RCaN : a software for Chance and Necessity modelling

Hilaire Drouineau¹ Benjamin Planque² Christian Mullan³

Abstract

Uncertainty is a challenge in modelling ecological systems and has been a source of misunderstandings between modelers and non-modelers. The "Chance and Necessity" (CaN) modelling approach has been proposed to address this issue, in the case of trophic network modelling. CaN modelling focuses exploring food-web trajectories that can satisfy fundamental physical and biological laws, while being compatible with observations and domain knowledge. This type of approach can facilitate discussion among actors as it promotes sharing of information and does not presuppose any knowledge of modelling practices. It is therefore suitable for participatory modelling , i.e. a modelling approach in which different actors can confront their knowledge and understanding of the marine system and of the associated uncertainties.

¹INRAE, UR EABX, 50 avenue de Verdun, CEDEX, 33612 Cestas, France, *Email: hilaire.drouineau@inrae.fr*

²Institute of Marine Research, Norway, P.O. Box 6606 Langnes, 9296 Tromsø, Norway. Corresponding author. *Email: benjamin.planque@hi.no*

³IRD, UMR MARBEC, Avenue Jean Monnet, Sete, France. *Email: christian.mullan@ird.fr*

One important ingredient to achieve participatory modelling is the availability of a modelling platform that is efficient, fast and transparent, so that all actors can understand and follow the modelling steps and choices, and can rapidly visualize and discuss the results. But, until now, there existed no software to easily perform CaN modelling. Here, we present RCaN and RCaNconstructor. Combined, these provide the first tool to build CaN models in an intuitive way that is 1) suitable within participatory frameworks, 2) transparent, 4) computationally efficient, 5) fully documented through the provision of meta-information and 6) supportive of exploratory analyses through predefined graphical functions.

Keywords: trophic food web, uncertainty, participatory modelling, polytope sampling, R, Java.

1 Introduction

Modelling complex ecological systems is often a trade-off between the accurate representation of the many components and interactions within an ecosystem and the reduction of ecosystem processes to few and important elements. Rather than attempting to model multiple ecological processes in detail, CaN modelling focuses on exploring possible ecological dynamics given a set of constraints. The principles of CaN modelling and the corresponding mathematical formulation have been outlined in [1]. The general idea is that it is easier for various actors (managers, scientists, stakeholders) to agree about ecological constraints, which separate what is possible from what isn't, than to agree on the mechanistic

39 formulation of detailed ecological processes. This is particularly true when mod-
 40 elling complex ecological systems for which available information/observations
 41 are limited. This approach is called “Chance and Necessity” modelling (CaN):
 42 chance refers to the indeterminacy of ecological processes while necessity reflects
 43 the constraints that delineate what is possible from what isn’t.

44 While the CaN principles can be applied to a wide range of ecological prob-
 45 lems, the first application was specifically designed to explore the temporal
 46 dynamics of marine systems [1]. This has been motivated by the shift from
 47 conventional single-species fisheries management towards ecosystem-based fish-
 48 ery management (EBFM, see for example [2, 3]). This shift implies up-scaling
 49 of the traditional population scale management into a complex set of biotic (e.g.
 50 trophic food interactions) and abiotic sets of interactions (e.g. interactions with
 51 physical habitats). This has led to the development of complex ecological mod-
 52 els, but which ability to help management is questioned by many actors who may
 53 have limited trust in such models [4]. In this context, there is a call for trans-
 54 parency in the science that supports management, and participatory modelling
 55 and assessment have been proposed as a way to support mutual trust. CaN
 56 is proposed as one tool to support transparency and participatory modelling
 57 by providing a simplified food-web modelling framework, in which the ideas of
 58 uncertainty and ecological limits are central. Assumptions about knowns and
 59 unknowns and about possible and impossible processes are presented explic-
 60 itly and all model constituents can be described in plain language (little or no
 61 modelling jargon).

By contributing to the transfer of biomass, energy and pollutants in ecosystems, trophic interactions play a major role in ecosystem functioning [5]. Trophic food web models have proved to be key tools for the quantification of these fluxes, they provide a holistic view of ecosystem functioning, and are increasingly used to assess ecosystem ecological status and to explore the effect of anthropogenic pressures such as fisheries [6]. Food web models, mostly Ecopath with Ecosim (EwE) [7, 8], have played an important role in the implementation of the EBFM. CaN modelling further contributes to this effort by providing a modelling framework for marine food-webs that primarily relies on ecological constraints which can be agreed upon by diverse actors.

While the equations behind CaN modelling are simple, one major technical challenge is to sample a polytope in a highly dimensional space. This is particularly useful if achieved inside the R framework [9], which is widely used by ecological modelers.

Until now, there existed no software to easily perform CaN modelling. The first models were constructed in an *ad hoc* fashion by assembling and structuring relevant data and developing tailor-made programs in Mathematica or R to run the model. This made it difficult to communicate and replicate CaN models. It also restrained the access of the model to non-modelers. Making CaN modelling transparent, replicable and efficient in a participatory modelling framework requires that model construction, sampling and output visualization be easily accessible by modelers and non-modelers alike, that the structure of CaN models be standardised and that the model sampling be robust and effi-

cient. A graphical user interface (GUI), which provides an easy entry level for users is also a highly desirable feature for a modelling approach targeting non-modelers. In this contribution, we present a software for the implementation of CaN modelling in R, called RCaN, and an associated GUI in Java, called RCaNconstructor. RCaN and RCaNconstructor can be used jointly or separately. Their combined use allows users to easily construct, document, sample and interpret the results of CaN food-web models. The relationship between RCaN and RCaNconstructor is shown in figure 1. The main features of RCaN and RCaNconstructor include:

1. a user-friendly interface that allow multiple users to jointly develop and revise models. Users can then focus on the discussion about their knowledge of the ecological system and how to best structure a model that fits their needs,
2. a file and data management system that ensures consistency between the different elements of a CaN model. For example, it must be guaranteed that each link connects existing components or that each constraint points to components, links or data series that are adequately specified in the model,
3. a standardized file format to fully specify a CaN model,
4. a Graphical User Interface that facilitate the handling and replicability of data use and the tracking of observations sources (data files),
5. a set of R functions that allow experimented programmers to construct,

- manipulate and visualize the outputs of CaN models in a flexible manner,
6. standardized graphical outputs of the model results, which promote discussions among modelers and other actors and ease the interpretation of model results,
7. a system for registering meta-information about the model, such as model version, authors, data sources, assumptions, and so on. This ensures transparency and reproducibility of modelling experiments.

In section 2 we summarize the elements of the CaN modelling approach. In section 3, we present the format of an RCaN file. In section 4 we present the R library RCaN which is designed to construct, sample and produce graphical outputs of CaN food-web models. In section 5 we introduce RCaNconstructor, the graphical user interface that facilitates the co-construction of CaN models and the inclusion of meta-information. In section 6, we briefly describe the Barents Sea trophic network which we use as a case study example. Finally,

we discuss the possible future developments of the RCaN and RCaNconstructor softwares.

2 Principles of the CaN approach

In CaN, food-web dynamics is defined by time-trajectories of biomass and fluxes between trophic groups (also termed components). The biomass of all trophic groups and the fluxes between them can usually not be precisely observed. Therefore, the food-web dynamics is indeterminate, and a range of dynamics is possible. However, not everything is possible because some ecological or physiological constraints are known to bound the temporal evolution of ecological system, and/or because some observations can inform about the past dynamics of some components of the food-web. These eco-physiological and observational constraints can be translated into mathematical constraints that need to be satisfied before a food-web trajectory can be considered possible. The ensemble of these mathematical constraints constitutes the “necessity” in a CaN model. It is by focusing on constraints that can be easily expressed and communicated between actors and by explicitly acknowledging indeterminacy in natural systems that CaN modelling facilitate exchanges among actors to support resource management.

2.1 Network structure

In CaN modelling, the food-web is defined as a network. In this network, each vertex/node is called a component which can be a species, a trophospecies,

142 i.e., a group or sub-group of species that share common prey and predators,
143 or another component exchanging biomass with the rest of the system (e.g.
144 a fishery). Components may be included within the model domain, in which
145 case their biomass trajectories are explicitly considered. Alternatively, they
146 can be outside the model domain, in which case it is only the fluxes from/to
147 these components that are considered, but not the components trajectories. An
148 edge/link corresponds to a flux of biomass between two components. Most fluxes
149 are trophic: from a prey to a predator. Some fluxes are non-trophic, such as
150 import/export of biomass from adjacent regions or removal of biomass through
151 fishing.

152 2.2 Components' characteristics

153 Components inside the system are characterized by the following biological prop-
154 erties. **Satiation** reflects the maximum consumption rate per unit biomass of
155 the predator. **Inertia** expresses that variations in biomass from one year to the
156 next are bounded between a maximum growth rate and a maximum mortality
157 rate. **Digestibility** is a correction factor that accounts for variations in energy
158 content between prey. **Assimilation efficiency** expresses the proportion of
159 the biomass ingested by a predator that is effectively assimilated; the product
160 of the potential assimilation efficiency by the digestibility correction factor is the
161 absorption efficiency (the proportion of prey biomass digested and absorbed).
162 **Refuge biomass** is the absolute minimum biomass that a trophospecies can
163 reach. **Other losses** is a mortality coefficient that account for losses, i.e.,

164 metabolic losses and other mortality, not explicitly accounted for in the model.

165 Components outside the model domain do not require input parameter ex-
166 cept when they constitute a prey, in which case digestibility should be provided.

167 **2.3 Observations**

168 CaN modelling builds from past observations of the dynamics of the food-web.
169 Observations can be catch or landings data, biological samples of specific popu-
170 lations, stomach samples or other sources of diet information, survey abundance
171 estimates, outputs from stock assessment models, estimated ranges of biomass,
172 and so on.

173 **2.4 Constraints**

174 Constraints that limit the dynamics of the food-web are at the heart of CaN
175 modelling and they are the way by which most of the knowledge and data enter
176 a CaN model. Some constraints are implicitly incorporated into CaN while
177 others can be explicitly formulated.

178 CaN models assume conservation of energy during trophic exchanges, as do
179 many other trophic models. A set of implicit constraints is standard for all CaN
180 models: (a) biomass of compartments are positive and above refuge biomass,
181 (b) trophic fluxes between components are positive, (c) the sum of fluxes into a
182 compartment is limited according to its biomass (satiation), (d) the proportional
183 changes in biomass of a compartment are bounded (inertia).

184 Additional explicit constraints can be provided. These are written in the

form of inequalities or equalities. The left side of the (in)equality must contains a reference to one or several compartments or fluxes. The right side of the (in)equality can contain fixed values, compartments, fluxes and observational data-series.

There are two kinds of additional explicit constraints. Some constraints are the expression of a knowledge about system functioning and relate the biomass of a component to its incoming or outgoing fluxes. Other constraints restrict the possible values of a biomass or a flux on the basis of available observations. These last constraints constitute the way to deal with observation uncertainties in CaN modelling.

2.5 Trajectories

A food-web trajectory is defined by the ensemble of biomass fluxes at each time-step and by the initial food-web configuration, i.e., the initial biomass in each component. Each trajectory can be expressed as a single point in a multi-dimensional space of dimension $P = nF \times nt + nB_0$, with nF the number of fluxes, nt the number of time steps and nB_0 the number of initial biomasses. The number of dimensions can be high. For example, following the dynamics of a food-web with 10 components and 20 fluxes over 15 time steps will results in $P = 20 \times 15 + 10 = 310$ dimensions.

All constraints are linear equalities or inequalities that restrict the space of possible food-web trajectories. As constraints are linear, the set of the trajectories that satisfy all constraints has the shape of a convex polytope in a high

dimensional space.

2.6 Sampling trajectories

The main objective of CaN modelling is to sample food-web trajectories that are possible given (1) the food-web structure, (2) the species biological properties, (3) available observations and (4) constraints. The diversity of sampled trajectories expresses the indeterminacy in the system.

Sampling food-web trajectories is achieved by sampling this high-dimension polytope using dedicated sampling algorithms.

3 RCaN file

In RCaN, the input information describing the model structure is provided in a spreadsheet in the xlsx format. In the worksheet **Components**, the first column corresponds to the name of a component. The second column indicates whether the component is inside our outside the model domain. Other columns correspond to components characteristics (i.e. input parameters) that have been presented in section 2.2).

In the worksheet **Trophic fluxes**, each flux is characterized by an Id (first column) and links a source component (column From) to a sink component (column To) and by its type, trophic or non trophic (last column); in a trophic flux, digestibility and assimilation should be applied to the flux.

In the worksheet **Observations**, the first column gives the identifier of time steps (most of the time years). Other columns correspond to the observational

228 data-series that can be used to constrain the system (see 2.3).

229 In the worksheet **Constraints**, each row describes a constraint. Each con-
230 straint has an Id (first column), a formula defining the linear constraints between
231 fluxes, biomass and observations (second column), a time period of validity
232 (third column) and a flag indicating whether the constraint should be active or
233 not (fourth column). The contents of the above worksheets are interrelated and
234 naming of components and fluxes across worksheets should be consistent.

235 The additional worksheet **INFO** contains meta-information about the model.
236 It is not compulsory to provide this worksheet to be able to run RCaN, but it
237 is highly recommended to do so. The worksheet contains information about the
238 model contributors, the model version, the source of data and information, etc.,
239 all of which contribute to transparency and replicability.

240 4 RcaN

241 4.1 Introduction

242 RCaN is an R package to build, sample and visualize the results of CaN food-
243 web models. It does this by 1) constructing the polytope from implicit/explicit
244 constraints, 2) solving the numerical challenges of polytope sampling and 3)
245 producing graphical outputs. The next sections detail the technical features of
246 the implementation of these different steps, in the R language.

247 4.2 The different steps to use RCaN

248 4.2.1 Build a polytope from RCaN-file

249 Once the RCaN-file has been built, a first step is to organize the information and
250 to build the polytope describing the model. More precisely, the mass conserva-
251 tion of biomasses and constraints can be summarized in matrix notation, and
252 all these matrices should be automatically generated whatever the constraints
253 and parameters used by the modeler.

254 Constraints can be summarized in a matrix form as $A \cdot x \leq b$ (inequality
255 constraints) and $C \cdot x = v$ (equality constraints) with x a solution (e.g. a
256 trajectory defined by components' biomass at initial step and fluxes at each
257 time step), A and C two matrices with as many columns as parameters in the
258 model (fluxes per time step and initial biomass per component) and as many
259 rows as constraints. Vectors v and b specify the bounds of the equalities and
260 inequalities.

261 Using the constraints formula and the symbolic links among variables, RCaN
262 automatically builds matrices C and A and vectors b and v for each active
263 constraint. This is done by a simple call to the function **buildCaN** which reads
264 the information from the RCaN-file and returns a single object that contains
265 the trophic network and the corresponding polytope matrices. For example, in
266 the Barents Sea case study (section 6), the function automatically constructs
267 the matrix A which has 2593 rows and 775 columns, and the matrix C which
268 has 54 rows (and by definition, the same number of columns).

269 4.2.2 Checking polytope characteristics

270 At first try, the specification of model can lead to an empty polytope (i.e., a
 271 model with no solution that satisfies all the constraints) or an unbounded poly-
 272 tope (i.e., a model in which the specified constraints are not sufficient to bound
 273 a given parameter that can vary subsequently to infinity). Before running the
 274 model, it is wise to check the status of the polytope (the model won't run with
 275 an empty polytope while results are generally not relevant with unbounded
 276 polytopes). The package provides different functions to check the properties
 277 of the polytope, of which the two most important are **checkPolytopeStatus**,
 278 which checks whether the polytope is closed and non-empty, and **getAllBound-**
 279 **sParam**, which estimates the bounds for all parameters. If the model is empty,
 280 the RCaN function **findingIncompatibleConstr** can be used to identify which
 281 constraints are incompatible with each other.

282 4.2.3 Sampling polytope

283 The aim of this step is to perform a uniform sampling within a high dimensional
 284 polytope. The function **sampleCaN** carries out the uniform sampling and
 285 returns results in the standard **coda** format of a **mcmc.list** [10]. It is possible
 286 to sample several chains in parallel.

287 4.2.4 Diagnostics about sampling

288 Once sampling is done, it is necessary to check whether the uniform sampling
 289 was successful. Since the result of **sampleCaN** is a **mcmc.list** object, we can

use all the diagnostics tools provided in the package coda, such **traceplot**,
summary or **gelman.diag** or autocorrelograms.

4.2.5 Graphical analysis of sampling

RCaN provides several functions to explore the model results. The function
ggSeries plots the time-trajectories of biomass and fluxes and **ggViolin** plots
their empirical distributions. The function **ggDiet** provides a summary of the
diet fractions of different prey for each predator. **ggPairsBiomass** displays the
pairwise relationships between species biomasses. The function **ggTrophicRe-**
lation displays the relation between prey abundance and ingestion. **ggGrowth**
plots annual growth against population biomass, which is useful to discuss
density-dependence. **ggSatiation** shows how much a predator ingests against
the predator biomass and displays the maximum possible feeding (satiation).
ggSatiatInertia shows the standardized growth against standardized inges-
tion and is useful to investigate possible bottom controls. **ggBottleneck** aims
to detect potential bottleneck species by showing whether a species was close
from satiation (i.e., has more resources that it can assimilate) whereas its preda-
tors were far from satiation (i.e., its predators could potentially have eaten more
if food was available).

4.3 Technical considerations

The development of the packages raises several challenges. The first challenge
was to improve computation performance. For this purpose, RCaN is interfaced

311 with **C++** using two R packages **Rcpp** [11] and **RcppEigen** [12]. The second
 312 challenge was to check the characteristics of the polytope. This was achieved
 313 by using standard linear programming library: RCaN uses **lp_solve** through
 314 the R package **lpSolveAPI** [13]). The third challenge was also to translate
 315 the content of a RCaN-file into matrices, and especially to allow user to specify
 316 constraints in a simple language. This was made possible by using the R package
 317 **symengine** [14], an interface to the **symengine** symbolic manipulation library
 318 (the syntax of explicit constraints is presented in Appendix A.

319 RCaN can performs uniform sampling with two methods, the hit-and-run
 320 and the Gibbs sampler. Hit-and-run is a well-known algorithm for uniform
 321 sampling of high-dimensional convex space [15][16]. Basically, hit-and-run sam-
 322 pling starts from a point in the polytope. Then at each iteration, a direction
 323 of the high dimensional space is randomly drawn, and a new point is uniformly
 324 sampled between the two extreme bounds defined by the intersection with the
 325 envelop of the polytope. At each iteration, the point is moved in a single di-
 326 rection of the multidimensional space, and consequently, when the sampler is
 327 "trapped" in a narrow zone of the polytope, it may take many iterations to pick
 328 the appropriate direction to exit the narrow zone resulting in autocorrelation.
 329 A variation of the hit-and-run sampler is referred to as the Gibbs sampler or
 330 "Coordinate Hit-and-Run" [16][17]: instead of picking a single direction and
 331 updating all coordinates in this direction, at each iteration the Gibbs sampling
 332 successively and independently updates the value of each parameter (i.e., at
 333 each iteration, the sampler moves successively in all directions of the space)

334 to generate a new parameter value that satisfies the constraints. By doing so,
335 it guarantees an efficient convergence [17]. By default, RCaN uses the more
336 efficient Gibbs's sampler.

337 5 RCaNconstructor

338 The main goal of RCaNconstructor is to help users to construct and revise CaN
339 models in a participatory context. Internally, it ensures the consistency between
340 the different worksheets of the RCaN-file (changing the characteristics of an
341 object somewhere triggers the corresponding changes elsewhere). In addition,
342 RCaNconstructor facilitates the elicitation and tracking of meta-information.
343 RCaNconstructor can also execute RCaN commands.

344 RCaNconstructor is a Java program using Javafx for building the graphical
345 interface and allowing the code to be imported and run on external computers;
346 it uses the Java library **RCaller** for interfacing R and Java.

347 5.1 Meta information

348 Users can document the meta-information relevant for their model, using the
349 RCaNconstructor Meta information menu. The meta information template in-
350 vites contributions relevant to model version and authors, domain coverage and
351 units, definition of components and fluxes, sources for input parameters and
352 observations, rationale for constraints, model uncertainties and limitations.

353 5.2 Views

354 The View menu opens a collection of graphical interfaces. With these, the user
 355 can graphically draw the food-web structure (View/Network) enter the input
 356 parameters (View/Components), edit trophic links (View/Fluxes), load obser-
 357 vation files (View/Observations) and specify the model constraints (View/Con-
 358 straints). The meta-information and the model setup are saved in a ready to
 359 use RCaN-file.

360 5.3 R interface

361 The RCaNconstructor is interfaced to R with the RCaller library [18]. From the
 362 RCaNconstructor menu, the user can interactively: (a) start a R session and
 363 load RCaN library, (b) Build the model polytope (corresponds to the **build-**
 364 **CaN** command of RCaN), (c) sample the polytope (corresponds to the com-
 365 mand **sampleCaN**), (d) retrieve sampling diagnostics and (e) produce graphical
 366 outputs of the model results.

367 6 Example: the Barents Sea food-web

368 To illustrate CaN modeling using RCaN and RCaNconstructor, we use a sim-
 369 plified food-web of the Barents Sea as an example.

370 In what we present below, we have chosen to build the RCaN-file using
 371 RCaNconstructor and then to run RCaN commands to construct the polytope,
 372 sample it, and analyze the properties of the sample.

373 A more detailed presentation of the Barents Sea example is provided in the
374 Supplementary material.

375 6.1 Monitoring of the Barents Sea

376 Monitoring of the Barents Ecosystem has been conducted for several decades
377 and a collection of data series are reported regularly and compiled by the ICES
378 working group on the integrated assessment of the Barents Sea (WGIBAR, [19]).
379 These time series cover the period 1988-2019 and include data on:

- 380 1. landings for pelagic and demersal fish, krill, shrimps and marine mammals,
- 381 2. survey-based biomass estimates of zooplankton, pelagic and demersal fish,
- 382 3. satellite-based estimate of Net Primary Production (NPP, from 1998),
- 383 4. consumption estimates by Atlantic cod (*Gadus morhua*) of krill, shrimps,
- 384 capelin (*Mallotus villosus*), herring (*Clupea harengus*), polar cod (*Bore-*
385 *ogadus saida*) and Atlantic cod.

386 In addition, there are estimates of the minimum and maximum plausible biomasses
387 of benthos, marine mammals and birds which are used as limits for the whole
388 time-period in the model ([20]).

389 6.2 Meta Information

390 Information about the model version, authors, funding sources, data sources
391 and pre-processing, assumptions and uncertainties are provided in the model
392 RCaN-file.

Figure 2: A selection of input/output windows from the RCaN-constructor applied to the Barents Sea food-web model. Top left: RCaNconstructor graphical interface to draw and specify the trophic network. Top right: the graphical interface used to specify the model constraints. Bottom left: sample diagnostic plot showing the autocorrelation of two chains of one flux. Bottom right: sampled trajectories (time-series) of selected fluxes and components.

6.3 Food-web Structure

The model of the Barents Sea food web is composed of 11 components. Four of them are external (two plankton standing stocks from the adjacent Norwegian Sea, primary producers and Fisheries), and seven are within the model domain, and include plankton (Hzzoo: herbivorous zooplankton, Ozoo: omnivorous zooplankton), benthos (bent), fishes (PelF: pelagic fishes, DemF: demersal fishes) and top predators (MM: marine mammals, birds). These 11 components are connected via 24 biomass fluxes among which 18 correspond to trophic relationships and 6 to imports (plankton) or export (fisheries catches) of biomass.

With RCaNconstructor the food-web structure is specified by drawing the

403 components and fluxes (menu View/Network). Using this graphical interface
404 (fig. 2, top-left) it is possible interactively add, delete and edit components and
405 fluxes. The graphical positioning of the components of the network can also be
406 modified to ease the visual interpretation of the food-web structure.

407 6.4 Input parameters

408 The input parameters for individual components of the Barents Sea food web
409 (menu View/Components) are taken from [20]. These were derived from the
410 Metabolic Theory of Ecology, Life-history theory or previous modeling exer-
411 cises using for example Ecopath [21]. These are summarized in Table 1, in the
412 supplementary material.

413 6.5 Observations

414 The observation data series are derived from to the Barents Sea monitoring
415 program (section 6.1). These includes time-series of biomass for all species
416 groups within the model domain as well as information about some of the fluxes
417 (e.g. fluxes from primary producers, fluxes towards fisheries, and fluxes between
418 species).

419 6.6 Explicit constraints

420 In addition to implicit constraints (satiation, inertia, positive fluxes and refuge
421 biomass), the Barents Sea RCan model includes explicit constraints. These
422 constraints reflect additional knowledge about the ecological system and can be

used to link the model to available observations. These are as follows:

1. The consumption of primary production by zooplankton and benthos is constrained by net primary production estimates ($\pm 30\%$) for the period when these estimates are available (1998 to 2019). For other years, this sum is bounded by estimates of absolute maximum and minimum primary production that are set to 1/2 million and 2 million tons respectively.
2. Time-series of landings provide accurate information about the fluxes from herbivorous and omnivorous zooplankton, pelagic and demersal fish, and marine mammals to corresponding fisheries. Therefore, these fluxes can be set to the values of the estimated landings.
3. The annual estimates of biomass for herbivorous and omnivorous zooplankton, and pelagic and demersal fish, derived from research surveys, can be used to constrain the biomass of these groups.
4. The estimates of minimum and maximum plausible biomasses for benthos, marine mammals and birds can be used as limits for these groups for whole time-period in the model.
5. The food consumption by demersal fish can be bounded based on consumption estimates by cod in the Barents Sea. These bounds are as follows:
 - (a) the consumption of demersal fish by demersal fish is greater than the consumption of cod by cod,
 - (b) the consumption of pelagic fish by demersal fish is greater than the consumption of capelin, herring and polar cod by cod;

- 445 (c) the consumption of omnivorous zooplankton by demersal fish is greater
446 than the consumption of krill and shrimp by cod
447 (d) the total consumption by demersal fish is not greater than twice the
448 the total consumption estimates by cod.

449 6.7 Polytope building

450 Once the components, input parameters, fluxes, observations and constraints
451 are provided, the model is fully specified. This information is gathered in the
452 corresponding RCaN file. The polytope can be constructed by running R com-
453 mands of the RCaN library. This can be achieved within the RCaNconstructor
454 environment or directly within the R environment. Both ways have their advan-
455 tages and limitations. Using RcaNconstructor is intuitive and does not require
456 prior knowledge of the R language. Using R provides a greater flexibility for
457 users who are familiar with the R language. We provide below the commands
458 in the R language.

459 We build the polytope with following R command :

```
460 1 library(RCaN)
461 2 CaNMod <- buildCaN("BarentsSeaDummy.xlsx")
```

462 In the Barents Sea case, the polytope has 775 dimensions (7 initial values
463 - one for each component within the model domain - and 24 fluxes times 32
464 years). There are 2593 constraints (both implicit and explicit). Constructing
465 this polytope on a laptop computer takes about 70s (MacBook Pro Retina,
466 13-inch, Early 2015, 3.1 GHz Dual-Core Intel Core i7).

467 It is possible to check the status of the polytope (bounded, empty or un-
468 bounded) and to compute the bounds of the polytope in every of its 775 di-
469 mensions using the following commands (the time taken to obtain the results is
470 indicated in round brackets):

```
471 1 checkPolytopeStatus(CaNMod) \\ (15s)
472 2 CaNBounds <- getAllBoundsParam(CaNMod) \\ (2900s)
```

473 6.8 Polytope sampling

474 Once the polytope is built, and if it is properly shaped (i.e. not empty or
475 unbounded), it is then possible to sample it. Each sample corresponds to a
476 trajectory of the food-web. It contains the initial biomass values for each com-
477 ponent inside the model domain and the values of the fluxes at each time step.
478 That is 775 values in the example of the Barents Sea model. Sampling the
479 polytope is achieved with one RCaN command:

```
480 1 CanSample <- sampleCaN(CaNMod, N=100, thin=100, nchain=2, ncore=2,
481   method = "gibbs") \\ (520s)
```

482 This command returns an R list object which contains the original model struc-
483 ture and two chains of 100 samples (i.e. 100 food-web trajectories). In total
484 10,000 samples were explored in each chain, but only every 100th sample was
485 retained. The polytope sampling algorithm used is the Gibbs sampler.

486 6.9 Sample diagnostics

487 An example of sampling diagnostics is shown in Figure 1, bottom left. This
488 shows the autocorrelation functions for the two chains, for the flux from primary
489 production to benthos (PP_Benthos) in the year 2011. This is obtained with
490 the following command:

```
491 1 acfplot(CaNSample$mcmc[, "PP_Benthos[2011]"], lag.max=30) \\ (<1s)
```

492 6.10 Sample Graphical outputs

493 An example of time-series outputs for 3 fluxes and 3 components is provided in
494 Figure 1, bottom right. This is obtained with the following command:

```
495 1 library(ggplot2) \\  
496 2 ggSeries(CaNSample, c("H zoo_PelF", "PelF_PelF", "PelF_DemF", "H zoo", "  
497 PelF", "DemF")) \\ (<1s)
```

498 In addition to the R function ggSeries, RCaN provides a collection of pre-
499 defined plotting functions which can be used to interpret the model outputs. In
500 the Supplementary Information file, we provide examples of graphical outputs
501 from these functions.

502 References

503 [1] Benjamin Planque and Christian Mullon. Modelling chance and necessity
504 in natural systems. *ICES Journal of Marine Science*, 2019.

- 505 [2] Stephen J Hall and B Mainprize. Towards ecosystem-based fisheries man-
506 agement. *Fish and Fisheries*, 5(1):1–20, 2004.
- 507 [3] Elizabeth A Fulton, Anthony DM Smith, David C Smith, and Penelope
508 Johnson. An integrated approach is needed for ecosystem based fisheries
509 management: insights from ecosystem-level management strategy evalua-
510 tion. *PloS one*, 9(1):e84242, 2014.
- 511 [4] Sigrid Lehuta, Raphaël Girardin, Stéphanie Mahévas, Morgane Travers-
512 Trolet, and Youen Vermard. Reconciling complex system models and fish-
513 eries advice: Practical examples and leads. *Aquatic Living Resources*, 29
514 (2):208, April 2016. ISSN 0990-7440, 1765-2952. doi: 10.1051/alr/2016022.
- 515 [5] Ross M. Thompson, Ulrich Brose, Jennifer A. Dunne, Robert O. Hall, Sally
516 Hladyz, Roger L. Kitching, Neo D. Martinez, Heidi Rantala, Tamara N.
517 Romanuk, Daniel B. Stouffer, and Jason M. Tylianakis. Food webs:
518 reconciling the structure and function of biodiversity. *Trends in Ecol-
519 ogy & Evolution*, 27(12):689–697, December 2012. ISSN 0169-5347.
520 doi: 10.1016/j.tree.2012.08.005. URL [https://www.sciencedirect.com/
521 science/article/pii/S0169534712002005](https://www.sciencedirect.com/science/article/pii/S0169534712002005).
- 522 [6] Éva E. Plagányi. Models for an ecosystem approach to fisheries. Report,
523 Food and Agriculture Organisation of the United Nations, 2007.
- 524 [7] Jeffrey J. Polovina. An overview of the Ecopath model. *Fishbyte*, 2:5–7,
525 1984.

- [8] Johanna Jacomina Heymans, Marta Coll, Jason S. Link, Steven Mackinson, Jeroen Steenbeek, Carl Walters, and Villy Christensen. Best practice in Ecopath with Ecosim food-web models for ecosystem-based management. *Ecological Modelling*, 331:173–184, 2016. ISSN 0304-3800. doi: <http://dx.doi.org/10.1016/j.ecolmodel.2015.12.007>. URL <http://www.sciencedirect.com/science/article/pii/S030438001500575X>.
- [9] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL <https://www.R-project.org/>.
- [10] Martin Plummer, Nicky Best, Kate Cowles, and Karen Vines. *Coda: Output Analysis and Diagnostics for MCMC*. 2010.
- [11] Dirk Eddelbuettel and James Joseph Balamuta. Extending R with C++: A Brief Introduction to Rcpp. *PeerJ Preprints*, 5:e3188v1, aug 2017. ISSN 2167-9843. doi: 10.7287/peerj.preprints.3188v1. URL <https://doi.org/10.7287/peerj.preprints.3188v1>.
- [12] Douglas Bates and Dirk Eddelbuettel. Fast and elegant numerical linear algebra using the RcppEigen package. *Journal of Statistical Software*, 52(5):1–24, 2013. URL <http://www.jstatsoft.org/v52/i05/>.
- [13] Kjell Konis and Florian Schwendinger. *lpSolveAPI: R Interface to lpSolve*, 2020. URL <https://CRAN.R-project.org/package=lpSolveAPI>. R package version 5.5.2.0-17.7.

- 547 [14] Jialin Ma, Isuru Fernando, and Xin Chen. *SymEngine: Interface to*
548 *the SymEngine Library*, 2020. URL [https://github.com/symengine/](https://github.com/symengine/symengine.R)
549 [symengine.R](https://github.com/symengine/symengine.R). R package version 0.1.0.
- 550 [15] Robert L. Smith. Efficient Monte Carlo Procedures for Generating Points
551 Uniformly Distributed over Bounded Regions. *Operations Research*, 32(6):
552 1296–1308, December 1984. ISSN 0030-364X. doi: 10.1287/opre.32.6.1296.
- 553 [16] Hans C. Andersen and Persi Diaconis. Hit and Run as a unifying device.
554 *Journal de la Société Française de Statistique*, 148(4):5–28, 2007.
- 555 [17] Aditi Laddha and Santosh Vempala. Convergence of Gibbs Sampling: Co-
556 ordinate Hit-and-Run Mixes Fast. *arXiv:2009.11338 [cs]*, September 2020.
- 557 [18] M Hakan Satman. RCaller: A software library for calling R from Java.
558 *Journal of Advances in Mathematics and Computer Science*, pages 2188–
559 2196, 2014.
- 560 [19] ICES. Working group on the integrated assessments of the barents sea
561 (WGIBAR). *ICES Scientific Reports*, 2(30):212pp, 2020. URL <https://doi.org/10.17895/ices.pub.5998>.
562
- 563 [20] Ulf Lindstrøm, Benjamin Planque, and Sam Subbey. Multiple patterns of
564 food web dynamics revealed by a minimal non-deterministic model. *Ecosys-*
565 *tems*, 20(1):163–182, 2017.
- 566 [21] J.L. Blanchard, J.K. Pinnegar, and S. Mackinson. Exploring marine

567 mammal-fishery interactions using Ecopath with Ecosim: modelling the
568 barents sea ecosystem. Report 117, CEFAS, 2002.

569 Code and Data archiving statement

570 The code of RCaN and RCaNconstructor is available on github at <https://github.com/inrae/RCaN>.

571 The code is published under a BSD3 license.

572 Funding statement

573 Benjamin Planque received funding from the Research Council of Norway through

574 the project The Nansen Legacy (RCN #276730).

575 Acknowledgements

576 The authors would like to thank Bérengère Husson, Elliot Sivel and Aurélien

577 Favreau for their help in testing earlier version of the RCaN and RCaNconstruc-

578 tor softwares.

579 Supplementary Information

580 A html file with a full description of the RCaN model implementation for the

581 Barents Sea is provided as supplementary material. The Rmarkdown code and

582 data sources used to generate the html file are available from the GitHub reposi-

583 tory <https://github.com/inrae/RCaN> (folder article_supporting/PaperRCaN).

584 A Appendix: Syntax of RCaN constraints

585 Constraints are at the heart of the Chance and Necessity modelling. They
 586 relate fluxes and biomass on the the trophic network. The basis is a mass
 587 balance equation: the variation of biomass of a trophic compartment is the
 588 result of assimilated input fluxes plus import fluxes minus output fluxes due
 589 to predation, losses due to somatic maintenance and other export fluxes. As
 590 somatic maintenance is proportional to biomass, this constraint is a linear one.
 591 The parameters: assimilation efficiency, trophic maintenance are given for all
 592 compartments in the RCaN-file. All linear constraints on fluxes and biomass can
 593 be expressed as constraints on trophic fluxes only. See [1] for precise formulation.

594 A.1 Implicit and explicit constraints

595 There are two types of constraints.

- 596 1. Standard or implicit constraints. These are part of any RCaN model by
 597 default. They are defined by the value of the parameters in the Compo-
 598 nents sheet of the RCaN-file. These constraints express that (a) biomass
 599 must be positive and above the refuge biomass, (b) fluxes between two
 600 compartments are always positive, (c) the sum of fluxes into a compart-
 601 ment, from time t to $t+1$, cannot exceed $B(t) * \sigma$, where $B(t)$ is the
 602 biomass at time t and σ is the satiation parameter, (d) the biomass of
 603 a compartment at time $t+1$ cannot be greater than $B(t)exp(\rho)$ or lower
 604 than $B(t)exp(-\rho)$, where ρ is the inertia coefficient.

2. Explicit constraints. These are specific to individual RCaN models. They express specific knowledge about how different biomass, fluxes and observations are related, i.e. constrained by each other. These explicit constraints are formulated in the constraints sheet of the RCaN-file, and must be written according to specific syntax rules. In this section we provide details and examples of this syntax.

A.2 Symbolic expression of explicit constraints

Constraints are symbolic expressions, i.e. equalities or inequalities. They relates elements of the model (trophospecies or fluxes) on the left side of the expression to observations or other elements of the model on the right side of the expression. To interpret these expressions in a numerical framework, RCaN uses the R interface to the library Symengine, a fast symbolic manipulation library in C++, [14].

A.3 Explicit constraints, principles and examples

The rules to write constraints explicitly are as follows.

1. Constraints are written in the form of inequalities or equalities.
2. The left side of the (in)equality must contains a reference to one or several model components or fluxes.
3. The right side of the (in)equality can contain fixed values, components, fluxes, observational time-series.

In the following constraints: spA and spB are the names (in the RCaN file) of two components, $fluxA_B$ is the name of the flux between the two components, $obsA$ is the name of an observational time-series of species spA . Some examples of standard constraints:

1. $spA \leq 100$

the biomass of species A must be lower or equal to 100.

2. $spA + spB \leq 100$

the combined biomasses of species A and B must be lower or equal to 100.

3. $fluxA_B \leq 50$

the flux from species A to species B must be lower or equal to 50.

4. $spA = obsA$

the biomass of species A must equate the observational time series of species A

5. $spA \leq spB$: the biomass of species A must be lower or equal to the biomass of species B

A.4 Explicit constraints using timeless absolute bounds

This type of constraint is useful when there are no data series available to inform on the temporal variations in certain biomass or fluxes but there is some knowledge about the maximum or minimum values that a compartment or a flux may take. For example, if the total biomass of species A (spA) is expected

to lie between 100 and 1000 tonnes for the whole time series, one can write the following constraints:

$$spA \geq 100$$

$$spA \leq 1000$$

A.5 Explicit constraints from imprecise biomass observations

Constraints using biomass time series, absolute estimates When times-series of absolute biomass estimates are available (e.g. from stock assessments) these can be used to constraint the corresponding modelled biomass. Consider a compartment inside the model (spA); a series of absolute biomass estimate with name $obsA$.

1. If we assume the observation to precisely reflect the true biomass, we can write the model constraint:

$$spA = obsA$$

2. If we assume that the observed biomass is uncertain by 10% we can write the 2 model constraints:

$$spA \geq obsA/1.1$$

$$spA \leq obsA * 1.1$$

3. If we assume that the observed biomass only represent a fraction of the population and the the true biomass lies somewhere between what is es-

666 timated and twice this amount, we can write:

$$667 \quad spA \geq obsA$$

$$668 \quad spA \leq obsA * 2$$

669 A.6 Explicit constraints on Fisheries catches

670 The specific case of fisheries catches can be handled in the following way. Con-
 671 sider a species A (spA) ; a fishery on species A (compartment outside the model)
 672 with name FA . Catches are a flux from the species to the fishery (non-trophic
 673 flux): spA_FA . There is a time series of reported catches of species A by the
 674 fishery (data series); its name is $CatchA$. All are expressed in the same units
 675 (e.g. tonnes).

676 1. If we assume that the reported catches reflect the true catches exactly, we
 677 can write the model constraint:

$$678 \quad spA_FA = CatchA$$

679 2. If we assume that the reported catches are uncertain by 10% we can write
 680 the model constraints:

$$681 \quad spA_FA \geq CatchA/1.1$$

$$682 \quad spA_FA \leq CatchA * 1.1$$

683 3. If we assume that the catches are under-reported and that the true catches
 684 are somewhere between what is reported and twice this amount, we can
 685 write:

$$686 \quad spA_FA \geq CatchA$$

$$spA_FA \leq CatchA * 2$$

A.7 Explicit constraints from relative estimates

When times-series of relative biomass estimates are available (e.g. from surveys) these can be used to constraint the corresponding modelled biomass. Consider compartment A (spA); a series of relative biomass estimate (data series) with name $surveyA$.

1. If we assume the observation to precisely reflect the relative variations in biomass over time, we can write the model constraint:

$$spA / \text{mean}(spA[f : l]) = obsA / \text{mean}(obsA[f : l]) \text{ where } f \text{ and } l \text{ are the first and the last indices of the time period of reference.}$$

2. If we assume that the observed relative changes in biomass are uncertain by 10% we can write the 2 model constraints:

$$(spA / \text{mean}(spA[f : l])) \geq (obsA / \text{mean}(obsA[f : l])) / 1.1$$

$$(spA / \text{mean}(spA[f : l])) \leq (obsA / \text{mean}(obsA[f : l])) * 1.1$$

A.8 Explicit structural constraints

In addition to constraining compartments/fluxes based on data or absolute bounds, it is also possible to express structural constraints within the model that are independent of observational time series or absolute bounds. For example, if species C (spC) can feed on species A and B (spA and spB), but we know that species A is always more abundant in the diet of species C , we can write.

708 $spA_spC \geq spB_spC$,

709 which expresses that the flux from spA to spC is always greater than the flux
710 from spB to spC .

711 A.9 Applying constraints over limited time periods

712 A time period is associated with each constraint. By default this is the period
713 from the first year of available observation to the last. However, constraints can
714 be applied to restricted time periods or specific years (or even a single year)
715 when necessary. The selected years are indicated in the third column of the
716 constraint table in the RCaN-constructor / RCaN-file.