

SR-02(2022)



frasyrを用いた 再生産関係の推定:実践編

- fit.SR関数を用いたSR関係の推定
- 結果のプロットと解釈

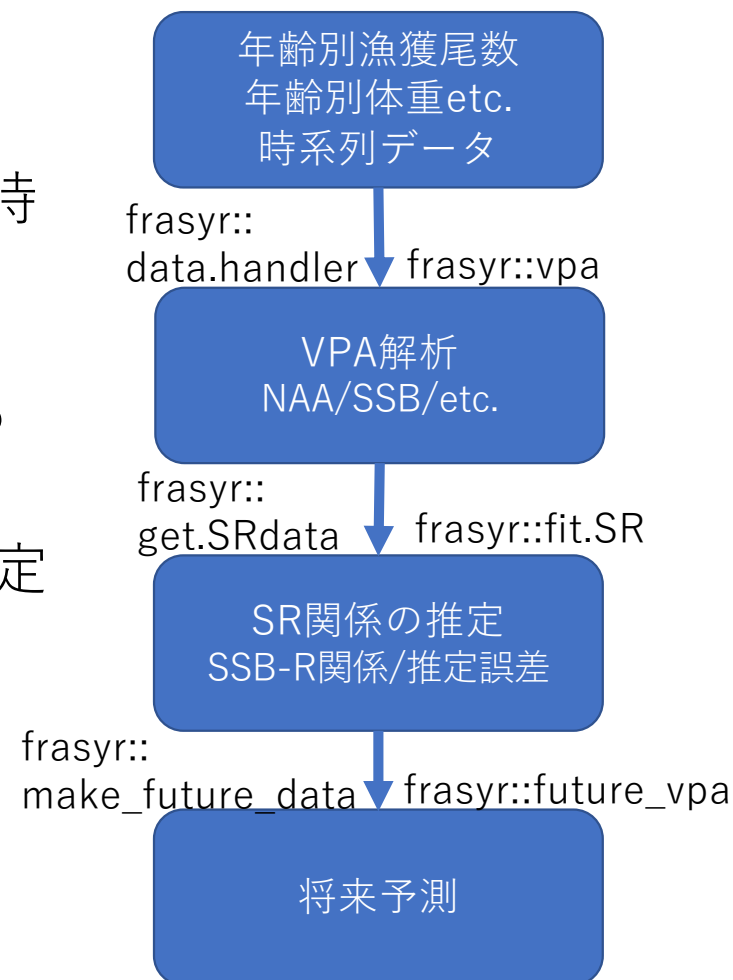
水産研究



動画製作者 漁業情報解析部 資源解析グループ 福井 眞
shinfukui@affrc.go.jp
fukui_shin87@fra.go.jp

VPA解析による資源の年齢別状態の時系列情報

- 再生産関係の推定のために、親魚量と加入量の時系列データが必要
- frasyrを使ってVPAを計算したことを前提とする
- fit.SR関数、fit.SRregime関数で再生産関係を推定する
- ここで紹介する手順は以下を参照
<https://ichimomo.github.io/frasyr/articles/fittingSR.html>



get.SRdataとderive_biopar

- frasyrでVPAの結果のオブジェクトには様々な結果が格納されているが、そのなかから再生産関係の推定に必要なSSB/Recruitmentの時系列を取り出す必要がある
→**get.SRdata**関数をつかう
- SR関係の推定後、モデル診断に生物パラメータ(bio.par)が必要
→**derive_biopar**関数をつかう
- SR関係を推定してみよう！
→**fit.SR**関数、**fit.SRregime**関数をつかう

get.SRdata と derive_biopar

- get.SRdata関数の引数
 - vpares : vpaの戻り値オブジェクト
 - weight.year : SR関係を推定するのに使う期間の指定 (0で全期間)
 - weighth.data : どのデータをSR関係推定に使うか
 - など
- derive_biopar関数の引数
 - res_obj : vpaの戻り値オブジェクト
 - derive_year : 生物パラメータに使う期間
 - stat : 生物パラメータを代表する統計量の種類 (デフォルトはmean)

get.SRdataでSRdataを作成

SR-02.R

```
3 # frasyrのインストールとライブラリーの読み込み
4 devtools::install_github("ichimomo/frasyr@dev")
5 library(frasyr)
6
7 # 例データの呼び出し
8 data("res_vpa_example")
9
10 # vpaオブジェクトからSRdataへの整形
11 SRdata_ex <- get.SRdata(vpares = res_vpa_example, weight.year = 0)
12
13 # 確認
14 names(SRdata_ex)
15 head(SRdata_ex)
16 plot_SRdata(SRdata_ex)
17
```

Environment History Connections Build Git

Import Dataset 244 MiB

R Global Environment

Data

| | |
|-----------------|------------------------|
| res_vpa_example | List of 28 |
| SRdata_ex | 30 obs. of 4 variables |

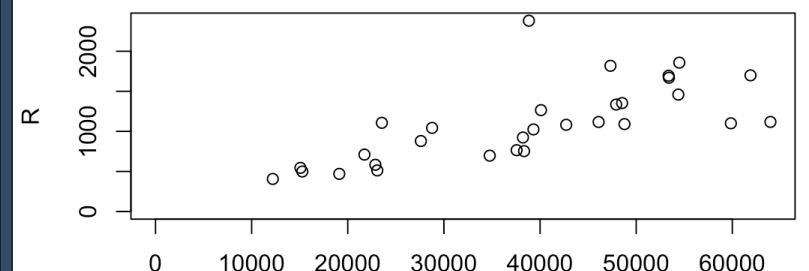
Console Terminal Background Jobs

R 4.1.2 ~ /git/frasyr/

```
> # 確認
> names(SRdata_ex)
[1] "year" "SSB" "R" "weight"
> head(SRdata_ex)
# A tibble: 6 × 4
  year  SSB  R weight
<dbl> <dbl> <dbl> <dbl>
1 1988 12199. 406. 1
2 1989 15267. 499. 1
3 1990 15072. 544. 1
4 1991 19114. 470. 1
5 1992 23544. 1107. 1
6 1993 28769. 1042. 1
```

Files Plots Packages Help Viewer Presentation

Zoom Export Publish



derive_bioparでbio_parを作成

The screenshot displays the RStudio environment with the following components:

- Source Editor:** Contains R code for creating the `bio_par` object. The code is as follows:

```
29 # bio_paroオブジェクトの作成(vpaへ入力したデータの最終年から前5年平均)
30 bio_par <- derive_biopar(res_obj = res_vpa_example,derive_year = c((max
31 (SRdata_ex$year)-4):max(SRdata_ex$year)),stat=mean)
32 # 確認
33 bio_par
34
```
- Console:** Shows the execution of the code and the resulting `bio_par` object. The output is:

```
> # 例データの呼び出し
> data("res_vpa_example")
> # vpaオブジェクトからSRdataへの整形
> SRdata_ex <- get.SRdata(vpares = res_vpa_example,weight.year = 0)
> # 確認
> names(SRdata_ex)
[1] "year" "SSB" "R" "weight"
> # bio_paroオブジェクトの作成(vpaへ入力したデータの最終年から前5年平均)
> bio_par <- derive_biopar(res_obj = res_vpa_example,derive_year = c((max(SRdata_ex
$year)-4):max(SRdata_ex$year)),stat=mean)
> # 確認
> bio_par
# A tibble: 4 × 4
      M    waa    maa    faa
<dbl> <dbl> <dbl> <dbl>
1  0.5    40     0  0.599
2  0.5   100   0.5  1.23
3  0.5   230     1  1.43
4  0.5   380     1  1.43
>
```
- Environment:** Lists the objects in the Global Environment:
 - `bio_par`: 4 obs. of 4 variables
 - `res_vpa_example`: List of 28
 - `SRdata_ex`: 30 obs. of 4 variables
- Terminal:** Shows the R version and the current directory:

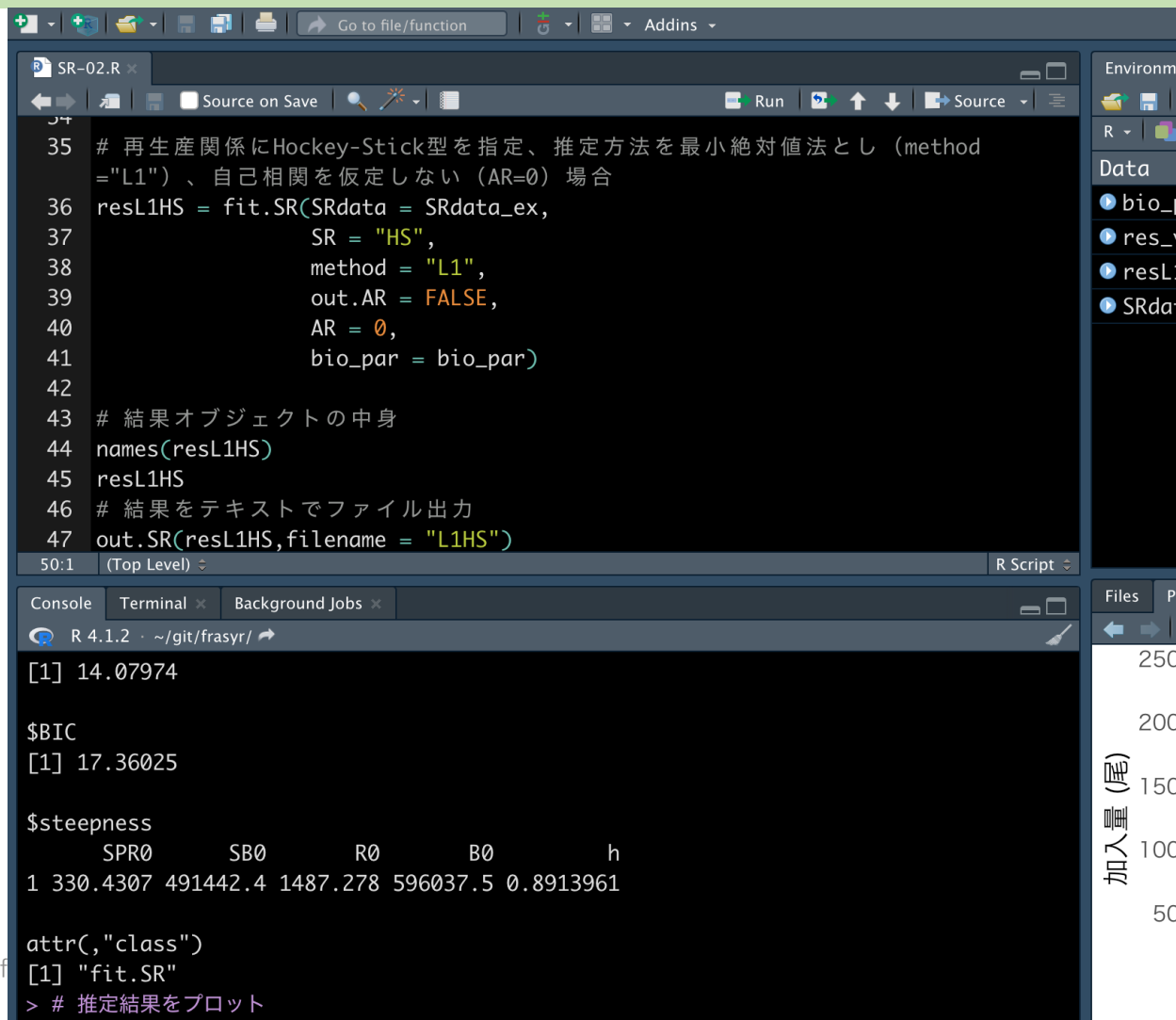
```
R 4.1.2 · ~/git/frasyr/
```

いざ再生産関係の推定へ

- frasyrでVPAの結果のオブジェクトには様々な結果が格納されているが、そのなかから再生産関係の推定に必要なSSB/Recruitmentの時系列を取り出す必要がある
→get.SRdata関数をつかう
- SR関係の推定後、モデル診断に生物パラメータ(bio.par)が必要
→derive_biopar関数をつかう
- **SRdata**、**bio.par**が生成できたら
- SR関係を推定してみよう！
→**fit.SR**関数、**fit.SRregime**関数をつかう

fit.SRを使ってみよう!

- fit.SR関数の引数
 - SRdata
 - SR="BH","RI","HS"
 - method="L1","L2"
 - AR=0 / 1
 - out.AR = FALSE / TRUE
 - bio.par



```
35 # 再生産関係にHockey-Stick型を指定、推定方法を最小絶対値法とし (method
36   = "L1")、自己相関を仮定しない (AR=0) 場合
37 resL1HS = fit.SR(SRdata = SRdata_ex,
38                  SR = "HS",
39                  method = "L1",
40                  out.AR = FALSE,
41                  AR = 0,
42                  bio_par = bio_par)
43 # 結果オブジェクトの中身
44 names(resL1HS)
45 resL1HS
46 # 結果をテキストでファイル出力
47 out.SR(resL1HS, filename = "L1HS")
```

Console

```
R 4.1.2 ~ /git/frasyr/
[1] 14.07974

$BIC
[1] 17.36025

$steepness
      SPR0      SB0      R0      B0      h
1 330.4307 491442.4 1487.278 596037.5 0.8913961

attr(,"class")
[1] "fit.SR"
> # 推定結果をプロット
```

Environment

- R
- Data
 - bio_p
 - res_y
 - resL1
 - SRda

Files

加入量 (尾)

- 250
- 200
- 150
- 100
- 50

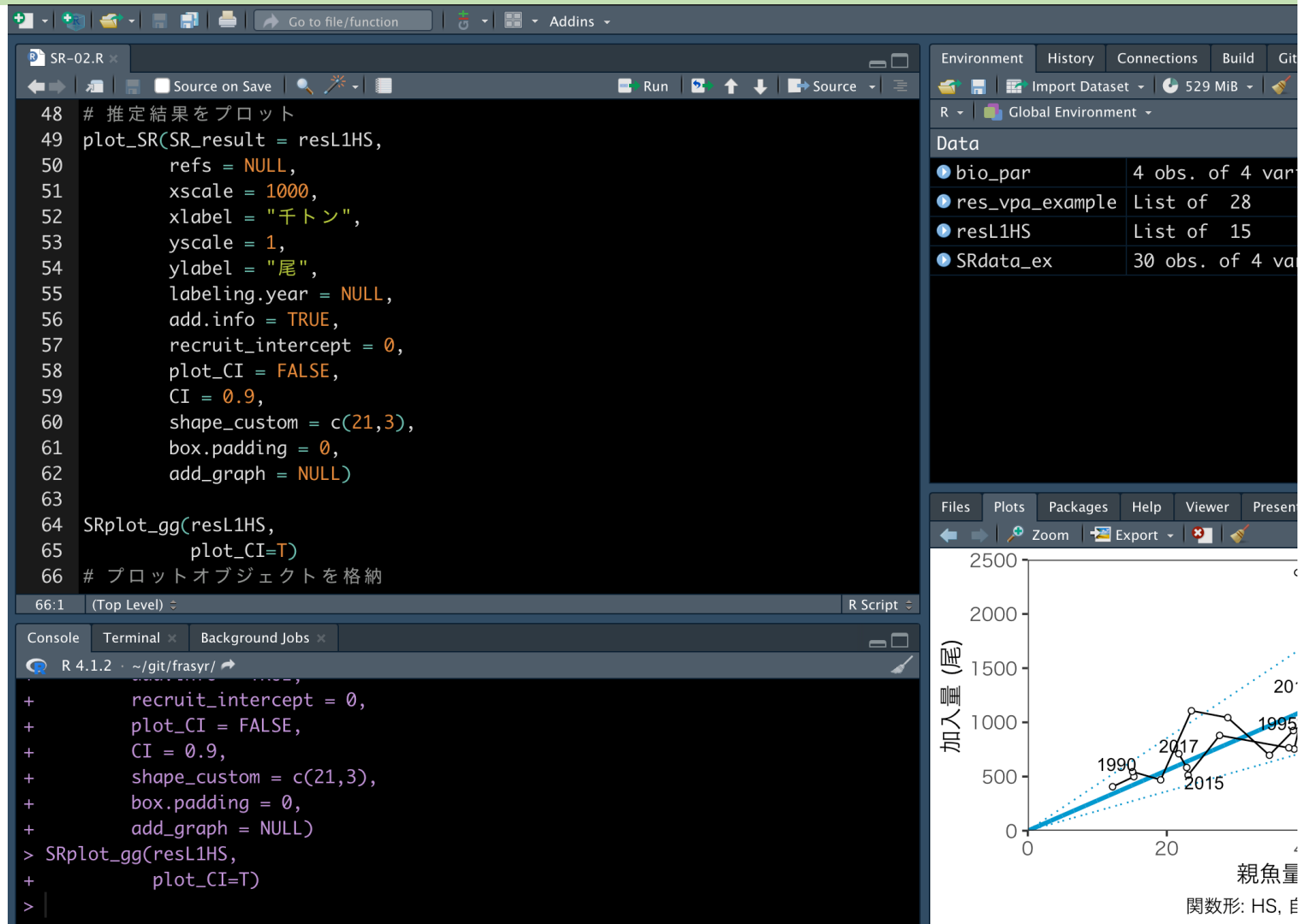
SR-02(2022) f

fit.SRの戻り値オブジェクトと出力

- fit.SR関数の戻り値
 - "input" "obj.f" "obj.f2" "opt" "resid" "resid2" "sd.pred" "pars"
"loglik" "pred" "k" "AIC" "AICc" "BIC" ["steepness"]
- out.SR関数でtxt出力
- plot_SR関数（plot.SR, SRplot_gg関数）で図のプロット
 - ggsave_SH関数で図の保存

plot_SR

- plot_SR関数
(plot.SR,
SRplot_gg)
のオプション



fit.SRregime設定

SR-02.R

```
72 # 再生産関係にHockey-Stick型を指定、推定方法を最小絶対値法とし (method  
73   = "L1")  
74 resR1_L1HS <- fit.SRregime(SRdata_ex, SR = "HS",  
75                           method = "L1",  
76                           regime.year = c(2005),  
77                           regime.par = c("a", "b", "sd")[1:3],  
78                           use.fit.SR = TRUE,  
79                           regime.key = c(0, 1),  
80                           bio_par=bio_par)  
81 # 結果をテキストでファイル出力  
82 out.SR(resR1_L1HS, filename = "R1L1HS")  
83 # 推定結果をプロット  
84 plot_SRregime(resR1_L1HS)  
85 SRregime_plot(resR1_L1HS, regime.name = c("A", "B"))  
86 # 再生産関係にHockey-Stick型を指定、推定方法を最小絶対値法とし (method  
87   = "L1")
```

84:51 (Top Level) R Script

Console

```
[13] "sd.pred" "pred" "pred_to_obs" "steepness"  
> View(resR1_L1HS)  
> # 再生産関係にHockey-Stick型を指定、推定方法を最小絶対値法とし (method="L1")  
> resR1_L1HS <- fit.SRregime(SRdata_ex, SR = "HS",  
+                           method = "L1",  
+                           regime.year = c(2005),  
+                           regime.par = c("a", "b", "sd")[1:3],  
+                           use.fit.SR = TRUE,  
+                           regime.key = c(0, 1),  
+                           bio_par=bio_par)  
> SRregime_plot(resR1_L1HS, regime.name = c("A", "B"))
```

Environment History Connections Build Git

R Global Environment

Data

| | |
|-----------------|------------------------|
| bio_par | 4 obs. of 4 variables |
| res_vpa_example | List of 28 |
| resL1HS | List of 15 |
| resR1_L1HS | List of 16 |
| SRdata_ex | 30 obs. of 4 variables |

Files Plots Packages Help Viewer Presentation

Zoom Export Publish

Weight

- weighted

Regime

- A
- B

SRregime_plot: regime name: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

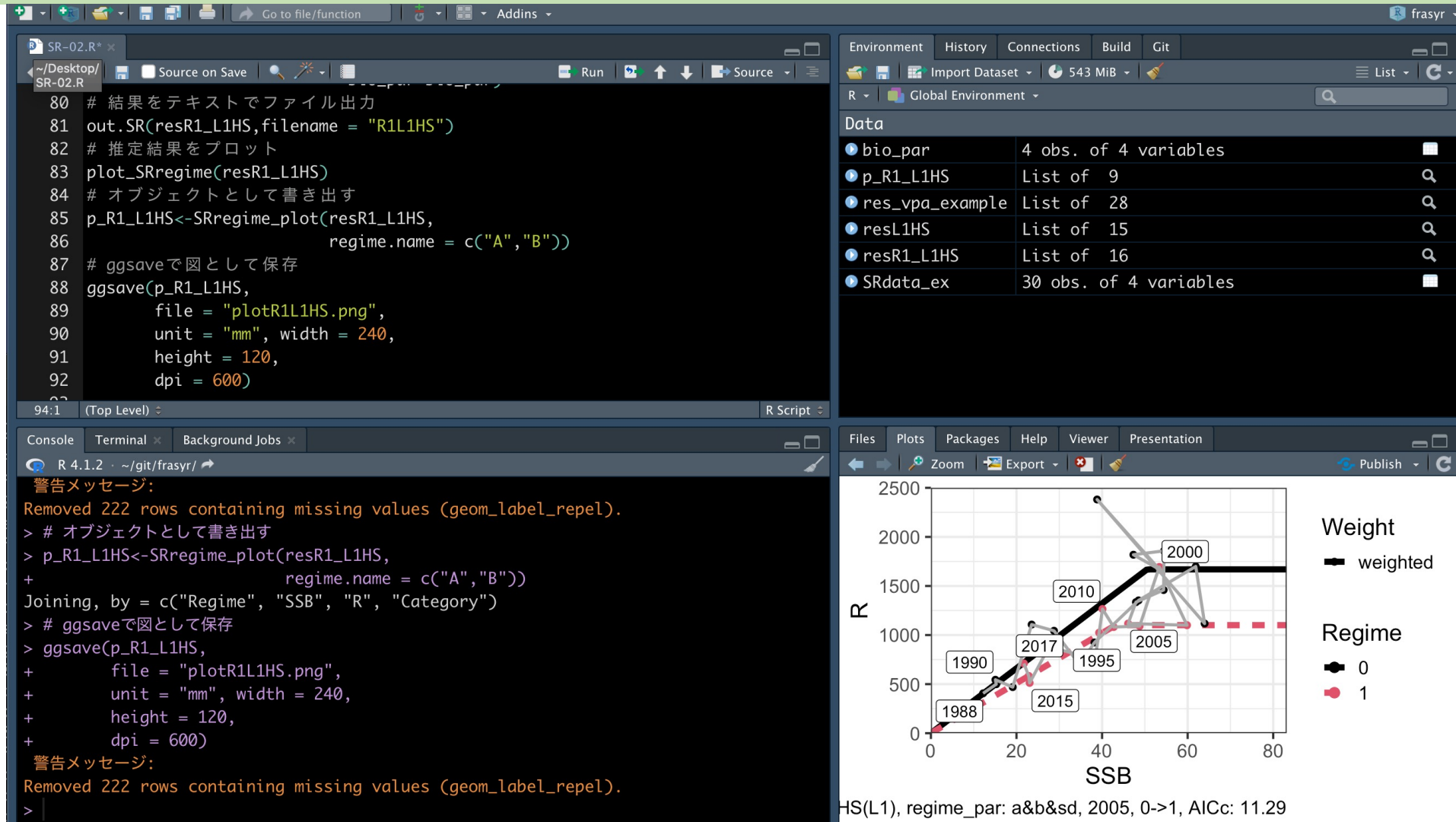
fit.SRregimeの戻り値オブジェクトと出力

- fit.SRregime関数の戻り値
 - "input" "opt" "obj.f" "obj.f2" "resid" "loglik" "k" "AIC"
"AICc" "BIC" "regime_pars" "regime_resid" "sd.pred" "pred"
"pred_to_obs" ["steepness"]
- out.SR関数でtxt出力
- plot_SRregime関数 (SRregime_plot関数) で図のプロット
 - ggsave関数をつかって図の保存

fit.SRregimeの戻り値オブジェクトと出力

- fit.SRregime関数の戻り値
 - "input" "opt" "obj.f" "obj.f2" "resid" "loglik" "k" "AIC"
"AICc" "BIC" "regime_pars" "regime_resid" "sd.pred" "pred"
"pred_to_obs" ["steepness"]
- out.SR関数でtxt出力
- plot_SRregime関数（SRregime_plot関数）で図のプロット
 - ggsave関数をつかって図の保存

plot_SRregime



SR関係を推定できた！



- frasyrを使ったSR関係の推定と結果・図の出力については以上です
- 次回「frasyrを用いた再生産関係の推定:診断編 SR-03(2022)」では推定した再生産関係の妥当性をチェックします

お疲れ様でした！

