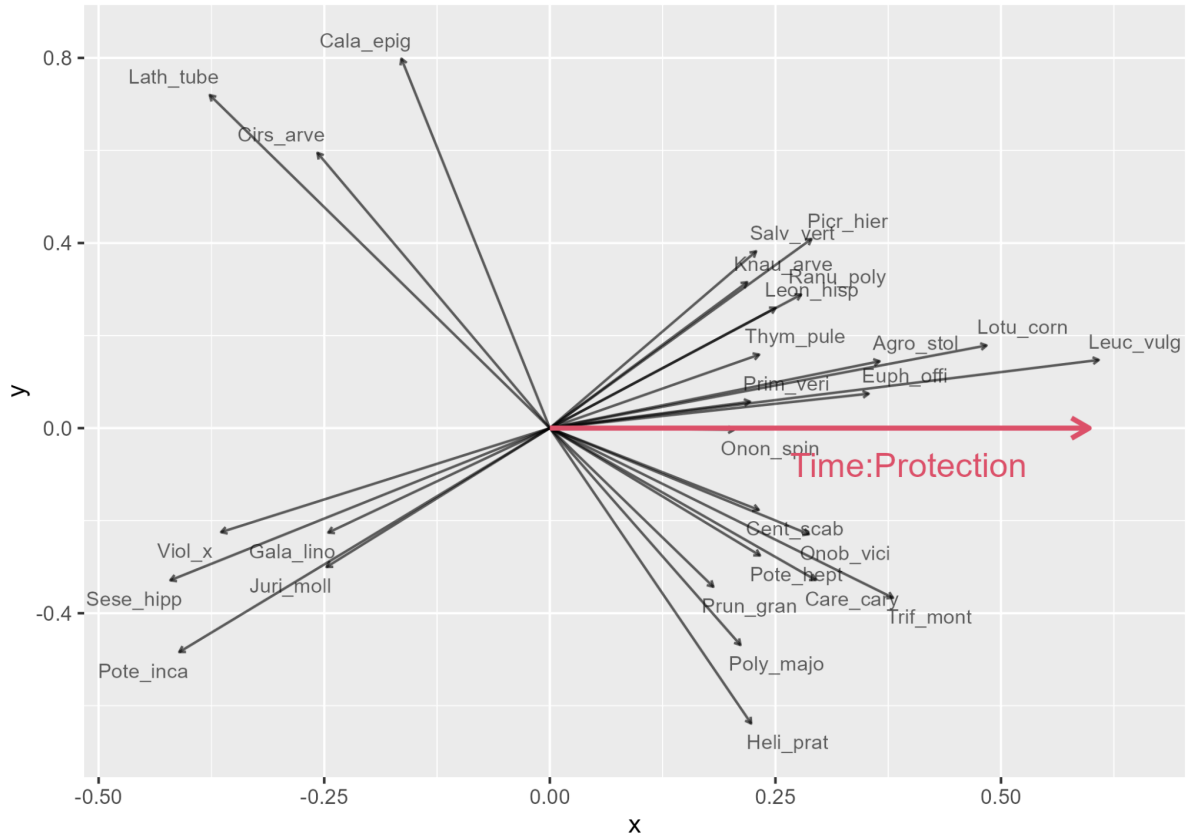
```
within = Within("none"), nperm = 999))
```

A výsledek je opět signifikantní. Můžeme tedy říct, že v chráněných územích probíhají změny rostlinných společenstev jinak než mimo chráněná území. Abychom získali trošku lepší obrázek o těchto rozdílech, podíváme se na druhy signifikantně korelované s omezenou osou, interakcí čas:ochrana.

```
ef.prot <- envfit(ord.prot, spe, choices = 1, permutations = how(plots =
Plots(head$Rs_plot, type = "free"), within = Within("none"), nperm = 999),
display = "lc")

 sig.spe.prot <- as_tibble(ef.prot$vectors$arrows, rownames = "SpecName_Layer")
   bind_cols(p = ef.prot$vectors$pvals, r = ef.prot$vectors$r)  |>
   filter(p < 0.05) |>
   arrange(CAP1, p)
```

A rovnou výsledek i zobrazíme v ordinačním diagramu.

```
species.scores.prot <- scores(ord.prot, scaling = 'sp', correlation = T, display
= 'sp', const = 30) |>
   as_tibble(rownames ='SpecName_Layer') |>
   semi_join(sig.spe.prot, join_by(SpecName_Layer)) |>
   mutate(short_name = str_c(SpecName_Layer |> word(1) |> str_sub(1, 4),
                             SpecName_Layer |> word(2) |> str_sub(1, 4),
                             sep = '_'))

 pred.prot <- scores(ord.prot, scaling = 'sp', display = 'bp') |>
   as_tibble(rownames = 'predictor')

 species.scores.prot |>
   ggplot(aes(x = 0, y = 0, xend = CAP1, yend = MDS1)) +
   geom_segment(arrow = arrow(length = unit(0.1, 'cm')), alpha = 0.6) +
   geom_text(aes(x = CAP1 + 0.04 * sign(CAP1), y = MDS1 + 0.04 * sign(MDS1),
label = short_name), size = 3, alpha = 0.6) +
   geom_segment(data = pred.prot, arrow = arrow(length = unit(0.3, 'cm')), color
= 2, linewidth = 1) +
   geom_text(data = pred.prot, aes(x = CAP1 - 0.2, y = MDS1 - 0.08), label =
'Time:Protection', color = 2, size = 5)
```



Na pravé straně ordinačního diagramu teď vidíme druhy, které mají trendy v chráněných územích pozitivnější než mimo chráněná území. To ale nemusí nutně znamenat, že přibývají, často jsou to ubývající druhy, které mají negativní trend v chráněných územích o něco mírnější. Vidíme tedy, že někteří specialisté suchých trávníků např. *Euphrasia officinalis, Leontodon hispidus, Potentilla heptaphyla, Prunella grandiflora* ubývají v chráněných územích méně. Na opačnou stranu podél omezené ordinační osy směřují šipky pro ty druhy, které mají v chráněných územích trend negativnější. Ne nutně negativní, trend může být jen méně pozitivní než mimo chráněná území. Vidíme tu např. *Calamagrostis epigejos, Cirsium arvense*, které více přibyly mimo chráněná území. Zároveň jsou tu také např. *Galatella linosyris, Jurinea mollis* anebo *Seseli hippomarathrum* ubývající hlavně v chráněných územích, což je ale způsobené tím, že mimo chráněná území od začátku nebyly, proto už tam nemají kam mizet.

Závěr z naší analýzy tedy je, že vegetace jihomoravských stepních trávníků se od 80. let signifikantně změnila. Ubývají zejména druhy specializované na suché trávníky, přibývají konkurenčně silné druhy trav a dřeviny. Tyto negativní trendy jsou o něco mírnější v chráněných územích, kde jsou sukcesní procesy zpomalovány ochranářským managementem.

Mnohorozměrnou ordinační analýzu můžeme dobře doplnit jednorozměrnými modely se stejnými prediktory pro různé vysvětlované proměnné. Můžeme např. zjistit, jestli se v průběhu času mění druhová bohatost, anebo otestovat změnu v podílu stanovištních specialistů pro silnější důkaz, že v průběhu času ubývají. Pro vážnější zájemce o změny vegetace jihomoravských stepních trávníků něco ke čtení tady: Significant decline in habitat specialists in semi-dry grasslands over four decades.

Příklad 2. Experiment.

V našem konkrétním případě tyto zásahy jsou: (1) kosení jednou za sezónu, (2) kosení dvakrát za sezónu, (3) kosení jednou za sezónu a výsev kokrhele, (4) kosení dvakrát za sezónu a výsev kokrhele.

Adresáti z předmětu:

Milí přátelé,

analýza dat a jejich vizualizace jsou velmi užitečné nástroje pro pochopení různých biologických i jiných souvislostí a velmi se hodí naučit se s nimi pracovat. Jednak se bez nich neobejde snad žádná diplomka u nás na ústavu, druhak se jejich prostřednictvím dostáváme od naivního pozorování přírody ke skutečné biologii a ekologii.

Je nám ale jasné, že je to občas trošku boj. Koneckonců jsme si tím poměrně nedávno prošli i my sami (Kryštof a Klára), a tak si pamatujeme, že začátky jsou obzvlášť těžké. V nějaký moment se to ale zlomí a najednou z trápení začne být radost. Důležité je to před tímto bodem nevzdat. Chtěli bychom vám tuto cestu trošku usnadnit, a proto jsme se rozhodli, že vytvoříme jednoduchý návod, jak na to. Nečekejte nadupanou učebnici statistiky včetně všech matematických detailů, takových už pár existuje, takže pokud budete stát o to, dozvědět se v tomto ohledu víc, snad víte, kde hledat. Naopak se snažíme co možná nejsrozumitelněji vysvětlit, kdy a jak kterou analýzu použít a jak úspěšně projít všemi kroky analýzy v Rku až po nakreslení pěkného grafu. Naše kuchařka zatím obsahuje jednu kapitolu o nepřímých ordinacích a časem se snad trošku rozroste minimálně ještě o přímé ordinace.

Kuchařku děláme pro vás a naší jedinou ambicí je, aby vám byla užitečná a pomohla vám naučit se analyzovat data. K tomu ale potřebujeme vaši pomoc. Nejsme učitelé a neumíme psát učebnice. Nevíme proto, jestli je kuchařka v této formě pro vás použitelná. Proto vás moc prosíme o zpětnou vazbu ve formě online komentářů nebo i odpovědi na tento email. Pokud se stydíte, že nerozumíte něčemu, čemu si myslíte, že byste rozumět měli, tak se zaprvé vůbec nestyďte, nebo prostě přidejte do dokumentu anonymní komentář, třeba tak, že si dokument otevřete v anonymním prohlížení.

Hodně zdaru přejí,

Klára & Kryštof