RCaN

Supplementary Information

Hilaire Drouineau[1], Benjamin Planque[2], and Christian Mullon[3]

[1]INRAE, Bordeaux, France
[2]HI, Tromsoe, Norway
[3]IRD, MARBEC, Sete, France

April 26, 2021

1 Introduction

In this Supplementary Information document, we present an example of a RCaN study. That is a concrete ecological question on a real ecosystem and then a sequence of RCaN-commands: their input, their output and their interpretation. The case study is that the Barents sea that has been presented in the main text. The RCaN file has been previously built, using the RCaN constructor; it is attached to this supplementary information. All following commands give rise to R-Markdown file that is also attached.

It is organized as follows:

1. We give a short glance at the RCaN-file.
2. Then we run main steps of the method after the construction of the RCaN file: build polytope, analyze polytope, compute sample, analyze sample.
3. Then after synthesizing the analysis of sample, we improve the set of constraints, and give the results of a second run after removing some constraints
4. Finally, we make comparisons between both runs and their interpretation

2 Preliminary: R Environment

Firstly RCaN has to be downloaded. This is done once for all and the installation instructions are given in the ReadMe file.

Before running RCaN commands; a few libraries are to be loaded.

```
> library(RCaN) #the main package
> library(ggplot2) #to draw results
> library(coda) #to explore mcmc
> library(dplyr) #to manipulate data frame
> library(xtable) #to create latex tables
> library(xlsx) # to import excel files
> library(cowplot)
```

3 The RCaN file

Parameters, observations and constraints have been gathered in an Excel file with a specific structure.

```
> setwd('/Users/christianmullon/gitC/article_supporting/Sweave')
> NAMEFILE <- 'BarentsSeaReconstructions_20210302.xlsx'
> # NAMEFILE <- 'CaN_template_miniS.xlsx'
```

3.1 Components

How the list of components has been established? From witch knowledge? By whom?

	Component	Inside	AssimilationE	Digestibility	OtherLosses
1	PP	0.00	0.00	0.65	0.00
2	Hzoo	1.00	1.00	0.90	8.40
3	Ozoo	1.00	1.00	0.90	5.50
4	Benthos	1.00	0.94	0.60	1.50
5	PelF	1.00	0.90	0.90	2.85
6	DemF	1.00	0.93	0.85	1.65
7	MM	1.00	1.00	0.00	5.50
8	Birds	1.00	0.84	0.00	60.00
9	NorSeaZoo	0.00	0.00	0.00	0.00
10	Fisheries	0.00	0.00	0.00	0.00

Table 1: Components

3.2 Fluxes

How the network has been designed? From which knowledge? By whom?

	Flux	From	To	Trophic
1	PP_Ozoo	PP	Ozoo	1.00
2	PP_Hzoo	PP	Hzoo	1.00
3	PP_Benthos	PP	Benthos	1.00
4	Hzoo_Ozoo	Hzoo	Ozoo	1.00
5	Hzoo_PelF	Hzoo	PelF	1.00
6	Ozoo_Ozoo	Ozoo	Ozoo	1.00
7	Ozoo_PelF	Ozoo	PelF	1.00
8	Ozoo_DemF	Ozoo	DemF	1.00
9	Ozoo_MM	Ozoo	MM	1.00
10	Ozoo_Birds	Ozoo	Birds	1.00
11	Benthos_Benthos	Benthos	Benthos	1.00
12	Benthos_DemF	Benthos	DemF	1.00
13	PelF_PelF	PelF	PelF	1.00
14	PelF_DemF	PelF	DemF	1.00
15	PelF_MM	PelF	MM	1.00
16	PelF_Birds	PelF	Birds	1.00
17	DemF_DemF	DemF	DemF	1.00
18	DemF_MM	DemF	MM	1.00
19	NorSeaZoo_Hzoo	NorSeaZoo	Hzoo	0.00
20	NorSeaZoo_Ozoo	NorSeaZoo	Ozoo	0.00
21	PelF_Fisheries	PelF	Fisheries	0.00
22	DemF_Fisheries	DemF	Fisheries	0.00
23	MM_Fisheries	MM	Fisheries	0.00
24	Ozoo_Fisheries	Ozoo	Fisheries	0.00

Table 2: Fluxes

nf is the number of fluxes.

3.3 Observations

How observations have been collected? From which inquiries? By whom?

	Year	Prod_Sat	Hzoo_Biomass	Ozoo_Biomass	Pelagics
1	1988.00		25432.12	24275.61	428.28
2	1989.00		31987.20	16130.85	864.52
3	1990.00		23027.73	7481.54	5831.66
4	1991.00		21188.34	16833.36	7288.56
5	1992.00		27314.02	7940.31	5152.50
6	1993.00		37612.31	11880.41	799.64
7	1994.00		72438.19	22699.62	203.94
8	1995.00		57941.78	23526.60	195.66
9	1996.00		38465.04	24633.25	504.21
10	1997.00		43364.75	19153.71	912.15
11	1998.00	799240.57	39682.56	20505.95	2057.46
12	1999.00	747073.95	36009.62	15105.06	2779.21
13	2000.00	1050000.00	41920.38	14611.77	4276.77
14	2001.00	1170000.00	31697.88	16117.23	3634.11
15	2002.00	1110000.00	35601.60	9402.57	2212.47
16	2003.00	874724.94	31731.20	14754.03	535.53
17	2004.00	1070000.00	36542.45	24127.27	960.22
18	2005.00	909010.84	42440.34	21917.78	574.61
19	2006.00	1100000.00	45739.13	25576.68	1027.10
20	2007.00	946826.75	39293.31	21387.11	2095.59

Table 3: Observations

nt is the number of time steps. It is the number of observation years.

3.4 Constraints

How constraints have been designed? From which knowledge? By whom?

	Id	Constraint
1	C01	PP_Hzoo + PP_Ozoo + PP_Benthos <= Prod_Sat * 1.5
2	C02	PP_Hzoo + PP_Ozoo + PP_Benthos >= Prod_Sat / 1.5
3	C03	PP_Hzoo + PP_Ozoo + PP_Benthos <= 2000000
4	C04	PP_Hzoo + PP_Ozoo + PP_Benthos >= 500000
5	C05	NorSeaZoo_Hzoo = 8 * 1600
6	C06	NorSeaZoo_Ozoo = 2 * 1600
7	C07	PelF_Fisheries >= Pel_landings
8	C08	PelF_Fisheries <= Pel_landings * 1.1
9	C09	DemF_Fisheries >= Dem_landings
10	C10	DemF_Fisheries <= Dem_landings * 1.1
11	C11	MM_Fisheries >= Mammals_landings / 1.1
12	C12a	MM_Fisheries <= Mammals_landings * 1.1 + 1.5
13	C12b	MM_Fisheries <= Mammals_landings * 1.1
14	C15	Ozoo_Fisheries >= Shrimp_Landings
15	C16	Ozoo_Fisheries <= Shrimp_Landings * 1.1
16	C17	Benthos <= 66 * 1600 * 3
17	C18	Benthos >= 66 * 1600 / 3
18	C19	MM <= 0.34 * 1600 * 3
19	C20	MM >= 0.34 * 1600 / 3
20	C21	Birds <= 0.007 * 1600 * 3

Table 4: Constraints

4 Building the polytope

This is the first step. The polytope is defined by two pairs of a matrix and and a vector. F being the vector of all flows at all timesteps, first one (A, b) is an equality $A.F = b$, second one (C, v) is an eqality $C.F \leq v$. This RCaN commands computes once for all thes matrices and vectors from the constrainst and the observations.

```
> system.time(POLYTOPE <- buildCaN(NAMEFILE))

   user  system elapsed
107.065   1.428 113.624

> summary(POLYTOPE)

                 Length  Class      Mode
components_param      10 data.frame list
species                7 -none-     character
fluxes_def             4 data.frame list
flow                  24 -none-     character
series                22 data.frame list
ntstep                 1 -none-     numeric
data_series_name      21 -none-     character
constraints            5 data.frame list
H                     49 -none-     numeric
N                    168 -none-     numeric
A                 2009575 dgCMatrix  S4
AAll              2009575 dgCMatrix  S4
C                   49600 dgCMatrix  S4
CAll                49600 dgCMatrix  S4
v                      64 -none-     numeric
vAll                   64 -none-     numeric
L                  173600 dgCMatrix  S4
b                    2593 -none-     numeric
bAll                 2593 -none-     numeric
symbolic_enviro       903 -none-     environment
```

4.1 Structure of the polytope

The polytope is defined inside a space with a very high number of dimensions. For the Barents sea example, we have:

```
> dim(POLYTOPE$A)

[1] 2593  775

> length(POLYTOPE$b)
```

```
[1] 2593

> dim(POLYTOPE$C)

[1]  64 775

> length(POLYTOPE$v)

[1] 64
```

In a general way: if $dim(C) = (x_C, y_C)$, $dim(A) = (x_A, y_A)$, $length(b) = l_b$, $length(v) = l_v$, then :

1. $y_A = y_C$ is the dimension of the underlying space. $x_A = l_b$ is the number of equality constraints; the proper dimension of the polytope is $y_A - x_A$;
2. $y_C = y_A$ is the number of inequality constraints; this is the number of faces of the polytope.

4.2 Checking the polytope

As it is defined in the RCaN file for the Barents' sea, the polytope is non-empty and bounded:

```
> checkPolytopeStatus(POLYTOPE)

[1] "polytope ok"
```

4.3 Bounds of the polytope

Limits of the Barents' sea polytope in all dimensions are obtained with getAllBoundsParam. This is a time consuming command. Results are a vector with $nf \times nt$ components.

```
> # system.time(BOUNDS <- getAllBoundsParam(POLYTOPE, progressBar = FALSE))
> # head(BOUNDS,10)
```

4.4 Slices the polytope

Function plotPolytope2D allows seeing the polytope in the plane defined by two parameters. In its first two dimensions, for the second 1990, the Barents sea polytope dimensions appears as.

```
> nX1 <- FLUXES[5,1]
> nY1 <- FLUXES[8,1]
> fluxX1 <- paste(nX1,'[1990]',sep="")
> fluxY1 <- paste(nY1,'[1990]',sep="")
> g1 <- plotPolytope2D(POLYTOPE, c(fluxX1, fluxY1), progressBar=FALSE)
> nX2 <- FLUXES[3,1]
> nY2 <- FLUXES[12,1]
> fluxX2 <- paste(nX2,'[1990]',sep="")
> fluxY2 <- paste(nY2,'[1990]',sep="")
> g2 <- plotPolytope2D(POLYTOPE, c(fluxX2, fluxY2), progressBar=FALSE)
> plot_grid(g1, g2)
```

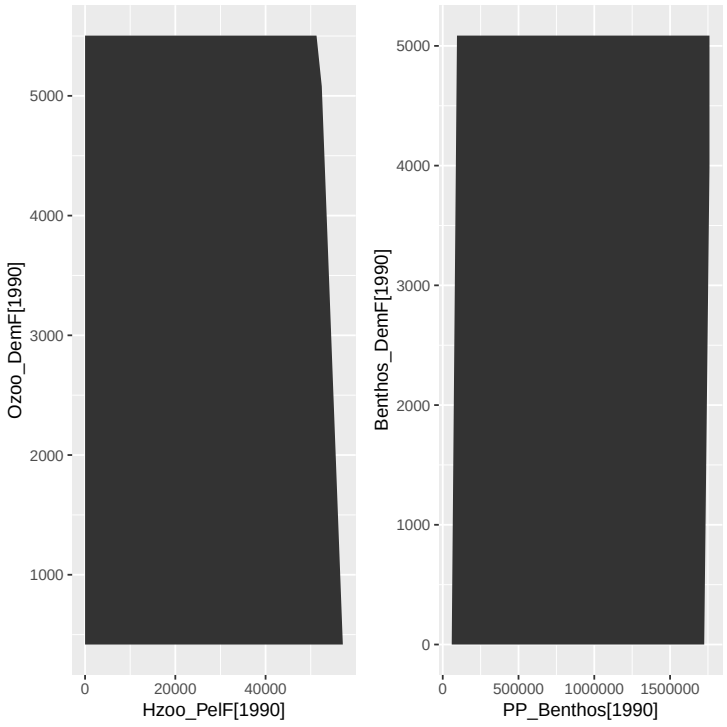

Figure 1: Figure 1

5 Sampling the polytope

5.1 Sampling

The most important step of the method is the sampling of the polytope, that is the building many random possible trajectories. This is accomplished with the following RCaN command:

```
> system.time(SAMPLE <- sampleCaN(POLYTOPE, N=100, thin=100, nchain=2, ncore=2))

   user  system elapsed
  0.554   0.421 605.065
```

5.2 Convergence of the sample chains

To check the mixing of the Monte Carlo Marko chain algorithm that is used to build the sample, we use commands from the coda library, for example nchain (dimensions of MCMC objects), pcramer (Cramer-von Mises distribution), etc.

```
> # nchain(SAMPLE$mcmc)
> # summary(SAMPLE$mcmc)
```

Gelman and Rubin (1992) propose a general approach to monitoring convergence of MCMC output in which $m > 1$ parallel chains are run with starting values that are overdispersed relative to the posterior distribution. Convergence is diagnosed when the chains have 'forgotten' their initial values, and the output from all chains is indistinguishable. The gelman.diag diagnostic is applied to a single variable from the chain. It is based a comparison of within-chain and between-chain variances, and is similar to a classical analysis of variance.

```
> fluxZ1 <- paste(FLUXES[3,1],'[1990]',sep="")
> fluxZ2 <- paste(FLUXES[17,1],'[1990]',sep="")
> fluxZ3 <- paste(FLUXES[22,1],'[1990]',sep="")
> fluxZ4 <- paste(FLUXES[13,1],'[1990]',sep="")
> gelman.diag(SAMPLE$mcmc[,fluxZ1])

Potential scale reduction factors:

     Point est. Upper C.I.
[1,]       1.27       1.87

> gelman.diag(SAMPLE$mcmc[,fluxZ2])

Potential scale reduction factors:

     Point est. Upper C.I.
[1,]      0.991      0.992

> gelman.diag(SAMPLE$mcmc[,fluxZ3])

Potential scale reduction factors:

     Point est. Upper C.I.
[1,]          1       1.01

> gelman.diag(SAMPLE$mcmc[,fluxZ4])

Potential scale reduction factors:

     Point est. Upper C.I.
[1,]       1.01       1.05
```

The autocorrelation function of the polytope sampling chains is illustrative of the independence of successive samples. If the samples are not serially correlated (which is desired) the autocorrelation should remain around zero (except for the first value which is always 1).

```
> fluxZ1 <- paste(FLUXES[3,1],'[1990]',sep="")
> fluxZ2 <- paste(FLUXES[17,1],'[1990]',sep="")
> fluxZ3 <- paste(FLUXES[22,1],'[1990]',sep="")
> fluxZ4 <- paste(FLUXES[13,1],'[1990]',sep="")
> g1 <- acfplot(SAMPLE$mcmc[,fluxZ1])
> g2 <- acfplot(SAMPLE$mcmc[,fluxZ2])
> g3 <- acfplot(SAMPLE$mcmc[,fluxZ3])
> g4 <- acfplot(SAMPLE$mcmc[,fluxZ4])
> plot_grid(g1, g2, g3, g4, ncol = 2, nrow = 2,
+           labels = c('fluxZ1', 'fluxZ2', 'fluxZ3', 'fluxZ4'), align="hv")
```

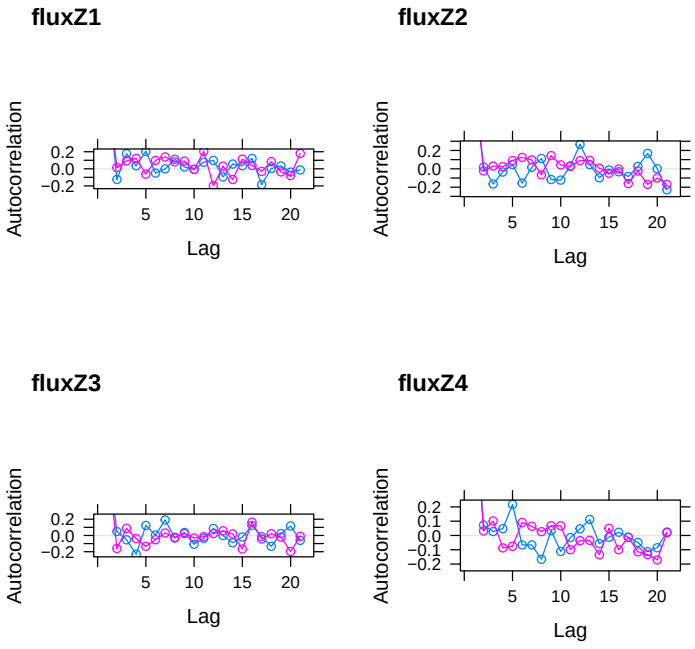

Figure 2: Figure 2

```
> g <- ggSeries(SAMPLE, POLYTOPE$species, TRUE)
> g + scale_y_log10() + guides(color = FALSE, fill = FALSE)
```

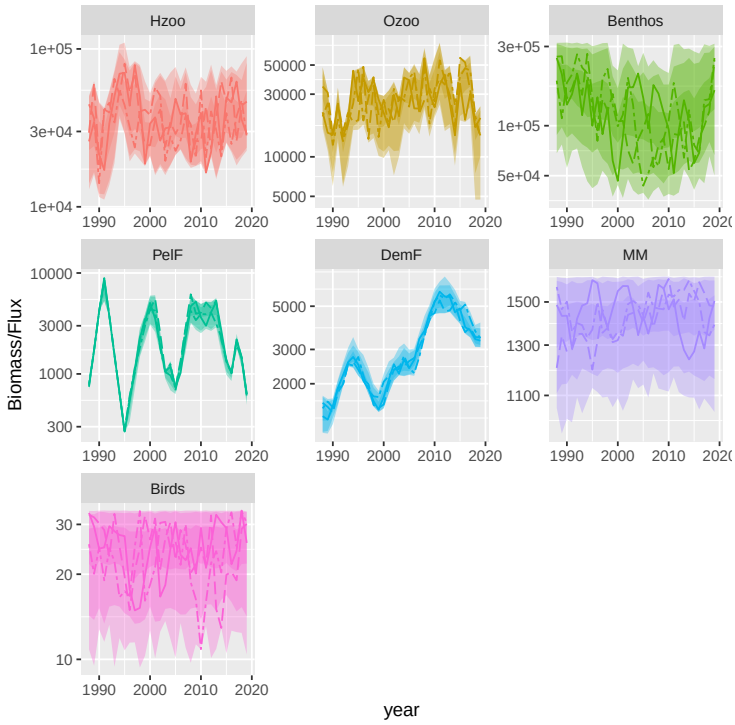

Figure 3: Figure 8

5.3 Food-web dynamics. Components

Temporal dynamics. The function 'ggSeries' plots the time series of individual or collections of components and/or fluxes. By default, 3 randomly picked samples are plotted (plain, dashed and dotted lines) together with the envelopes containing 100%, 95% and 50% of the samples. The example below is for 2 fluxes (Primary production to omnivorous and herbivorous zooplankton) and one component (herbivorous zooplankton).

```
> g <- ggSeries(SAMPLE, POLYTOPE$flow, TRUE)
> g + scale_y_log10() + guides(color = FALSE, fill = FALSE)
```

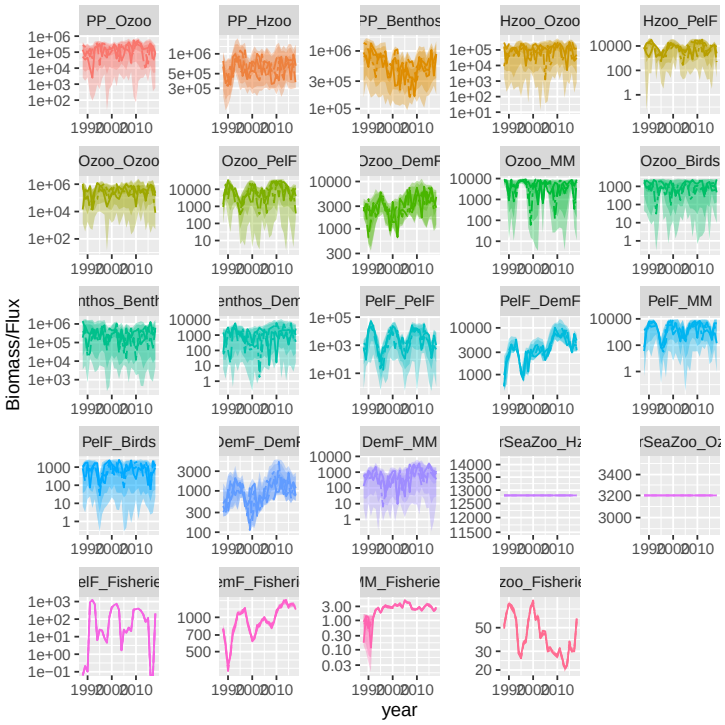

Figure 4: Figure 9

5.4 Food-web dynamics. Fluxes

```
> ggViolin(SAMPLE,POLYTOPE$species,year=1990,TRUE) + xlab('Component') + ylab('Biomass')
```

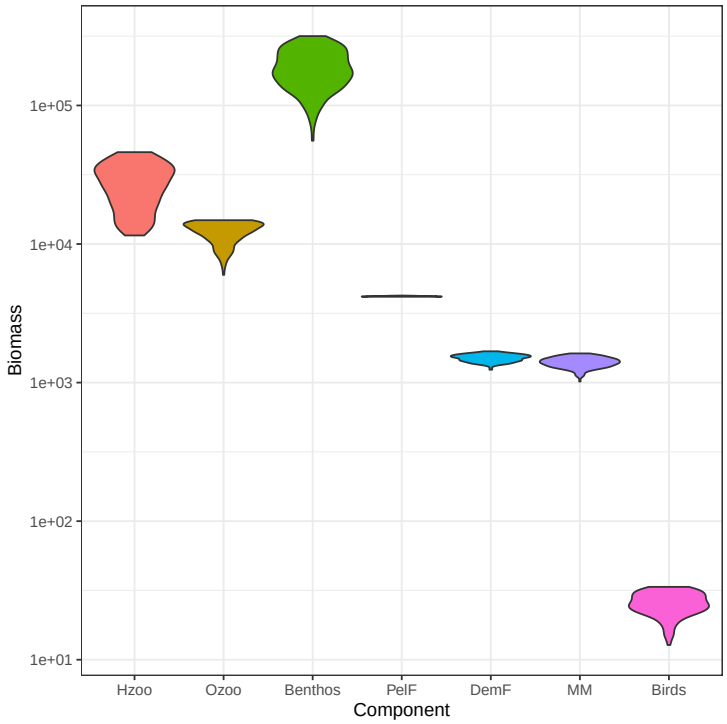

Figure 5: Figure 10

5.5 Distribution. Components

Distribution of biomass and fluxes. The function 'ggViolin' plots the empirical distribution of the sampled biomass or fluxes, for a given year. Below is an illustration of the violin plots for the biomass of all components in the year 1990.

```
> ggViolin(SAMPLE,POLYTOPE$species,year=1990,TRUE) + xlab('Component') + ylab('Biomass')
```

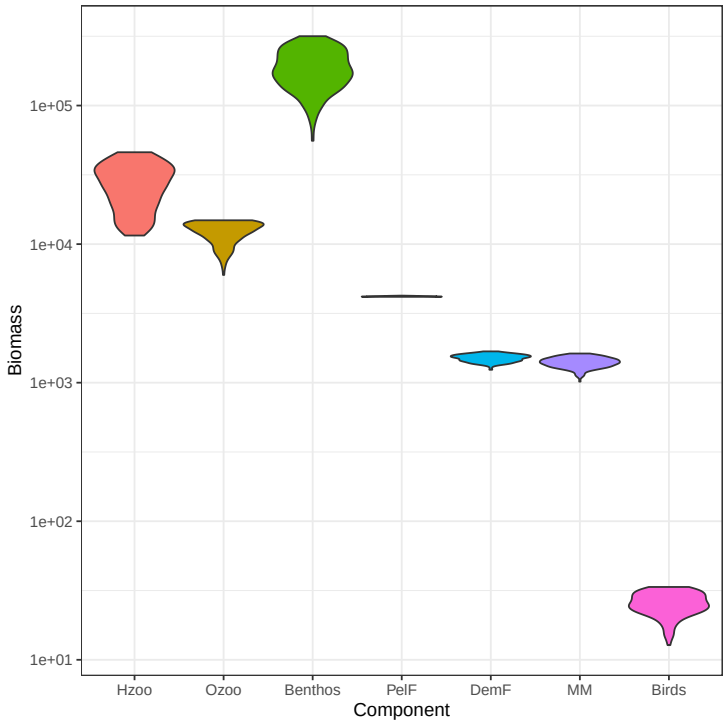

Figure 6: Figure 12

```
> ggViolin(SAMPLE,POLYTOPE$flow,year=1990,TRUE) + xlab('Component') + ylab('Biomass')
```

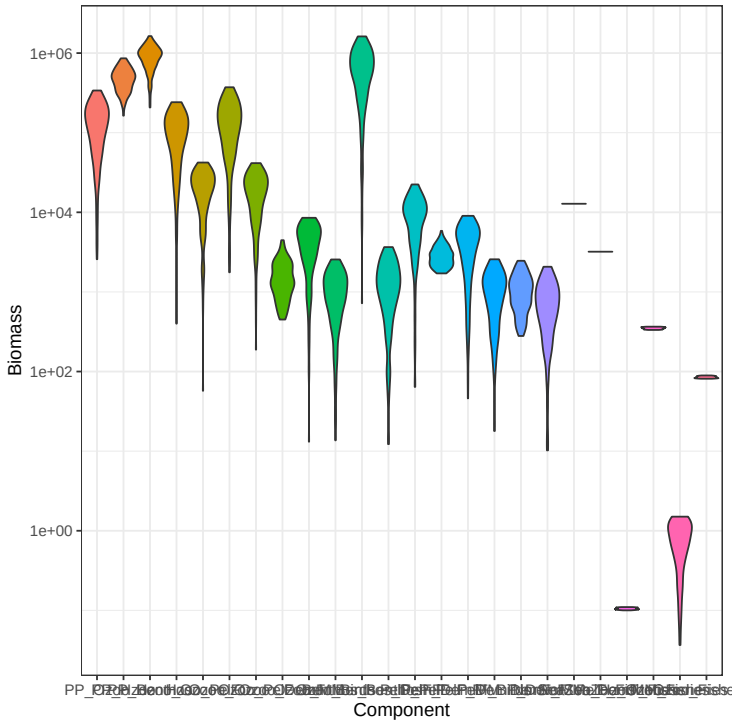

Figure 7: Figure 11

5.6 Distribution. Fluxes

```
> ggDiet(SAMPLE, POLYTOPE$species) + ylab('prey proportion in the diet')
```

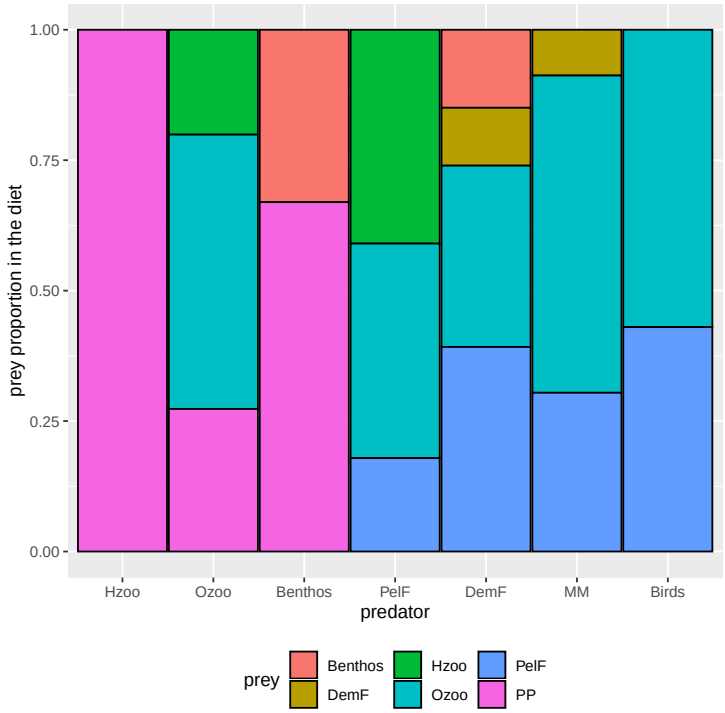

Figure 8: Figure 13

5.7 Diets

Diet composition. The function 'ggDiet' plots the mean proportions of prey eaten by specific predator(s). Below is an example of the diet fractions for all the components over the entire sampling period.

```
> ggGrowth(SAMPLE, POLYTOPE$species)
```

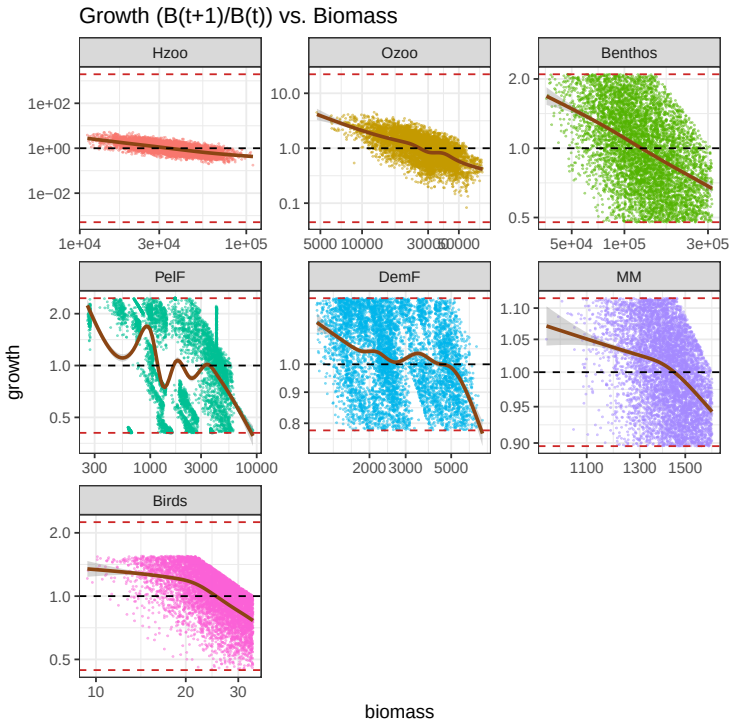

Figure 9: Figure 14

5.8 Growth

Growth versus biomass. The function 'ggGrowth' plots the growth rate (measured as the ratio of biomass at time $t+1$ over the biomass at time t) as a function of the biomass (at time t). This figure illustrate how density-dependence of single components can emerge from the food-web dynamics. The upper and lower red-dotted lines visualise the minimum and maximum growth rates set by the inertia constraints for each component. Below is an example of growth vs. biomass for all components over the entire sampling period.

```
> ggTrophicRelation(SAMPLE, POLYTOPE$species[3:5])
```

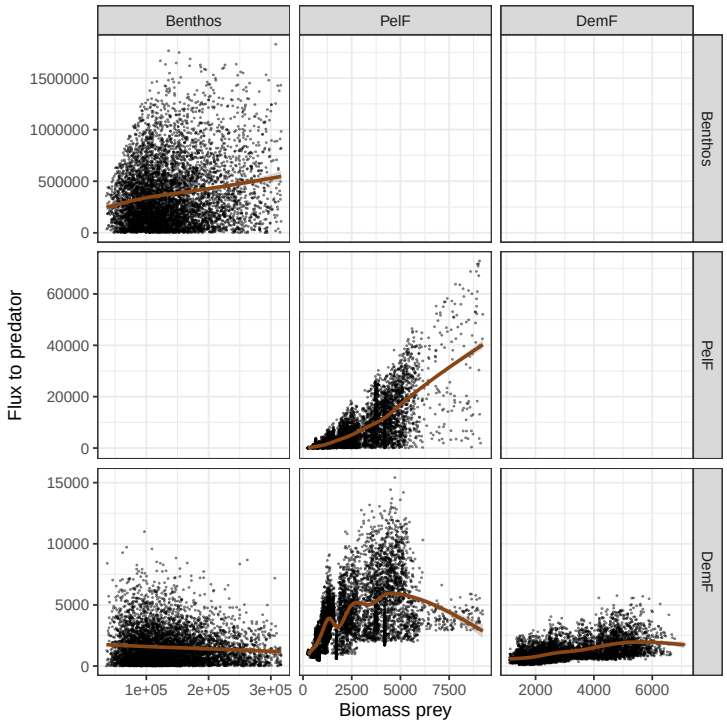

Figure 10: Figure 14

5.9 Trophic functional relationships

The function 'ggTrophicRelation' plots the empirical relationships between the biomass of a prey and the quantity of that prey consumed by a predator. Below is an example of trophic functional relationships between benthos, pleagic fish and demersal fish over the entire sampling period.

```
> ggTrophicRelation(SAMPLE, POLYTOPE$species)
```

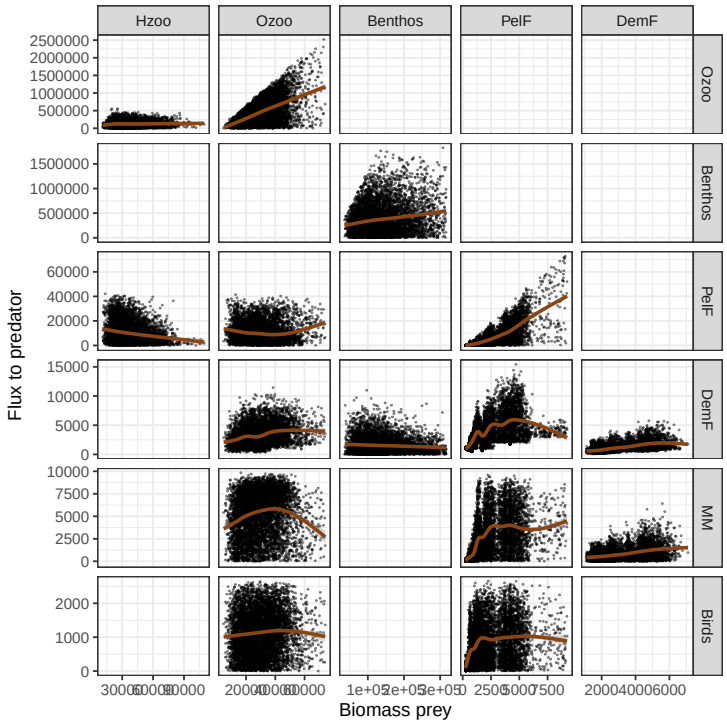

Figure 11: Figure 15

```
> ggSatiation(SAMPLE, POLYTOPE$species)
```

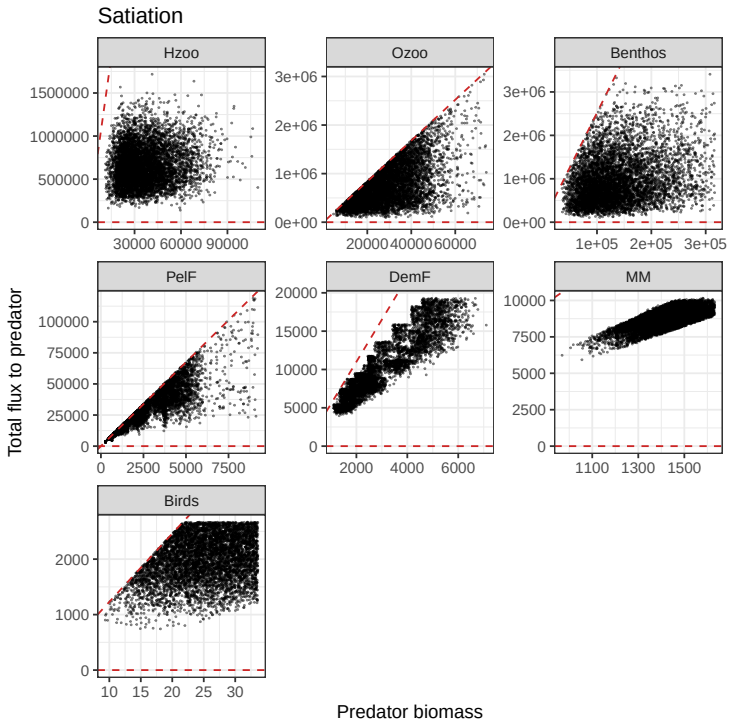

Figure 12: Figure 16

5.10 Satiation

The function 'ggSatiation' plots the total biomass of prey eaten by a predator, as a function of the biomass of the predator. In addition the plot includes an indication (upper red dotted line) of the maximum possible consumption rate, derived from the satiation parameter of the predator species. An example is illustrated below for all model components and over the entire sampling period.

```
> ggSatiatInertia(SAMPLE, POLYTOPE$species)
```

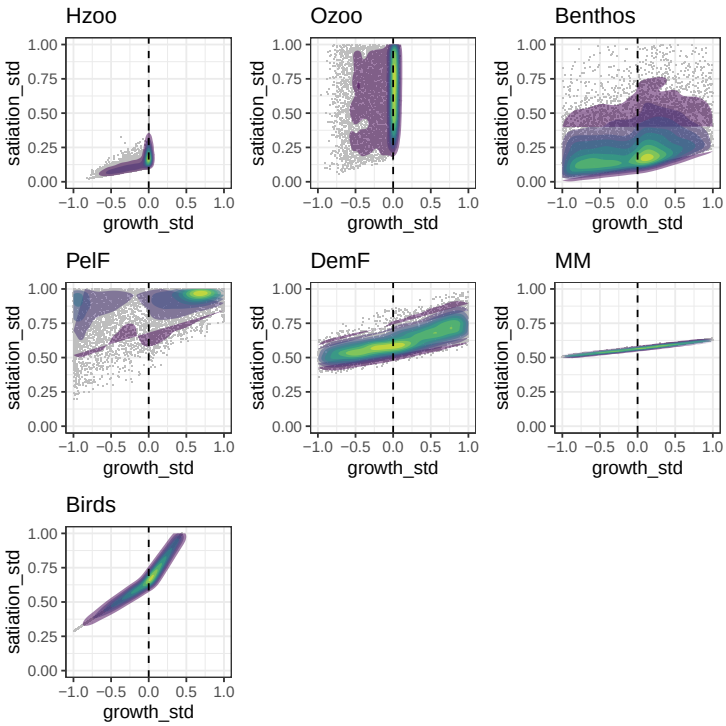

Figure 13: Figure 17

5.11 Feeding and Growth

The function 'ggSatiatInertia' is used to explore the relationships between population growth and feeding. The plot shows standardised total consumption(0 = no feeding, 1 = feeding to satiation) as a function of standardised population growth/mortality (-1 = maximum mortality, 0 = no change in biomass, 1 = maximum growth). the total biomass of prey eaten by a predator, as a function of the biomass of the predator. A positive relationship between growth and satiation is indicative of bottom up control. An example is illustrated below for all model components and over the entire sampling period.

```
> ggPairsBiomass(SAMPLE,logscale=FALSE)
```

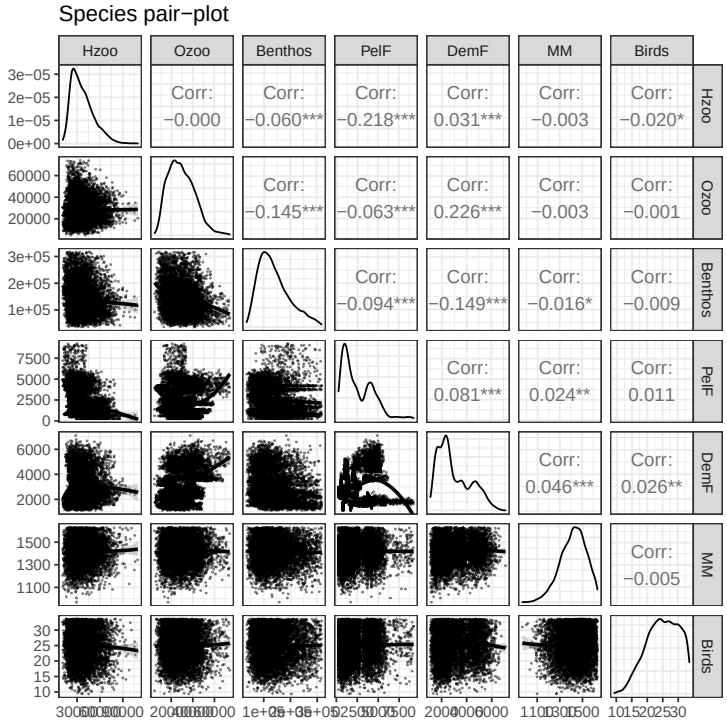

Figure 14: Figure 19

5.12 Pair plots

The function 'ggPairsBiomass' is used to explore the relationships between the biomass of different food-web components. The plot shows the individual component density function in the diagonal, the scatterplots of one species against the other in the lower triangle and the Kendall correlation coefficient in the upper triangle. An example is illustrated below for all model components and over the entire sampling period.

```
> ggPairsBiomass(SAMPLE,POLYTOPE$species[c(5,6)],logscale=FALSE)
```

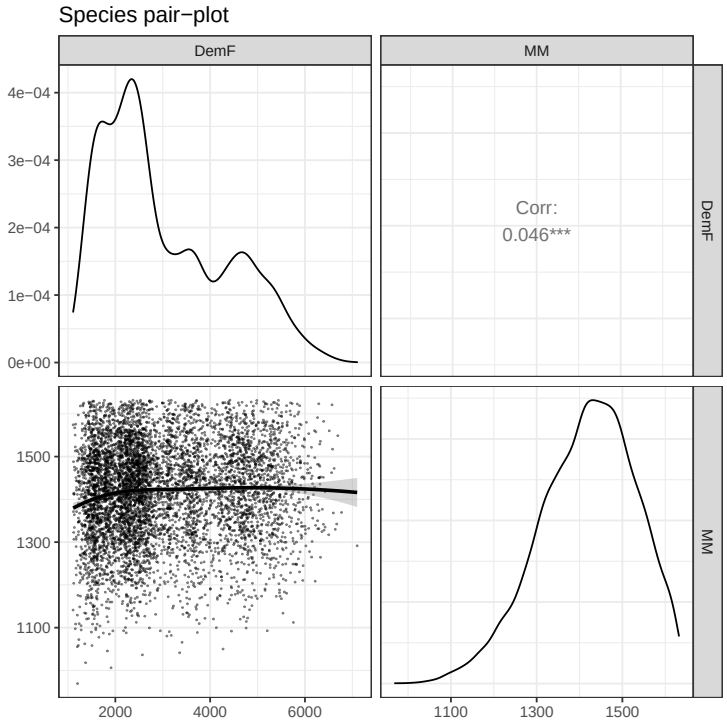

Figure 15: Figure 18

6 Try and errors

6.1 Activating and desactivating constraint

```
> # disactivate constraints C08 : Max pelagic catches
> constA <- CONSTRAINTS[8,1]
> constA

[1] "C08"

> POLYTOPEA <- toggleConstraint(POLYTOPE, constA)

 [1] "disactivate inequality C08 : 1988" "disactivate inequality C08 : 1989"
 [3] "disactivate inequality C08 : 1990" "disactivate inequality C08 : 1991"
 [5] "disactivate inequality C08 : 1992" "disactivate inequality C08 : 1993"
 [7] "disactivate inequality C08 : 1994" "disactivate inequality C08 : 1995"
 [9] "disactivate inequality C08 : 1996" "disactivate inequality C08 : 1997"
[11] "disactivate inequality C08 : 1998" "disactivate inequality C08 : 1999"
[13] "disactivate inequality C08 : 2000" "disactivate inequality C08 : 2001"
[15] "disactivate inequality C08 : 2002" "disactivate inequality C08 : 2003"
[17] "disactivate inequality C08 : 2004" "disactivate inequality C08 : 2005"
[19] "disactivate inequality C08 : 2006" "disactivate inequality C08 : 2007"
[21] "disactivate inequality C08 : 2008" "disactivate inequality C08 : 2009"
[23] "disactivate inequality C08 : 2010" "disactivate inequality C08 : 2011"
[25] "disactivate inequality C08 : 2012" "disactivate inequality C08 : 2013"
[27] "disactivate inequality C08 : 2014" "disactivate inequality C08 : 2015"
[29] "disactivate inequality C08 : 2016" "disactivate inequality C08 : 2017"
[31] "disactivate inequality C08 : 2018" "disactivate inequality C08 : 2019"

> checkPolytopeStatus(POLYTOPEA)

[1] "polytope not bounded"
```

6.2 Building and analyzing sample

```
> system.time(SAMPLEA <- sampleCaN(POLYTOPEA, N=100, thin=100, nchain=2, ncore=2))

   user  system elapsed
  0.610   0.464 746.084
```

```
> fluxX <- FLUXES[1,1]
> fluxY <- FLUXES[2,1]
> compA <- COMPONENTS[2,1]
> compB <- COMPONENTS[3,1]
> c(fluxX,fluxY,compA)

[1] "PP_Ozoo" "PP_Hzoo" "Hzoo"

> g <- ggSeries(SAMPLEA, POLYTOPEA$species, TRUE)
> g + scale_y_log10() + guides(color = FALSE, fill = FALSE)
```

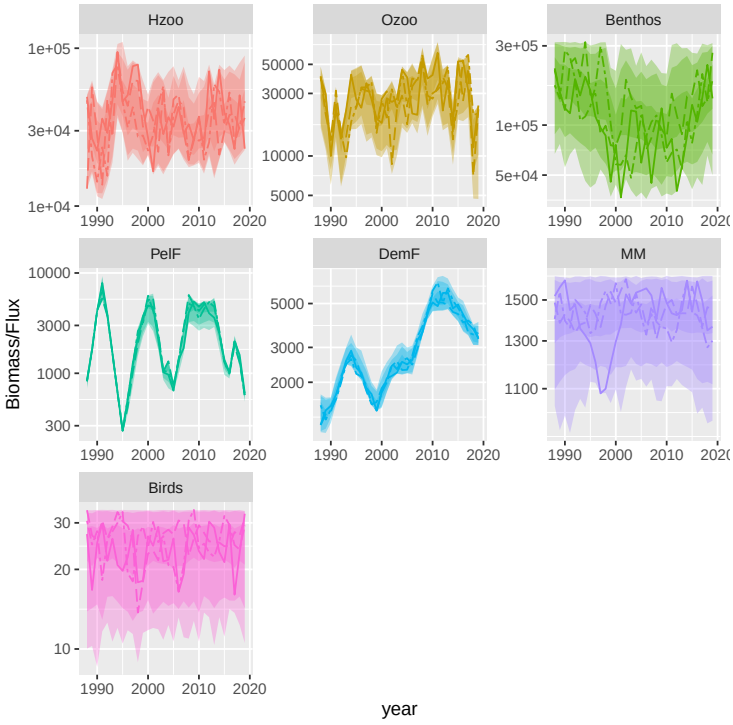

Figure 16: Figure 21

7 Iterpreting differences