



A flexible, extensible interface between compiled languages and the statistics system **R**

Michael H. Prager
Erik H. Williams
Population Dynamics Team
NOAA Beaufort Laboratory

Presented within NMFS: March, 2004
Revised for Web: August, 2005



Preface

- This talk was given at U.S. National Marine Fisheries Service (NMFS) national stock-assessment meeting, 2004.
- *Stock assessment* is the modeling of a fish population to determine its status & the impacts of fishing.
- This version of the talk has been edited slightly for a general modeling audience.



Modeling methods

- Much of our modeling work is done with commercial package *AD Model Builder*.
 - Optimization and modeling package
 - Uses automatic differentiation
 - C++ code generator—makes compiled models
- Models tend to be large and time consuming.



Problem definition

- *Situation:* Modeling work in ADMB often results in poorly structured ASCII output.
- *Problem:* Assessment workshop participants can spend hours making diagnostic plots and tables of runs.
- *Needed:* A quicker way to examine model results.
- *Also:* A way to make figures and tables more expeditiously for reports.



Criteria for solution

- Minimal user effort
 - Solution as automated as possible
- Generality
 - Minimal reprogramming for new but similar stock assessments
- Open, multiplatform, archival
 - Platform-independent
 - Free of proprietary software
 - Using stable technologies
 - Compatible with ADMB, C++, Fortran

Solution has 3 elements

1. **Data writing** after stock-assessment model estimation is finished
2. **Data reading** by suitable graphics and analysis software (**R**)
3. **Graphics generation** from the data, both of—
 - Rapid onscreen display graphics
 - Optional graphics in file storage



Concepts of solution

- Save results to structured ASCII file
- Use **R** as graphics engine
 - Open-source version of **S** language
 - Provides graphics & analyses
- **R** graphics are—
 - modern, extensive, attractive,
 - programmable,
 - exportable to many formats

Data transfer...

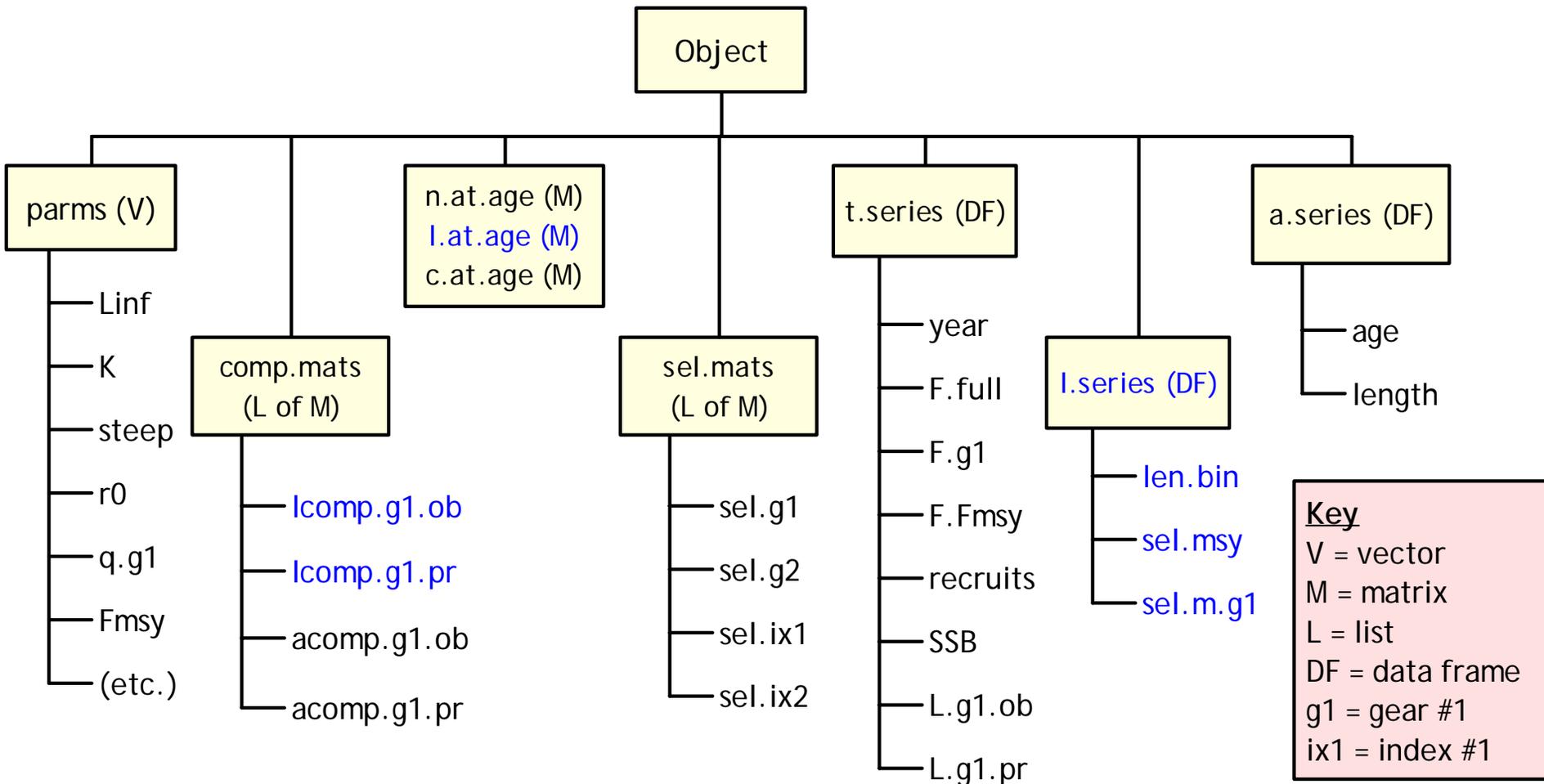
- Transfer data as an **R** object of type “list”
 - Arbitrarily complex list of items
 - Recursive (may contain other lists)
 - Items, row, columns may be named
 - Typical items:
 - Vectors (parameter estimates)
 - Matrices (numbers of fish at age)
 - Data frames (time series of abundance indices, catches, etc.)

Data transfer (2 of 2)

- Make ADMB write data in **R** “dput” format
 - Structured ASCII file
 - Contains **R** object(s)
 - Compatible with **R** and **S-Plus**
 - Entire data structure can be read into **R** with one function call:

> **myobj = dget("mydat.rdat")**

Sample R object structure (vermillion snapper)



Note: This was a length-structured stock assessment, unusual in fishery modeling. In a more typical age-structured assessment, the items in blue would not be present.

Implementation: 3 elements

- **Helper C++ functions**
 - Help write ADMB data to file in format expected by **R**
- **User code** in REPORT_SECTION of ADMB
 - Writes specific items needed
 - Repeatedly calls C++ functions
- **Set of generalized R functions**
 - Each fn makes group of related plots
 - Each searches for appropriate data in object
 - They make diagnostics and results plots
 - Optionally, export all plots & tables to files

Implementation: one trick

- Use standard names in **R** object.
 - `n.at.len`: n-at-length matrix (year × N)
 - `sel.x`: list of fishing-gear-selection matrices (year × proportion)
 - `lcomp.x.ob`, `lcomp.x.pred`: matrices of length compositions (year × length-bin)
 - `L.x.ob`, `L.x.pred`: annual vectors (length)
- Then, standard graphics functions work with most assessments.
 - **R** functions can find data elements that match patterns.

To use framework (1 of 3)

Include C++ in ADMB template

```
GLOBALS_SECTION
```

```
#include "admodel.h"
```

```
#include "mhp-s-funcs.cpp"
```



To use framework (2 of 3):

- Add code to REPORT_SECTION
 - Similar code for each assessment.
 - A recent assessment used 142 lines (309 including comments)
 - Partial example follows...

Writing some matrices....

```
// ----- START LIST OF LENCOMP and AGECOMP MATRICES -----  
// Open the list  
sfile << "comp.mats = structure(list(  
  
// Commercial HAL length comps  
write_s_matrix(sfile, value(obs_lenc_CHAL), "cm1", styr, endyr, 1, nlens);  
write_s_rownames_vec(sfile, len, 1, nlens, "ma", 0);  
write_s_matrix(sfile, value(pred_lenc_CHAL), "cm2", styr, endyr, 1, nlens);  
write_s_rownames_vec(sfile, len, 1, nlens, "ma", 0);  
  
...similar blocks omitted...  
  
// Close the list  
sfile << ")", .Names = c('lcomp.c.hal.ob', 'lcomp.c.hal.pr' [...])," <<  
endl;  
  
// ----- END LIST OF LENCOMP and AGECOMP MATRICES -----
```

To use (3 of 3):

- Run the ADMB model.
- Read data into **R** & make graphs.

```
> v41 = dget("vsnap41.rdat")
```

```
> go(v41)
```

- go() is user function that calls all the graphics functions for this type of analysis
- Preceding creates ~200 graphs in 10 seconds & optionally writes to files.



Demonstration

NOTE: A demonstration was run at this point in the presentation, generating 174 graphs.

To replace that, the following slide illustrates reading the data into R. Later slides show some typical graphs from a fish stock assessment.

The following graphs are illustrative only and do not show the status of any actual fish population!

```

> vs = dget("vsnap41.rdat")
> str(vs,max.level=1)
List of 9
 $ parms      : Named num [1:17] 405.875  0.406  1.000  57.992  56.988 ...
  .. attr(*, "names")= chr [1:17] "Linf" "K" "meanlen.R" "stdlen.a" ...
 $ like       :List of 3
 $ n.at.age   : num [1:39, 1:13] 2376790 2301320 2333560 2343830 2378800 ...
  .. attr(*, "dimnames")=List of 2
 $ n.at.len   : num [1:39, 1:51] 791778 768083 777122 780570 791721 ...
  .. attr(*, "dimnames")=List of 2
 $ sel.mats   :List of 7
 $ comp.mats  :List of 26
 $ t.series   : `data.frame':      39 obs. of  33 variables:
 $ l.series   : `data.frame':      51 obs. of  6 variables:
 $ a.series   : `data.frame':      13 obs. of  2 variables:
> print(vs$parms)
      Linf      K  meanlen.R  stdlen.a  stdlen.R  q.mm.cvt  q.mm.flr
4.05875e+02 4.05663e-01 1.00000e+00 5.79920e+01 5.69876e+01 1.16636e-06 3.47869e-07
      q.mm.hal  q.r.hb  q.r.mr  steep  r0  dmsy  msy
3.05897e-07 6.79697e-07 1.07751e-06 8.36112e-01 3.49529e+06 0.00000e+00 6.19725e+05
      Fmsy  eggs.msy  epr.FO
4.22093e-01 1.59009e+10 1.47459e+04
> names(vs$like)
[1] "values" "wgts" "names"
> dimnames(vs$n.at.age)
[[1]]
 [1] "1964" "1965" "1966" "1967" "1968" "1969" "1970" "1971" "1972" "1973" "1974" "1975"
 [13] "1976" "1977" "1978" "1979" "1980" "1981" "1982" "1983" "1984" "1985" "1986" "1987"
 [25] "1988" "1989" "1990" "1991" "1992" "1993" "1994" "1995" "1996" "1997" "1998" "1999"
 [37] "2000" "2001" "2002"

[[2]]
 [1] "0" "1" "2" "3" "4" "5" "6" "7" "8" "9" "10" "11" "12"

> names(vs$sel.mats)
[1] "sel.c.hal" "sel.c.twl" "sel.c.xxx" "sel.r.hb" "sel.r.mr"
[6] "sel.c.hal.disc" "sel.hb.disc"
> names(vs$a.series)
[1] "age" "length"
>

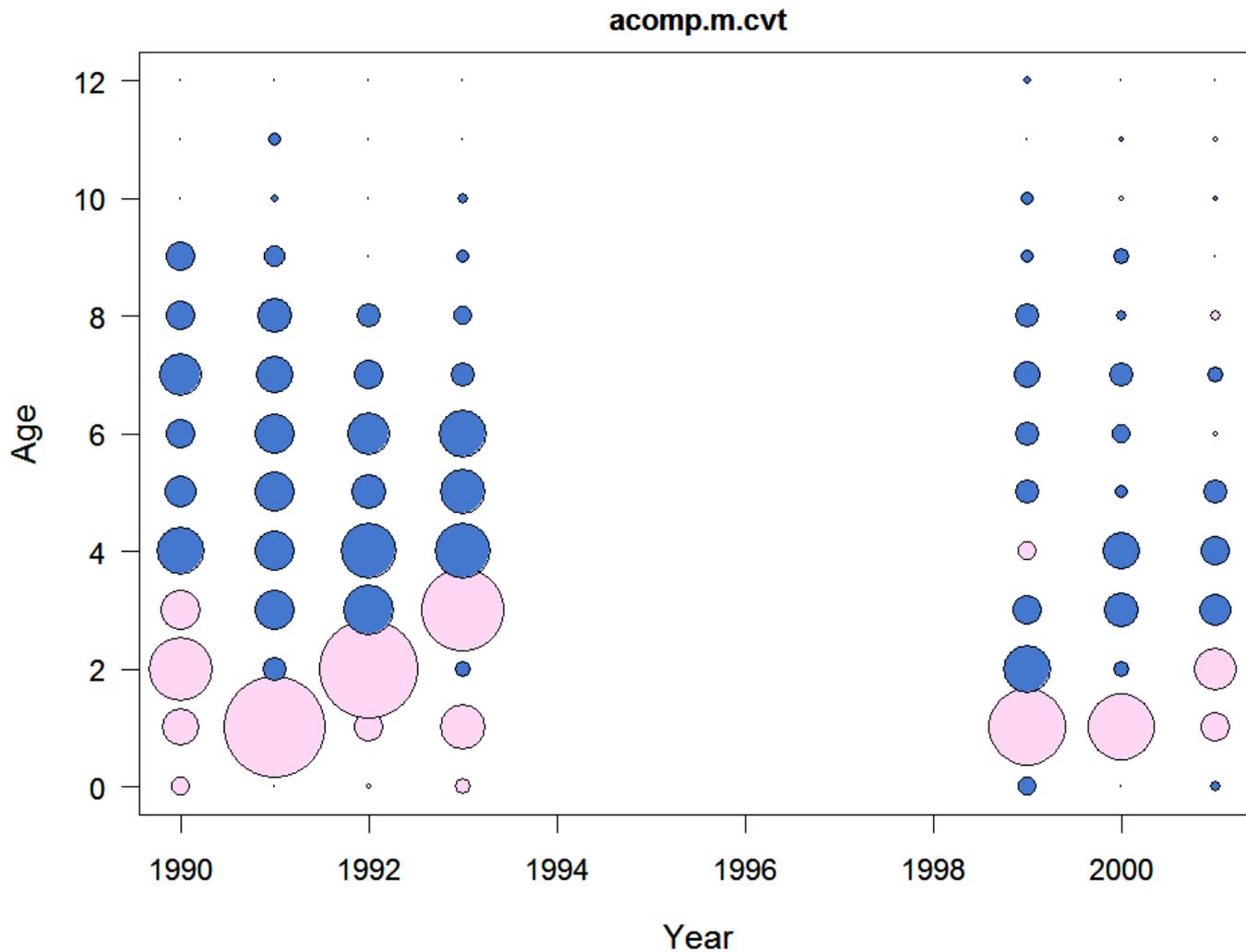
```

► The R console to the left shows reading a model output file & printing some information about it with standard R functions.

► The data object is read with the first line. The “str” function shows its components.

► The “parms” vector holds 17 named parameter estimates.

► The “n.at.age” matrix has yrs for its row names, ages for column names.

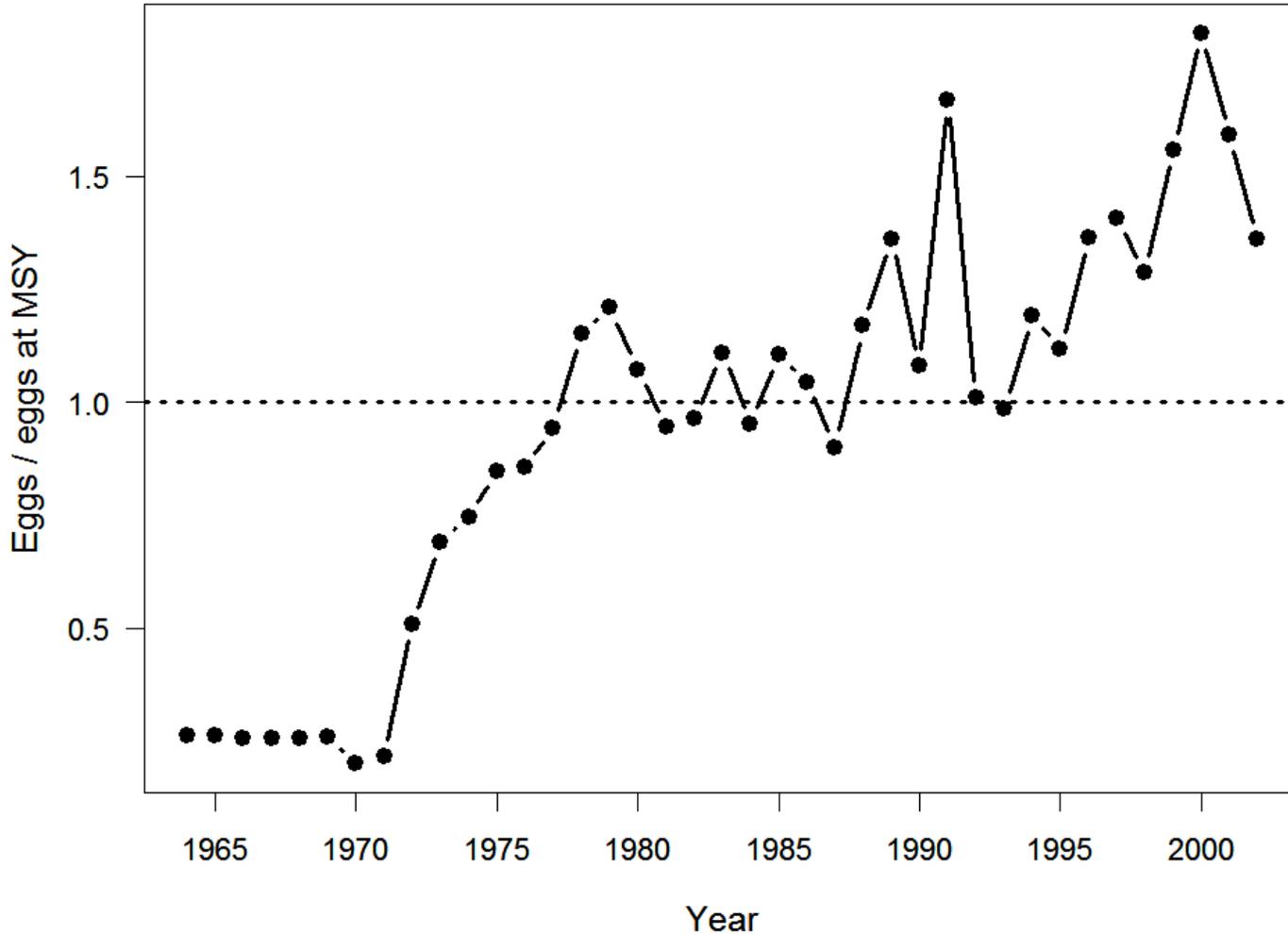


► The graph to the left shows residuals in fitting age distribution by year.

► One such graph is generated for each series of age-composition data found.

► Graph titles are generated automatically to match data found in the data object.

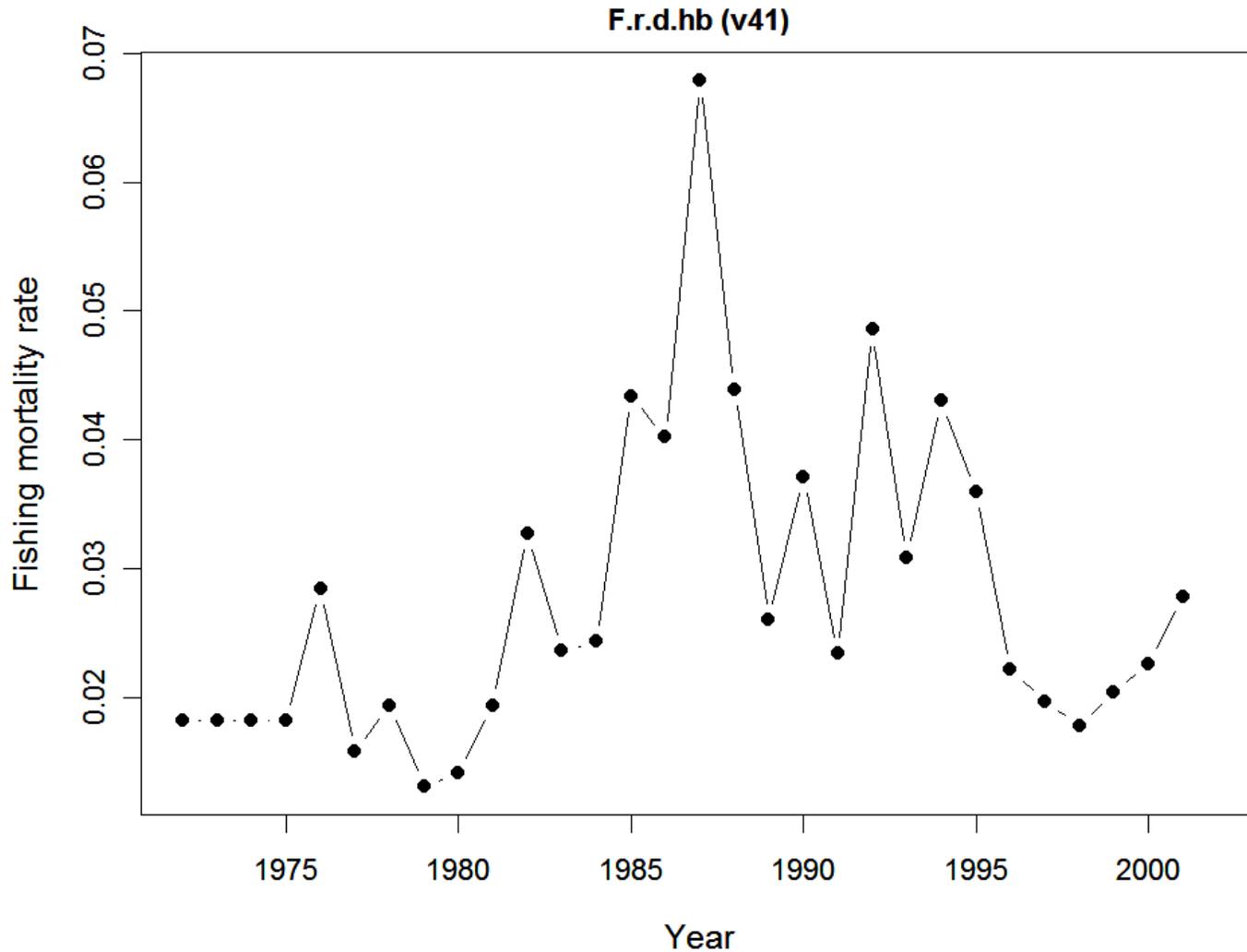
v41 Recruitment etc.



► The graph to the left shows time trajectory of the population's egg production (a measure of spawning stock).

► The trajectory is normalized to eggs at maximum sustainable yield.

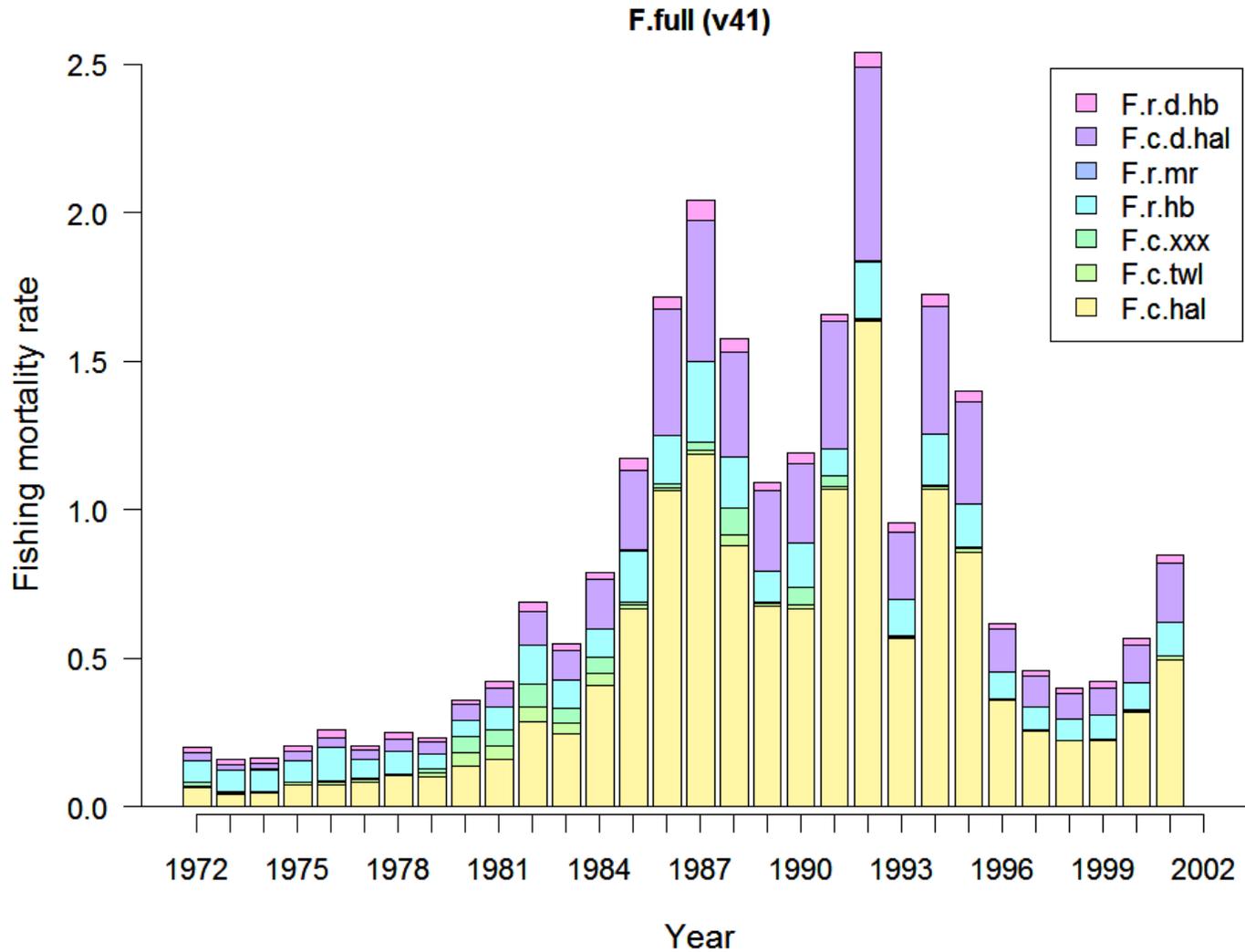
► A subdirectory named from the data object (v41-graphs) is created automatically to hold all graphics files.



► The graph to the left shows time trajectory of fishing mortality rate in the recreational headboat fishery.

► One such graph is made automatically for each fishery.

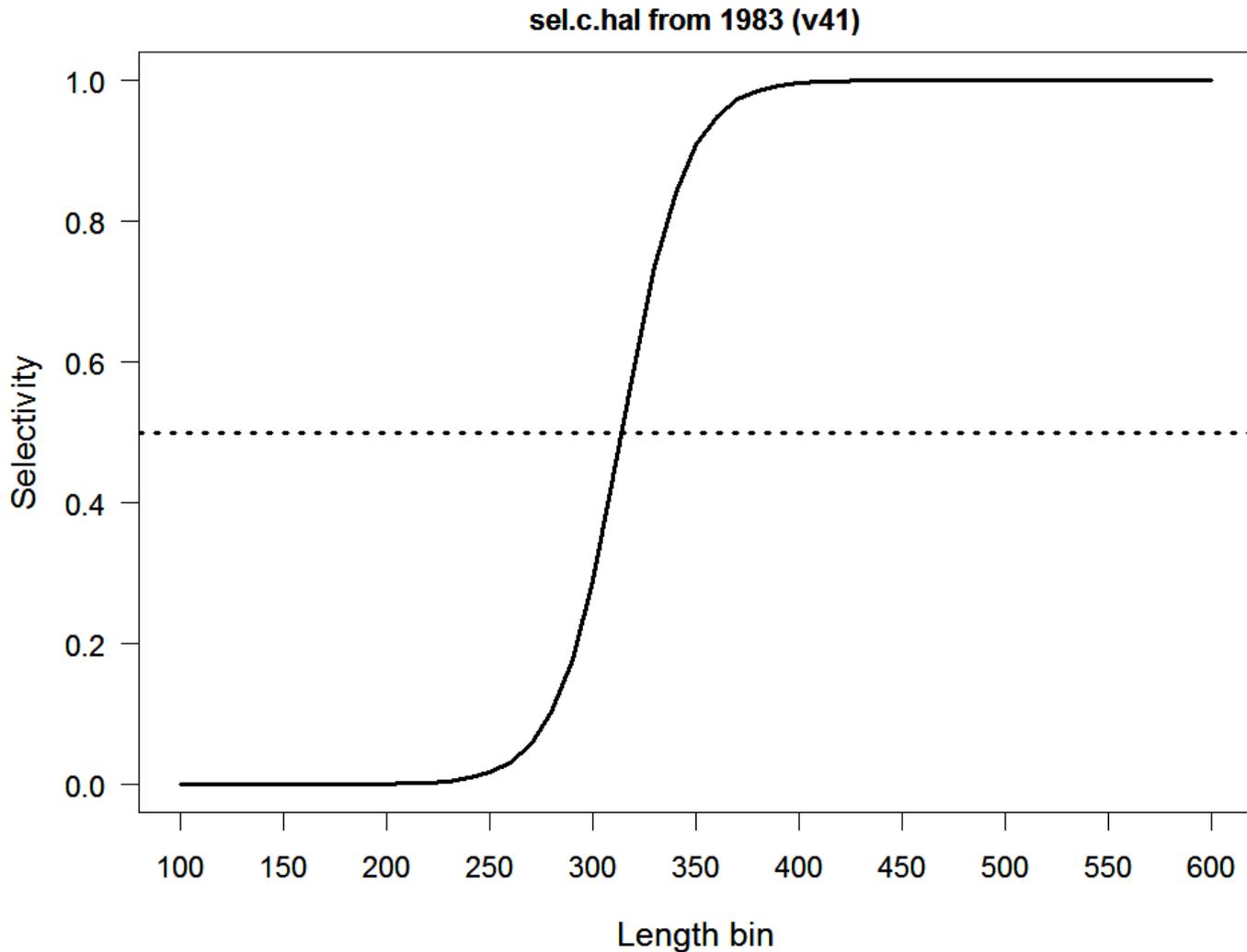
► All graphics files are named automatically similarly to their titles.



► The graph to the left shows time trajectory of fishing mortality rate in all fisheries.

► The legend is generated automatically, and shows fishery type and gear.

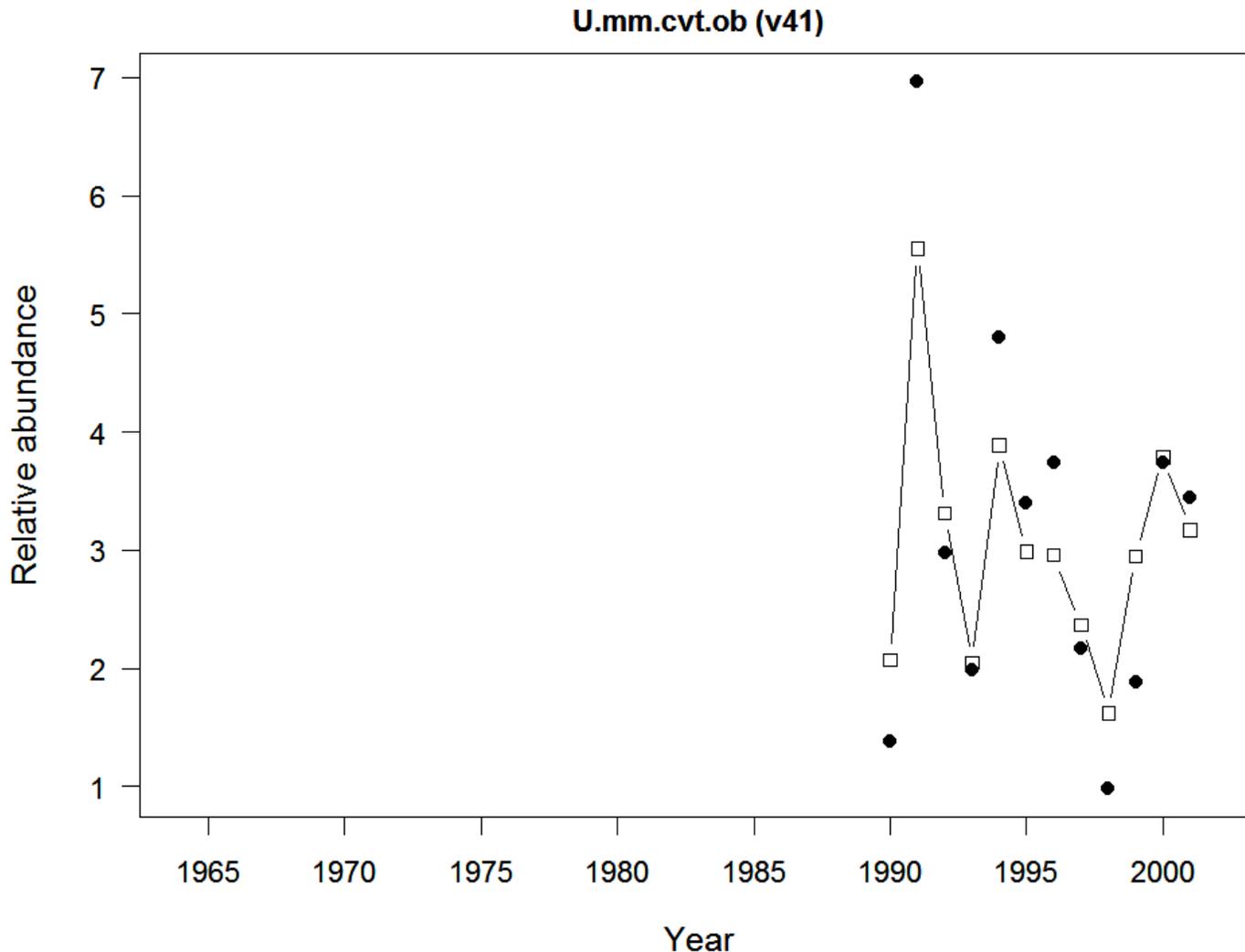
► All graphics files are saved in .eps and .png formats when saving is turned on.



► The graph to the left is estimated selectivity of the fishing gear (by fish length) in a commercial hook-and-line fishery.

► One such graph is generated for each selectivity curve estimated.

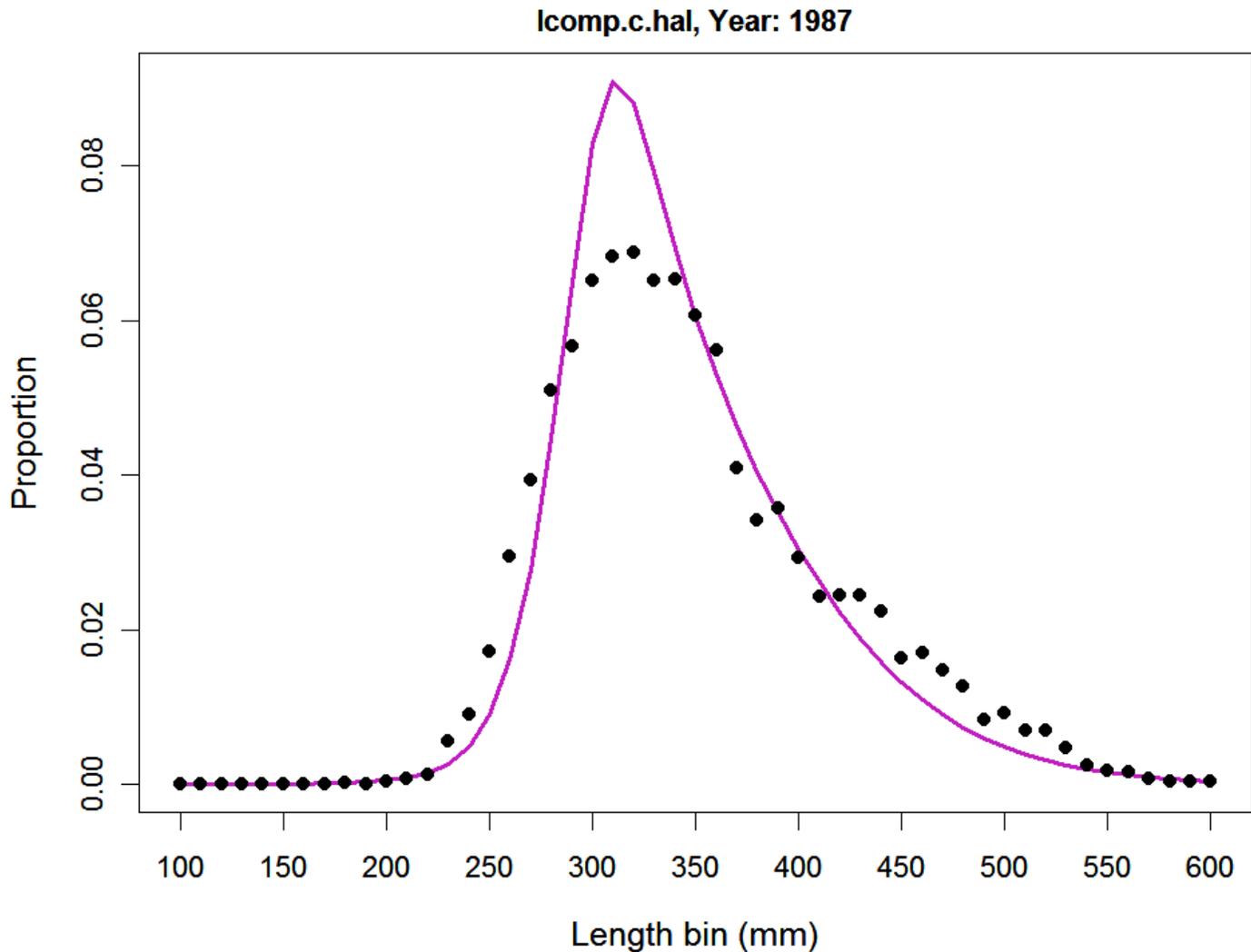
► As in other graphs, titles are generated automatically.



► The graph to the left shows estimated and observed (solid) relative abundance in a fishery-independent survey.

► One graph is generated for each abundance index used.

► As with other data, abundance indices are found within the data object automatically using R's `grep()` function.



► The graph to the left shows estimated (line) and observed (dots) length composition for one fishery–year combination.

► One such graph is generated for each combination.

► Graphs like this form over 50% of the generated graphs per model run.

Table 24. Black sea bass: Estimated abundance (numbers) at age

Year	0	1	2	3	4	5	6	7	8	9	10	11
1978	10522600	6831010	5506280	2812580	1209810	651957	268240	114552	47032	17696	6285	8826
1979	14868100	7764340	4785890	2892790	1431290	596668	318576	129875	55007	22437	8403	7144
1980	11665300	10781100	5267360	2344800	1332980	613628	243571	123684	48197	19696	7830	5308
1981	16551700	8377890	7159430	2423150	1001080	521482	223665	82554	39266	14528	5720	3699
1982	10126100	11796900	5578130	3575080	1091650	399091	190292	74539	25356	11305	3993	2491
1983	12499000	7273070	7691140	2264030	1325300	363994	123976	54982	20178	6518	2800	1558
1984	10552000	9249540	5358220	3934970	977988	544329	149467	50908	22577	8286	2677	1790
1985	10804900	7809770	6813200	2072290	1194120	277982	154676	42472	14466	6416	2354	1269
1986	7508180	7996000	5753610	3053760	749010	404048	94032	52322	14367	4893	2170	1226
1987	7357640	5555400	5892310	3157060	1416550	326495	176078	40978	22801	6261	2132	1480
1988	8882310	5446010	4095710	3025630	1381460	593723	136818	73786	17172	9555	2624	1514
1989	7306930	6571630	4008240	1626650	921627	389671	167415	38579	20806	4842	2694	1167
1990	8057600	5402470	4829010	1508230	434058	216488	91482	39304	9057	4885	1137	906
1991	5067850	5950930	3964520	2099770	425026	98588	49125	20759	8919	2055	1108	464
1992	5100770	3743210	4369840	1510300	492937	76764	17786	8862	3745	1609	371	284
1993	4851310	3768190	2751620	1844580	406186	102961	16016	3711	1849	781	336	137
1994	9863940	3583960	2771730	1197230	527273	95101	24085	3747	868	433	183	110
1995	6581610	7279250	2629800	934009	223835	72705	13096	3317	516	120	60	40
1996	8149480	4853550	5329740	723985	124426	20421	6622	1193	302	47	11	9
1997	5431190	6013300	3560890	1922820	115531	10844	1775	576	104	26	4	2
1998	5366830	4009420	4416400	1437290	430531	17521	1642	269	87	16	4	1
1999	5720080	3966710	2952760	2215380	472113	103962	4225	396	65	21	4	1
2000	4394370	4237510	2925510	2023300	498939	32119	7017	285	27	4	1	0
2001	6578780	3255410	3128770	2035590	519873	71735	4602	1005	41	4	1	0
2002	5590490	4873640	2399370	2110440	238402	29360	4035	259	57	2	0	0
2003	6099440	4141520	3598230	1658530	459731	16295	1991	274	18	4	0	0
2004	6406360	4518540	3053150	2418930	207048	24407	861	105	14	1	0	0

► The table at left is from a recent final report.

► It was generated as LaTeX code from the data object by the R “Hmisc” package by Frank Harrell.

► We are now automating production of such tables.

► Tables can be output as HTML for MS- Word users.



End of demo...

- That concludes the demonstration of typical graphs & tables generated.
- Thank you for your attention.
- The talk itself continues for several more slides!



Status in March, 2004

- We use this framework regularly.
- All code is well commented, though not well documented.
- It's a good prototype, not a finished product.



Extensions since March, 2004

- Graphics routines were modified to make final graphs for reports.
- New graphs were programmed in the **R** language for specific needs.
- **R** used to generate LaTeX tables (using Harrell's Hmisc package).
- Output-data object is used for initializing projections written in **R**
 - **R** includes a complete matrix language.



Further developments

- We are attempting to formalize the framework so that it can be distributed and used by others within and outside NMFS.
- Contact MHP for more information:
<http://shrimp.ccfhrb.noaa.gov/~mprager/>



New helper routines in C

- A nicer set of C helper routines is being written as of Aug 2005.
 - Work by experienced C programmer.
 - Part of NMFS assessment toolbox.
 - Will make it easier to use this framework for new ADMB or C models.
 - Will not require hand coding of punctuation, etc., by the analyst.
 - Expected completion date: fall, 2005.

Fortran functions written (For2R)

- A set of Fortran-95 routines has already been completed, called For2R.
 - They allow writing an **R** object from any Fortran modeling code.
 - They use Fortran 95, a superset of Fortran 77.
 - Free Fortran 95 compilers are now available.
- This allows output-compatibility of Fortran and ADMB models.
 - Common graphics routines can be shared.
- Fortran source is available from MHP.

Summary (1/2)

- Our framework provides quick diagnostics for stock-assessment runs.
- It also makes data available in **R** for
 - Projections
 - One-off graphs
 - Other statistical analyses
- The time tradeoff is positive.
 - Several hours' setup per model
 - Savings of hours per model run
- This time savings allows us to have many additional diagnostic views.

Summary (2/2)

- Use of **R** gives great flexibility and automation ability.
- Use of “dput/dget” file format allows storing complex data, with attributes such as names and labels, in a single file.
- Helper subroutines for writing files give easy integration w/ ADMB, C++, Fortran.
- This combination provides a flexible, extensible solution to postprocessing and storage of model output.