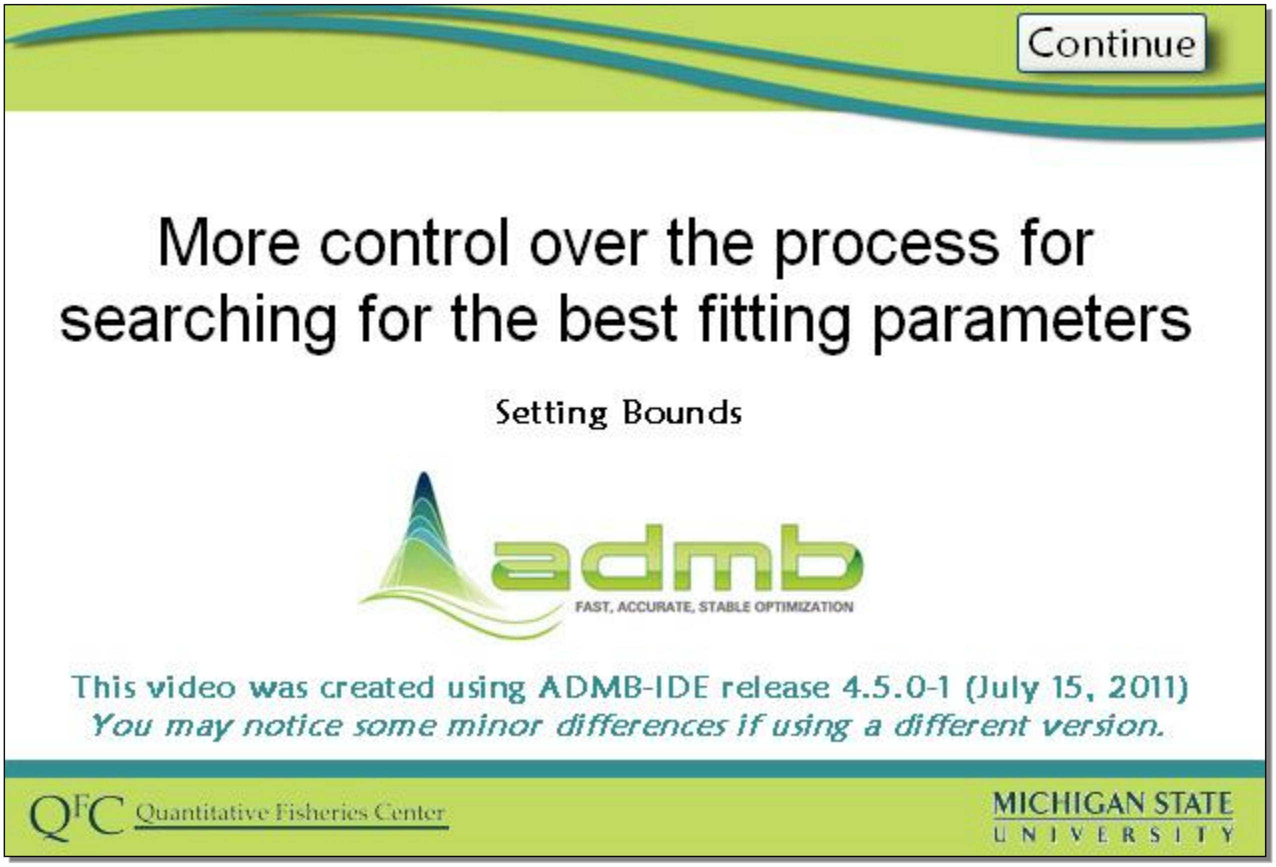


Continue

More control over the process for searching for the best fitting parameters

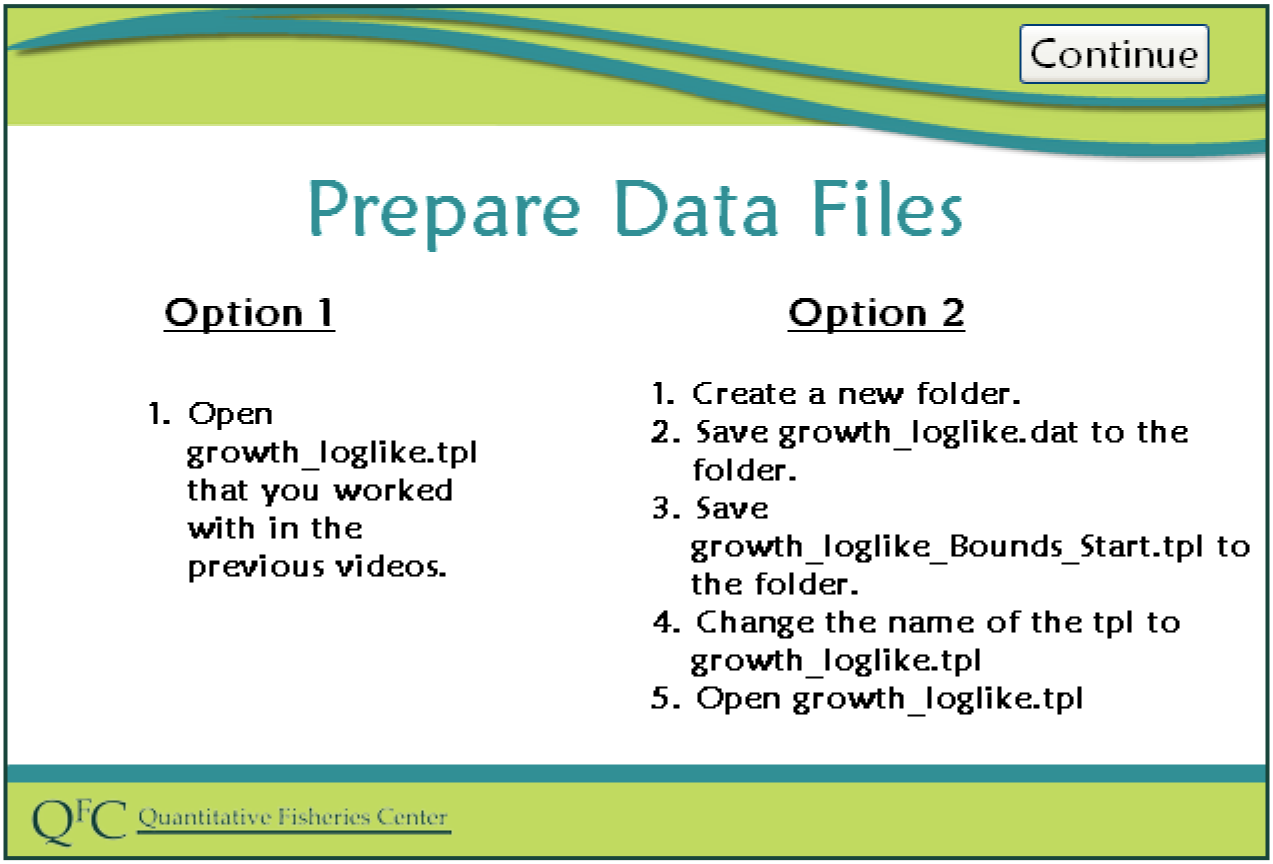
Setting Bounds

admb
FAST, ACCURATE, STABLE OPTIMIZATION

This video was created using ADMB-IDE release 4.5.0-1 (July 15, 2011)
You may notice some minor differences if using a different version.

QFC Quantitative Fisheries Center
MICHIGAN STATE UNIVERSITY

In this video you will learn some ways to better control the process of searching for the best fitting parameters used by AD Model Builder. When you set bounds AD Model Builder does not look at parameters above or below the bounds during the search process. This can help avoid regions of parameter space that are unreasonable, but are hard to escape from once entered.



Prepare Data Files

Option 1

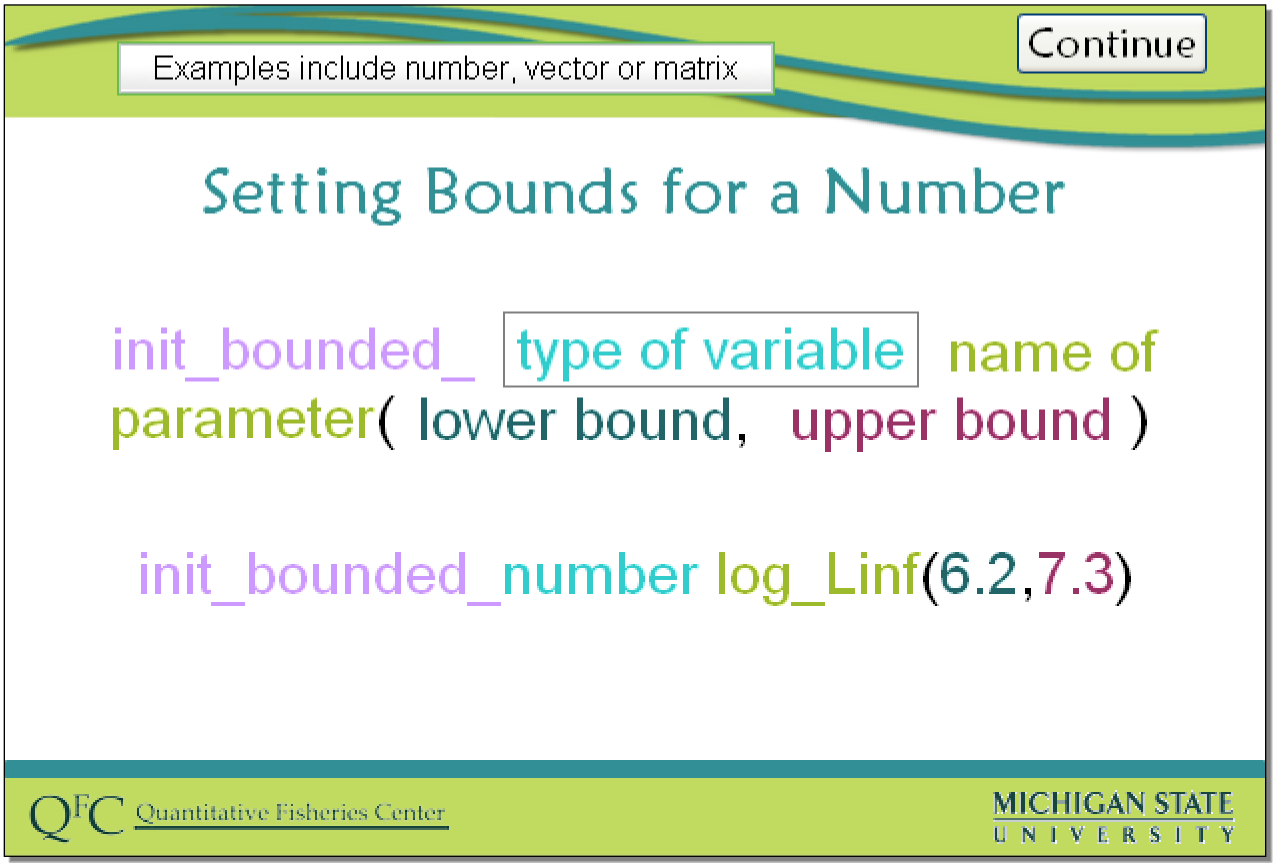
1. Open growth_loglike.tpl that you worked with in the previous videos.

Option 2

1. Create a new folder.
2. Save growth_loglike.dat to the folder.
3. Save growth_loglike_Bounds_Start.tpl to the folder.
4. Change the name of the tpl to growth_loglike.tpl
5. Open growth_loglike.tpl

Prepare and open your files. Click continue when you are ready.

1. Either open growth_loglike.tpl that you worked with in the previous videos or Create a new folder.
2. Save growth_loglike.dat to the folder.
3. Save growth_loglike_Bounds_Start.tpl to the folder.
4. Change the name of the tpl to growth_loglike.tpl
5. Open growth_loglike.tpl



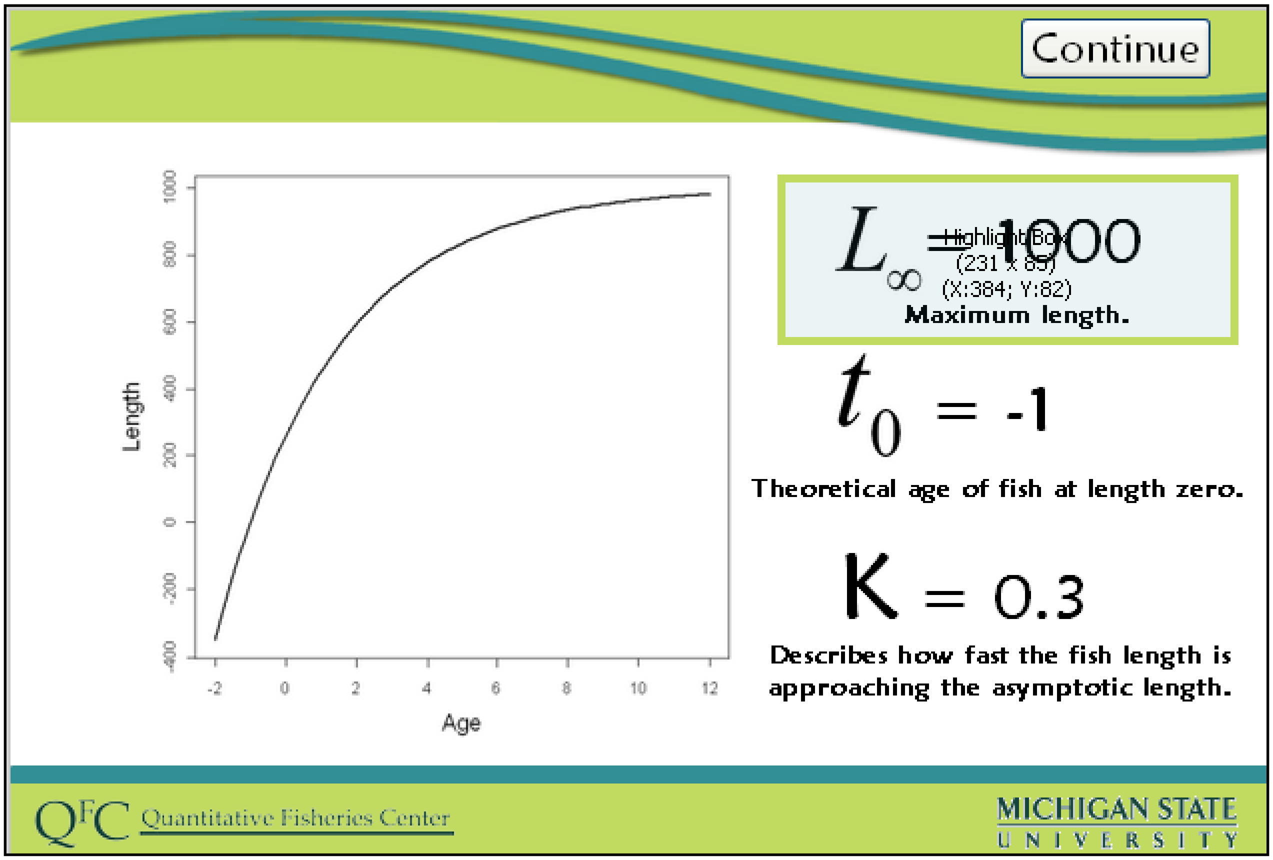
Examples include number, vector or matrix

Setting Bounds for a Number

init_bounded_ type of variable name of parameter(lower bound, upper bound)

```
init_bounded_number log_Linf(6.2,7.3)
```

The basic syntax and approach for specifying bounds for parameters is straightforward and can be applied to scalars, vectors, and matrices. First you specify that the estimated parameter is bounded by starting the name with “init_bounded” you then complete the definition by the type of variable, just as in defining a regular parameter but you follow the name of the parameter, with parentheses inside which you give the lower and upper bounds. For example, here we specify the lower and upper bounds for the parameter log_Linf which is the natural log of asymptotic length.



$L_\infty = 1000$

Maximum length.

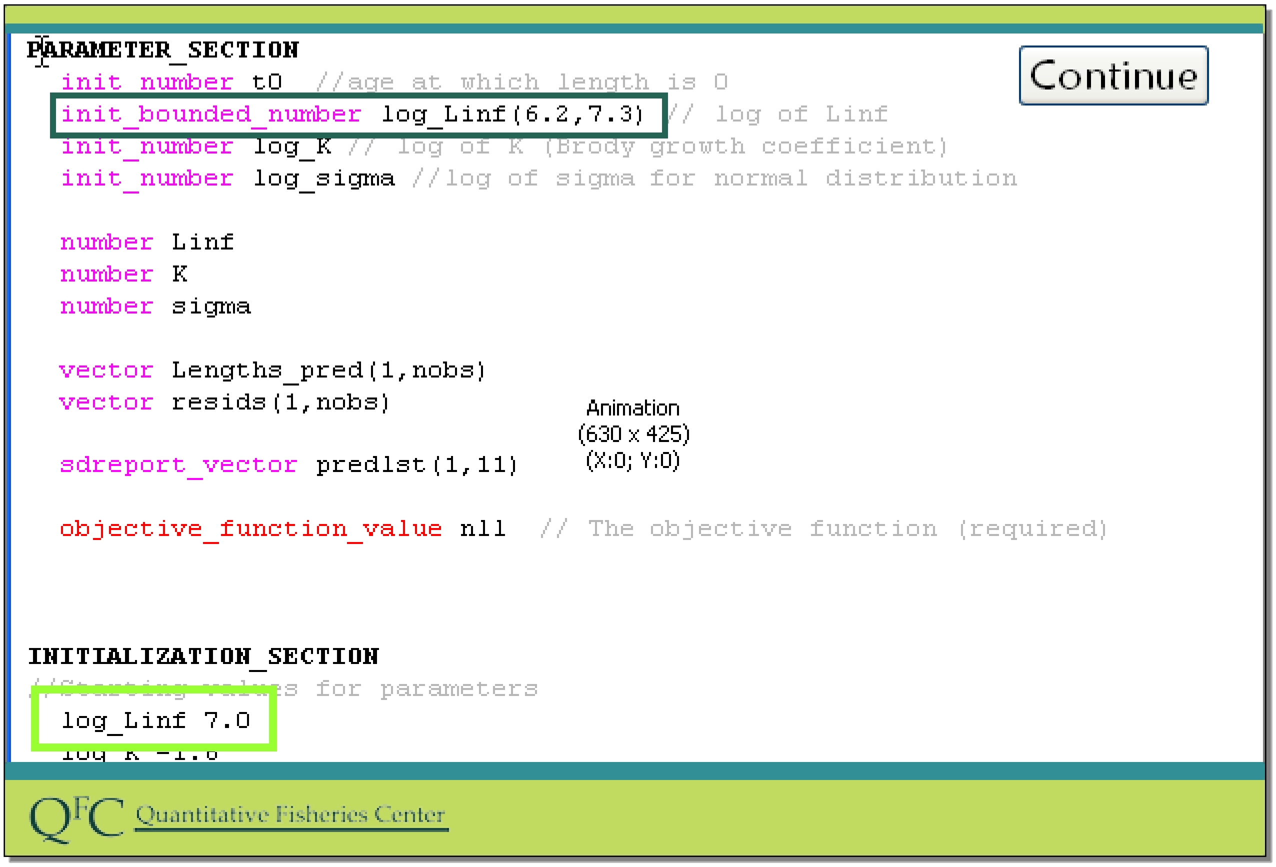
$t_0 = -1$

Theoretical age of fish at length zero.

$K = 0.3$

Describes how fast the fish length is approaching the asymptotic length.

Next we will set bounds on the working template named growth_loglike.tpl that we developed and used in the previous videos. But first recall we used the von Bertalanffy function for expected length given age, with these parameters. We are going to add bounds to constrain the L-infinity parameter between 500mm and 1500mm. Taking natural logs the gives us lower and upper bounds for the log scale parameter of 6.2 and 7.3.



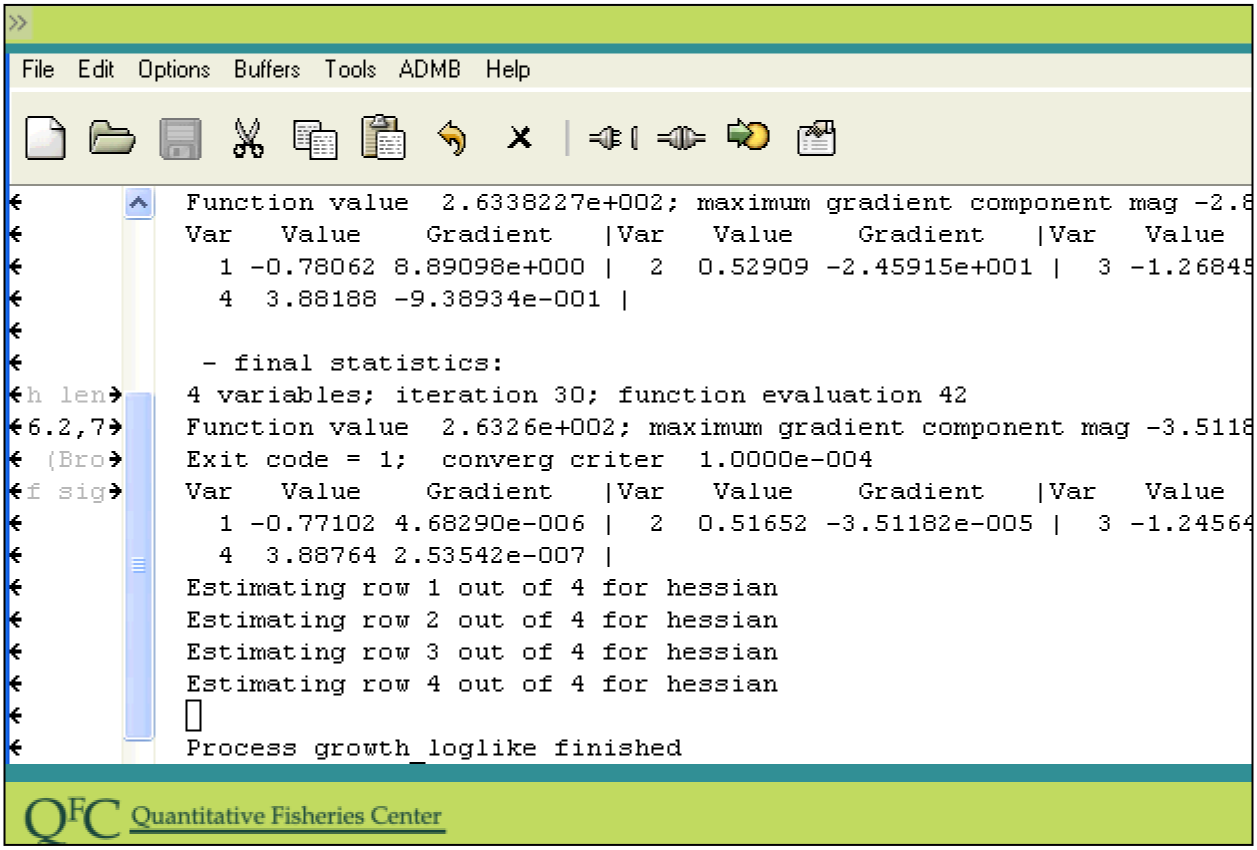
```
PARAMETER_SECTION
  init_number t0  //age at which length is 0
  init_bounded_number log_Linf(6.2,7.3) // log of Linf
  init_number log_K // log of K (Brody growth coefficient)
  init_number log_sigma //log of sigma for normal distribution

  number Linf
  number K
  number sigma

  vector Lengths_pred(1,nobs)
  vector resids(1,nobs)

  sdreport_vector predlst(1,11)

  objective_function_value nll  // The objective function (required)


INITIALIZATION_SECTION
//Starting values for parameters
  log_Linf 7.0
  log_K -1.6
```

1. So we will open the existing tpl file and
2. scroll to the PARAMETER_SECTION where are parameters are defined
3. then modify the log_Linf parameter to be bounded with 6.2 as the lower bound and 7.3 as the upper bound.

When using bounds you need to be careful to make sure that the starting value for a parameter is within any bounds you set. Otherwise unexpected behavior can result. We are ok here because the starting value is 7.0.

Slide Code:

Init_bounded_number log_Linf (6.2,7.3)

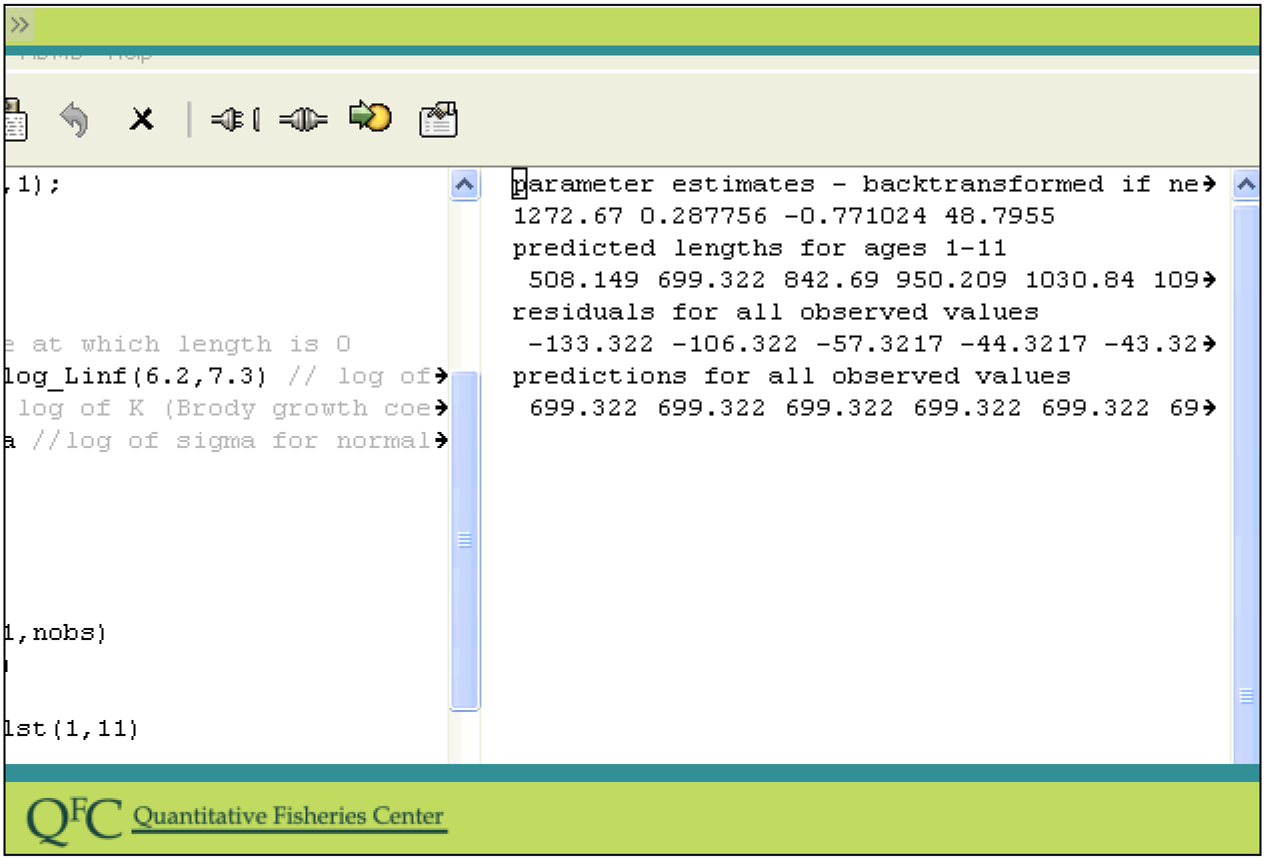


```
Function value  2.6338227e+002; maximum gradient component mag -2.8
Var    Value     Gradient    |Var    Value     Gradient    |Var    Value
  1 -0.78062 8.89098e+000 |  2  0.52909 -2.45915e+001 |  3 -1.26845
  4  3.88188 -9.38934e-001 |

 - final statistics:
4 variables; iteration 30; function evaluation 42
Function value  2.6326e+002; maximum gradient component mag -3.5118
Exit code = 1;  converg criter  1.0000e-004
Var    Value     Gradient    |Var    Value     Gradient    |Var    Value
  1 -0.77102 4.68290e-006 |  2  0.51652 -3.51182e-005 |  3 -1.24564
  4  3.88764 2.53542e-007 |
Estimating row 1 out of 4 for hessian
Estimating row 2 out of 4 for hessian
Estimating row 3 out of 4 for hessian
Estimating row 4 out of 4 for hessian

Process growth_loglike finished
```

1. Then we build
2. and run the program.

Note that the values shown on the screen for parameters with bounds are actually transformations of your original parameters. It is these transformed values that are what is actually changed during the parameter search. The main thing is not to worry when you see values on the screen that are outside the bounds you set.

Slide 13 - Slide 13

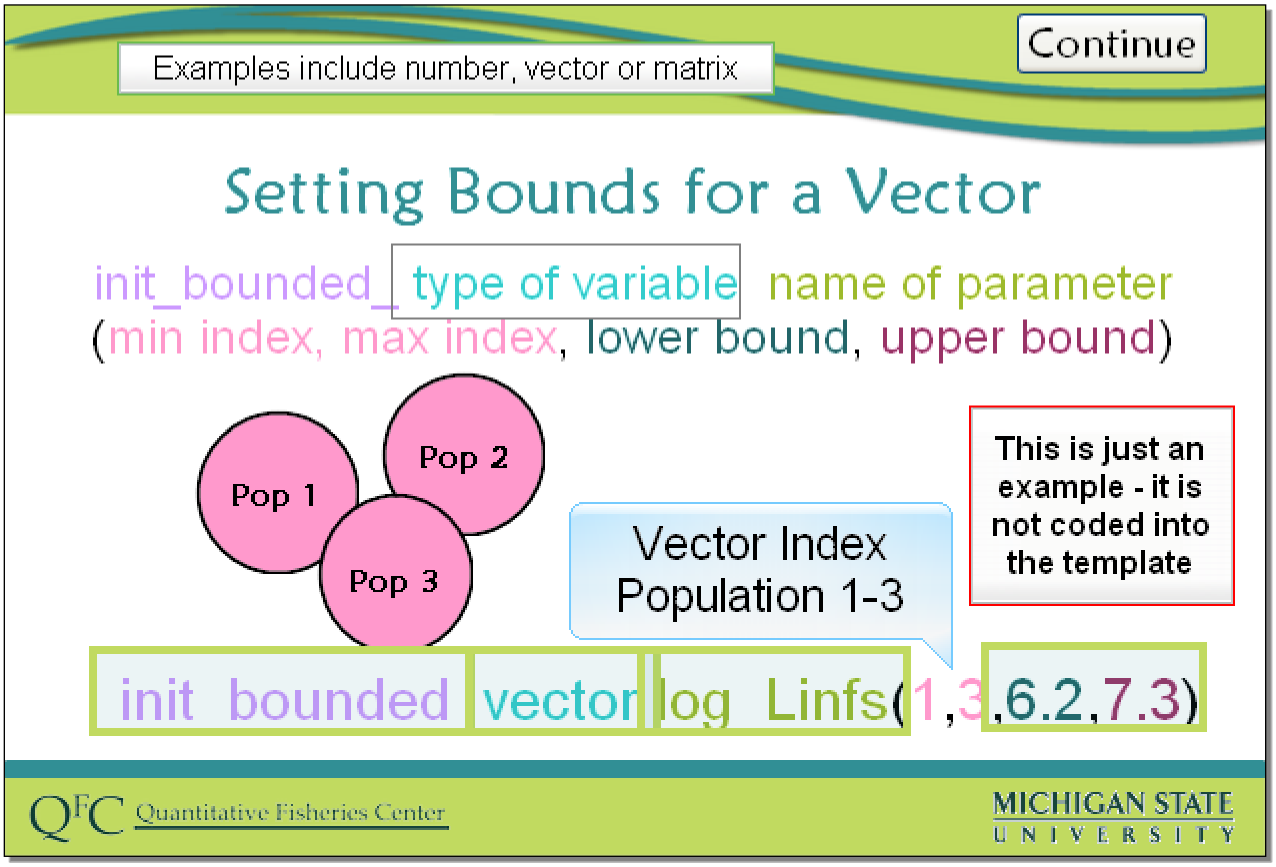


```
,1);



e at which length is 0
log_Linf(6.2,7.3) // log of
 log of K (Brody growth coe
a //log of sigma for normal





1,nobs)


lst(1,11)
```

```
parameter estimates - backtransformed if ne
1272.67 0.287756 -0.771024 48.7955
predicted lengths for ages 1-11
 508.149 699.322 842.69 950.209 1030.84 109
residuals for all observed values
 -133.322 -106.322 -57.3217 -44.3217 -43.32
predictions for all observed values
 699.322 699.322 699.322 699.322 699.322 69
```

1. Go to the **ADMB** menu
2. To **view report**

A quick peek at the results in the report file shows that this works. We won't dwell on the results because this is a case where things were already working ok before we constrained estimates to be between bounds. Our resulting estimate is not at one of the bounds. If it was we would need to either reconsider the bounds or fix some other problem that is causing the model to move toward implausible estimates. Some users make it a practice to always provide bounds on parameters. This may avoid numerical problems during searches and it can be especially useful if ultimately you plan to use Bayesian approaches, because by specifying bounds you are implicitly specifying as uniform distribution as a proper prior distribution. If you are unfamiliar with Bayesian approaches don't worry about this potential additional benefit.



Examples include number, vector or matrix

Setting Bounds for a Vector

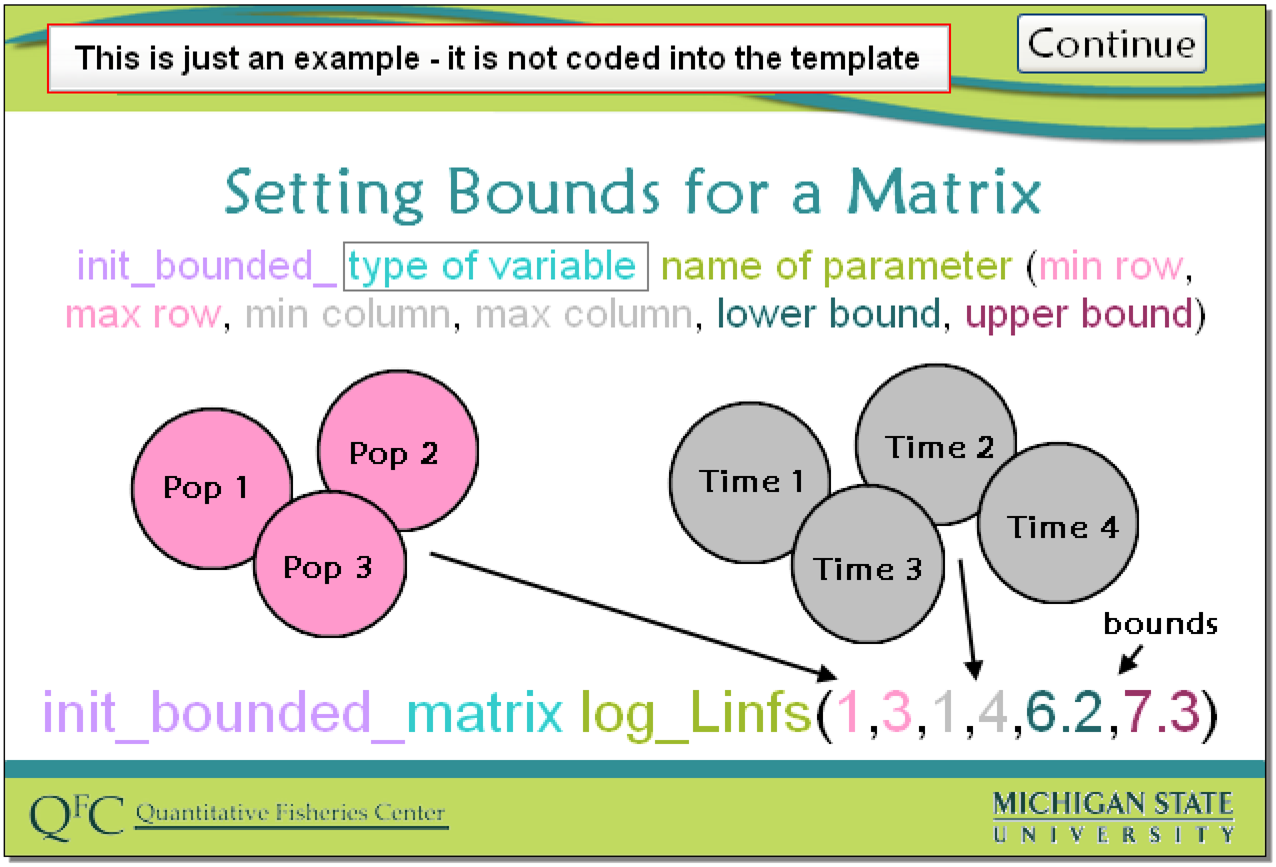
init_bounded_ type of variable name of parameter
(min index, max index, lower bound, upper bound)

Vector Index
Population 1-3

This is just an example - it is not coded into the template

```
init_bounded_vector log_Linfs(1,3,6.2,7.3)
```

Now we will show you how to created bounded versions of vectors of parameters. For example maybe we are fitting growth models for three populations with different asymptotic lengths. Then we might define a bounded vector of parameters. Note after the parameter name, log_Linfs, you see the index of the vector. Since we have 3 populations, our first two numbers would be 1 and 3 to index the population of 1-3. Then follows the lower and upper bounds.



This is just an example - it is not coded into the template

Setting Bounds for a Matrix

init_bounded_type of variable name of parameter (min row, max row, min column, max column, lower bound, upper bound)

init_bounded_matrix log_Linfs(1,3,1,4,6.2,7.3)

Now assume we have different time periods and different populations, and each combination could have a different asymptotic length. We might then have a matrix of parameters with columns representing time periods and rows indicating populations.