

Updated Projections for Gulf of Mexico Gray Triggerfish (*Balistes capriscus*)

by

Joshua Sladek Nowlis

14 December 2006

NOAA Fisheries
Southeast Fisheries Science Center
75 Virginia Beach Drive
Miami, FL 33149

Sustainable Fisheries Division Contribution No. SFD-2006-051

INTRODUCTION

Projections were not the focus of the most recent assessment of the Gulf of Mexico gray triggerfish (*Balistes capriscus*) stock (SEDAR 9-Stock Assessment Report 1). Consequently, they received little attention at the time, and two issues have come up since the review workshop. Both issues pertain to Table 7 in the Post-Review Workshop Addendum. The first issue was that the spawning stock biomass ratios and fishing mortality ratios in the pre-2005 years of the table do not agree precisely with the base run discussed earlier in the document. The second issue was that catches in the table were reported as the estimated values, not the observed values. This report was generated to resolve these issues.

The problem behind the minor disagreement between the base run and the projections is attributable to some simplifications that occurred when transferring from the assessment fitting software (SSASPM) to the projection software (PRO-2BOX). In particular, though the assessment fitting software modeled selectivity of each fleet independently and seasonally (in the case of shrimp bycatch), only an aggregate annual selectivity pattern was used as an input into PRO-2BOX. Because PRO-2BOX models all years in the sequence, not just the projected years, this minor simplification produced a minor discrepancy with the years that were also modeled in SSASPM.

METHODS

To resolve this problem, a C-based deterministic projection was developed that began with 2005. In other words, values were taken from SSASPM through 2004, and the projection software was used to generate values from 2005 onwards (see Appendix for code). This software read in detailed values from a modified version of SSASPM, and used these to project forward using the same algorithms of SSASPM, including seasonal fisheries.

Projections all assumed that new management measures would take place at the beginning of 2007, and affect all sectors of the fishery (including shrimp bycatch) equally. The years 2005 and

2006 were modeled using average fishing mortality rates per fleet from the final three years of the assessment (2002-04), and projection scenarios differed on fishing mortality rates (F) applied from 2007 onwards. Six scenarios were run: F=0, current F (average from 2002-04), F_{MSY} , 75% of F_{MSY} , F_{30} , and 75% of F_{30} .

RESULTS

Projections indicate that substantial reductions in catch are necessary to end overfishing, but more limited reductions would be sufficient to avoid an overfished condition (Table 1; Fig. 1). Fishing rates equal to 75% of F_{MSY} would be sufficient to maintain a population above the overfished threshold of B_{20} . However, this rate is well above the overfishing threshold of F_{30} . To achieve F_{30} , catches would have to be scaled back by nearly 40%. Other scenarios and their consequences are presented in Table 1 and Fig. 1.

TABLE 1—Projections Under Various Fishing Mortality Scenarios.
All projections based on new management measures starting in 2007.

Catches (t lbs)

Year	No Fishing	Current F	Fmsy	75% Fmsy	F30	75% F30
2004	1248	1248	1248	1248	1248	1248
2005	1100	1100	1100	1100	1100	1100
2006	1060	1060	1060	1060	1060	1060
2007	0	1023	1081	841	686	526
2008	0	996	1031	873	750	605
2009	0	979	996	903	809	681
2010	0	968	972	929	860	749
2011	0	961	956	950	902	806
2012	0	956	946	966	935	853
2013	0	953	939	976	958	888
2014	0	951	934	983	975	913
2015	0	949	930	988	986	932
2016	0	948	928	992	994	945
2017	0	947	927	994	1000	955
2018	0	947	925	996	1004	962
2019	0	947	925	997	1007	967
2020	0	946	924	998	1009	971
2021	0	946	924	998	1011	974
2022	0	946	923	999	1012	976

Shrimp Bycatch (t lbs)

Year	No Fishing	Current F	Fmsy	75% Fmsy	F30	75% F30
2004	167	167	167	167	167	167
2005	616	616	616	616	616	616
2006	607	607	607	607	607	607
2007	0	605	636	507	419	326
2008	0	603	627	521	442	352
2009	0	601	624	522	446	357
2010	0	600	621	525	450	362
2011	0	599	619	527	453	366
2012	0	598	617	528	455	369
2013	0	597	616	529	457	371
2014	0	597	615	530	459	373
2015	0	597	614	530	460	374
2016	0	596	614	531	461	375
2017	0	596	613	531	462	376
2018	0	596	613	531	462	377
2019	0	596	613	532	462	377
2020	0	596	613	532	463	377
2021	0	596	613	532	463	378
2022	0	596	613	532	463	378

TABLE 1 (cont.)—Projections Under Various Fishing Mortality Scenarios.

All projections based on new management measures starting in 2007.

SSB/SSB20

Year	No Fishing	Current F	Fmsy	75% Fmsy	F30	75% F30
2004	1.02	1.02	1.02	1.02	1.02	1.02
2005	1	1	1	1	1	1
2006	0.97	0.97	0.97	0.97	0.97	0.97
2007	1.07	0.95	0.94	0.97	0.99	1.01
2008	1.38	0.94	0.91	1.01	1.07	1.14
2009	1.72	0.92	0.89	1.04	1.15	1.26
2010	2.07	0.91	0.87	1.06	1.21	1.37
2011	2.42	0.91	0.86	1.08	1.25	1.46
2012	2.77	0.9	0.85	1.09	1.3	1.54
2013	3.12	0.9	0.84	1.11	1.33	1.61
2014	3.47	0.89	0.84	1.11	1.36	1.67
2015	3.8	0.89	0.83	1.12	1.38	1.72
2016	4.1	0.89	0.83	1.13	1.39	1.75
2017	4.34	0.89	0.83	1.13	1.4	1.78
2018	4.54	0.89	0.83	1.13	1.41	1.8
2019	4.7	0.89	0.83	1.13	1.42	1.81
2020	4.83	0.89	0.83	1.14	1.42	1.82
2021	4.94	0.89	0.83	1.14	1.42	1.83
2022	5.02	0.89	0.83	1.14	1.43	1.84

F/F30

Year	No Fishing	Current F	Fmsy	75% Fmsy	F30	75% F30
2004	1.62	1.62	1.62	1.62	1.62	1.62
2005	1.56	1.56	1.56	1.56	1.56	1.56
2006	1.56	1.56	1.56	1.56	1.56	1.56
2007	0	1.56	1.67	1.25	1	0.75
2008	0	1.56	1.67	1.25	1	0.75
2009	0	1.56	1.67	1.25	1	0.75
2010	0	1.56	1.67	1.25	1	0.75
2011	0	1.56	1.67	1.25	1	0.75
2012	0	1.56	1.67	1.25	1	0.75
2013	0	1.56	1.67	1.25	1	0.75
2014	0	1.56	1.67	1.25	1	0.75
2015	0	1.56	1.67	1.25	1	0.75
2016	0	1.56	1.67	1.25	1	0.75
2017	0	1.56	1.67	1.25	1	0.75
2018	0	1.56	1.67	1.25	1	0.75
2019	0	1.56	1.67	1.25	1	0.75
2020	0	1.56	1.67	1.25	1	0.75
2021	0	1.56	1.67	1.25	1	0.75
2022	0	1.56	1.67	1.25	1	0.75

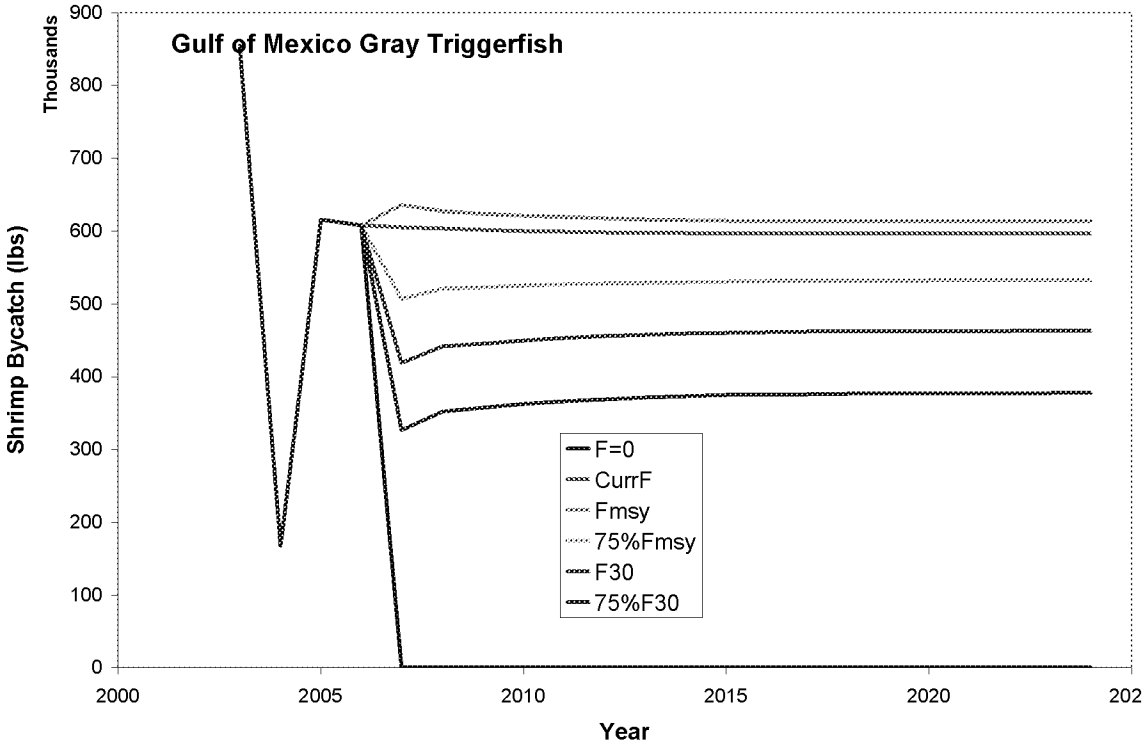
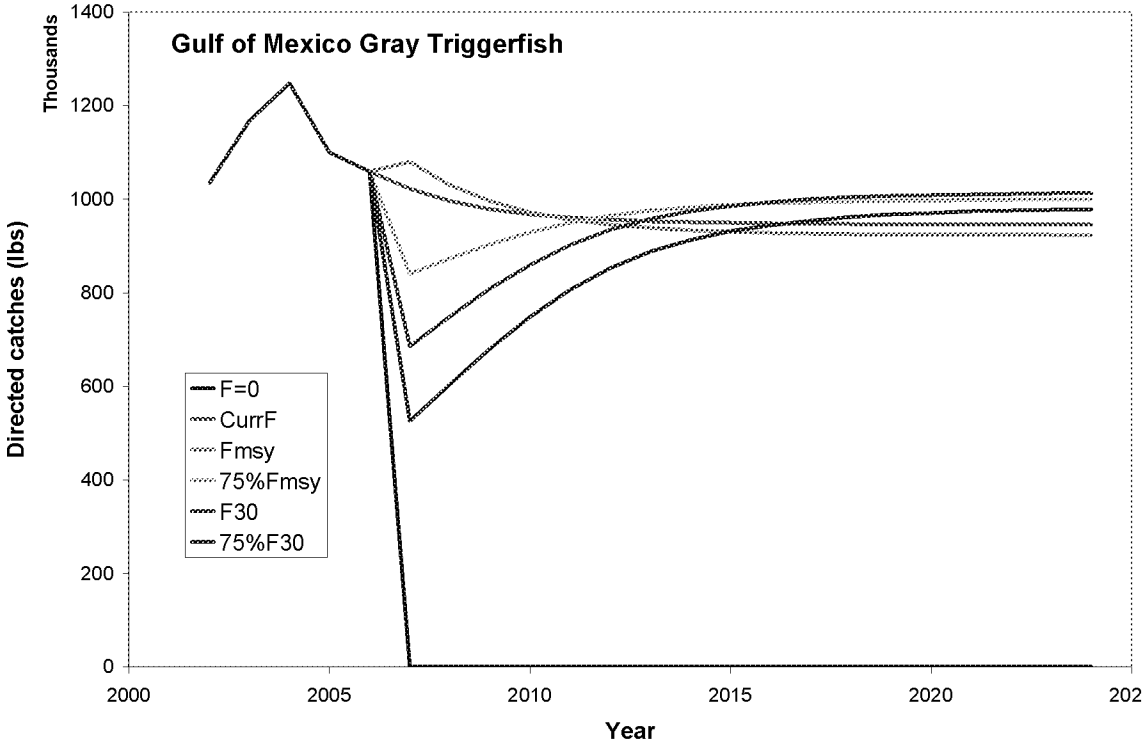


FIG. 1—Projections Under Various Fishing Mortality Scenarios.
All projections based on new management measures starting in 2007. Red lines for SSB and F ratios indicate current legal thresholds for overfishing and overfished status.

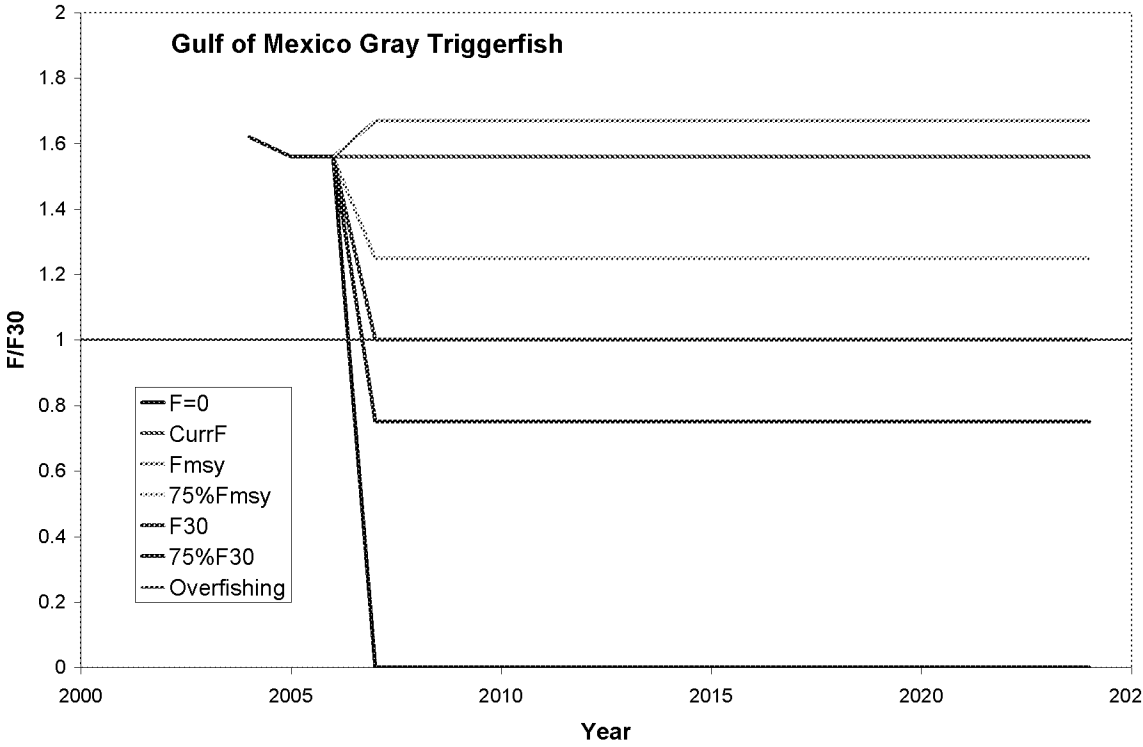
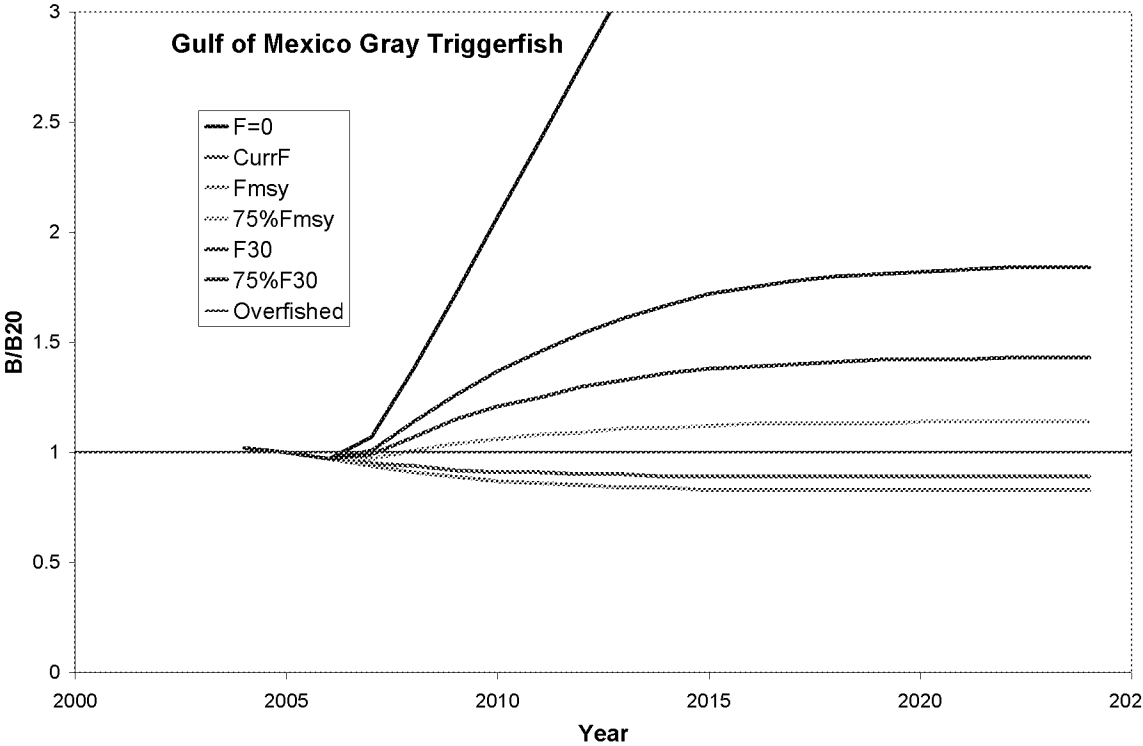


FIG. 1 (cont.)—Projections Under Various Fishing Mortality Scenarios. All projections based on new management measures starting in 2007. Red lines for SSB and F ratios indicate current legal thresholds for overfishing and overfished status.

APPENDIX

```

/*****
 *
 *          SSASPM projection.cpp
 *
 *          by Josh Sladek Nowlis
 *
 *          Last modified 14 DEC 2006
 *
 * This program does projections using the same age-structured population
 * model that is the heart of SSASPM. It reads versions of SSASEM output
 * that have been modified to include additional information (principally
 * fleet specific faa values, waa, fec, and maturity vectors, and scalars for
 * M and spawning increment).
 * At present, one must design the projections within this code and pay
 * attention to constants defined at the top of this code.
 *
 * Need to add (to SSASPM and here) preferred fishery units (yield or $s).
 * Should also change so age indexing doesn't assume start at 1.
 *
 *****/

#include "stdafx.h"
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#define FNAME "ssaspmLinear"
#define MAXSTEPS 12
#define MAXAGES 40
#define MAXSERIES 10
#define PROJYRS 20
#define PRINTFLAG 0 // 1 prints progress to console

main (void)
{

int flag, series, dummy, spawning, a, y;
int fishseason[MAXSERIES+1][2];
int firstyr, lastyr, nsteps, nages, nseries;
double dummy1, dummy2, dummy3, dummy4;
double SSB0, R0, Fmsy, MSY, SSBmsy, F20, SSB20, F30, SSB30, F40, SSB40, F50, SSB50;
double alpha;
double m[MAXAGES+1], p[MAXAGES+1], fec[MAXAGES+1];
double F[MAXAGES+1][PROJYRS+3][MAXSERIES+1];
double n[MAXAGES+1][PROJYRS+4][MAXSTEPS+1];
double total_yield[PROJYRS+3][MAXSERIES+1][2];
double w[MAXAGES+1][4];
char marker[4], fin[17], fout[17];
FILE *inp, *outp;

int i;
double steep, Tho;
double ssb[PROJYRS+3];
double avg_f[PROJYRS+3];
double avgn[MAXAGES][PROJYRS+3];
double total_catch[MAXAGES][PROJYRS+3];

/*-----*
 * INPUTS read from SSASEM output files that include fleet specific F values *
 * and other key parameters *
 *-----*/

```

```

sprintf (fin, "%s.rep", FNAME);
inp = fopen (fin, "r");
printf ("\n\nReading from %s...\n\n", fin);
flag = 0;

while (flag ==0)
{
    fgets (marker, 4, inp);

    if (marker[0]=='A' && marker[1]=='1' && marker[2]=='.')
    {
        while (getc(inp) != '\n') continue; // finishes rest of line
        while (getc(inp) != '\n') continue; // finishes 1st title line
        fscanf (inp, "%d %d %d", &firstyr, &lastyr, &nsteps);
        while (getc(inp) != '\n') continue; // finishes prev line
        while (getc(inp) != '\n') continue; // finishes 2nd title line
        fscanf (inp, "%d %d", &nages, &nseries);
        while (getc(inp) != '\n') continue; // finishes prev line
        while (getc(inp) != '\n') continue; // finishes 3rd title line
        while (getc(inp) != '\n') continue; // finishes 4th title line
        for (series=1; series<=nseries; series++)
            fscanf (inp, "%d %d %d", &dummy, &fishseason[series][0],
                &fishseason[series][1]);
        while (getc(inp) != '\n') continue; // finishes prev line
        while (getc(inp) != '\n') continue; // finishes 5th title line
        fscanf (inp, "%d", &spawning);

        if (PRINTFLAG == 1)
        {
            printf ("%d %d %d\n", firstyr, lastyr, nsteps);
            printf ("%d %d\n", nages, nseries);
            for (series=1;series<=nseries; series++)
                printf ("%d %d %d\n", series, fishseason[series][0],
                    fishseason[series][1]);
            printf ("%d\n\n", spawning);
        }

        if ((nsteps > MAXSTEPS) || (nages > MAXAGES) || (nseries >
            MAXSERIES))
        {
            printf ("ERROR: array boundary exceeded (nsteps, nages, or
                nseries)\n\n");
            exit(1);
        }
    }
}
else if (marker[0]=='A' && marker[1]=='2' && marker[2]=='.')
{
    while (getc(inp) != '\n') continue; // finishes rest of line
    while (getc(inp) != '\n') continue; // finishes title line
    for (a=1; a<=nages; a++)
        fscanf (inp, "%d %lf %lf %lf", &dummy, &m[a], &p[a], &fec[a]);

    if (PRINTFLAG == 1)
    {
        for (a=1; a<=nages; a++)
            printf ("%d %6.4lf %6.4lf %6.4lf\n", a, m[a], p[a], fec[a]);
        printf ("\n");
    }
}
else if (marker[0]=='A' && marker[1]=='3' && marker[2]=='.')
{
    while (getc(inp) != '\n') continue; // finishes rest of line

```

```

for (series=1; series<=nseries; series++)
{
    while (getc(inp) != '\n') continue; // finishes 1st title line
    while (getc(inp) != '\n') continue; // finishes 2nd title line
    while (getc(inp) != '\n') continue; // finishes 3rd title line
    for (y=1; y<lastyr-firstyr-1; y++)
        while (getc(inp) != '\n') continue; // skips all years but last 3
    for (y=0; y<3; y++)
    {
        fscanf (inp, "%d", &dummy);
        if (PRINTFLAG == 1) printf ("%d\n", dummy);
        for (a=1; a<=nages; a++)
            fscanf (inp, "%lf", &F[a][y][series]);
    }
    while (getc(inp) != '\n') continue; // finishes title line
    while (getc(inp) != '\n') continue; // finishes title line
} // next series

if (PRINTFLAG == 1)
    for (series=1; series<=nseries; series++)
    {
        printf ("Series %d\n", series);
        for (y=0; y<3; y++)
        {
            for (a=1; a<=nages; a++)
                printf ("%6.4lf", F[a][y][series]);
            printf ("\n");
        }
        printf ("\n");
    }
}
else if (marker[0]=='L' && marker[1]=='I' && marker[2]=='F')
{
    while (getc(inp) != ':') continue; // finishes rest of text
    fscanf (inp, "%lf", &alpha);

    if (PRINTFLAG == 1) printf ("%5.2lf\n\n", alpha);
}
else if (marker[0]=='C' && marker[1]=='.' && marker[2]==' ')
{
    while (getc(inp) != '\n') continue; // finishes rest of line
    while (getc(inp) != '\n') continue; // finishes 1st title line
    while (getc(inp) != '\n') continue; // finishes 2nd title line
    for (a=1; a<8; a++) getc(inp); // skips label
    fscanf (inp, "%lf %lf %lf %lf %lf %lf", &dummy1, &dummy2, &dummy3, &SSB0,
        &dummy4, &R0);
    for (a=1; a<8; a++) getc(inp); // skips label
    fscanf (inp, "%lf %lf %lf %lf %lf %lf", &Fmsy, &MSY, &dummy1, &SSBmsy,
        &dummy2, &dummy3);
    while (getc(inp) != '\n') continue; // finishes line
    while (getc(inp) != '\n') continue; // skips 1st line
    while (getc(inp) != '\n') continue; // skips 2nd line
    for (a=1; a<8; a++) getc(inp); // skips label
    fscanf (inp, "%lf %lf %lf %lf %lf %lf", &F20, &dummy1, &dummy2, &SSB20,
        &dummy3, &dummy4);
    while (getc(inp) != '\n') continue; // finishes line
    for (a=1; a<8; a++) getc(inp); // skips label
    fscanf (inp, "%lf %lf %lf %lf %lf %lf", &F30, &dummy1, &dummy2, &SSB30,
        &dummy3, &dummy4);
    while (getc(inp) != '\n') continue; // finishes line
    for (a=1; a<8; a++) getc(inp); // skips label
    fscanf (inp, "%lf %lf %lf %lf %lf %lf", &F40, &dummy1, &dummy2, &SSB40,
        &dummy3, &dummy4);
}

```

```

while (getc(inp) != '\n') continue; // finishes line
for (a=1; a<8; a++) getc(inp); // skips label
fscanf (inp, "%lf %lf %lf %lf %lf %lf", &F50, &dummy1, &dummy2, &SSB50,
        &dummy3, &dummy4);

if (PRINTFLAG == 1)
{
    printf ("%8.0lf %8.0lf %6.4lf %8.0lf %8.0lf\n\n", SSB0, R0, Fmsy, MSY,
            SSBmsy);
    printf ("%6.4lf %8.0lf\n%6.4lf %8.0lf\n", F20, SSB20, F30, SSB30);
    printf ("%6.4lf %8.0lf\n%6.4lf %8.0lf\n\n", F40, SSB40, F50, SSB50);
}
}
else if (marker[0]=='E' && marker[1]=='.' && marker[2]==' ')
{
    while (getc(inp) != '\n') continue; // finishes rest of line
    while (getc(inp) != '\n') continue; // finishes 1st title line
    while (getc(inp) != '\n') continue; // finishes 2nd title line
    for (y=1; y<lastyr-firstyr-1; y++)
        while (getc(inp) != '\n') continue; // skips all years but last 3
    for (y=0; y<3; y++)
    {
        fscanf (inp, "%d", &dummy);
        if (PRINTFLAG == 1) printf ("%d\n", dummy);
        for (a=1; a<=nages; a++)
            fscanf (inp, "%lf", &n[a][y][1]);
    }

    if (PRINTFLAG == 1)
    {
        for (y=0; y<3; y++)
        {
            for (a=1; a<=nages; a++)
                printf ("%10.0lf", n[a][y][1]);
            printf ("\n");
        }
        printf ("\n");
    }
}
else if (marker[0]=='H' && marker[1]=='.' && marker[2]==' ')
{
    while (getc(inp) != '\n') continue; // finishes rest of line
    while (getc(inp) != '\n') continue; // finishes 1st title line
    while (getc(inp) != '\n') continue; // finishes 2nd title line
    for (series=1; series<=nseries; series++)
    {
        for (y=1; y<lastyr-firstyr-1; y++)
            while (getc(inp) != '\n') continue; // skips all years but last 3
        for (y=0; y<3; y++)
        {
            fscanf (inp, "%d %d %lf %lf", &a, &dummy,
                    &total_yield[y][series][0], &total_yield[y][series][1]);
            fscanf (inp, "%lf %lf %lf", &dummy1, &dummy2, &dummy3);
            if (PRINTFLAG == 1) printf ("%d %d\n", a, dummy);
        }
    }
    while (getc(inp) != '\n') continue; // finishes 2nd title line
} // next series

if (PRINTFLAG == 1)
    for (series=1; series<=nseries; series++)
    {
        printf ("Series %d\n", series);
        for (y=0; y<3; y++)

```

```

        {
            printf ("%d %9.0lf %9.0lf\n", (y+lastyr-2),
                total_yield[y][series][0], total_yield[y][series][1]);
        }
        printf ("\n");
    }
}
else if (marker[0]=='L' && marker[1]=='.' && marker[2]==' ')
{
    while (getc(inp) != '\n') continue; // finishes rest of line
    while (getc(inp) != '\n') continue; // finishes 1st title line
    while (getc(inp) != '\n') continue; // finishes 2nd title line
    for (y=1; y<lastyr-firstyr-1; y++)
        while (getc(inp) != '\n') continue; // skips all years but last 3
    for (y=1; y<=3; y++)
    {
        fscanf (inp, "%d", &dummy);
        if (PRINTFLAG == 1) printf ("%d\n", dummy);
        for (a=1; a<=nages; a++)
            fscanf (inp, "%lf", &w[a][y]);
    }

    if (PRINTFLAG == 1)
    {
        for (y=1; y<=3; y++)
        {
            for (a=1; a<=nages; a++)
                printf (" %7.3lf", w[a][y]);
            printf ("\n");
        }
        printf ("\n\n");
    }

    flag = 1; // signal to close the input file and move on
}
} // end of file

fclose (inp);
printf ("Finished reading %s\n\n", fin);

/*.....*
- Preliminary parameter calculations -
*.....*/

steep = alpha / (4 + alpha);
Tho = 0;
for (a=1; a<nages; a++)
{
    Tho += p[a]*fec[a]*exp(-m[a]*((float(spawning)-1)/float(nsteps)+(a-1)));
    if (PRINTFLAG == 1) printf ("%lf %lf\n", ((float(spawning)-1)/float(nsteps)),
        exp(-m[a]*((float(spawning)-1)/float(nsteps)+(a-1))));
}
Tho += p[nages]*fec[nages]*exp(-m[a]*((float(spawning)-1)/float(nsteps)+
(a-1)))/(1-exp(-m[nages]));
if (PRINTFLAG == 1) printf ("%lf %lf\n", ((float(spawning)-1)/float(nsteps)),
    exp(-m[a]*((float(spawning)-1)/float(nsteps)+(a-1)))/(1-exp(-m[nages])));

for (a=1; a<=nages; a++)
{
    w[a][0] = 0;
    for (y=1; y<=3; y++)
        w[a][0] += w[a][y];
    w[a][0] /= 3;
}

```

```

}

if (PRINTFLAG == 1)
{
    printf ("%lf %lf\n\n", steep, Tho);
    for (a=1; a<=nages; a++) printf ("%7.3lf", w[a][0]);
    printf ("\n\n");
}

/*-----*
- Constructing F series for projections -
*-----*/

for (y=3; y<5; y++)          // using average Fs for the first two proj years
    for (a=1; a<=nages; a++)
        for (series=1; series<=nseries; series++)
            F[a][y][series] = (F[a][0][series]+F[a][1][series]+F[a][2][series])/3;

sprintf (fin, "%s.prj", FNAME);
inp = fopen (fin, "r");
fseek(inp, 0L, SEEK_SET ); // Set pointer to beginning of file
printf ("\n\nReading from %s...\n\n", fin);
flag = 0;

while (flag == 0)
{
    fscanf (inp, "%4d", &y);
    y = y - lastyr + 2;
    for (series=1; series <=nseries; series++)
    {
        fscanf (inp, "%lf", &dummy1);
        for (a=1; a<=nages; a++)
            F[a][y][series] = dummy1 * F[a][4][series]; // scaling avg F
    }
    if (y == PROJYRS + 2) flag = 1;
}

fclose (inp);
printf ("Finished reading %s\n\n", fin);

if (PRINTFLAG == 1)
{
    for (series=1; series<=nseries; series++)
    {
        for (y=0; y<PROJYRS+3; y++)
        {
            printf ("%4d", (y+lastyr-2));
            for (a=1; a<=nages; a++)
                printf (" %6.4lf", F[a][y][series]);
            printf ("\n");
        }
        printf ("\n");
    }
    printf ("\n\n");
}

/*-----*
- Population projected into the future -
*-----*/

for (y=2; y<PROJYRS+3; y++)
{
    ssb[y] = 0; // initialization
}

```

```

avg_f[y] = 0; // initialization
for (series=1; series<=nseries; series++)
  total_yield[y][series][1] = 0; // initialization
for (a=1; a<=nages; a++)
{
  avgn[a][y] = n[a][y][1]; // initialization
  total_catch[a][y] = 0; // initialization
  for (i=1; i<=nsteps; i++)
  {
    if (i == spawning) ssb[y] += p[a]*fec[a]*n[a][y][i];
    n[a][y][i+1] = n[a][y][i]*exp(-0.5*m[a]/nsteps); // 1/2 int M
    for (series=1; series<=nseries; series++) // catch by series
      if ((i>=fishseason[series][0]) && (i<=fishseason[series][1]))
      {
        n[a][y][i+1] -= F[a][y][series]*n[a][y][i+1];
        total_catch[a][y] += F[a][y][series]*n[a][y][i+1];
        total_yield[y][series][1] +=
          F[a][y][series]*n[a][y][i+1]*w[a][0];
      } // end series loop
    n[a][y][i+1] = n[a][y][i+1]*exp(-0.5*m[a]/nsteps); // 2/2 int M
    avgn[a][y] += n[a][y][i+1];
  } // end i loop
  avgn[a][y] /= (nsteps+1);

  if (a != nages) // next step begins new year
    n[a+1][y+1][1] = n[a][y][nsteps+1];
  else // plus group
    n[a][y+1][1] += n[a][y][nsteps+1];

  if (total_catch[a][y]/avgn[a][y] > avg_f[y]) avg_f[y] =
    total_catch[a][y]/avgn[a][y];
} // end a loop

n[1][y+1][1] = 4*steep*(ssb[y]/Tho)/(1-steep+(ssb[y]/Tho)*((5*steep-1)/R0));

/*if (PRINTFLAG == 1)
{
  printf ("%d", y+lastyr-2+1);
  for (a=1; a<=nages; a++)
    printf (" %8.0lf", n[a][y+1][1]);
  printf ("\n");
} // end if*/
} // end y loop

/*-----*
- KEY OUTPUTS written to output file -
*-----*/

sprintf (fout, "%s.out", FNAME);
outp = fopen(fout, "w");
printf ("\nOpened %s for writing...\n\n", fout);

fprintf (outp, "YIELDS BY YEAR AND FLEET\n\nYEAR"); // output file label
for (series=1; series<=nseries; series++)
  fprintf (outp, " Fleet %d", series);
fprintf (outp, "\n");

printf ("Writing yields...\n\n", fout);
for (y=0; y<PROJYRS+3; y++)
{
  fprintf (outp, "%4d", (y+lastyr-2));
  for (series=1; series<=nseries; series++)
    if (y < 3) fprintf (outp, "%11.0f", total_yield[y][series][0]);
}

```

Gray Trigger Updated Projections

```
        else      fprintf (outp, "%11.0f", total_yield[y][series][1]);
    fprintf (outp, "\n\n");
}

printf ("Writing status trajectories...\n\n", fout);
fprintf (outp, "F/F30 AND B/B20 BY YEAR\n\n"); // output file label
fprintf (outp, "YEAR      Fratio      Bratio\n");

for (y=2; y<PROJYRS+3; y++)
    fprintf (outp, "%4d      %6.4f      %6.4f\n", (y+lastyr-2), (avg_f[y]/F30),
(ssb[y]/SSB20));

fclose (outp);
printf ("Finished reading %s\n", fout);

} // end main
```