

VPA-02(2020)

Frasyrを用いたVPA:実践編①

～チューニングなしVPA～

- 実行方法の紹介
(データの読み込み方から結果のプロットの仕方まで)



動画作成者 漁業情報解析部 宮川光代
(mmiyagawa@affrc.go.jp)

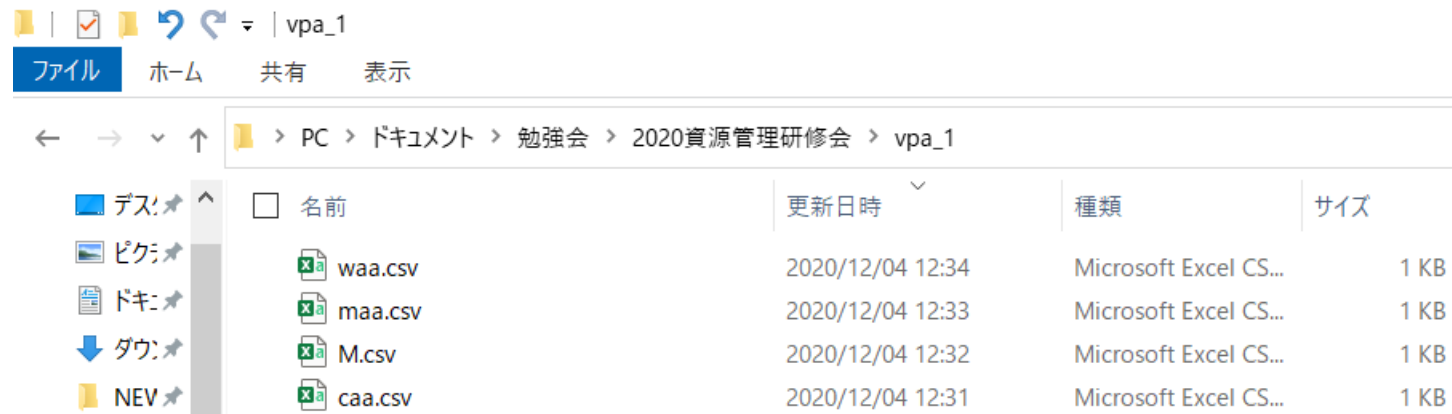
Frasyrを用いたVPAの実践（チューニングなしVPA編）

1) 基礎的な設定

- Rのインストール（インストール方法は動画R-01参照）
- Rstudioのインストール（必須ではないですが、インストールすると便利：インストール方法も動画R-01参照）
- Frasyrのインストール方法は <https://github.com/ichimomo/frasyr> に従って下さい。（動画Tool-03参照）
- インストール後, library(frasyr) でパッケージを呼び出しすることで使えるようになります。

2) VPAに用いるデータファイルの作成

- 色々やり方がありますが、今回は、4つのcsvファイルを作成してRで読み込むようにします
 1. caa.csv ---年別年齢別漁獲尾数のデータファイル
 2. waa.csv ---年別年齢別体重のデータファイル
 3. maa.csv ---年別年齢別成熟率のデータファイル
 4. M.csv ---年別年齢別自然死亡係数のデータファイル



例えば、vpa_1というフォルダーに4つのcsvファイルを置く

4つのcsvファイルの中身の例

- 例:0-3+歳まで, 2011-2020年までのデータを用いる場合

caa.csv : 年別年齢別漁獲尾数 (↓)

[illegible]

waa.csv : 年別年齢別体重 (↓)

[illegible]

maa.csv : 年別年齢別成熟率 (↓)

[illegible]

M.csv : 年別年齢別自然死亡係数 (↓)

[illegible]

データの読み込み（チューニングなしVPA編）

3) VPAに用いるデータファイルの読み込み

```
1 #2020年VPA実践研修
2
3 devtools::install_github("ichimomo/frasyr@dev") #最新版のFrasyrのインストール
4 library(frasyr) #Frasyrの呼び出し
5
6
7 #データの読み込み----
8
9 caa <- read.csv("caa.csv", row.names=1) #caaの読み込み
10 waa <- read.csv("waa.csv", row.names=1) #waaの読み込み
11 maa <- read.csv("maa.csv", row.names=1) #maaの読み込み
12 M <- read.csv("M.csv", row.names=1) #Mの読み込み
13
14 dat <- data.handler(caa=caa, waa=waa, maa=maa, M=M) #データをVPA用の関数に変換する
15
```

R上にこのように5行
書いて、実行させる

#以降はコメント文な
ので省略してよい

データ読み込みの確認

4) データがきちんと読み込まれたことの確認

```
1 #2020年VPA実践研修
2
3 devtools::install_github("ichimomo/frasyr@dev") #最新版のFrasyrのインストール
4 library(frasyr) #Frasyrの呼び出し
5
6
7 #データの読み込み----
8
9 caa <- read.csv("caa.csv", row.names=1) #caaの読み込み
10 waa <- read.csv("waa.csv", row.names=1) #waaの読み込み
11 maa <- read.csv("maa.csv", row.names=1) #maaの読み込み
12 M <- read.csv("M.csv", row.names=1) #Mの読み込み
13
14 dat <- data.handler(caa=caa, waa=waa, maa=maa, M=M) #データをVPA用の関数に変換する
15
16 dat
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
```

↑ datと書いて実行する

datの中身が表示される

\$caa

\$maa

\$waa

\$M

が与えたデータと一致しているか確認

一致していればOK

Console Jobs x

~/勉強会/2020資源管理研修会/vpa_1/

```
> dat <- data.handler(caa=caa, waa=waa, maa=maa, M=M) #データをVPA用の関数に変換する
> dat
$caa
  2011  2012  2013  2014  2015  2016  2017  2018  2019  2020
0 199.87428 216.59647 267.69057 218.69839 165.56032 249.24846 187.90069 130.28690 79.91960 77.11111
1 129.46178 121.41525 162.96263 158.66633 154.13567 95.75337 144.15519 91.39456 62.34938 36.74810
2 72.37195 71.29575 82.08322 85.46280 98.07561 76.98423 47.82475 61.08600 39.04364 26.22548
3 26.55117 32.79557 38.99656 33.76093 40.75214 36.62719 28.75043 15.40572 20.81536 13.74911

$maa
  2011 2012 2013 2014 2015 2016 2017 2018 2019 2020
0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3
2 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8
3 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

$waa
  2011  2012  2013  2014  2015  2016  2017  2018  2019  2020
0 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254 0.03583254
1 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471 0.16698471
2 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750 0.34124750
3 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731 0.50836731

$index
NULL

$M
  2011 2012 2013 2014 2015 2016 2017 2018 2019 2020
0 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
1 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
2 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
3 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4

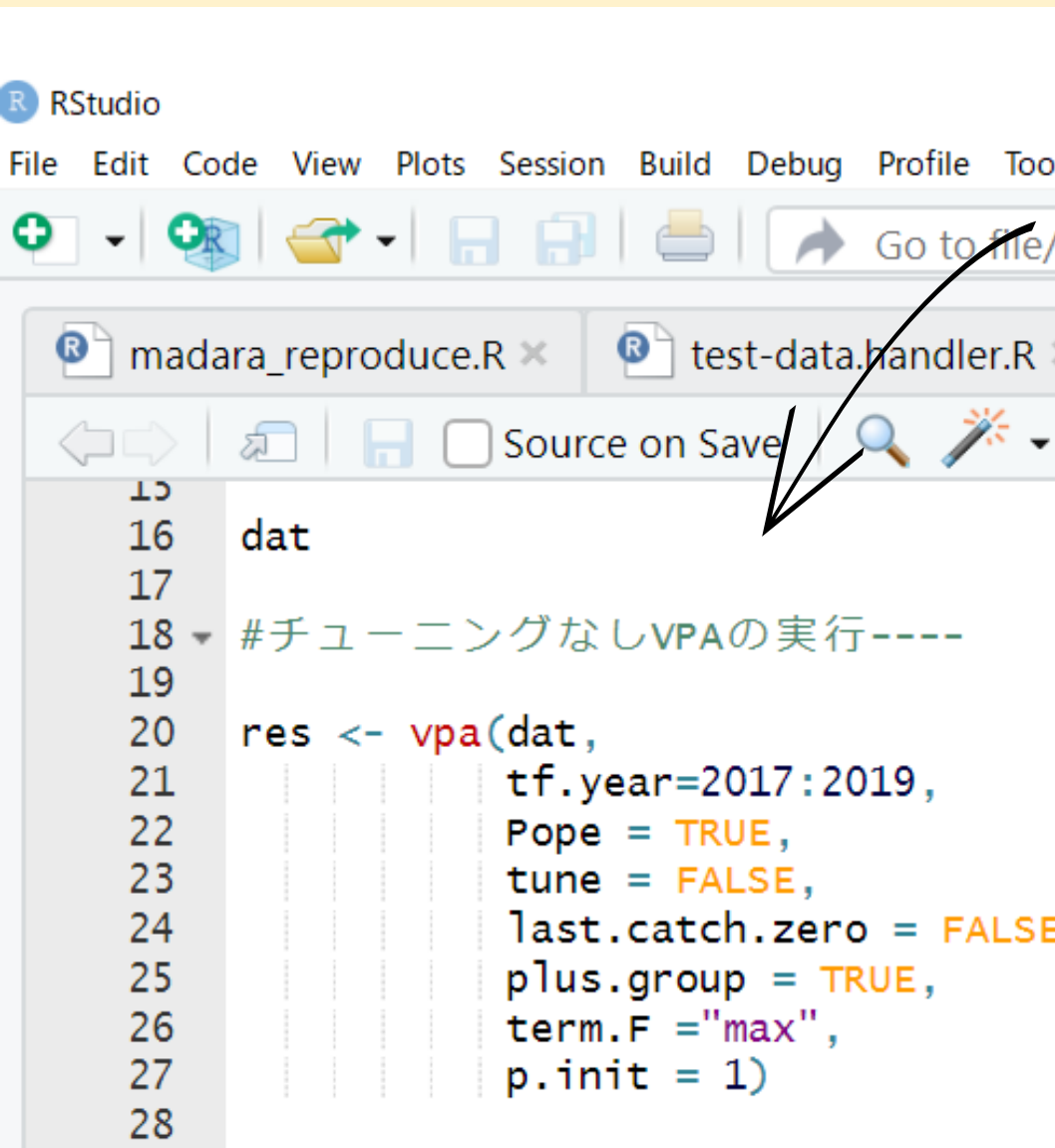
$maa.tune
NULL

$waa.catch
NULL

$catch.prop
NULL
```

VPAの実行（チューニングなしVPA編）

5) VPAを走らせる



```
15  
16 dat  
17  
18 #チューニングなしVPAの実行----  
19  
20 res <- vpa(dat,  
21           tf.year=2017:2019,  
22           Pope = TRUE,  
23           tune = FALSE,  
24           last.catch.zero = FALSE,  
25           plus.group = TRUE,  
26           term.F = "max",  
27           p.init = 1)  
28
```

R上にこのように8行書いて、実行させる
（#以降はコメント文なので省略してよい）

動画VPA-01『**frasyr**を用いたVPA：概要編』で解説しているように、自分のしたい解析に合わせて以下のことを指定する：

1. 最終年の漁獲係数(F)はどの年の平均に等しいとするか
2. Pope近似式を使うか、Baranov方程式にするか
3. チューニングはするのかしないか
4. 最高齢はプラスグループなのか
5. ターミナルFは最高齢だけ推定する

#VPAを行うデータ

#ターミナルFをどの年の平均にするか

#Popeの近似式を使うならTRUE, Baranovを用いるならFALSE

#チューニングはしないのでFALSE（デフォルトはFALSE）

#最終年の漁獲尾数が全部0の場合はTRUE（デフォルトはFALSE）

#最高齢はプラスグループか否か（デフォルトはTRUE）

#ターミナルFの最高齢だけ推定なら"max", 全年齢なら"all"

#ターミナルFに与える初期値

VPAの結果をみる（チューニングなしVPA編）

6) VPAの結果をみる

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source on Save Run Source

```
47
48 #チューニングなしVPAの結果----
49
50 res$term.f #推定されたターミナル F の値をみる
51
52 res$faa #推定された年別年齢別漁獲係数
53
54 res$naa #推定された年別年齢別資源尾数
55
56 res$saa #推定された年別年齢別選択率
57
58 res$baa #推定された年別年齢別資源重量（バイオマス）
59
60 res$ssb #推定された年別年齢別産卵親魚量
61
```

ここを実行すると
推定された様々な値が確認出来る

Console Jobs

~/勉強会/2020資源管理研修会/vpa_2_nidan/

```
>
> #チューニングなしVPAの結果----
>
> res$term.f #推定されたターミナル F の値をみる
[1] 1.046609
> res$faa #推定された年別年齢別漁獲係数
      2011      2012      2013      2014
1 0.4786438 0.4471648 0.5312455 0.5138490 0.6
0.6812031 0.7064806
2 0.7029733 0.7018771 0.8409400 0.7951212 0.9
1.1171955 0.9601168
```

推定された
ターミナルF

VPAの結果を可視化する (I)

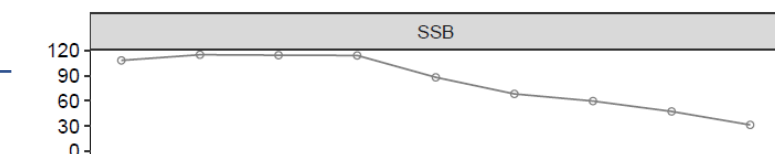
7) VPAの結果をプロットしてみる

```
33 #VPAの結果を可視化する  
34 plot_vpa(res, plot_year=2012:2020)
```

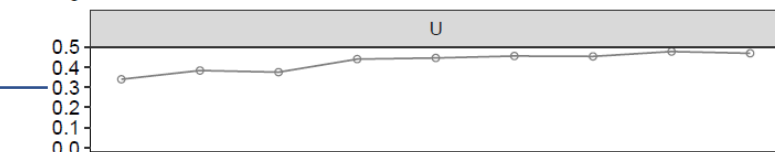
R上にこのように書いて、実行させる

2012年～2020年までの様々な結果をプロット (下図)

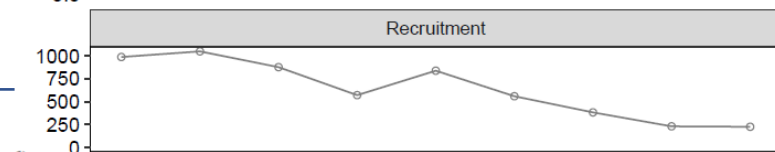
産卵親魚量



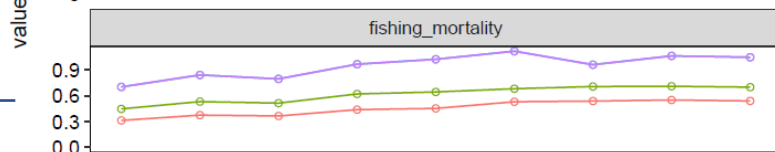
漁獲量/資源量



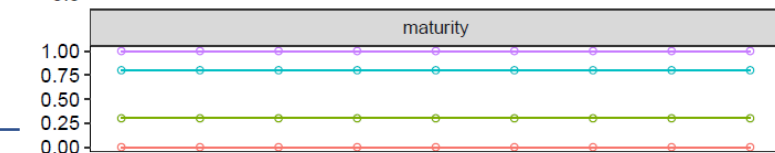
加入量



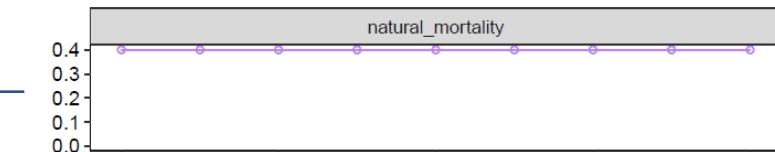
漁獲係数



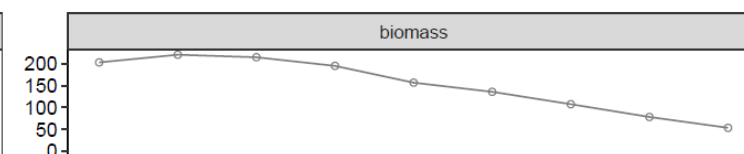
成熟率



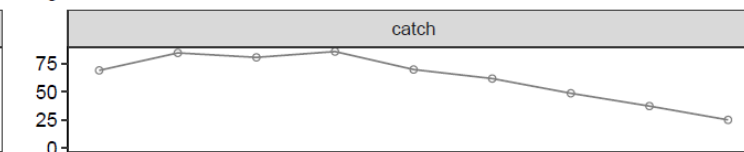
自然死亡係数



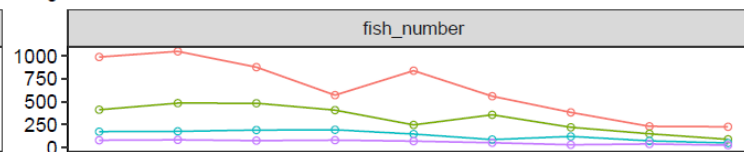
id 1 age 0 1 2 3 NA



資源量



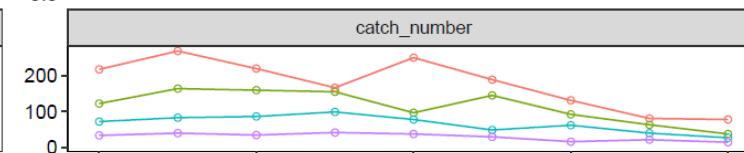
漁獲量



資源尾数



体重



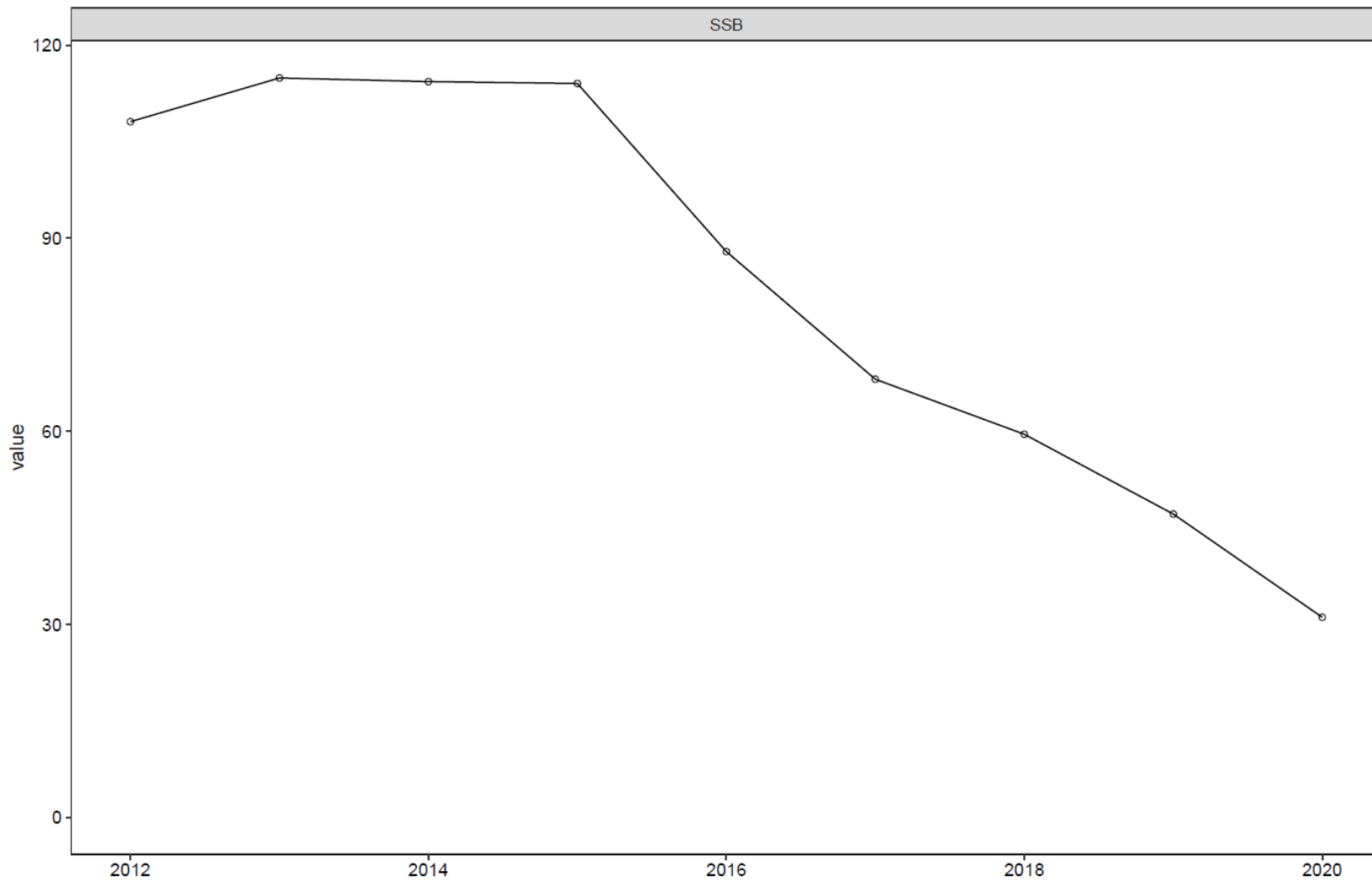
漁獲尾数

VPAの結果を可視化する (2) 特定のプロットのみ

```
36 plot_vpa(res, what.plot="SSB", plot_year=2012:2020)
```

R上にこのように書いて、実行させる

産卵親魚量



What.plotで指定したSSBの結果のみをプロット (左図)

SSB以外にも、前のスライドに図で示してある様々な指標を指定できる：

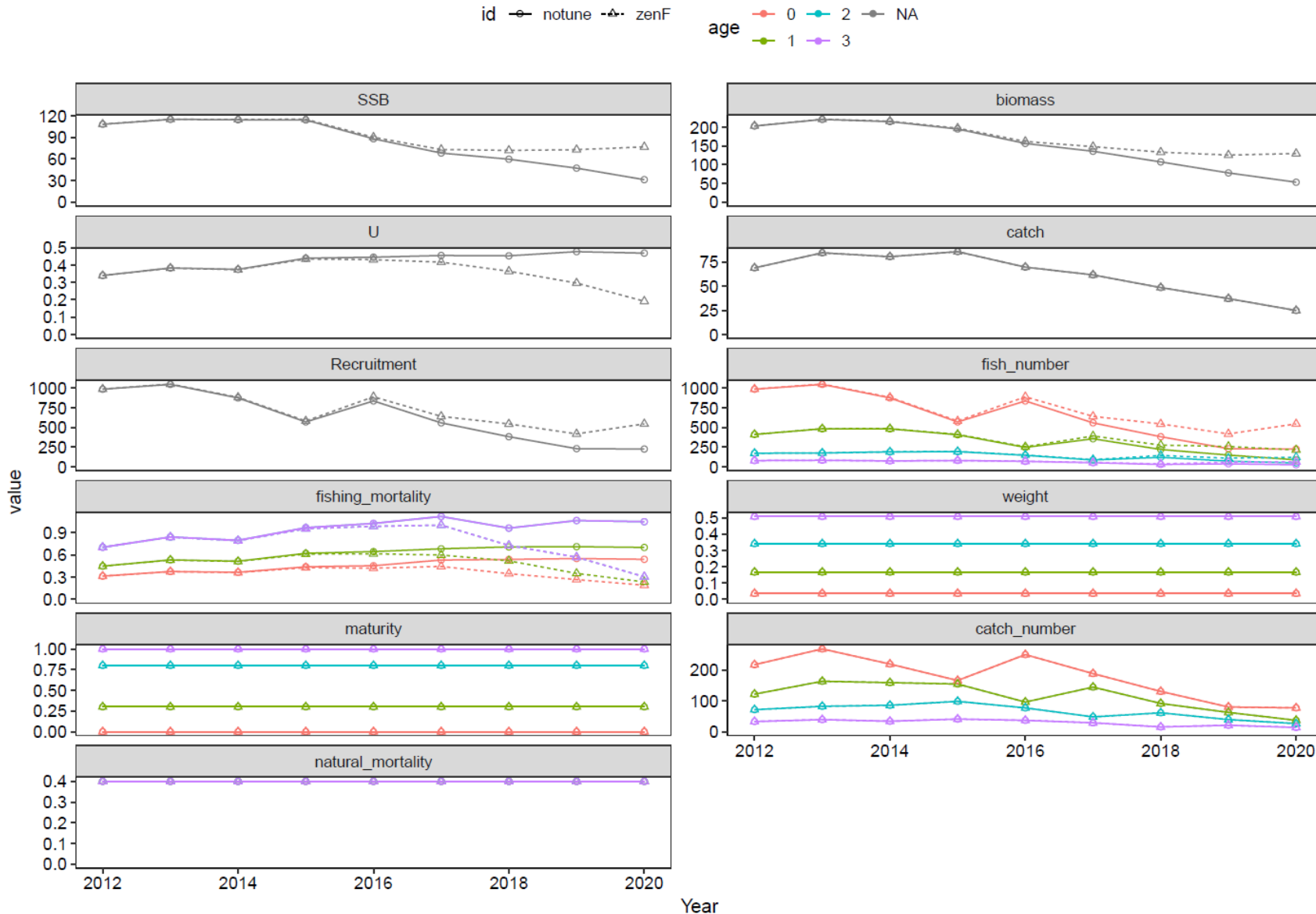
U
Recruitment
fishing_mortality
maturity
natural_mortality
biomass
catch
fish_number
weight
catch_number

VPAの結果を可視化する (3) 比較したい

R上にこのように書いて、実行させる

```
38 plot_vpa(list(res,res2), plot_year=2012:2020, vpaname=c("notune","zenF"))
```

二つ以上のVPA結果を比較したい
ときなどに便利



これで、チューニングなしVPA実践編（動画VPA-02）の解説は終わりです
こちらでご紹介したデータやコードはファイル名：vpa_02_data_codeにあります。
引き続き、チューニングありVPA実践編（動画VPA-03）をご覧ください。

