rtmbGMACS design considerations

# Flow

1. Read configuration file
   1. Define model dimensions
      1. years, seasons
      2. population characteristics (region, sex, maturity state, post-recruitment age, post-molt age, size
      3. fleets
   2. identify data, control files
2. read data file(s)
3. read control files(s) to specify functions and associated parameters (parameter types [fixed, RE] initial values, bounds, priors, estimation phases) for
   1. allometry
   2. initial abundance
   3. recruitment
   4. natural mortality
   5. molting
   6. growth
   7. maturation
   8. selectivity/catchability
   9. fishing mortality
   10. survey catchability
   11. aging
   12. movement
4. create "inputs" list with
   1. dimensions info,
   2. data

# Dimensions

Model population indices are:

* y: year
* s: season
* r: region (spatial area)
* x: sex
* m: maturity state
* a: post-recruitment age
* p: post-molt age
* z: size

Fleets (fisheries, surveys) are indexed by "f".

# Functions

* multiple advectors can be returned from a function in a list
  + see: testing/RTMB\_Capabilities/Lists\_With\_AD\_Elements\_Functionality

# Parameters

## Mirroring and fixing

Mirroring or fixing parameters within a parameter vector ("p", say)is done by creating a list ("map", say)

with an element with the same name as the parameter which is a factor vector with the same

number of elements as in the parameter vector and levels from 1:length(p). Thus

map=list(p=factor(1:length(p)))

This is included when MakeADFun'ing the model by specifying

model = MakeADFun(objfun,parameters,map=map,...),

where `objfun` is the objective function, `parameters` is the parameters list (with `p` as an element),

and `map` is the list above. Only elements of parameters specified in `map` can be mirrored or fixed.

To "mirror" parameter values within "p", simply change the level for each of the mirrored elements

to the same level. Thus,

map=list(p=factor(1,2,1,2))

will mirror the 1st and 3rd elements as one parameter and the 2nd and 4th elements as another parameter,

essentially reducing the parameter vector "p" to a two-element vector, which is what is actually returned

in the model parameter list.

To fix an element's value to its initial value, simply assign "NA" to the level of the element when

creating the factor in the map. This needs to be done when creating the factor, not after creation--

see https://mail.google.com/mail/u/0/?shva=1#inbox/KtbxLwghhxtwvVMtJccgJzWTvkSnQhmQzg

## Bounds

Bounds are placed on a parameter when optimizing not model, not when MakeADFun'ing it.

Using the optimizer `nlminb`, the lower and upper bounds are specified as values of

the `lower` and `upper` function arguments. The values are replicated to be "as long as

`start`", the vector of initial parameter values.

From the run\_map\_example, it looks like bounds can't be placed on random effects.

Given RTMB's strategy for mirroring and fixing parameters, some care needs to be

taken with creating the vectors specifying the upper and lower bounds.