The RGadget environment

A tidyverse inspired approach to model development work flow

Bjarki Þór Elvarsson, Jamie Lentin and Pamela J. Woods

HAFRANNSÓKNASTOFNUN

Shuttle Thread

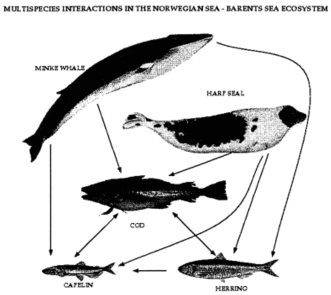
The outline of the talk

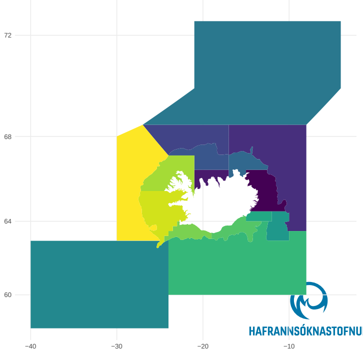
- Gadget
    - Origins and technical description
- Gadget's user interface
    - Flexible model structure encaptured with setting files
- The RGadget environment:
    - Rgadget and MFDB
- Future work and concluding remarks

HAFRANNSÓKNASTOFNUN

The origins of Gadget

- Gadget is an open source age and length–based, multi–species, multi-fleet, multi–area assessment modelling framework
- Gadget traces it origins to parallel modelling efforts in the North Atlantic in the early 90's
- Originally known as BORMICON, (**BOR**eal **MI**gration and **CON**sumption model) and was developed a part of a multi–species research project in the mid–1990s at the Marine Research Institute in Iceland.
- BORMICON was a conceptual descendant of MULTSPEC, a Norwegian multi-species model for the Barents sea ecosystem.
- The BORMICON code base was branched off to Norway where a single-area, single–species variant of BORMICON, FLEKSIBEST (**FLEKSI**bel **BEST**andsvurdering), was developed.
- In the early 2000s Gadget, the **G**lobally applicable **A**rea **D**isa**G**gregated **E**cosytem **T**oolbox, was created from the merger of the two forks


MULTISPECIES INTERACTIONS IN THE NORWEGIAN SEA - BARENTS SEA ECOSYSTEM
MINKE WHALE
HARP SEAL
COD
CAPELIN
HERRING



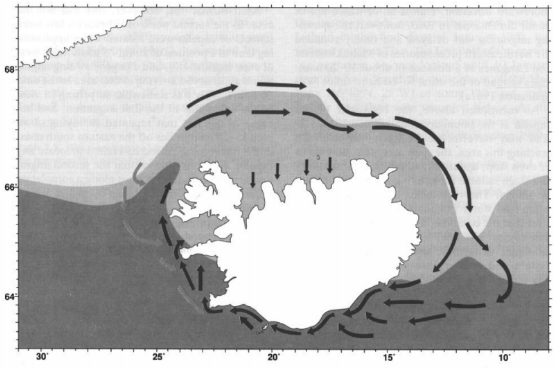
HAFRANNSÓKNASTOFNUN

Motivation for the Framework

The original design goals of BORMICON were to (flexibly) model:

- Migrations
- Consumption of predators and fleets
- Natural mortality and growth in size
- Maturity and spawning
- External recruitment into the stock(s)
- Flexibly compare with minimaly processed data, where and when available

Where the motivating example was the cod–capelin interaction in Icelandic waters



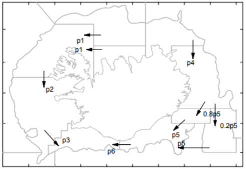


Figure 8.4.2: Migration of mature cod in January-April.

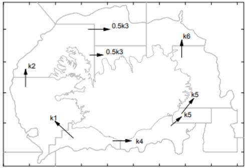


Figure 8.4.3: Migration of mature cod in June-December.

s

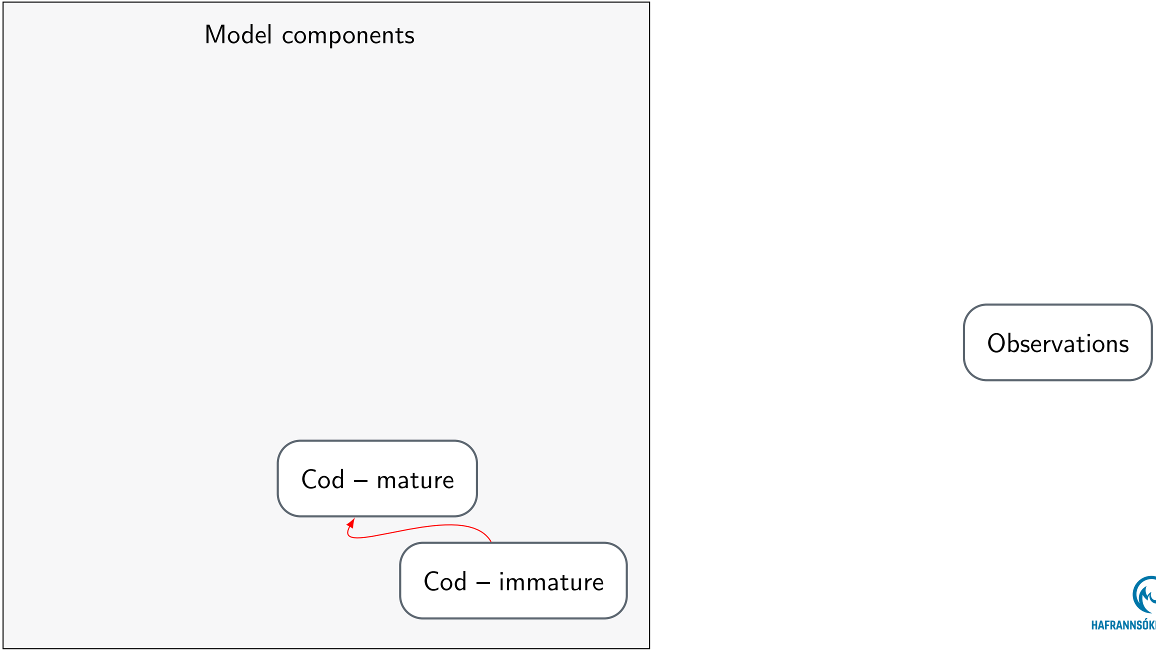
HAFRANNSÓKNASTOFNUN

Gadget – Technical description

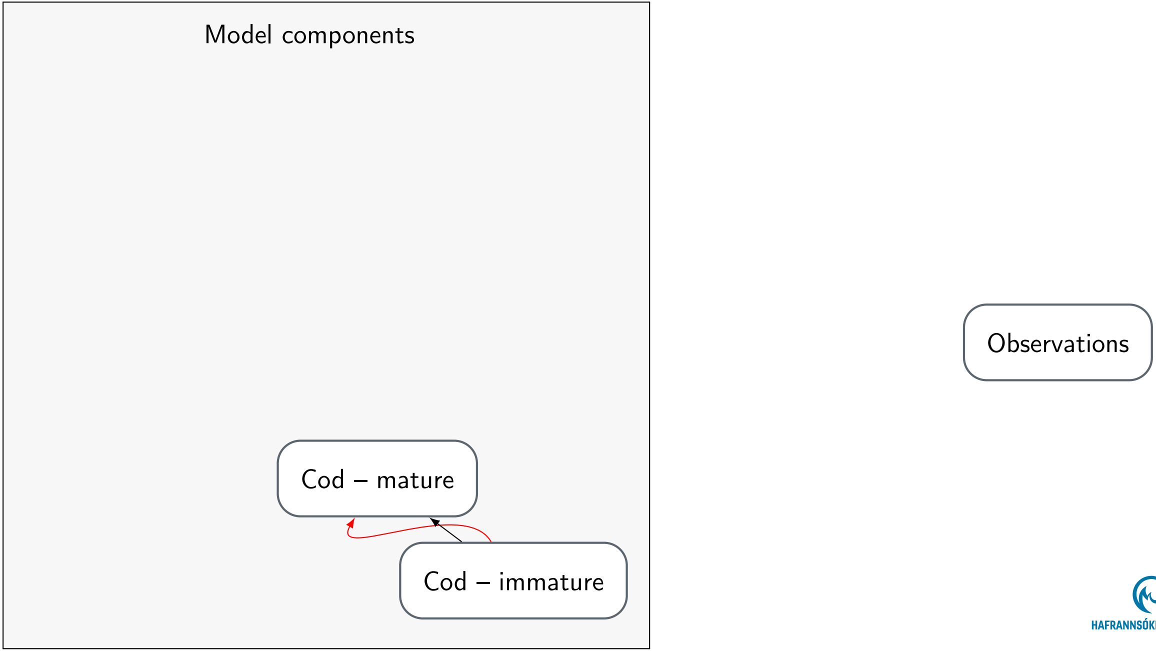
Gadget is designed as a forward simulator:

- The fundamental block in Gadget is the **substock**, an entity that is homogeneous with respect to various processes
    - For each substock the number of fish at a certain length and age is stored
    - Interaction between stock can be defined in numerous ways, such as spawning, maturation and predation.
- The substocks live in an **area**, or areas, where they optionally migrate to and from
    - Areas have various properties such as size and temperature
- Harvesting of the substocks by area and time is defined through **fleets** that fish according to either effort or total catch and use length based selection functions by stock
- The state of the ecosystem that is simulated in Gadget is observed through the substock abundance, consumption and harvest at each **timestep**
    - Length of timesteps is arbitrary and can be uneven within the year

Example model diagram



Example model diagram



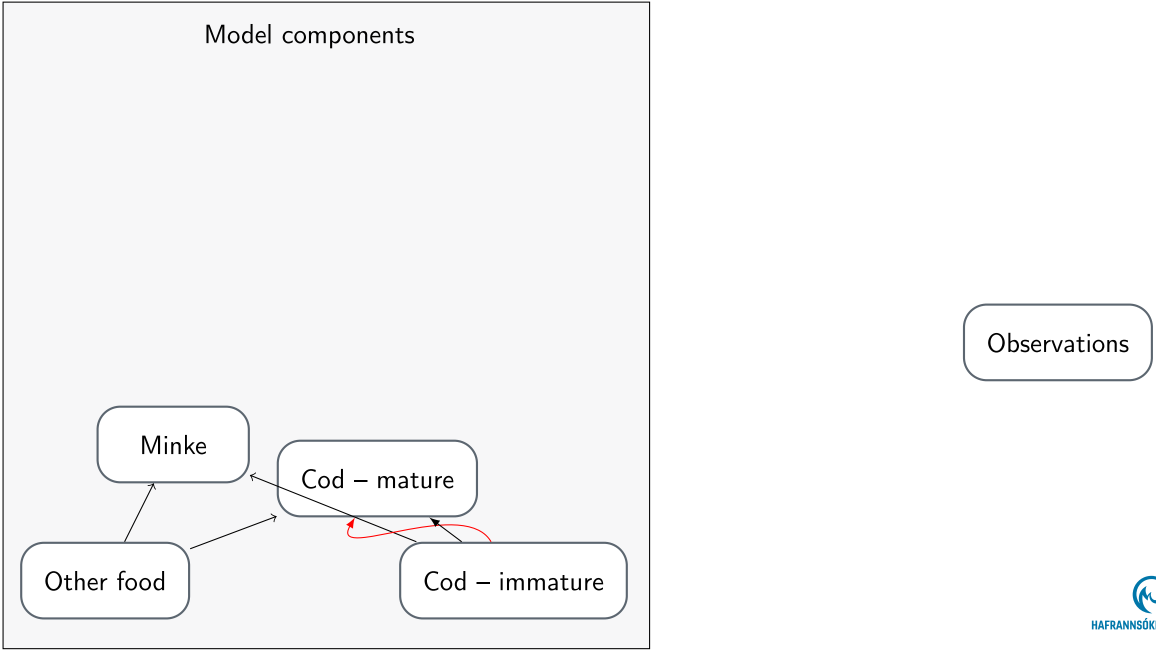
Model components

Cod – mature

Cod – immature

Observations

Example model diagram



Model components

Minke

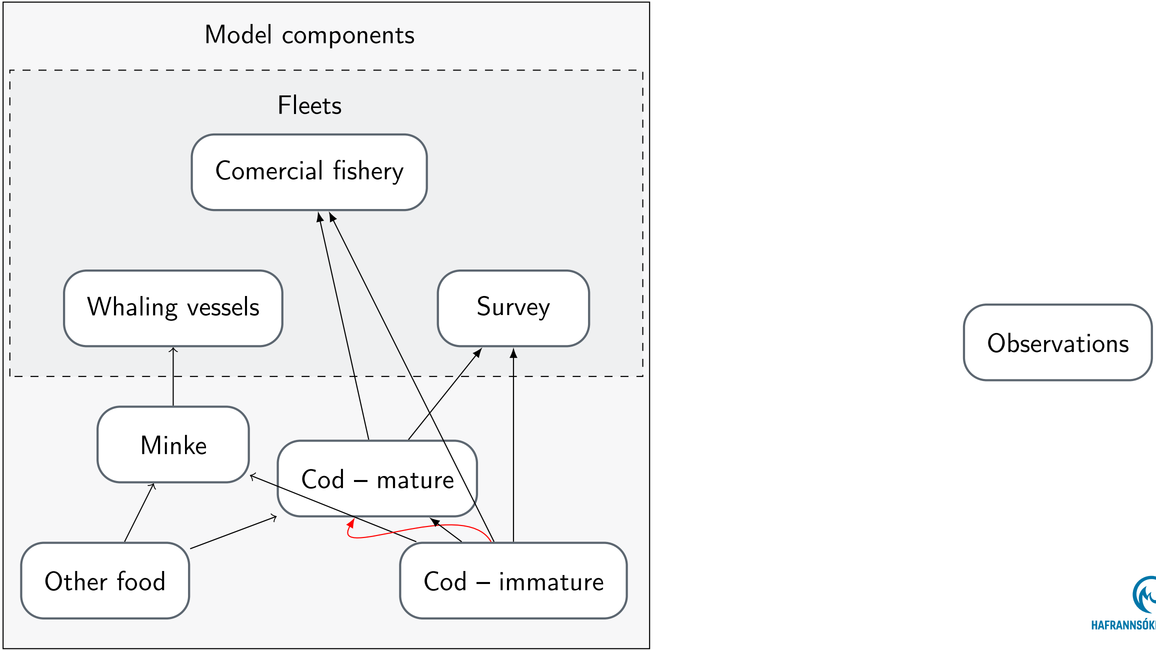
Cod – mature

Other food

Cod – immature

Observations

Example model diagram


Model components

Fleets

Comercial fishery

Whaling vessels

Survey

Observations

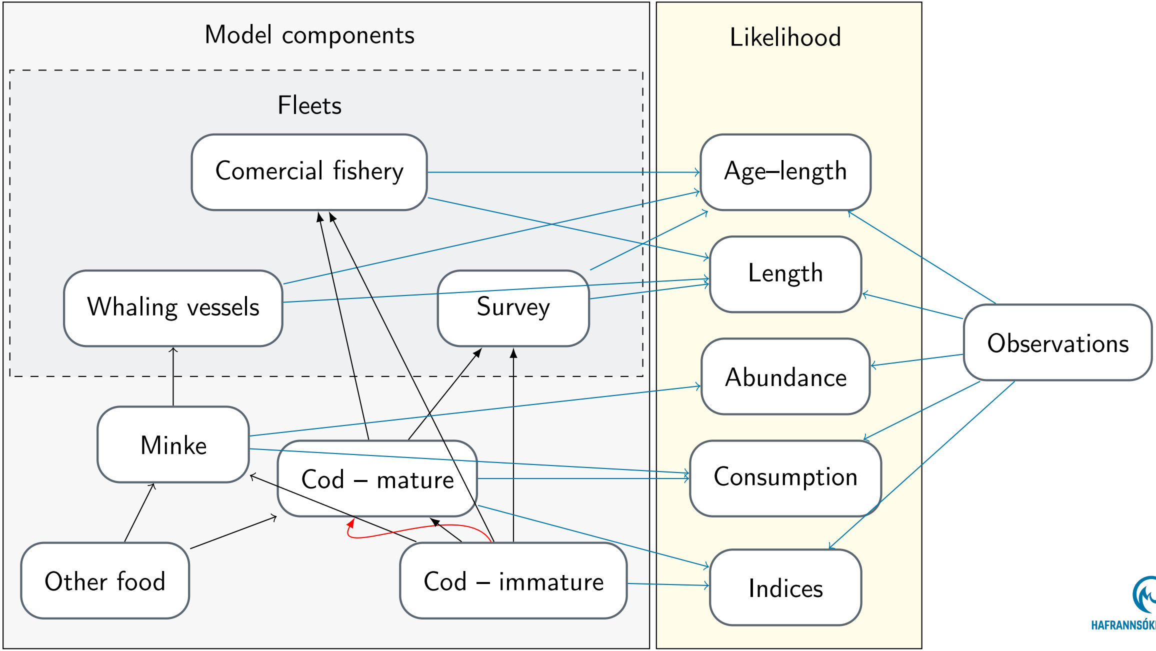
Minke

Cod – mature

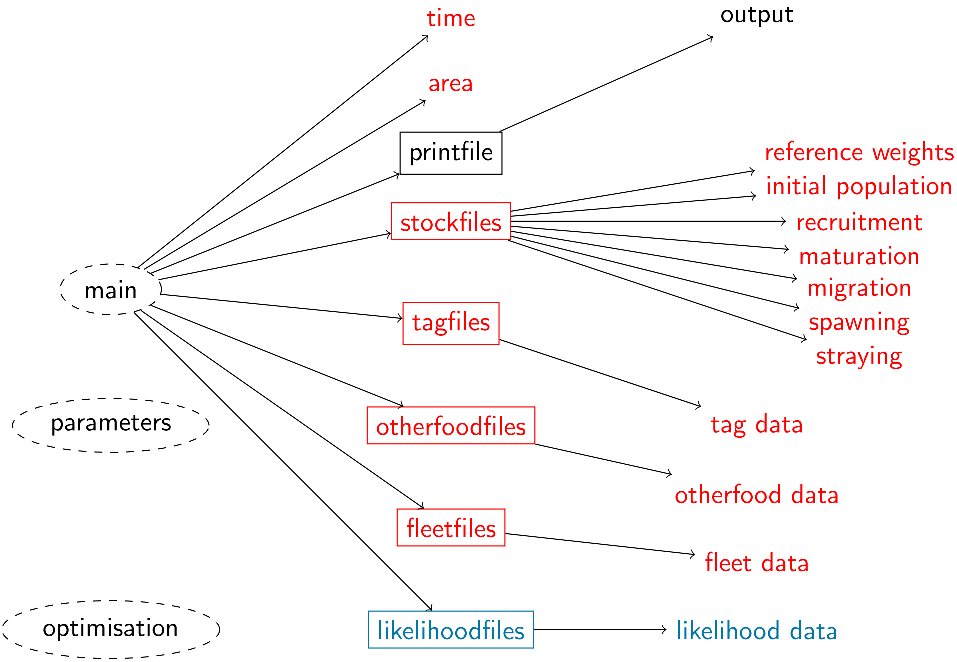
Other food

Cod – immature

Example model diagram



User interface



The stock file

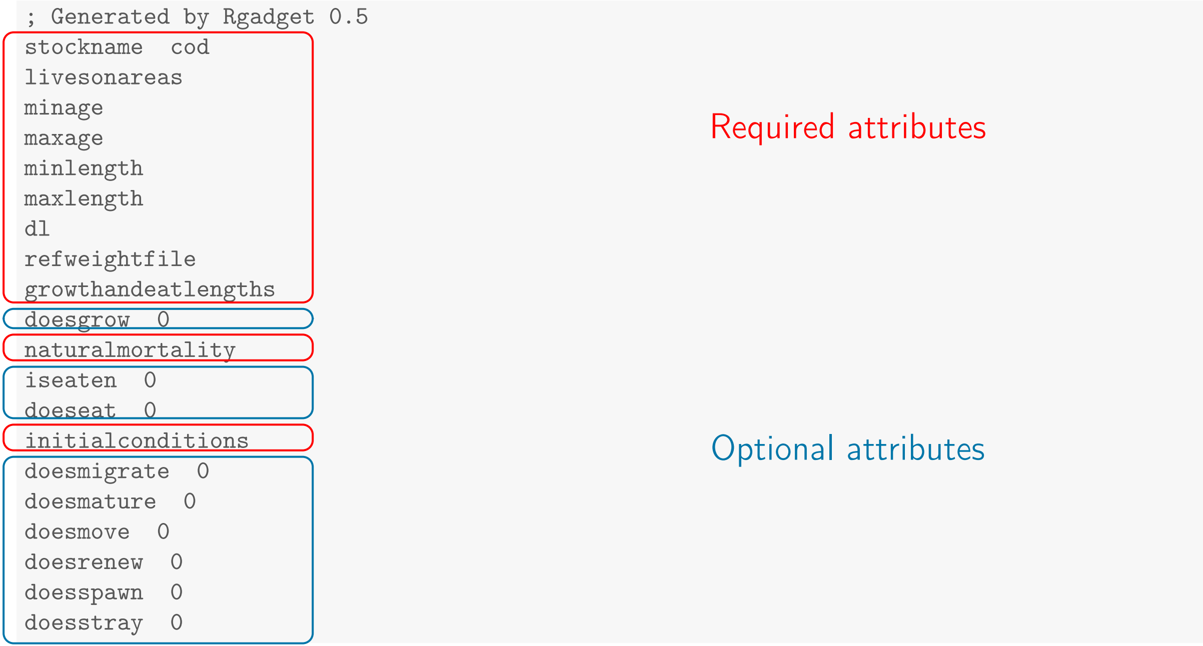
```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  0
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

The stock file

```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  0
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

Required attributes

The stock file



```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  0
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

Required attributes

Optional attributes

The stock file

```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  1
growthfunction  lengthvbsimple
growthparameters  #cod.Linf  (* 0.001 #cod.k)  #cod.walpha  #cod.wbeta
beta  (* 10 #cod.bbin)
maxlengthgroupgrowth  15
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

The stock file

```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  1
growthfunction  lengthvbsimple
growthparameters  #cod.Linf  (* 0.001 #cod.k)  #cod.walpha  #cod.wbeta
beta  (* 10 #cod.bbin)
maxlengthgroupgrowth  15
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

Optional attribute: Growth

The stock file

```
; Generated by Rgadget 0.5
stockname  cod
livesonareas
minage
maxage
minlength
maxlength
dl
refweightfile
growthandeatlengths
doesgrow  1
growthfunction  lengthvbsimple
growthparameters  #cod.Linf  (* 0.001 #cod.k)  #cod.walpha  #cod.wbeta
beta  (* 10 #cod.bbin)
maxlengthgroupgrowth  15
naturalmortality
iseaten  0
doeseat  0
initialconditions
doesmigrate  0
doesmature  0
doesmove  0
doesrenew  0
doesspawn  0
doesstray  0
```

Optional attribute: Growth

Gadget variables:

- #name indicates a model variable in Gadget
- Can be used instead of numbers
- Functions of model variables are entered as equations using reverse Polish notation
- Can be defined to be variable with time
- Values are controlled via the parameter file

Parameter values

```
                switch        value lower upper optimise
1             cod.Linf 1.633348e+02 1e+02   200        1
2                cod.k 9.616510e+01 4e+01   100        1
3           cod.walpha 2.275674e-06 1e-10     1        0
4            cod.wbeta 3.202005e+00 2e+00     4        0
5             cod.bbin 2.669141e+01 1e-08   100        1
6             codimm.M 1.500000e-01 1e-03     1        0
7   codimm.init.scalar 1.661737e+00 1e+00   300        1
8           cod.init.F 2.193912e-01 1e-01     1        1
9        codimm.init.3 3.663376e+02 1e-03  1000        1
10       codimm.init.4 3.772913e+02 1e-03  1000        1
11       codimm.init.5 2.820772e-01 1e-03  1000        1
12       codimm.init.6 1.000000e-03 1e-03  1000        1
13       codimm.init.7 1.000003e-03 1e-03  1000        1
14       codimm.init.8 1.000009e-03 1e-03  1000        1
15       codimm.init.9 1.000028e-03 1e-03  1000        1
```

The likelihood file

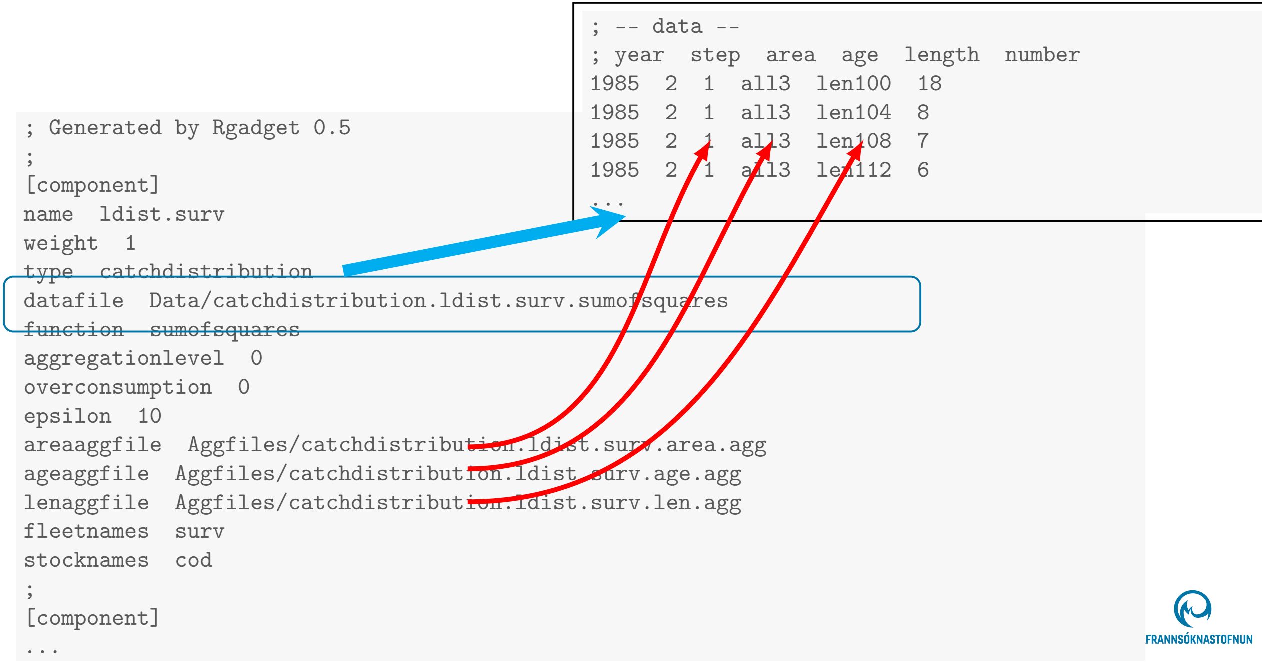
```
; Generated by Rgadget 0.5
;
[component]
name  ldist.surv
weight  1
type  catchdistribution
datafile  Data/catchdistribution.ldist.surv.sumofsquares
function  sumofsquares
aggregationlevel  0
overconsumption  0
epsilon  10
areaaggfile  Aggfiles/catchdistribution.ldist.surv.area.agg
ageaggfile  Aggfiles/catchdistribution.ldist.surv.age.agg
lenaggfile  Aggfiles/catchdistribution.ldist.surv.len.agg
fleetnames  surv
stocknames  cod
;
[component]
...
```

The likelihood file

```
; Generated by Rgadget 0.5
;
[component]
name  ldist.surv
weight  1
type  catchdistribution
datafile  Data/catchdistribution.ldist.surv.sumofsquares
function  sumofsquares
aggregationlevel  0
overconsumption  0
epsilon  10
areaaggfile  Aggfiles/catchdistribution.ldist.surv.area.agg
ageaggfile  Aggfiles/catchdistribution.ldist.surv.age.agg
lenaggfile  Aggfiles/catchdistribution.ldist.surv.len.agg
fleetnames  surv
stocknames  cod
;
[component]
...
```

```
; -- data --
; year  step  area  age  length  number
1985  2  1  all3  len100  18
1985  2  1  all3  len104  8
1985  2  1  all3  len108  7
1985  2  1  all3  len112  6
...
```

The likelihood file



```
; -- data --
; year  step  area  age  length  number
1985  2  1  all3  len100  18
1985  2  1  all3  len104  8
1985  2  1  all3  len108  7
1985  2  1  all3  len112  6
...
```

```
; Generated by Rgadget 0.5
;
[component]
name  ldist.surv
weight  1
type  catchdistribution
datafile  Data/catchdistribution.ldist.surv.sumofsquares
function  sumofsquares
aggregationlevel  0
overconsumption  0
epsilon  10
areaaggfile  Aggfiles/catchdistribution.ldist.surv.area.agg
ageaggfile  Aggfiles/catchdistribution.ldist.surv.age.agg
lenaggfile  Aggfiles/catchdistribution.ldist.surv.len.agg
fleetnames  surv
stocknames  cod
;
[component]
...
```

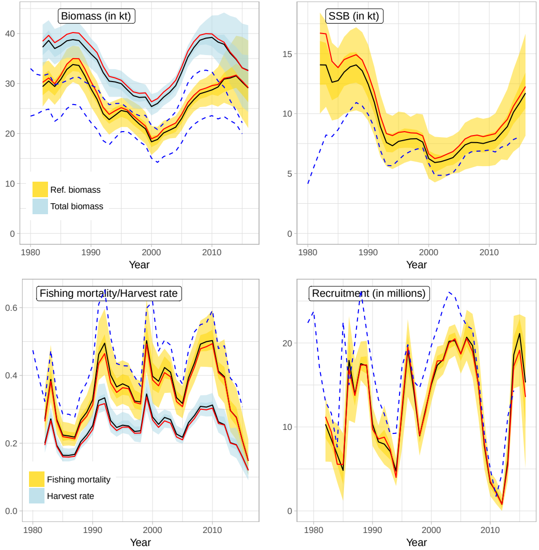
The likelihood file

```
len4   4   8
len8   8   12
len12   12   16
len16   16   20
len20   20   24
...
```

```
; Generated by Rgadget 0.5
;
[component]
name  ldist.surv
weight  1
type  catchdistribution
datafile  Data/catchdistribution.ldist.surv.sumofsquares
function  sumofsquares
aggregationlevel  0
overconsumption  0
epsilon  10
areaaggfile  Aggfiles/catchdistribution.ldist.surv.area.agg
ageaggfile  Aggfiles/catchdistribution.ldist.surv.age.agg
lenaggfile  Aggfiles/catchdistribution.ldist.surv.len.agg
fleetnames  surv
stocknames  cod
;
[component]
...
```

The problem

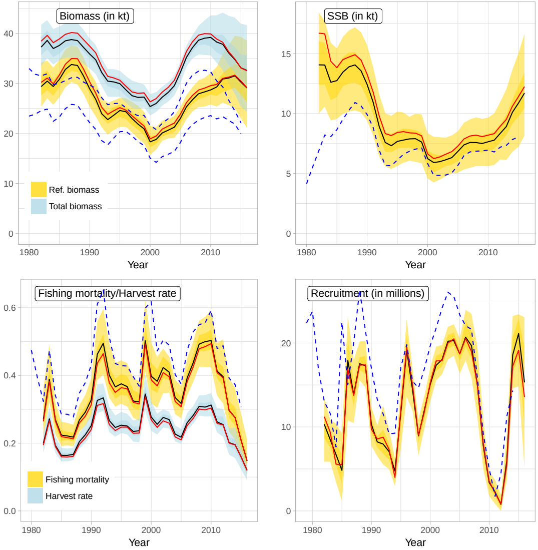
- Building the model by "hand"requires a lot of manual labour
    - People often resort to copy/pasting files between models
    - Model errors are not easy to track as the input files may be valid
- Disparate data types and sources
    - Aggregation files can be difficult to maintain
    - Number of likelihood components typically exceeds 10
- Reproducability!!



HAFRANNSÓKNASTOFNUN

The problem

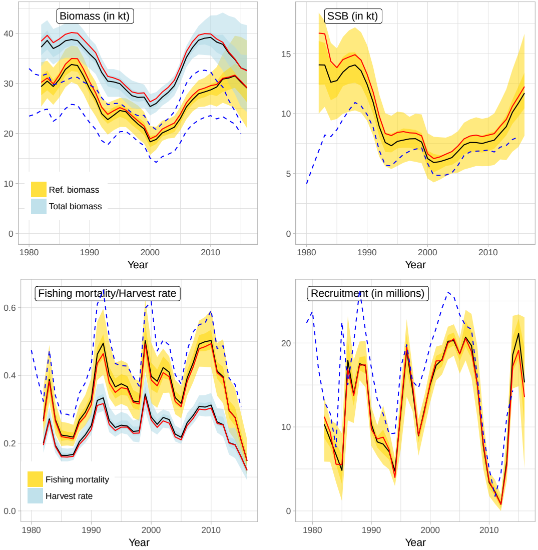
- Building the model by "hand"requires a lot of manual labour
    - People often resort to copy/pasting files between models
    - Model errors are not easy to track as the input files may be valid
    - **Solution**: generate model files using scripts
- Disparate data types and sources
    - Aggregation files can be difficult to maintain
    - Number of likelihood components typically exceeds 10
- Reproducability!!



HAFRANNSÓKNASTOFNUN

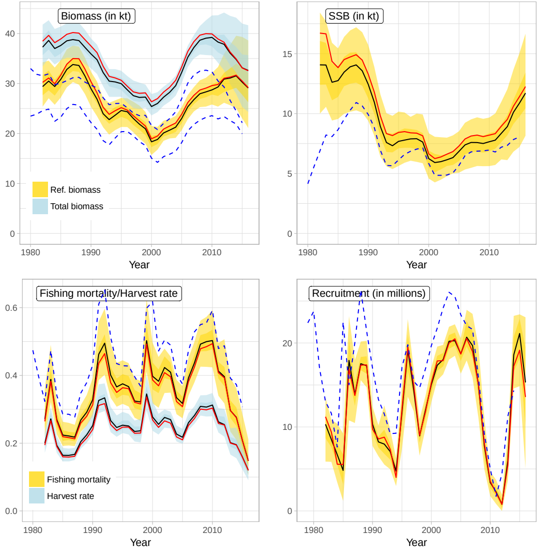
The problem

- Building the model by "hand"requires a lot of manual labour
  - People often resort to copy/pasting files between models
  - Model errors are not easy to track as the input files may be valid
  - **Solution**: generate model files using scripts
- Disparate data types and sources
  - Aggregation files can be difficult to maintain
  - Number of likelihood components typically exceeds 10
  - **Solution**: Use a database system
- Reproducability!!

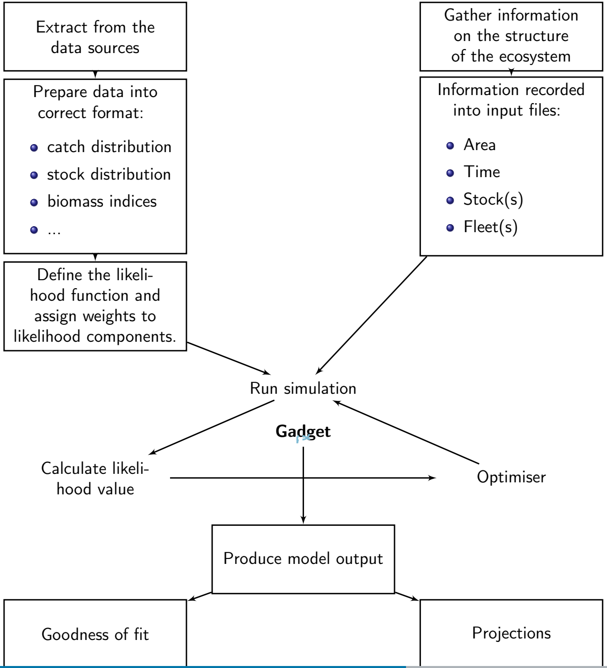


The problem

- Building the model by "hand"requires a lot of manual labour
    - People often resort to copy/pasting files between models
    - Model errors are not easy to track as the input files may be valid
    - **Solution**: generate model files using scripts
- Disparate data types and sources
    - Aggregation files can be difficult to maintain
    - Number of likelihood components typically exceeds 10
    - **Solution**: Use a database system
- Reproducability!!
    - **Solution**: store the model generating scripts in a version control system



The RGadget environment



Extract from the data sources

Prepare data into correct format:

- catch distribution
- stock distribution
- biomass indices
- ...

Define the likelihood function and assign weights to likelihood components.

Gather information on the structure of the ecosystem

Information recorded into input files:

- Area
- Time
- Stock(s)
- Fleet(s)

Run simulation

Gadget

Calculate likelihood value

Optimiser

Produce model output

Goodness of fit

Projections

The general objectives of Rgadget are to capture this entire process via R - scripts from data to TAC. This includes:

- Data formating
- Definition of relevant processes, both biological and fishing
- Fitting a model to data
- Producing diagnostics and biomass estimates
- Setup projections

Essentially we want to be able to redo the entire process either when new data are availble or significant changes have been made to the data

File manipulation

- Rgadget file manipulation routines rely on the modular Gadget file structure
  - This allows for updates of only parts of file, leaving others in place
- Files are organised into components and sub-components
- Files are linked via model directory structure
- Additional lookup files are created behind the scenes where appropriate

Fleet file:

```
[component]
linearfleet  simple_fleet
livesonareas  1
multiplicative  1
suitability
simple_stock  function  constant  0.2
amount  Data/fleet.simple_fleet.data
;
[component]
...
```

Stock file:

```
...
growthandeatlengths
doesgrow  1
growthfunction  lengthvbsimple
growthparameters  #cod.Linf  (* 0.001 #cod.k)  #cod
beta  (* 10 #cod.bbin)
maxlengthgroupgrowth  15
naturalmortality
```

Rgadget - setting up the stock file

```
library(Rgadget)

gd <- gadget.variant.dir('cod_model')

cod_stock <-
  gadgetstock('cod',gd,missingOkay = TRUE) %>%
  gadget_update('stock',
                livesonareas = 1,
                maxage = 1,
                minage = 1,
                minlength = 0,
                maxlength = 2,
                dl = 1) %>%
  gadget_update('doesgrow',0) %>%
  gadget_update('naturalmortality',0) %>%
  gadget_update('refweight',data=tibble(length=0:2,mean=0:2)) %>%
  gadget_update('initialconditions',
                normalparam = tibble(age = 1,
                                     area = 1,
                                     age.factor = 1,
                                     area.factor =1,
                                     mean = 1,
                                     stddev = .1,
                                     alpha = 1,
                                     beta = 1))
cod_stock %>%
  write.gadget.file(gd)
```

Rgadget - setting up the stock file

```
library(Rgadget)

gd <- gadget.variant.dir('cod_model')

cod_stock <-
  gadgetstock('cod',gd,missingOkay = TRUE) %>%
  gadget_update('stock',
                livesonareas = 1,
                maxage = 1,
                minage = 1,
                minlength = 0,
                maxlength = 2,
                dl = 1) %>%
  gadget_update('doesgrow',0) %>%
  gadget_update('naturalmortality',0) %>%
  gadget_update('refweight',data=tibble(length=0:2,mean=0:2)) %>%
  gadget_update('initialconditions',
                normalparam = tibble(age = 1,
                                     area = 1,
                                     age.factor = 1,
                                     area.factor =1,
                                     mean = 1,
                                     stddev = .1,
                                     alpha = 1,
                                     beta = 1))

cod_stock %>%
  write.gadget.file(gd)
```

Location of the model

Rgadget - setting up the stock file

```
library(Rgadget)

gd <- gadget.variant.dir('cod_model')

cod_stock <-
  gadgetstock('cod',gd,missingOkay = TRUE) %>%
  gadget_update('stock',
                livesonareas = 1,
                maxage = 1,
                minage = 1,
                minlength = 0,
                maxlength = 2,
                dl = 1) %>%
  gadget_update('doesgrow',0) %>%
  gadget_update('naturalmortality',0) %>%
  gadget_update('refweight',data=tibble(length=0:2,mean=0:2)) %>%
  gadget_update('initialconditions',
                normalparam = tibble(age = 1,
                                     area = 1,
                                     age.factor = 1,
                                     area.factor =1,
                                     mean = 1,
                                     stddev = .1,
                                     alpha = 1,
                                     beta = 1))

cod_stock %>%
  write.gadget.file(gd)
```

Location of the model

Location of the stock file

Rgadget - setting up the stock file

```r
library(Rgadget)

gd <- gadget.variant.dir('cod_model')

cod_stock <-
  gadgetstock('cod',gd,missingOkay = TRUE) %>%
  gadget_update('stock',
                livesonareas = 1,
                maxage = 1,
                minage = 1,
                minlength = 0,
                maxlength = 2,
                dl = 1) %>%
  gadget_update('doesgrow',0) %>%
  gadget_update('naturalmortality',0) %>%
  gadget_update('refweight',data=tibble(length=0:2,mean=0:2)) %>%
  gadget_update('initialconditions',
                normalparam = tibble(age = 1,
                                     area = 1,
                                     age.factor = 1,
                                     area.factor =1,
                                     mean = 1,
                                     stddev = .1,
                                     alpha = 1,
                                     beta = 1))

cod_stock %>%
  write.gadget.file(gd)
```

Location of the model

Location of the stock file

Updates the stock file, calls to gadget_update can be chained together

Example uses

Create related stocks:

```
cod_north <-
  cod_stock %>%
  gadget_update('stock',
                name = 'cod_north',
                maxage = 2,
                livesonarea = 2)
cod_north %>%
  write.gadget.file(gd)
```

Define model variants

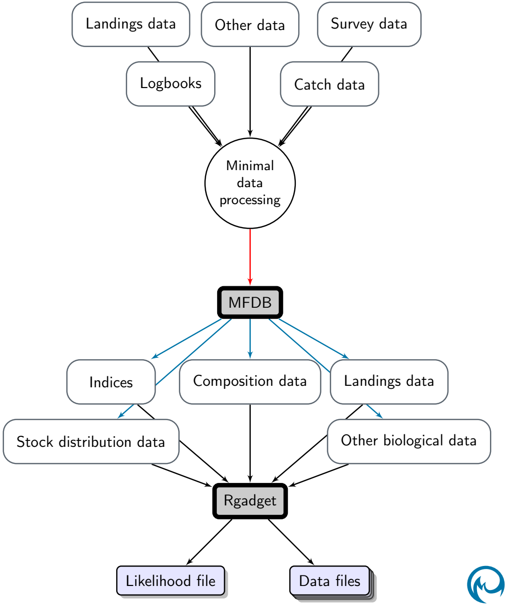
```
gd.variant <-
  gadget.variant.dir('cod_model',
                     variant_dir = 'variant_1')

## only writes files that are updated
gadgetstock('cod',gd) %>%
  gadget_update('naturalmortality',0.2) %>%
  write.gadget.file(gd.variant)
```

- Other files can similarly be edited by gadget_update
    - Including Fleet and Likelihood files
- Other files, which are simpler, such as the area and time files can be created using the gadgetfile function
- As Gadget is a forward projection model setting up projections into the future is done in the same way as for "regular"models

MFDB

The MareFrame Database (MFDB) is a R-package that has

- Automatic set-up and configuration of a PostgreSQL database on the users machine
- Functions to ingest data automatically from files or other database APIs
- Functions to transform and aggregate the data
- Functions to create input files for ecosystem modelling tools.



Preparing composition data

```
library(mfdb)
mdb <- mfdb('Iceland')

aggdata <-
  mfdb_sample_count(mdb, c('age', 'length'), list(
    age = mfdb_unaggregated(),
    length = mfdb_step_interval("len", by = 20, to = 100),
    year = 1981:2019,
    step = mfdb::mfdb_timestep_quarterly,
    area = mfdb_group("1" = unique(reitmapping$SUBDIVISION)),
    data_source = 'survey',
    species = 'COD')
```

```
gadgetlikelihood('likelihood',gd$dir,missingOkay = TRUE) %>%
  gadget_update("catchdistribution",
                name = "ldist.igfs",
                weight = 1,
                data = aggdata,
                fleetnames = c("igfs"),
                stocknames = 'cod')
```

HAFRANNSÓKNASTOFNUN

Preparing composition data

```r
library(mfdb)
mdb <- mfdb('Iceland')

aggdata <-
  mfdb_sample_count(mdb, c('age', 'length'), list(
    age = mfdb_unaggregated(),
    length = mfdb_step_interval("len", by = 20, to = 100),
    year = 1981:2019,
    step = mfdb::mfdb_timestep_quarterly,
    area = mfdb_group("1" = unique(reitmapping$SUBDIVISION)),
    data_source = 'survey',
    species = 'COD')
```

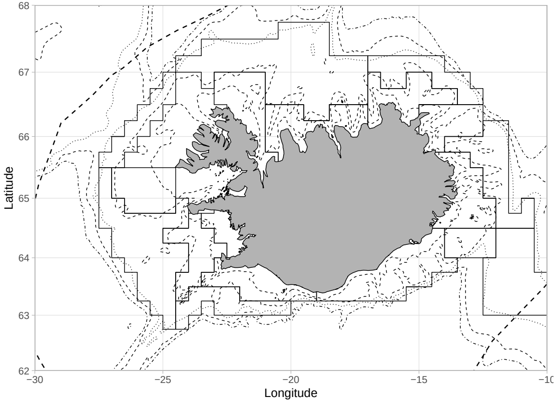
- Retains aggregation information from MFDB
- writes appropriate lookup information into Agg-files

```r
gadgetlikelihood('likelihood',gd$dir,missingOkay = TRUE) %>%
  gadget_update("catchdistribution",
                name = "ldist.igfs",
                weight = 1,
                data = aggdata,
                fleetnames = c("igfs"),
                stocknames = 'cod')
```

HAFRANNSÓKNASTOFNUN

Uncertainty estimates

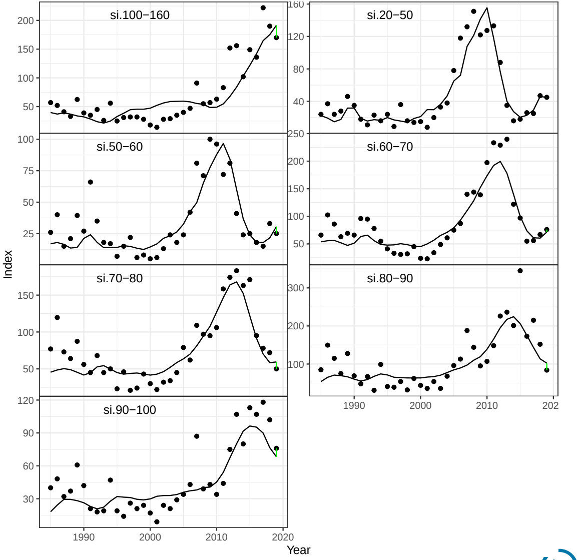
- MFDB implements a spatial bootstrap to estimated uncertainty
    - Samples recorded into MFDB are divided into spatial **divisions**
    - The divisions are treated as sampling units, that is all the samples within the division are treated as a unequal sized vector
    - New data replicates are created using resampling of the divisions, with replacement
- Model agnostic and allows comparisons between models and model platforms
- Rgadget code requires minimal modification to run a bootstrap



Additional helpers

Apart from defining the model files Rgadget has other useful tools:

- Likelihood weights can be assigned using an iterative reweighting heuristic via `gadget.iterative`
- Model diagnostics and population estimates can be collected from a model using `gadget.fit`
- Simple model projections and yield per recruit are provided via `gadget.forward` and `gadget.ypr` respectively



Next steps:

- Having the whole procedure scripted has increased the speed of model development
    - Proper processing of data is key
- We aim for tighter integration between Rgadget and Gadget:
    - Remove legacy file I/O code and pass objects directly from R
    - Text based file inputs will still be used, but read via R instead
- Automatic differention and random effects will hopefully happen
    - Just received funding from the Icelandic infrastructure fund
- More focus on the ability to define full feedback MSEs (Project on-going at the IMR in Bergen)

HAFRANNSÓKNASTOFNUN

Useful links

- hafro.github.io/gadget
- github.com/fishvice/gadget-models
- github.com/mareframe/mfdb
- Gadget course availble on line at heima.hafogvatn.is/∼bthe/gadget_course.html

Acknowledgements

MareFrame

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 613571

BECAUSE EU project "Critical Interactions Between Species and their Implications for a Precautionary Fisheries Management in a Variable Environment - a Modelling Approach"502482

HAFRANNSÓKNASTOFNUN

HAVFORSKNINGS INSTITUTTET

DST^2 EU project "Development of structurally detailed statistically testable models of marine populations"QLK5-CT1999-01609

CAPAM