

Adding constraints

- SURGE and linear models

Up until now, we've used SURGE to test fairly simple hypotheses. We've seen how to use SURGE to help with model selection (Chapter 4), and how the process of model selection can be viewed in an analysis of variance context, by comparing models with and without grouping factors (Chapter 5).

- However, suppose, for example, you want to test the hypothesis that annual variation in survival (which you've previously determined to be significant) correlates with some weather variable. How would you do this analysis? Intuitively, you might decide to take your estimates of survival (from SURGE, or some other source) and simply use a correlation analysis to see if survival rate and "weather" are related. Unfortunately, such an approach is not valid statistically, since the annual estimates of survival are not independent (due to sampling covariances among years).
- The correct way to test this hypothesis is to use an extension of the basic ideas we've covered so far. In fact, you might be able to correctly guess the basic logic of what you would do: fit a model where survival is a function of time and weather, and compare it with a LRT to the fit of a model where survival simply varies over time. In other words,

$$\Phi_{f(weather)} \text{ VERSUS } \Phi_{f(time)}$$

- More specifically, you would compare the fit of a time-dependent model to the fit of a time-dependent model where the estimates of survival from this model are **constrained** to be a linear function of weather. The concept of "constraints", and how to use them to apply linear models to SURGE, is one of the most important and powerful extensions of what we have covered so far.

- What do we mean by "constraint"? In the presence context, we are referring to a mathematical constraint - "forcing" SURGE to estimate survival and recapture rates after imposing a specific set of *linear* constraints on the structure of the *underlying* model.
- In our example, the underlying model was a time-dependent model (for example, a CJS model). To this model, we applied a specific linear constraint - we forced the time-dependent survival rates to be estimated as linear functions of weather. It is important to note that variations in survival over time as a function of weather is a particular "case" of time variation - and as such, the model where temporal variation in survival is modelled as a function of weather is, in fact, nested within the general model where survival varies simply with time. Recall that such nesting between models is necessary to use a LRT.
- While this example is very simple, you can use SURGE to apply *any* linear model, however complex, as a constraint on estimation of survival and/or recapture. In fact, *any* ANOVA-type analysis (additive models, nested ANOVA etc) can be expressed in terms of linear models. Thus, if you can conceive of a linear model (ANOVA, ANCOVA, multiple regression etc), you can apply it to mark-recapture data using SURGE. The only thing you'll need to know is how to construct the "constraint" - the linear model.

Review of Basic Linear Models

- If you have a background in linear models, then much of this material will be familiar. If you're a statistician, obviously we're leaving out a lot of the "details" (to say the least). Our purpose is to provide a minimum level of background, so even newcomers to linear models have a "feel" for the approach (however, if you *are* new to linear models, we **strongly** suggest you look at one of the many good textbooks on this subject, especially any material on dummy variables in regression. Neter, Wasserman & Kutner (1985) and Kleinbaum, Kupper & Müller (1988) are particularly good. If nothing else, we hope you'll see that, once the linear model has been constructed, implementing it in SURGE is fairly straightforward.
- First, to construct linear models using SURGE, and many other

software applications, the **qualitative** or **categorical variables** (i.e., the “grouping” variables - often referred to in textbooks as the “classification” factors) in the model must be “coded” by a series of “0” (zero) or “1” values. Consider an example where a skull circumference of young pre-school children is measured, and we’re interested in knowing if this structure is on average larger in males than in females (we’ll assume for the moment that all of the children were the same chronological age). Let’s suppose we measure 7 male and 7 female children, and analyze our data using a normal single-classification ANOVA. Here is the data set.

male	7.2	7.1	9.1	7.2	7.3	7.2	7.5
female	9.8	8.5	8.7	8.6	8.4	7.7	8.2

- First, the results from a “standard” ANOVA.

Source	df	SS	MS	F	P
SEX	1	3.806	3.806	8.33	0.0137
Error	12	5.485	0.457		
Total	13	9.292			

- In this example, we see that there is a significant difference between our two groups (males and females) in skull circumference.
- However, what if our statistics package was limited only to a regression subroutine? Could we have analyzed our data using a linear regression model, instead of ANOVA, and arrived at the same result? The answer is, indeed, yes, we can. What we do is simply take the classification factor (SEX) and “code” it as a “0” or “1” dummy variable. For example, let “0” represent females, and “1” represent males. Thus, every individual in our data set is assigned a “0” or a “1”, depending upon their gender. Let’s call this dummy variable SEX.

Now, all we need to do is regress our response variable (the skull circumference) on SEX. Here are the results of the regression analysis:

Source	df	SS	MS	F	P
SEX	1	3.806	3.806	8.33	0.0137
Error	12	5.485	0.457		
Total	13	9.292			

- No, it’s not a typo - it is in fact the exact same table as in the left-hand column. The two approaches are entirely synonymous, yielding identical results.
- How can this be? The answer lies in the structure of the models actually being tested. In the first case, where we used ANOVA, we were using the following linear model - in ANOVA (single-classification, Model I), any single variate Y can be decomposed as follows:

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

- Each variate is the sum of the global mean, the deviation due to the classification factor, and the random error. In our example, with 2 groups (males and females), we are testing for differences of the type $\alpha_1 - \alpha_2$. If $\alpha_1 - \alpha_2 = 0$ (the null hypothesis), then we conclude no significant group effect (i.e., no significant difference in group means between the sexes).
- In contrast, in our regression approach, each variate Y can be decomposed as:

$$Y_i = \beta_1 + \beta_2 x_i + \epsilon_i \quad (\text{for } i=1,..n)$$

- In this case, each variate is the sum of product of the slope (β_2) and the variable x, the intercept (β_1), and a random error term (ϵ). In this case, the hypothesis being tested is whether or not the estimate of the slope is significantly different from 0 ($H_0: \beta_2=0$). This is undoubtedly familiar.
- But, what is the factor “x”? In our case, it is the dummy coding for SEX (“0” for females, “1” for males). We regress the response variable on SEX. If the slope is not different from 0, then we interpret this as evidence that the numerical value of the dummy variable does

not significantly influence variation in our data. In other words, no difference in slope from 0 means no difference between the sexes.

- As it turns out, it is possible to analyze **any** ANOVA-type design using the combination of regression models and “dummy” coding for the different categorical or grouping factors (although interpretation of interactions in linear models is often more complex). In the examples we will explore in this chapter, you will learn the basic steps of creating these linear “dummy variable” models, and how to use them with SURGE to test a variety of hypotheses.
- The only thing we now need to consider is - how can we use “regression models” for analysis of mark-recapture data, since both survival and recapture are not “normal” response variables - normal in the sense that they are both constrained to be values from 0 → 1? If you simply regressed “live=1/dead=0” or “seen=1, not seen=0” on some set of explanatory variables **x**, it is quite conceivable that for some values of **x** the estimates value of survival or recapture would be >1 or <0, which clearly can't be correct! However, we clearly want to be able to bring the full power of ANOVA-type analyses to bear on capture-recapture studies.
- The way around this problem is to transform the probability of survival or recapture, such that the transformed probabilities have been mapped from [0,1] to [-∞,+∞], which is of course the “assumption” for normal linear regression models. To accomplish this, SURGE defaults to what is known as a “**logit link**” or “**logit transform**”. The logit transform is frequently used in analysis of probabilities, and has been shown to have good properties. However, other transforms are possible, and may be more appropriate under some circumstances (see Chapter 9). For the remainder of this chapter, however, when we refer to a “*linear constraint*” or “variable *x* is a *linear* function of variable *y*”, we are referring to a *logit linear* relationship.
- Using these transformed probabilities, we can use linear regression models analogous to the one we just considered in the skull circumference example. We will now consider a simple example in detail, to demonstrate how linear models are constructed using SURGE.



What is the “logit link function”, and how does it work?

Suppose you want to express a dichotomous (i.e., binary) response variable **Y**, such as survival or recapture, as a function of one or more explanatory variables. Let **Y**=1 if alive or present; otherwise **Y**=0. Let **x** be a vector of explanatory variables, and $p = \Pr(Y=1|\mathbf{x})$ is the probability of the response variable you want to model. We can construct a linear function of this probability by using a logit transform of the probability, *p*.

$$\log \text{it} (p) = \frac{e^{a + bx}}{1 + e^{a + bx}} = \ln \left(\frac{p}{1 - p} \right) = a + bx$$

where **α** is the intercept, and **β** is the vector of slope parameters. In other words, we can express the probability of the event (survival or recapture) as a linear function of a vector of explanatory variables.

The logit (or logistic) model is a special case of a more general class of linear models where are function $f = f(\mu)$ of the mean of any arbitrary response variable is assumed to be linearly related to the vector of explanatory variables. The function **f** is the “link” between the random component of the model (the response variable) and the fixed component (the explanatory variables). For this reason, the function $f(\mu)$ is often referred to as a “link function”.

What SURGE does is to estimate the intercept and vector of the slope parameters, using the logit link, and then reconstitutes the values of ϕ or *p* from the values of the explanatory variables, **x**. It does this in 2 steps: (1) first, SURGE reconstitutes estimates of ϕ or *p* from **α**, **β** and **x**, and then (2) SURGE computes values of ϕ or *p* from **f** using the back transform **f**⁻¹. Further details of the implementation of the logit link function in SURGE can be found on p. 77 of Lebreton *et al.* (1992).

The European Dipper - the effects of flood

- We return to our analysis of the European dipper data set. Now, we will examine the effects of a specific climatic event (flood conditions on the breeding ground) on survival and recapture estimates.

- As you may recall from Chapter 3 and Chapter 4, this data set involves 7 occasions of mark recapture, of both male and female individuals. In those earlier chapters, we focussed on the males exclusively. In this chapter, we'll reanalyze this data set including both males and females.
- Each year in the study was characterized by the presence or absence of flood conditions. Are years with high (or low) values of either survival or recapture (or both) associated with years when there was a flood? Does the relationship between flood and either survival or recapture differ significantly between male and female dippers?
- In order to address these questions, we will use the following "logic sequence":

<i>Step 1</i>	• Is there a significant interaction of sex and the covariate (flood) on variation in either survival or recapture?
<i>Step 2</i>	• if there is no interaction, then is there a significant difference between the sexes in survival?
<i>Step 3</i>	• in the absence of interaction, is there any evidence of a significant linear relationship between survival (or recapture) and the covariate (flood)?

- This is the basic sequence of steps used in analysis of covariance (ANCOVA). This is a very basic (and hopefully familiar) analytical design in statistical analysis, and we will demonstrate that the very same approach can be used to analyze variation in survival or recapture.
- Before we begin, let's recast our analysis in terms of linear models. For the moment, let's use the simplified expression of linear models used in earlier chapters. Our basic model, including our classification variable (SEX) is

$$\phi \text{ (or } p) = \text{SEX} + \text{FLOOD} + \text{SEX.FLOOD} + \text{error}$$

- One thing you might ask at this point is - why isn't TIME included in the model? The answer lies in the fact that when we talk about constraints, we are speaking about "applying" a constraint to a particular starting model. For example, we could start with the standard CJS model, with time-dependence of both survival and recapture rates. Then, we could apply a specific constraint to this model. In this example, we replace the 'random' effect of time in the CJS model by the 'specific' temporal effect of FLOOD. FLOOD is a particular case of time-dependence, because years with the same flood condition will share the same survival value. Thus, models with the FLOOD factor are 'nested' in the corresponding CJS model with the 'random' TIME factor. Of course, FLOOD, just like TIME, can be crossed (interaction) with SEX.
- Our first step in this analysis is to test for the significance of the interaction term: SEX.FLOOD. If the interaction term is not significant, we can proceed to test for the significance of the other factors. How do we test for significance of the interaction term? Clearly, we test the model with the interaction against the same model without the interaction - using a LRT. The significance of the difference in fit of these two models is a formal test of the significance of the interaction term.

$$\phi \text{ (or } p) = \text{SEX} + \text{FLOOD} + \text{SEX.FLOOD} + \text{error}$$

versus $\phi \text{ (or } p) = \frac{\text{SEX} + \text{FLOOD}}{\text{SEX.FLOOD}} + \text{error}$

- How do we do this? The basic mechanics are the same as were described in Chapters 4 and 5 - we use SURGE to fit the 2 models we want to compare. But, in this case, there is a subtle difference; although the SEX term clearly corresponds to the 2 groups on our analysis, how to we incorporate the information specified by the FLOOD variable? In other words, how do we "constrain" our estimates for either sex to be a linear function of flood?
- To apply a linear constraint to a particular model using SURGE, you will need to construct 2 different files: the "VAR" file, and the "PAR" file. The VAR file is the file which contains the structure and elements of the linear model. The PAR file will contain the index numbers of the

parameters you want to apply the constraint to.

VAR file	<ul style="list-style-type: none"> the file containing the VARIables, and the model structure, of the linear constraint
PAR file	<ul style="list-style-type: none"> the file containing the index values of the PARameters in the model you want to constrain using the model specified in the VAR file.

- So, the first thing we need to do is learn how to create our starting model - using the VAR and PAR files. For the moment, let's concentrate on modeling the survival in terms of SEX and FLOOD. Here are the steps you need to follow:

Step 1 - specifying the parameters you want to constrain (the PAR file)

- The first thing you need to do is decide on your starting, underlying model. For example, your starting model might include simple time-dependence in both parameters - the CJS model. How do you decide on the starting model? By using the techniques discussed in the receding chapters, and the GOF procedures outlined in the Appendix. Remember - you *apply a constraint to a particular underlying (or starting) model* - if this model doesn't adequately fit the data, then applying a constraint will not yield a particularly powerful test.
- For the moment, let's assume that the CJS model is a good (and valid) starting model.
- Once you've determined the starting model, you need to determine the parameter indexing that SURGE will use. For the Dipper data set, we have 2 groups, and 7 occasions. Thus,

survival

1	2	3	4	5	6
	2	3	4	5	6
		3	4	5	6
			4	5	6
				5	6
<i>males</i>					6

7	8	9	10	11	12
	8	9	10	11	12
		9	10	11	12
			10	11	12
				11	12
<i>females</i>					12

recapture

13	14	15	16	17	18
	14	15	16	17	18
		15	16	17	18
			16	17	18
				17	18
<i>males</i>					18

19	20	21	22	23	24
	20	21	22	23	24
		21	22	23	24
			22	23	24
				23	24
<i>females</i>					24

- For the moment, we're concentrating on survival, so the parameters we're interested in are in the top "survival" matrices. Thus, we want to constrain 12 parameters: 6 survival estimates for males, and 6 for females.
- Now, all we need to do is create the PAR file which contains these parameter indexes. Let's call the file FLOOD.PAR. Using the extension .PAR will help you remember what is contained in the file.
- Using your favorite editor, create a simple text (ASCII) file containing the following:

```
12
1 2 3 4 5 6 7 8 9 10 11 12
```

- This is all there is in the FLOOD.PAR file for this model. The first number, "12" in the first line of the file, is the number of parameters

which will be constrained. It is NOT the value of the last parameter (the fact that 12 is also the value of the last parameter in this example is coincidental).

- The lines after the first line contain the index numbers of the parameters themselves - each entered in turn, with one or more spaces separating them. The only rule is that each parameter index must be separated from others by a space. Otherwise, you can put them all on a single line, or on multiple lines. The order of the index values must match that of the rows of the VAR file (as we will see later). SURGE is very flexible in its ability to handle varying file structures. The only thing that SURGE requires is that the first line of the PAR file have the total number of parameters, and that the index numbers be separated by at least one space.

Step 2 - defining the model structure of the linear constraint (the VAR file)

- There is nothing particularly difficult about building the VAR file, once you've figured out the correct dummy-variable structure for the linear model you want to fit. Simple designs tend to be very easy, more complex designs obviously less so.
- Think back to our earlier comparison of a simple ANOVA versus a simple regression analysis of skull measurements of a sample of pre-school children (p. 6-2). Recall that the first step in the process of translating the ANOVA analysis into the equivalent linear regression model was to "code" the classification factor (SEX) as either "0" (for females) or "1" (males).
- In this example, we have 2 variables of interest in our model: SEX, and FLOOD. Clearly, we can again code SEX as either "0" or "1". As noted in Lebreton *et al.* (1992), FLOOD is a "yes or no" variable: there is either a flood in a given breeding season, or there isn't. Thus, we can also code FLOOD using a simply "0" or "1" dummy variable system. In this study, the flood occurred during the 2nd and 3rd breeding seasons only.
- We need to build a file that has dummy variables to characterize the SEX, and the FLOOD state, in each of the years. There are 7 occasions, and therefore 6 survival parameters, for each sex.
- Start by writing out terms of the linear model to the right of the equal

sign, and then underneath, write in the appropriate dummy variables for each cell in the table. We'll start with males.

<i>males</i> (year)	SEX	FLOOD	SEX.FLOOD
1	1	0	0
2	1	1	1
3	1	1	1
4	1	0	0
5	1	0	0
6	1	0	0

- The top of the table is simply the model (without the "+" signs). Along the left side of the table are the intervals or years after the first occasion. If we remember that males are indicated by the dummy variable "1", then it is clear that the SEX column in the table is simply a column of the number "1".
- Presence of a FLOOD is indicated by "1", and no flood by "0". If the flood occurred only during the second and third years of this study, then this explains the sequence of dummy variables in the FLOOD column in the table.
- The final column is the interaction of SEX and FLOOD (SEX.FLOOD). The values in this column are derived by simply multiplying the dummy variable for SEX by the dummy variable for FLOOD (we will see what to do if our classification and covariate variables have more than one column later on). Since the value for the SEX dummy variable for males is "1", then clearly the SEX.FLOOD column will be exactly the same as the FLOOD column alone.
- What about for females? Well, all that really changes are the values in the SEX column, and as a result, the values in the S.F interaction column. The table for females would be:

<i>females</i> (year)	SEX	FLOOD	SEX.FLOOD
1	0	0	0
2	0	1	0
3	0	1	0
4	0	0	0
5	0	0	0
6	0	0	0

1	0	0
1	1	1
1	1	1
1	0	0
1	0	0
1	0	0
0	0	0
0	1	0
0	1	0
0	0	0
0	0	0
0	0	0
0	0	0

- Since the sex column is now all “0”, the interaction column will also be a column of “0”s, regardless of what is in the FLOOD column.
- Now, in our analysis, we are looking at both sexes together, so in our VAR file, we need to include the dummy structures we’ve just created for both males and females. So, again using your favorite editor, create a file called FLOOD.VAR, and put in the male and female dummy variable structures (i.e., recreate the two preceding tables in a single file). It should look something like the box at the top of the opposite column.
- Note that we do not include the “labels” - i.e., we don’t write the word “SEX” or “FLOOD” over the corresponding column. We also do not include the left hand column of the number of the year.

- Each column is separated from each other column by 1 or more spaces. The order of the columns does not matter. It is also valid to leave a blank line or two in the file separating the male and female “parts” of the file (this may help you keep track of what is going on). However, the order of the rows **must** correspond to the order in which the constrained parameters are listed in the PAR file.
- Are we finished with the VAR file? Almost. Just like the PAR file, we also need to add a first line which “tells” SURGE something about “how many elements” are in the file. In the VAR file, the first line has 2 parts: the number of occasions times the number of groups, and the number of elements in the model. In this example, we have six occasions we’re estimating over for each sex, so $6 \times 2 = 12$. We have 3 elements in the model (sex, flood and the interaction). So, the first line of FLOOD.VAR would be: “12 3”
- Finally, on the 2nd and 3rd line, we have to add a pair of control characters - 2 “\$” signs, one on each line. Why? Don’t ask! All you

need to know is you have to add these two “\$” signs, or SURGE won’t properly read in the VAR file. The final version of the FLOOD.VAR file then looks like:

12	3	
\$		
\$		
1	0	0
1	1	1
1	1	1
1	0	0
1	0	0
1	0	0
0	0	0
0	1	0
0	1	0
0	0	0
0	0	0
0	0	0

- You might find it helpful to note that the values on the first line (“12 3”) also refer to the number of rows and the number of columns (following the “\$ \$” signs), respectively.
- Also, SURGE does not require you to use the file extensions .PAR or .VAR. We simply find that it is a useful way to keep track of what the files contain.
- That’s it! You’ve now created the 2 files needed to constrain the CJS model. All that is left is to learn how to apply these constraints using

SURGE.

Step 3 - Running SURGE with constraints

- We are now ready to run SURGE. For this example, have SURGE send the output to the file FLOOD.LST.
- We shall begin by fitting the general model without constraints. We’re going to use CJS - full time-dependence (i.e., model $\phi_{s,t}p_{s,t}$), as our starting point. We have interactions of sex and time for both parameters.
- We have 2 groups, contained in the files MALES.SUR, and FEMALES.SUR, respectively.
- At this point you should be pretty comfortable with handling the model specification screens. We clearly want full time dependence, so we select choice 2, hit the <enter> key, and hit the <enter> key again to accept the default “full dependence” condition.
- Now, think back to what we did in Chapter 5. We are fitting model $\phi_{s,t}p_{s,t}$. So, what do we answer to the question concerning “Do you want the same parameter values across groups?”. The presence of sex in the interaction in the model implies we want SURGE to estimate survival (and recapture) separately for each sex. So, we answer “no”. If this is confusing, go back to Chapter 5.
- We want the same parameter structure (time-dependence) for both sexes, so we answer “yes” to the next question. The same sequence of answers is also used for the recaptures.
- Next, the question concerning the constraints. At this stage, we are not going to apply our constraints, so we simply hit the <enter> key to accept the default condition (“0”).
- Next, the question concerning fixing parameters. Again, we don’t want to fix parameters for this model, so we accept the default of “0” by hitting the <enter> key.
- By now, you’ve run through SURGE enough times you might realize that from this point on, we can usually bypass the remaining questions by simply hitting the <enter> key. In this book, we will never be changing the starting estimates, or the link function (until Chapter 10), so you can simply jump to the screen where we run the analysis by

hitting the <enter> key 4 times in succession - each time accepting the default.

- Run the analysis. When finished, select choice 2 to return to the model selection part of the program.
- Now, we want to enter a new title. Rather than using our short-hand to represent the model in terms of “phi” and “p” - simply enter something informing you that you are now going to add the linear constraint. One thing you might use is the linear model itself. For example: “Constrained CJS time-dependent model by SEX: model = SEX FLOOD S.F”. This will give you sufficient information in your output file to help you “remember” what you’ve done. As you’ll see, it will get progressively more difficult to simply “figure out” from the output what is going on without a useful title.
- Again, we come to the model specification menus. Here is one of the first things you must keep in mind: we are applying the constraint to a particular model. The constraint itself is a model, but one we are imposing upon another, underlying model. Basically, we fit a model with and without these constraints, and see which one fits better (of course, taking into account the differences in the number of parameters).
- In this example, we’re using the CJS time-dependent model as our “starting point” - as our underlying model. So, we use precisely the same choices and answers for the survival and recapture screens as we just did. In other words, for both survival and recapture, we select choice 2 (for time-dependence), then answer “no” to the question concerning identity of parameters over groups, and “yes” to using the same parameter structure.
- Now, we come to the constraints screen. We’ve obviously seen this screen many times before, but in general have either bypassed it, or used option “-1” to look at the model structure on-screen. Now we are going to use it to actually apply the constraints to our CJS model. To do this, we need to tell SURGE how many constraints we want to add. SURGE allows you to apply multiple constraints (for example, one set of constraints to survival, and a different one to recaptures). For the moment, we’re only applying our constraints once, to the survival model, so we enter the number “1”, and press <enter>.
- SURGE will now start asking you a series of questions concerning the

files where the constraint model(s) is stored: the PAR and VAR files you constructed earlier.

- First, SURGE asks you for the name of the file with the parameters that are to be constrained. This is obviously the FLOOD.PAR file you created so enter “FLOOD.PAR” <enter> (Fig. 6.3).

```

C* CONSTRAINING PARAMETERS

C* There are already 0 constraints
C* HOW MANY MORE CONSTRAINTS? <display model=-1; back to M=-2> : 1
C* CONSTRAINT # 1
  C* FILE OF PARAMETERS TO BE CONSTRAINED?      : flood.par
  
```

Fig. 6.3

- Next SURGE asks you how many variables there are in the constraint. In other words, how many terms are there in your constraint model (or, if you prefer, how many columns in your VAR file do you want to use?). In this example, there are 3 elements or terms in the constraint (SEX, FLOOD and the S.F interaction), so at most we would enter the number “3” here. However, as we will see shortly, it is not always necessary to enter the maximum number of terms here. Why? As a hint as to what is to come, if you enter the number “3” at this stage, we are telling SURGE that we want to use ALL of the terms in the model in the constraint.
- For the moment, we do want to use the full constraint model (SEX + FLOOD + S.F), so we enter “3”, and press <enter> (Fig. 6.4).

```

C* CONSTRAINING PARAMETERS

C* There are already 0 constraints
C* HOW MANY MORE CONSTRAINTS? <display model=-1; back to M=-2> ? 1
* CONSTRAINT # 1
  C* FILE OF CONSTRAINED PARAM. LIST <PAR= <---> ? flood.par
  C* NUMBER OF VARIABLES IN CONSTRAINT ? 3
  
```

Fig. 6.4

- Next, SURGE will ask you for the name of the variable file. It also

notes that the file must be in BIOMEKO format. Not to worry. The FLOOD.VAR file you created is in BIOMEKO format (thanks in part to the magical “\$ \$” signs). So, we enter “FLOOD.VAR”, and press <enter>. SURGE then asks if you want to use the logistic transformation. We do not want to change the transformation, so we accept the default condition by pressing the <enter> key (Fig. 6.5).

```

C* CONSTRAINING PARAMETERS

C* There are already 0 constraints
C* HOW MANY MORE CONSTRAINTS (display model=-1; back to M=-2) ? 1

* CONSTRAINT # 1
  C* FILE OF CONSTRAINED PARAM. LIST (PAR= <—>) ? flood.par
  C* NUMBER OF VARIABLES IN CONSTRAINT ? 3
  C* VARIABLE FILE (BIOMEKO FORMAT) (VAR=<—>) ? flood.var
  C* LOGISTIC TRANSFORMATION (YES=<—>)?
  
```

Fig. 6.5

Index	Estimate
1	0.696970
2	0.423077
3	0.505288
4	0.609402
5	0.570818
6	0.763763
7	0.742857
8	0.446841
9	0.453813
10	0.640424
11	0.628045
12	0.692820

- You’ve now applied the constraint to the CJS model! After hitting the enter key, SURGE will next ask you if you want to fix any parameters. Since we don’t want to at this stage, we simply start hitting the <enter> key until we’re at the screen where we run the analysis. Run the analysis, and then when finished, quit SURGE by selecting choice “5” from the final menu.
- Now, let’s look at the output in FLOOD.LST, to see what has happened.
- The first model we fit was the underlying CJS time-dependent model. Immediately after the title, SURGE prints out the parameter structure for survival and recapture, for both males and females. These should look the same as the matrices on p. 6-5.
- Next, SURGE gives you the estimates for survival and recapture, for the CJS model. SURGE lists the estimates in order of the parameter index value. For example, for survival, SURGE lists the estimates in the following manner (we’ve edited out the 95% CI and standard errors of the estimates for brevity).

- It is important to remember that the male estimates are indexed 1 →6, and the female estimates are indexed 7 → 12. Also, since this is a CJS model, remember that the final estimates of survival for both sexes are not estimable separately from the corresponding final estimates of recapture rate (only the product $\phi_6 p_7$ for males and females, respectively, is identifiable).
- The deviance for this model is 653.951.
- How many parameters are estimated? For each sex alone there are 11 parameters: 5 survival parameters, 5 recapture parameters, and one β term = 11 parameters. Thus, for the 2 sexes together, a total of 22 estimated parameters. Therefore, the AIC for this model is $653.951 + 2*22 = 697.951$.
- Now, let’s look at the results from our constrained analysis. It is next in the output file. However, the first thing you’ll notice is that,

immediately after the title, the parameter structure matrices look the same as for the previous model. Does this make sense? Yes - since what we have done is apply a constraint to the same CJS model.

- Thus, the parameter structure matrices show only the underlying structure of the model we are fitting. They give no indication about whether or not constraints were added, what the model structure of the constraints might have been, or anything else - they only indicate the “starting point”.
- Our first indication that we have done something to the analysis comes from the model deviance, which is 654.903. This is different from the starting model deviance - it is 0.952 larger. So, although the difference is small, the model fit is somewhat worse, in terms of the absolute model deviance. But of course, this ignores any possible change in the number of parameters. How many parameters were there in the constrained analysis?
- The answer to this is indicated in the next part of the output file. The first thing you’ll notice is that SURGE prints a short header “CONSTRAINT #1”. This is the first indication that you’ve applied a constraint.
- The second indication follows immediately after this head. Now, instead of a simple tabulation of estimates and standard errors, you see a list of “coefficient estimates” and associated 95% CI and standard errors. In other words, a list of linear model parameter estimates, rather than estimates of survival and recapture alone (Fig. 6.6). Again, the 95% CI and standard errors have been edited for brevity.

coeff.	estimate
INTERCEPT	0.576929D+00
SLOPE N 1	-0.18377D+00
SLOPE N 2	-0.76269D+00
SLOPE N 3	0.25935D+00

Fig. 6.6

- First, what are the “slopes” and “intercepts” tabulated under the “CONSTRAINT #1” header? Well, as you might imagine, from our earlier example of using regression to analyze differences in skull measurements in pre-school offspring, these values are the coefficients for each of the terms in our linear model: one each for SEX, FLOOD, and the SEX.FLOOD interaction. The INTERCEPT was not explicitly included in the model (i.e., we didn’t have a column in our FLOOD.VAR file for the intercept) - SURGE builds it in automatically, and estimates it along with the coefficients for the other terms in the model. In fact, if you’re familiar with linear modelling, you’ll realize that the full design matrix does, in fact, have an additional column of “1”.
- When we apply a constraint, the parameters which are estimated are the slopes, and the intercept. It is from these slopes and intercept (or, rather, from the linear model they define), that we “reconstitute” our estimates for survival.
- A simple example will make this clear. Suppose you were given the equation $Y = 3.2x + 4$. Now, if you were then given some value x , you could interpolate what the value of Y will be (on average, if the equation is a regression line). For example, if $x=4$, then $Y = 16.8$.
- The same thing applies in our constraint analysis. We now have an equation of the form:

$$\phi = \text{INTERCEPT} + \beta_1\text{SEX} + \beta_2\text{FLOOD} + \beta_3(\text{SEX.FLOOD})$$

- Now, from this equation, we can predict, or “reconstitute” estimates of survival for any value of SEX, FLOOD and the interaction (SEX.FLOOD). We have “slopes” for each term, and an intercept. Note however, that the slopes and intercept are estimated on the logit scale. Thus, the estimates are first derived from this equation, then back-transformed to the standard 0 → 1 scale (see the ‘thought box’ on p. 6-3, and below).
- Immediately under the list of “slopes” and “intercepts” are these reconstituted estimates of survival (Fig. 6.7).

Index	Estimate	Index	Estimate
1	0.597044	7	0.640360
2	0.472482	8	0.453693
3	0.472482	9	0.453693
4	0.597044	10	0.640360
5	0.597044	11	0.640360
6	0.597044	12	0.640360

males

females

Fig. 6.7

- To make sure you really understand what is happening, let's consider how the reconstituted estimate for male survival over the fifth interval (i.e., ϕ_5) is obtained. We must first compute the estimate of survival on the logit scale using the linear formula noted above, where the values of INTERCEPT, β_1 , β_2 and β_3 are INTERCEPT, SLOPE1, SLOPE2 and SLOPE3 (respectively) from Fig. 6.6. For males, SEX is "1" (this is our convention). As the fifth interval is a "non-flood" year, FLOOD is "0", and thus the interaction term (SEX.FLOOD) is also "0". Therefore,

$$\begin{aligned} \text{logit}(\phi_5) &= 0.576929 + (-0.18377)x(1) + (-0.76269)x(0) + (0.25935)x(0) \\ &= 0.393159 \end{aligned}$$

- The reciprocal of the logit transform is $e^x/(1+e^x)$ (see p. 6-3). Thus, the "reconstituted" value of $\phi_5 = e^{0.393159}/(1+e^{0.393159}) = 0.597043$. This is the result given by SURGE (up to the level of the rounding error).
- We further distinguish between "reconstituted" parameter estimates and "free parameters" in the next section. In the output file, immediately under the table of reconstituted survival estimates, is a header "FREE PARAMETERS". This is followed by another table of estimates - this time, the recapture estimates. These estimates are "free" in the sense that they were estimated without constraint. Recall that we applied the constraint to the survival estimates only. Thus, the

recapture rates were estimated "the normal way", although their value reflects the influence of the constrained survival estimates - this is why they are not the same as the estimates from the preceding CJS analysis.

- Now, 2 important things to look at. First, go back to the previous page, and look at the reconstituted estimates for survival, for each sex. There is something notably different about these estimates, compared to the unconstrained CJS estimates (p. 6-10). Notice that the 2nd and 3rd "reconstituted" estimates are the same (i.e., have the same numerical value), while the 1st, 4th, 5th and 6th share the same value, but one that is different from 2 and 3. In other words, there are 2 discrete "groups" of estimates. They differ in numerical value between the sexes (significantly? We shall see..), but the structure is the same.
- Does this make sense? It should! Go back to the structure of your FLOOD.VAR file (p. 6-8). Look carefully at the FLOOD column in the file (the second column, as we've constructed it). Notice there are only two dummy variables: "1" - flood, "0" - no flood. Thus, there are only 2 possibilities for this factor in the analysis, and, just as with our example of the skull measurements of children (p. 6-3), there are only two "group means", or "group estimates". Our model constrains SURGE to estimating only 2 values: the estimate of survival when there is a flood, and the estimate of survival when there is no flood. In this example, it appears as if both male and female survival declines during flood years.
- The second important consideration is the number of parameters. We noted previously that the deviance of the constrained model was only marginally larger than the deviance of the unconstrained CJS model. However, do we have the same number of parameters. The key to counting parameters in a constrained model is to recall that we count only "estimated" parameters. In this example, we have estimated 16 parameters: 1 intercept, 3 slopes, and 12 recapture rates. Thus, when compared to the preceding CJS model alone, this model has 6 fewer parameters. Thus, the constrained CJS model is a reduced-parameter model. There are no β terms, since survival is effectively constant within each "flood" group (for the effect of a constant parameter on β terms, see Chapter 4, p. 15).
- How does this affect our interpretation of the relative fit of the 2 models?

Model	deviance	# parameters	AIC
ϕ_{s*t}, p_{s*t}	653.951	22	697.951
cnstr - ϕ_{s*t}, p_{s*t}	654.903	16	686.903
	$\Delta = 0.952$		

- The difference between the 2 models ($\Delta = 0.952$) is not significantly different ($P=0.984$). Thus, we conclude that there is no significant difference between the two models in terms of fit to the data. Since the constrained model has fewer parameters, we choose this model as the most parsimonious.
- We would have also reached this conclusion if we'd used the AIC rather than the LRT - the AIC value for the constrained model is clearly smaller than the unconstrained CJS model.
- Where should we proceed next? Well, the answer to this question is always "it depends on the hypothesis you want to test", but for the moment, let's continue with the logic presented on p. 6-4. First, let's test if there is a significant interaction of sex with flood - in other words, does the effect of flood on survival differ as a function of the sex of the organism? Recall that this is the prerequisite analysis in either multi-factorial ANOVA or ANCOVA. We need to test the interaction terms before going on to the main effects (which may be what we are most interested in).
- So, we'll start by comparing the following models:

ϕ (or p) = SEX + FLOOD + SEX.FLOOD + error
versus ϕ (or p) = <u>SEX + FLOOD</u> + error
SEX.FLOOD

- So, from top to bottom, we see that we first fit the "full model", with both main effects and the interaction. We then follow this by fitting the model without the interaction term. The comparison of the deviances

between these models is a formal test of the significance of the interaction term.

- We've just finished fitting the full model, so we'll proceed directly to fitting the second model - without the interaction term.

Selecting terms to include in the constraint

- The execution of this analysis is very similar to what we've just done. The major difference occurs when we apply the constraint. Thus, we'll skip the description of the preliminary steps (the title, the model selection), and discuss only the steps in applying the constraint. Again, remember we are still applying the constraint to the underlying CJS time-dependent model.
- Again, we want to apply only 1 constraint, so we enter the number "1" and hit the <enter> key.
- We are still going to make use of the PAR file we previously created, so we tell SURGE to use the FLOOD.PAR file.
- Now, we SURGE asks us how many variables to include in the constraint, we have to think about 2 things: which of the variables in our model do we want to include, and in what columns in the VAR file are these variables? The 2 variables we are including are the SEX and FLOOD variables - we are not including the interaction term. So, clearly, the answer to the question is "2".
- Next, SURGE asks you to enter the name of the file containing the model variables - the VAR file. Enter FLOOD.VAR, and hit the <enter> key>.
- SURGE then asks you what columns contain the 2 variables we've just told it we want to include. Here is where we must remember something about how we constructed the FLOOD.VAR file. While it didn't matter which order we entered the columns (i.e, just as in a linear model, the order of the terms in the model doesn't matter in most cases), it IS important to remember which elements went into which columns!
- In FLOOD.VAR, we put SEX first, followed by FLOOD, and finally the interaction term (SEX.FLOOD). So, the terms we want to include are in columns 1 and 2. So, to indicate this to SURGE, you enter the

number “1”, followed by a space, and then the number “2” (of course, you could enter these 2 numbers in reverse order as well - what is important is the numbers of the columns, not the order you enter them). (Fig. 6.8).

```

C* CONSTRAINING PARAMETERS

C* There are already 0 constraints
C* HOW MANY MORE CONSTRAINTS (display model=-1; back to M=-2) ? 1

* CONSTRAINT # 1
C* FILE OF CONSTRAINED PARAM. LIST (flood.par= <—>) ? flood.par
C* NUMBER OF VARIABLES IN CONSTRAINT ? 2
C* VARIABLE FILE (BIOMEKO FORMAT) (flood.var=<—>) ? flood.var
C* NUMBER OF THE VARIABLES [COLUMNS] ? 1 2
C* LOGISTIC TRANSFORMATION (YES=<—>)?
    
```

Fig. 6.8



Does you have to use the same .VAR file? No. We could have “eliminated” the SEX.FLOOD interaction term simply by building a new, different VAR file containing only 2 columns: one for SEX, and one for FLOOD. However, SURGE gives you the option of “re-using” the original VAR file, by letting to select which terms (columns) you want to use. This is convenient, since you only have to create the VAR file once, and it also “reinforces” the idea of “dropping” terms from the model. Use whichever approach you find most comfortable.

- Accept the logistic transformation, and proceed with the rest of the analysis. What do our results look like?
- First, we notice that instead of 3 slopes and 1 intercept, we now only have 2 slopes and one intercept. The slopes correspond to the SEX and FLOOD terms in our model, respectively. We have 1 fewer slope, since we eliminated the interaction term (SEX.FLOOD) from the model.

coeff.	estimate
INTERCEPT	0.51623D+00
SLOPE N 1	-0.66673D+00
SLOPE N 2	-0.63611D+00

- Next, we examine the “reconstituted” survival estimates. Again, we notice that there are effectively 2 estimates for each sex: one each for the flood or non-flood years.

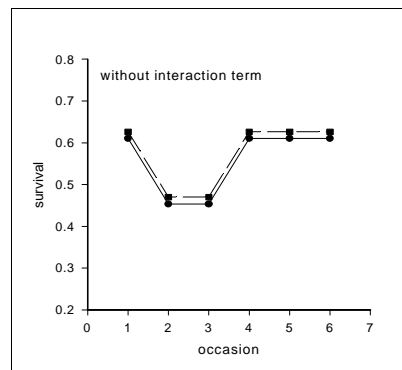
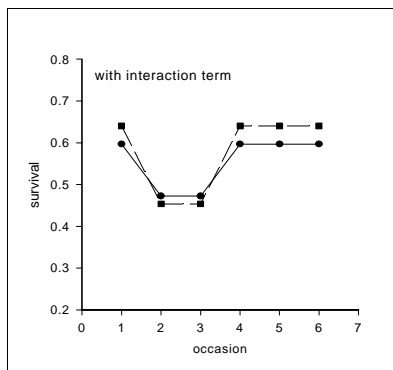
Index	Estimate
1	0.610534
2	0.453497
3	0.453497
4	0.610534
5	0.610534
6	0.610534

Index	Estimate
7	0.626265
8	0.470006
9	0.470006
10	0.626265
11	0.626265
12	0.626265

- Notice that the estimates are similar to, but not exactly the same, as the estimates with the full constraint (i.e., including the interaction term) (p. 6-12).
- However, the key question is, are they significantly different? We test this by comparing the relative fits of the two models, keeping track of the number of parameters difference between the 2 models.

Model	deviance	# parameters	AIC
with (S.F)	654.903	16	686.903
without (S.F)	655.190	15	685.190
$\Delta = 0.287$			

- The difference between the 2 models is not significant ($\chi^2=0.287$, $df=1$, $P=0.990$). Therefore, we conclude that there is no significant interaction of sex and flood on variation in survival, since the models do not differ significantly in their fit with or without the interaction term being present.
- What about the AIC values? The AIC for the reduced model (without the SEX.FLOOD interaction) is marginally smaller (by 1.713). This is consistent with the results of our LRT - the model without the interaction has the lower AIC, and is thus the most parsimonious of the two. Thus there is no significant interaction between SEX and FLOOD on survival in this data set. This is shown graphically in the following figures .



- How did we arrive at 15 parameters for the model without the interaction term? Well, the full model, with both SEX, FLOOD and

the Interaction term (S.F) had 16 parameters. Since we have eliminated 1 term from the model, we now have 15 parameters.

- Since the interaction term is not significant, we can now proceed with testing the significance of the main effects: SEX and FLOOD. We do so easily by using exactly the same process as we just completed above: we modify the constraint to include one of these two remaining terms, and compare the fit.
- However, while this allows us to test for significance of both terms, we must remember that we will not be able to use LRT to determine if the model containing SEX alone is a better model than one containing FLOOD alone. Why? Because these are not nested models. Thus, for comparison of these 2 models, we will have to use the AIC method. Even if the nesting isn't obvious (if it isn't, think about it!), the necessity of using AIC for these 2 models will be obvious when you compare the number of parameters.
- In the following table, we list the remaining models, and their associated statistics.

Model	deviance	# parameters	AIC
SEX + FLOOD	655.190	15	685.190
SEX	662.250	14	690.250
FLOOD	655.275	14	683.275

- The comparison of (SEX + FLOOD) versus (SEX) is interesting. Even though (SEX) has fewer parameters, the AIC value is larger. This tells us right away that the (SEX) model is not as good a model as (SEX + FLOOD). We can quantify this disparity by using a LRT, since these models are nested. There is a significant difference in fit between the models ($\chi^2 = 7.06$, $df=1$, $P=0.008$). Thus, there is a significant influence of FLOOD on model fit, since eliminating from the model significantly reduces the fit to the data.
- What about (SEX + FLOOD) versus (FLOOD)? Inspection of the AIC values suggests that (FLOOD) is a better model than (SEX+ FLOOD).

In other words, we expect no significant difference between the fit of (SEX + FLOOD) and (FLOOD). We confirm this using a LRT, which indicates no significant difference in fit ($\chi^2 = 0.085$, $df=1$, $P=0.77$).

- At this point, we have already deduced that model (SEX) is a poor model, and that model (FLOOD) is the most parsimonious acceptable model among those tested. However, for completeness, if we had wanted to compare (FLOOD) with (SEX), we would have to compare the AIC values. Since the AIC for (FLOOD) is smaller than the value for (SEX), we conclude that (FLOOD) is a better model for these data.
- In other words, we conclude that there is no significant difference between the sexes, and that flood significantly influences variation in survival.
- Does this mean that this is our best model overall? The answer to this question is no. What we have done is simply to test a set of hypotheses under specific conditions. What were our main conditions? The conditions in this example were the use of the CJS model structure for both survival and recapture, prior to adding the constraints.
- The remaining question is - would we have come up with a different result if we had made recapture rate constant? What if we had left time-dependence in recapture rate, but used the same parameter values between the sexes? How does our current model compare to one where survival is assumed to be constant?
- Where we go from here, then, very much depends upon what we're after. We have to keep in mind the various purposes of model testing. At one level, we are seeking to test specific biological hypotheses. At the other, we may also be trying to find the most parsimonious model, which will provide us with the most precise and least biased estimates for modeling purposes.
- Again, our recommended strategy is to use the process of model selection to identify the most parsimonious acceptable model containing the effect(s) that you want to test, and then proceed to use LRT or AIC to compare this model with reduced parameter models where one or more of these terms have been eliminated. Remember, by "acceptable" we mean a model which fits the data.
- In fact, we can't emphasize this enough - the first step in analyzing your data must be to ensure that your starting model (CJS, for example) adequately fits the data. SURGE does not do goodness of fit

testing on its own - you must use some other program (e.g., RELEASE) to test goodness of fit before using SURGE. Again, this important topic is introduced briefly in the Appendix.

- If this seems like a long and involved process, it frequently is, especially for large data sets. You have little alternative but to think through the questions you have in mind carefully before starting, and then making a good plan for which models you want to test. Obviously, biological insight is an important tool in helping streamline this process.

Linear trend

- We will now show how to use SURGE, along with VAR and PAR files, to test for "trend" in the data - linear increase or decrease in survival or recapture.
- We created a simple data set (LINEAR.SUR), which contains simulated data which we will use for our analysis of linear trend. There are 8 occasions in the data set. Assume that the "simulated animals" are all marked as adults.
- Start with fitting the basic CJS model. Assume that this is an acceptable model, and fits the data. Since there is only one group, this analysis is very easy, and should take you only a few minutes with SURGE. At this stage, we'll assume you know the steps for fitting this model, so we'll proceed directly to the results. The deviance of the CJS model for these data was 1243.961.
- The estimates for both survival and recapture rates are tabulated on the next page. Note that the value of the last estimate for both survival and recapture is the same (0.495434). As you might have remembered, this is the square-root of the β_8 term (i.e., the square-root of the product $\phi_8 p_8$). Thus, for this model, we have 13 total identifiable parameters. The AIC for this model is therefore 1269.961 (i.e., $1243.961 + 2*13$).

Index	Estimate
1	0.793932
2	0.614494
3	0.459556
4	0.576422
5	0.369512
6	0.625467
7	0.495434

Index	Estimate
8	0.604580
9	0.229809
10	0.455414
11	0.764706
12	0.710396
13	0.417219
14	0.495434

- Now, let's proceed to see how to use SURGE to fit a model with a linear trend - this will allow us to formally test our hypothesis that there may be a decline in survival over time.
- Doing this in SURGE is relatively straightforward. Mechanically, we again create VAR and PAR files, and apply them to our CJS model.
- The PAR file is straightforward - 7 parameters, with parameter indexing 1 → 7. Call it LINEAR.PAR (in fact, it is identical in structure to FLOOD.PAR, which we created earlier, except for the increased number of parameters in this example).
- While the PAR file is straightforward, there is a slight "twist" to the logic needed to create the VAR file. Let's call it LINEAR.VAR. Think back to the VAR file we created in our earlier example using the Dipper data set - the FLOOD.VAR file. We had 3 columns of numbers in that file: one for SEX, one for FLOOD, and one for the interaction term (SEX.FLOOD). Concentrate on the FLOOD column. We coded FLOOD as a simple binary variable: there was either a flood ("1") or there wasn't ("0"). In our present example, however, things aren't quite so simple. We are trying to build a model with a linear trend. In other words, a systematic change in survival (up or down) through time.
- What do we know about a "trend", and how does it differ from the flood example? By definition, a trend has a slope which is significantly different from 0. Given that the slope differs from 0,

then on average, $y_{(i-1)} < y_i < y_{(i+1)}$ for an increasing trend with (i), and the reverse for a decreasing trend. Second, a trend (if linear) is "continuous" through time - it is not a simple binary condition, as was the case with flood. Thus, we need to code a "trend" through time in such a way that it meets these 2 conditions.

- As it turns out, it is very simple to do this, although you may have to take a few minutes to grasp the logical connection.
- To code for a linear trend, all you need to do is write a series of increasing (or decreasing) numbers, 1 through n (where n is the number of occasions you want to fit the "trend" to). You don't have to start with the number 1, but you do need to use the sequence {starting value}+1, {starting value}+2, and so on.
- Here is what the LINEAR.VAR file would look like, using "1" as our starting value:

```

7 1
$
$
1
2
3
4
5
6
7

```

LINEAR.VAR

- The top row is the number of intervals in our example (7), and the number of columns (1). Next, the "\$ \$" signs, and finally, the coding for a trend. The order of the numbers (1 to 7, or 7 to 1) makes no difference - SURGE will simply use the numbers to fit a linear trend - it will let the data determine if the trend is up or down (if any trend exists).

- Before we go on, as a quick test, how would we have coded the dipper data set to test if there was a trend in survival, and if the trend differed between males and females?
- Well, if you look carefully at the structure of the FLOOD.VAR file we created earlier, you should see right away that it is only a slight variant of that file. Here is the “linear trend” VAR file for the dippers:

```

12  3
$
$
1   1   1
1   2   2
1   3   3
1   4   4
1   5   5
1   6   6
0   1   0
0   2   0
0   3   0
0   4   0
0   5   0
0   6   0
    
```

- In this case, the second column is the “linear trend” column, and the final column is the interaction of sex with the linear trend.
- Now, one subtle variation in this theme: suppose that we want to test for a non-linear trend. How would we do this? Well, there are several ways to analyze non-linear relationships. Perhaps the easiest is to use multiple regression, fitting a series of power terms to the function.

- For example, a comparison of the model $Y = X + X^2$ to model $Y = X$ is a formal test of the significance of the X^2 term. If the X^2 term is not significant, and if the model $Y=X$ fits the data, then we can conclude that the relationship is linear.
- How would we construct a VAR file for model $Y = X + X^2$? In fact, it is very simple. All you need to do is add a second column to your VAR file to accommodate the X^2 term.
- Let’s edit our LINEAR.VAR file to do just this. If X is represented in the file by the sequence 1,2,3...7, then X^2 is represented by the sequence 1,4,9...49. Thus, our new LINEAR.VAR file would look like:

```

7  2
$
$
1  1
2  4
3  9
4 16
5 25
6 36
7 49
    
```

- There are really only 2 notable changes in the file: first, the first line is now “7 2”. The change from “1” to “2” reflects the fact that we now have a second column (below the “\$ \$” signs). The second change is, obviously, the addition of the second column. The 2 columns now correspond to X and X^2 respectively in a model $Y = X + X^2$.
- Let’s now use SURGE to test for both non-linear and linear trend in the data, and compare the fit to the CJS model where survival simply varies over time.
- Start SURGE, and send the output to file LINEAR.LST. For a title, “Phi($X+X^2$),p(t)”, to denote we are going to use both the “X” and

“X²” terms in our constraint. There is only 1 data file to be analyzed: LINEAR.SUR.

- For model selection, for both survival and recapture, we choose option 2 - time-dependence. In both cases, we accept the default condition of complete time dependence.
- Now, when prompted by SURGE for the number of constraints we want to add, we enter “1”. The name of the file containing the parameters to be constrained is LINEAR.PAR.
- Next, SURGE asks for the number of variables in the constraint (i.e., the number of columns in the VAR file). We enter “2”.
- Then, SURGE asks for the name of the VAR file. We enter LINEAR.VAR.
- Finally, we accept the logistic transformation (the default).
- Since we don’t want to fix any parameters, we hit the <enter> key enough times to proceed directly to the analysis menu. We run the analysis, and then choose option “2” from the final menu, to specify another model.
- We proceed exactly as before. First, we select the time dependent starting models for both survival and recapture. Then, we add 1 constraint, and use the LINEAR.PAR file.
- Now, when SURGE asks us how many variables in the constraint, we say “1”.
- Next, SURGE asks us for the name of the VAR file. We enter LINEAR.VAR.
- Since the number of variables we’ve chosen to include in the constraint (“1”) is less than the total number of variables in the LINEAR.VAR file (“2” variables or columns), SURGE will then ask us to identify the column we want to use. The column we want is the first column, so we enter “1”.
- We accept the logistic transformation, and proceed to the end, where we run the analysis. Once completed, exit SURGE, and examine the output (in file LINEAR.LST).
- Again, as with our first example using the dippers and the flood covariate, we see that SURGE does not indicate in the parameter structure that we have added a constraint - the model structure printed is the CJS starting model.

- Next, we see the slopes and intercept estimate for the $Y = X + X^2$ model, followed by the reconstituted estimates of survival. The deviance for this model is 1249.160. With 10 estimated parameters (2 slopes, 1 intercept, and 7 recapture rates), this gives an AIC of 1269.160.
- The second model, without the “X²” term, has a deviance of 1252.837. The difference in deviance from the model with the “X²” term is 3.677. Since we have eliminated 1 term from the model (i.e., 1 slope for the X² term), the second model has 9 parameters. Thus, the LRT between these 2 models has 1 degree of freedom, and is marginally non-significant ($\chi^2=3.677$, $df=1$, $P=0.055$).
- For the moment, let’s accept the fact that this difference is non-significant, and therefore accept the second model as the more parsimonious of the two.
- However, how well does this second model, which has a simple linear trend in survival (a negative trend, since the slope is negative), compare with the starting CJS model? Recall that the first CJS model had a deviance of 1243.961, with 13 parameters. Thus, the AIC for this model is 1269.961. The difference in deviances between the 2 models is 8.875, with 4 degrees of freedom, which is marginally non-significant ($P=0.0643$). This marginal difference is also reflected in the comparison of AIC values: 1269.961 for the CJS model, 1270.837 for the constrained “linear decline” model, a difference of <1.
- In general, when the difference in AIC is <2, there is probably little reason to differentiate between 2 models (K. Burnham pers. comm.). Thus, in this example, perhaps the best we can do is say” although there is a strong suggestion of a negative trend in survival in this study, we did not have sufficient statistical power to differentiate a model with a “trend” from a model where survival was allowed to vary randomly over time.
- Of course, one thing that might affect our power is the structure of the recapture model, which we have not considered at this stage. So, before we would draw final conclusions, we would proceed to test different models for recapture rate.
- Let’s assume that our model with the linear decline is in fact the correct model (i.e., we aren’t making a Type II error).



One of the more problematic issues in “step-down” model selection is the question of Type II error. Recall from your basic statistics classes that we distinguish between Type I and Type II error. Type I (α) error occurs when we reject the null hypothesis when, in fact, it is true. Type II (β) error occurs when we accept the null hypothesis when, in fact, it is false. Frequently, researchers seem most concerned with Type I error - the rejection of true null hypotheses. We “guard” against this type of error by selecting a significance level α which is sufficiently small such that the probability of making a Type I error is reduced (i.e., $\alpha=0.05$).

However, in the case of “step-down” model selection, the issue is not so clear. When we are looking for the most parsimonious model, we are, in fact, “seeking” non-significance. If a LRT between 2 nested models is non-significant, we accept the reduced model as being the more parsimonious of the two. In other words, we use the non-significance of this test as the basis for selecting the reduced model, and for drawing conclusions about the one or more terms which differ between the models. And yet, in this process, we clearly run the risk of making a Type II error - accepting the null hypothesis (no difference) when, in fact, it is false.

The question becomes one of which is more of a problem: making a Type I error, or making a Type II error? In general, it is probably wise to err on the conservative side, and increase the α -level from 0.05 to 0.1, decreasing the risk of making a Type II error at the expense of a slightly increased risk of a Type I error. In simplest terms, if your LRT significance is ≤ 0.10 , especially with “typical” sample sizes, there is probably good reason to be cautious in accepting the reduced model. In fact, most stepwise-regression routines suggest using a P value of 0.10 to 0.15.

The issue of significance testing, model selection, and the role of alternate selection criterion (like AIC) is currently an area of intense research.

- What do our estimates look like? Well, we can clearly see that the constrained estimates decline systematically through time:

Index	Estimate
1	0.645318
2	0.605577
3	0.564389
4	0.522293
5	0.479878
6	0.437751
7	0.396503

- One further subtle, but important point to be made here. Accept that we haven’t made a Type II error and that the general CJS model does not fit the data appreciably better than the trend model. Would it then be correct to say that there is a significant trend - i.e., that the slope is significantly different from 0? No! To make this claim, we would need to compare the trend (df=1) with a model where survival was constant over time. Comparing the “trend” model with the CJS model tells us that the variation around the trend is not significant.

Constraining with real covariates

- The last section, where we constrained estimates of survival to be linear functions of time (i.e., show trend), leads directly into our next example - constraining parameters to be functions of “real” variables (in the mathematical sense of real), as opposed to simple “dummy” or other integer variables.
- For example, suppose we have measured some other variable, such as total precipitation, or measures of annual food abundance, which can take on “real” or “fractional” values. Clearly, we might want to test the hypothesis that a model where one or both parameters are constrained to be linear functions of this type of covariate might be extremely useful.
- Fortunately, we have to learn absolutely nothing new in order to do this -

12	3	
\$		
\$		
1	12.1	12.1
1	6.03	6.03
1	9.1	9.1
1	14.7	14.7
1	18.02	18.02
1	12.12	12.12
0	12.1	0
0	6.03	0
0	9.1	0
0	14.7	0
0	18.02	0
0	12.12	0

- One you have the VAR and PAR files created, you proceed in precisely the same fashion you did with either the “flood” or “linear trend” examples we just covered - the only difference is that, in this example, you’re concentrating on recapture rates, rather than survival.

More than 2 groups

- It is not uncommon to have more than 2 groups in an analysis. For example, you may have a control and 2 or more treatment groups.
- How would you code the VAR file for such a situation? In fact, it’s easy, if you remember some basic principles from analysis of variance (ANOVA).

- The number of columns used to characterize group differences (e.g., belonging to one group or not) will always equal the number of dummy variables (“coded “0” or “1””) that you need to characterize group differences. For any variable that we treat as “categorical” or “classification” (i.e., both COLONY and TIME in this example) with k levels, the number of columns needed is $(k-1)$, which happens to be the numbers of degrees of freedom associated with a factor in a standard ANOVA (not - it’s not a coincidence). In short, the number of columns (hence, the number of dummy variables) needed equals the degrees of freedom associated with that variable. So, the minimum number of columns of dummy variables needed to specify our model are defined by the number of degrees of freedom for each factor, plus any interaction terms.
- Of course, it IS possible to specify a model with more columns than the minimum set we’ve just described. Would this be wrong? Not exactly - your estimates would be “correct”, but it becomes very difficult to count (separately) identifiable parameters. And since counting parameters is essential to model testing, using more columns than necessary in your VAR file to specify the model should be avoided.
- Consider an example of a study over 5 occasions, where we have 3 “groups”, or levels of our “main effect”. Thus, we need $(3-1)=2$ columns of “dummy variables” to specify group association. Suppose we also have a quantitative covariate (say, hours of observation). Linear terms have 1 degree of freedom, so one column of dummy variables. Finally, for the interaction, we need $(3-1) \times (1) = 2$ columns.
- The VAR file is shown at the top of the next page. We have formatted the table slightly to emphasize the “logical connection” amongst the columns. The 2 columns on the left indicate group: group is identified depending upon the pair of dummy variables across these 2 columns: “0 0”, “0 1”, and “1 0”. The middle column (of the 5 total columns), is the “quantitative covariate” column.

12	5			
\$				
\$				
0	0	1.1	1.1	1.1
0	0	0.2	0.2	0.2
0	0	3.4	3.4	3.4
0	0	4.1	4.1	4.1
0	1	1.1	0	1.1
0	1	0.2	0	0.2
0	1	3.4	0	3.4
0	1	4.1	0	4.1
1	0	1.1	1.1	0
1	0	0.2	0.2	0
1	0	3.4	3.4	0
1	0	4.1	4.1	0

- The last 2 columns are the interaction columns. Remember, the interaction term can be thought of as a “multiplication term” - the product of the various factors contained in the interaction. Since this interaction is the interaction of “group” (2 columns) and “quantitative covariate” (1 column), then we have (2) x (1) = 2 columns for the interaction. We simply right multiply each element of the group column vectors by its corresponding element in the “trend” column vector. Therefore, 5 columns total. This is reflected in the first line of the VAR file as well (“12 5”).
- Now, if we were applying this VAR file, and we wanted to test for the significance of the interaction term, we would first run the constraint using all 5 columns in the VAR file, and then a second time using only the first 3 columns - 3 because we want to drop the interaction term,

which is “stored” in columns 4 and 5.

- Note that it is important to use the minimum number of columns of “0” or “1” dummy variables to characterize your groups. Why? Because each column in your VAR file becomes a slope, which is an estimated parameter. Our goal is to use as few parameters as possible to specify a model. Extra columns wouldn’t make your model “wrong”, but would make the counting of (separately) identifiable parameters more difficult.

Time and Group - building additive models

- Most newcomers to SURGE find building VAR files the most daunting part of the “learning curve”, and building VAR files for what we refer to as “additive models” the most difficult of all. However, if you’ve understood everything we’ve covered up to now, you should find building VAR files for additive models only slightly more involved than what we’ve already done.
- What do we mean by an “additive model”? As you may remember from Chapter 5, an additive model is one where we express variation in survival or recapture as a function of the additive contributions of 2 or more factors. In other words, a multi-factor ANOVA.
- Recall our comparison of the “good” and “poor” colonies. Our linear model was represented by

$$\phi = \text{COLONY} + \text{TIME} + \text{COLONY.TIME} + \text{error}$$

- In this model, we have two “factors” - COLONY (“good” and “poor”) and TIME ($\phi_1, \phi_2, \dots, \phi_7$). Each “time interval” is considered as a different level of the TIME factor. In this case, we are treating time as a “categorical” variable, as opposed to a quantitative covariate as we did in the “trend” example presented earlier.
- You may have noted that this is, in fact, the default SURGE CJS model with 2 groups - as long as you tell SURGE to use the same parameter structure between groups, but to let the parameter values differ, then SURGE uses the “full model”, with both factors (COLONY and TIME), and the interaction term (COLONY.TIME).

- When we run SURGE and use the same parameter values between groups, we are testing model

$$\phi = \text{TIME} + \text{error}$$

- The difference between these two models is, in fact, the effect of COLONY + C.T, not just COLONY alone. As noted in Chapter 5, we are, in fact, leaving out the “intermediate model”:

$$\phi = \text{COLONY} + \text{TIME} + \text{error}$$

- In this model, we are considering the additive effects of the 2 factors - hence, we refer to it as an “additive” model. To fit this model we need to build the appropriate VAR file. If you understood the logic outlined on p. 6-22, you should find this task relatively straightforward.
- First, how many columns will there be in our VAR file (lets call it ADD.VAR)? We have 2 factors - COLONY, with 2 levels, and TIME with 7 levels. Since we need (k-1) columns of dummy variables to specify a factor, where k is the number of levels of a given factor (see p. 6-22), we therefore need (2-1) x (7-1) = 6 columns of dummy variables to specify the TIME parameter.
- What goes into the columns? Let’s start with the easy one - the COLONY factor, with only two levels, and thus only one column of dummy variables. By now, you should recognize that all we need for COLONY is a single column of “1” for one colony, and “0” for the other.
- However, specifying the dummy structure for the TIME effect is perhaps not so obvious. We have 7 levels for TIME, so 6 degrees of freedom, or 6 columns. How do we decide which columns should have “1” or “0”?
- To derive the TIME columns, it helps to remember the connection between regression models and ANOVA (pp. 6-2 & 6-3). That connection involves dummy variables; remember, all ANOVA-type analyses can be carried out (and in statistical software ARE carried out) using dummy variables. In this example, to specify each of the 7 factor levels, we need 6 dummy variables; call them T₁, T₂...T₆ respectively. The specify the first level of time, we make all 6 dummy variables (T₁...T₆) equal to zero. Call this the “baseline” level. For the second

level of TIME, we let T₁=1, and leave all other dummy variables (T₂...T₆) equal to zero. For the third level of TIME, we let T₂=1, and set all other dummy variables (T₁, T₃..T₆) equal zero. And so on, until we get to the seventh level of TIME, for which T₆=1, but all other dummy variables (T₁..T₅) equal 0. We repeat this process for both the “good” and “poor” colonies. Here is what the VAR file should look like:

ADD.VAR

COLONY	TIME (T ₁ →T ₆)					
1	0	0	0	0	0	0
1	1	0	0	0	0	0
1	0	1	0	0	0	0
1	0	0	1	0	0	0
1	0	0	0	1	0	0
1	0	0	0	0	1	0
1	0	0	0	0	0	1
0	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	1

- What SURGE does with the ADD.VAR file is to estimate a coefficient (“slope”) for each dummy variable. This coefficient tells use how any one particular level differs from the baseline level (i.e., the first time interval). We can put all the coefficients together in one regression equation (for COLONY and TIME):

$$\phi = \text{INTERCEPT} + \beta_0 \text{COLONY} + \beta_1 T_1 + \beta_2 T_2 + \beta_3 T_3 + \beta_4 T_4 + \beta_5 T_5 + \beta_6 T_6$$

- β_0 tells us how much, on average, survival in the “good” colony differs from survival in the “poor” colony. Note that when COLONY=0 (say, for the “poor” colony), the β_0 term drops out of the regression equation, estimating survival in the “good” colony. Each of the remaining β -terms specifies how much survival in one year period (average over both colonies) differs from the baseline year. The greater the magnitude of a particular β the greater that year’s survival rate differs from the baseline year, and the greater the statistical significance of a particular β , the greater the contribution to the overall significance of the TIME factor. It should be easy to see that, for example, for the “poor” colony (COLONY=0) in year 3, the β_0 term drops out of the equation, and we are left with

$$\phi = \text{INTERCEPT} + \beta_2$$

because $T_2=1$, and all other T values ($T_1, T_3..T_6$) are equal to zero. Thus, β_2 tells us how much survival in year 3 differed from the baseline year (year 1).

- Similarly, the equation for year 5 in the “good” colony (COLONY=1) would be

$$\phi = \text{INTERCEPT} + \beta_0 + \beta_4$$

- Note that the coding of dummy variables is arbitrary. You could just as easily and intuitively coded year 7 as the baseline level (all dummy variables equal to zero) and made the coding for year 1, $T_1=1, T_2..T_6=0$. If you had, then the first row of the TIME columns in the ADD.VAR file would read “1 0 0 0 0” instead of “0 0 0 0 0”. Obviously, all the other rows would change as well.
- We admit that this is somewhat more complicated, but with a bit of thought, and a glance at a good text on linear model (see references noted earlier in this chapter), you should see the connection.
- With the VAR file we created on the previous page, we could run the full CJS model - all you need to do is add the columns for the

(COLONY.TIME) interactions!

- Remember, for each factor we need $(k-1)$ columns of dummy variables. If we have two factors, one with k levels, and the other with j levels, then we need $(k-1) \times (j-1)$ columns for the interactions. In our present example, we have 2 levels for colony, and 7 levels for time, so $(2-1) \times (7-1) = 6$ columns for the interactions. The values of the dummy variables in the interaction columns are obtained simply by multiplying each element in each of the “factor” columns (COLONY and TIME) together.



If you are using CR, which was supplied with the “commercial” version of SURGE, you may wonder why go through all this trouble - CR (which is a simple front end interface to SURGE and several of the associated utilities), has an option to build additive models for CJS or age models. As such, why go through the trouble of building it manually? The reason is simple - the more you rely on short-cuts, the less you’ll really understand. The emphasis in this chapter has been on drawing the connection between ANOVA and ANCOVA (which most biologists are familiar with), and linear models (which they more often are less familiar with). While CR does provide you with a “quick and dirty” way of generating the VAR file for some additive models, there are many, often more complicated designs it doesn’t address, for which you’d have to build the VAR files manually anyway. Given this, it is probably worth the effort to learn how to do things by hand first, before relying on automated “short-cuts”.

- Here is what the “full CJS” model VAR file would look like - it is simply the ADD.VAR file we constructed on the previous page with the interaction columns included.

COL	TIME (T ₁ → T ₆)	COLONY.TIME
1	0 0 0 0 0 0	0 0 0 0 0 0
1	1 0 0 0 0 0	1 0 0 0 0 0
1	0 1 0 0 0 0	0 1 0 0 0 0
1	0 0 1 0 0 0	0 0 1 0 0 0
1	0 0 0 1 0 0	0 0 0 1 0 0
1	0 0 0 0 1 0	0 0 0 0 1 0
1	0 0 0 0 0 1	0 0 0 0 0 1
0	0 0 0 0 0 0	0 0 0 0 0 0
0	1 0 0 0 0 0	0 0 0 0 0 0
0	0 1 0 0 0 0	0 0 0 0 0 0
0	0 0 1 0 0 0	0 0 0 0 0 0
0	0 0 0 1 0 0	0 0 0 0 0 0
0	0 0 0 0 1 0	0 0 0 0 0 0
0	0 0 0 0 0 1	0 0 0 0 0 0

- Lets see how we would use this new VAR file to go through the entire model testing sequence described on p. 6-23.
- Start SURGE, and use the 2 swift data sets: G.SUR and P.SUR (for “good” and “poor” respectively).
- We want to start with full-time dependence for both survival and recapture rates. Of course, we could fit this model using the built-in menu selections, as we’ve discussed in detail previously. For the moment, lets use our “full” VAR file (above) to confirm the equivalence of the two approaches. We need to make sure that the first line of our “full” VAR file has the correct number of “rows and columns” - “14 13”, followed by the two “\$ \$” signs.
- If this VAR file does truly represent the CJS model (as we're claiming it does), then applying it as a constraint to the CJS model we just

selected (using the model choice menus) should have no effect. We’ll check this.

- Off course, we also need the correct PAR file (call it ADD.PAR). We have 7 occasions for each colony, so if we apply the constraint to the survival estimates, the PAR file would look like:

12
1 2 3 4 5 6 7 8 9 10 11 12

- First, run the “full” CJS model using just the menus (by now you should be able to do this without instruction). We will use this results as our “control”.
- Then, run the same full CJS, but this time applying 1 constraint. Tell SURGE to use the ADD.PAR file, and then use all 13 columns of the “full” ADD.VAR file we constructed on the previous page. Accept the logistic transformation, and proceed to run the analysis. Once finished, use choice 2 to go back to the model selection menus.
- Again, we will run the CJS model, and apply the “additive model” constraint. Our ADD.PAR file is the same, except now, we tell SURGE that we only want to use 7 columns of the ADD.VAR file (1 column for COLONY, 6 columns for TIME). SURGE will ask you which columns to use - these elements are contained in columns 1 to 7, so you enter the numbers 1 to 7, separated by a space (Fig. 6.9).

```

C* CONSTRAINING PARAMETERS
C* There are already 0 constraints
C* HOW MANY MORE CONSTRAINTS <display model=-1; back to M=-2> ? 1
* CONSTRAINT # 1
C* FILE OF CONSTRAINED PARAM. LIST <flood.par= <—> ? add.par
C* NUMBER OF VARIABLES IN CONSTRAINT ? 7
C* VARIABLE FILE (BIOMECO FORMAT) <flood.var=<—> ? add.var
C* NUMBER OF THE VARIABLES [COLUMNS] ? 1 2 3 4 5 6 7
    
```

Fig. 6.9

- Run the analysis, and examine the output. The first thing you'll notice is that when we constrained the CJS model using the "full" ADD.VAR file (which corresponds to the CJS model), we should see no difference in our estimates. A comparison of the model deviances shows that both models give the same deviance (339.705). Further, the parameter estimates are also identical, with one important difference. When you ran the CJS model using just the menu choices in SURGE, the final estimate of survival and recapture for both colonies was, in fact the estimate of the β_8 term (i.e., the square-root of $\phi_7 p_8$). As such, the total number of parameters was 26.
- However, look closely at the results from using the ADD.VAR file. You'll notice that SURGE has estimated separate survival and recapture rates for this last interval. However, these clearly are not individually identifiable, so we must ignore their actual values.
- However, because SURGE has "tried" to estimate them, it includes 2 extra "slopes" - one for each group. Thus, you need to discount the number of slopes by 2 to get the correct number of parameters.
- Recall that when we ran our analyses without the constraints, the final term (the square-root of the β_8 term), was the same for both survival and recapture. This was an artifact of the choice of initial values - it acted as a reminder that only β_8 was estimated, not ϕ_7 and p_8 separately. However, with the addition of the constraint, there is no such "reminder", and we need to be aware of the "non-identifiability" problem ourselves. Although this is not "obvious", some slope(s) in the survival constraint, and the last recapture rates, are not separately identifiable. We will see how to discount non-estimable parameters below.
- These complications aside, the 2 models are identical - if nothing else, they must be in order to lead to precisely the same model deviance.
- Now, let's turn to the comparison with the additive model. The model deviance for the additive model is 339.705. What about the number of parameters?
- Counting parameters for the additive model is not quite as simple as for other models. The easiest way to do it is to remember what we're doing - we're testing a model where there is time variation in survival for both colonies, but that the difference between colonies is due to a constant, additive, component. This additive component is simply 1 more parameter (like estimating a constant). This should be surprising, especially if you think of the additive model in the ANCOVA example we dealt with earlier - in the Lebreton *et al.* (1992) monograph, they give you an explicit "hint" when they use the word "parallelism". If any pair of lines in an X-Y plane are parallel then for any value X, the two lines differ in Y by some constant amount. Look back at the "thought box" on p. 5-2 of Chapter 5. In the lower right hand panel you see a representation of "no interaction" - the two lines parallel each other. It is the constant "difference" between the lines which constitutes one of the estimable parameters.
- So, for our present example, simply count the number of parameters you would expect for 1 colony alone, for the underlying model (CJS — 13 parameters for either colony alone). Then, add 1 for the additivity (i.e., the "constant difference") in survival (you would add 2 if you had additivity in survival and recapture simultaneously).
- Now, the tricky part - if survival in one colony is simply survival in the other colony plus a constant, then the survival rate is identifiable (by linear interpolation) for all intervals in the second colony, and thus all of the recapture rates are estimable (no β terms). So, since there are 7 recaptures, we add 7, bringing our total to $13 + 1 + 7 = 21$ total parameters.
- Still don't get it? Here's another way to think of it. First, in this example, we are constraining the CJS model by colony. What was estimable in this starting model will remain estimable in the same model with a constraint - in this case, the additive model. Thus, if some parameters are not identifiable in the additive model, they must be those from the last time interval: ϕ_{7g} and ϕ_{7p} and p_{8g} and p_{8p} (where "g" = good, and "p" = poor). Let's focus our attention on them. In the unconstrained CJS model, we could identify the products $\beta_{8g} = \phi_{7g} p_{8g}$ and $\beta_{8p} = \phi_{7p} p_{8p}$. In essence, we had 2 equations and 4 unknowns. If we pick some value for (say) ϕ_{7g} , then we can solve for p_{8g} . Similarly, we could pick an arbitrary value for ϕ_{7p} and solve for p_{8p} . Thus, we have 2 arbitrary parameters to discount from the initial total of 28 parameters in the model - in other words, $28 - 2 = 26$ identifiable parameters. Of course, we knew this already - we simply want to confirm that this approach yields the same results.
- Now, let's apply this same line of reasoning to the additive model case.

We still have the same 2 equations as before, plus 1 new one; $\phi_{7g} = \phi_{7p} + c$ (from the constraint). “c” is a known constant, because “c” is common to all intervals. Therefore, if we pick a value for ϕ_{7g} then ϕ_{7p} is known, and thus also both p_{8g} and p_{8p} . Thus, we have just one arbitrary parameter to discount from the total number of parameters originally included in the additive model; for survival, 7 slopes + 1 intercept = 8, and for recapture, 14. Thus, $8+14 = 22$, -1 (the arbitrary parameter) = 21 identifiable parameters.

- Lebreton *et al.* (1992) analyze a large number of different models for this data set (see Table 14). In fact, they find that the most parsimonious model has constant survival within colony, but different colony values, and additivity (“parallelism”) in recapture rates (ϕ_c, p_{c+t}).
- Before we leave this chapter, it is worth noting that the “parallelism” we have been discussing refers only to the logit scale - i.e., it is not linear parallelism, but logit parallelism. Thus, if you plot the reconstituted values from an additive model, they may not “look” to be parallel, but on the logit scale, they indeed are.
- Suppose we had found additivity of survival between colonies for the swift data. What would this mean? It would mean that whatever factor made the survival differ overall between the colonies had a constant additive effect over time. This would imply that there is some “real difference”, possibly genetic or age, between the two colonies such that, although both of them are subject to fluctuations over time, one colony always does relatively better (or worse) than the other.

We’re done...at last! Chapter 6 is a **long** chapter, with many concepts and technical details. However, it is also one of the most important chapters, since it covers one of the most useful tools available with SURGE. It is very important that you understand this material, so if you’re at all unsure of the material, go through it again. Your efforts will ultimately be rewarded - you’ll find that using the linear models approach with SURGE will enable you to fit a wide variety of complex analytical models, quickly and easily.

