CUBIC SPLINE ESTIMATE OF FITNESS SURFACE

USING PROJECTION PURSUIT REGRESSION

Version 1.3, January 22, 2003

Dolph Schluter
Department of Zoology
University of British Columbia
Vancouver, B.C. Canada V6T 1Z4
schluter@zoology.ubc.ca

and

Douglas Nychka
Geophysical Statistics Project
Climate and Global Dynamics Division
1850 Table Mesa Dr., Boulder, CO 80305
nychka@ucar.edu

A) Note on version 1.3

Version 1.3 is only a slight modification of version 1.2 but is provided here because newer PC's were having difficulty running the old version. Small wonder: version 1.2 was compiled in 1993 for DOS machines. Version 1.3 makes use of extended and expanded memory of Windows and hence is free of the conventional memory restriction that limited the size of data sets that could be handled by version 1.2 (and that in some cases prevented the program from running at all on Windows machines). Version 1.3 is compiled to handle up to 10,000 cases. This value can be increased if necessary by modifying the source code and recompiling (instructions given below).

The only other difference is that lambda has been rescaled in version 1.3 to match that now used in the most recent version of the univariate program, GLMS.

B) Outline of method

This section assumes some prior familiarity with univariate regression using cubic splines. The univariate program GLMS is available from D.S. at www.zoology.ubc.ca/~schluter/splines.html. Example applications of the univariate method are given in Schluter (1988). The present method is based on Schluter and Nychka (1994) which should be consulted for details.

The present program estimates fitness surfaces (and other regression surfaces) using the projection pursuit approximation. The true surface f is a function of the independent variables $\mathbf{z} = [z_1, z_2, z_3, \ldots z_k]'$ such that

$$Y = f(\mathbf{z}) + \text{random error.}$$

Y is individual survival or reproductive success. Our goal is to estimate the surface f without making any prior assumptions about its shape.

The projection pursuit approximation of the surface is the additive model

$$f(\mathbf{z}) = f_1(\mathbf{a}_1'\mathbf{z}) + f_2(\mathbf{a}_2'\mathbf{z}) + \ldots f_m(\mathbf{a}_m'\mathbf{z})$$

$$= f_1(x_1) + f_2(x_2) + \ldots f_m(x_m)$$

where $x_1, x_2, \ldots, x_m$ are new 1-dimensional variables created as linear combinations of the original set. The coefficients of these combinations are the m vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_m$, known as the projections ($'$ indicates transpose). The above approximation simplifies the task of estimating the full multivariate surface to one of estimating a small number of univariate functions $f_1, f_2, \ldots f_m$ in the directions described by the coefficients $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_m$. The $x_1, x_2, \ldots, x_m$ are new 1-dimensional variables created as linear combinations of the original set. The coefficients may be loosely though of as "principal components," but they are not chosen to maximize explained variation in $\mathbf{z}$; instead they are chosen to maximize explained variation in Y. The $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_m$ then contain the "factor loadings" for the components.

To estimate each univariate function $f_1, f_2, \ldots f_m$, we use the cubic spline, a flexible method of nonparametric regression. For simplicity, we use the same value of the smoothing parameter λ for each function. We choose $\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_m$, to minimize the weighted residual sum of squares

$$\text{WSS} = \Sigma\ (Y - Y_{\text{hat}})^2 / \sigma^2_{Y,}$$

Where Y_{hat} is the Y-value predicted by the spline fit and σ^2_Y is the variance in Y. The method thus estimates the true surface by taking cross sections in only the few most interesting directions (the directions that explain maximal variation in Y). These directions, and the functions $f_1, f_2, \ldots f_m$, are estimated simultaneously using an iterative algorithm known as backfitting. The first minimization for each direction is done using a coarse search over many randomly-constructed projections. Subsequent steps refine the search using a simplex algorithm.

Survival or reproductive success Y is not assumed to be normally distributed for a given phenotype $\mathbf{z}$. Survival is binary, taking on the values 0 or 1. Mating or reproductive success may be approximately Poisson (0, 1, 2, ...). To accomodate these possibilities, estimates of the surface use the iterative methods of generalized linear models. For example, for survival data fitness $g(\mathbf{z})$ is the probability of survival as a function of $\mathbf{z}$

$$g(\mathbf{z}) = \text{Prob}(Y \mid \mathbf{z}) = \exp\{f(\mathbf{z})\} / (1 + \exp\{f(\mathbf{z})\})$$

For Poisson data (e.g., number of mates),

$$g(\mathbf{z}) = \exp\{f(\mathbf{z})\}.$$

C) What's included in this package

---

pp.exe — The main compiled program.

surface.exe — A program that prompts the user for desired parameter settings, and then runs pp.exe.

pp13.f — Source code for the main program. These two files must be linked when compiling.

surface11.f — Source code for the program surface.exe

demo.dat — Example data set, taken from Karn and Penrose. Data are survival to 28 days of male human infants, their gestation time, and their birthweights. A single outlying observation (birthweight 13 lbs) has been deleted. Both variables were standardized to mean 0 and variance 1. Variables are listed in columns as Y, Z_1, Z_2, N, where N is the number of individuals having the same gestation and birthweight, and y is their mean survival.

demo.in — An example input file to analyse demo.dat with example parameter settings. The parameter settings are explained in the following section. Demo.in was initially created using the program surface. In this example, the program is instructed to find a single projection $\mathbf{a}_1$ describing a linear combination of gestation and birthweight. The value –2 is used for the

smoothing parameter λ (natural log scale).

demo.out — Output from the program pp.exe using demo.in as input file.

pp.manual.pdf —The present manual.

D) Running pp.exe

To run the program pp.exe using the demonstration data set, open a command window and type

```
pp < demo.in
```

to see output on screen, or

```
pp < demo.in > outfile
```

to send results to a file. Until you are proficient at editing the input file, it is best to begin a new analysis by running surface.exe. This will create an input file for your data, which you can edit later if you wish to rerun the analysis with altered parameter settings. The following section explains the input parameters.

Don't be surprised if the program takes a long time to finish. Running the program using demo.dat and demo.in as input took about 50 seconds on my 300 MHz Pentium II. Expect the time required to increase with the number of variables, cases, projections, and iterations.

E) Running surface.exe to input parameters

Type

```
surface
```

and reply to the prompts. This section explains the input requested, and other parameter settings. The values are written to the file pp.in which is created by this program.

File structure—Data should be listed in columns beginning with *Y* (survival or reproductive success), and followed by the *z*-variables (measured traits). An optional final column is for *N*, the number of individuals having the same values of the *z*-variables. In this case, *Y* is the mean survival or reproductive success of the *N* individuals.

The maximum number of cases that can be read from a file is set to 10000 in the general version of the program. See the program compilation notes below for information on how to increase this number.

Z-variables—These refer to the measured phenotypic traits. If the different traits are not in the

same units, or have very different variances, then they should probably be standardized prior to analysis. Presently, the maximum number of traits that can be analysed is 12.

Number of projections—The desired number of directions, or cross-sections of the surface. For any data set it is best to begin with 1 projection in order to keep things tractable. This is especially important if you intend to test and compare a series of values of λ . In later analyses you can add extra dimensions if you are curious or the data seem to warrant further analysis. We strongly recommend that you use a fixed value of λ when more than one direction is sought (use the best λ obtained from the preliminary analysis that sought only 1 projection) otherwise matters get very complex. The maximum number of directions that can be sought is presently set to 4.

Number of random directions—To find the best directions (those minimizing WSS) the program begins by evaluating the fit at a specified number of random directions. If the specified number is too small, then the search will not reliably find the best direction, and separate searches using different random number seeds will not converge. Hence the number of random directions should be large. When there are only two traits, and only a single projection is sought (as in the example data set demo.dat), we have found that at least 2000 random directions is needed. When looking for 2 projections from 6 original variables we found 6000 to be a minimum. A good strategy is to start with a reasonable number (e.g., 5000), and run the program several times using different random number seeds. If the final solutions from different seeds are not close to one another, then it would be a good idea to increase the number of random directions until you get convergence.

Integer seed—Input an integer to begin the random projection search. Searches begun with the same seed (and the same number of random directions and projections) will be identical.

Model type—Choose option 1 if the residuals are approximately normally distributed with equal variance. Choose option 2 if the data are survival (between 0 and 1). Choose option 3 if the data are number of mates, recruits, etc—i.e., always greater than 0, with the variance of the residuals increasing with increasing mean Y. Note that if option 2 is chosen then Y must be between 0 and 1 (inclusive), and if option 3 is chosen then Y must be greater than or equal to 0.

One vs range of lambdas—A range of λ-values may be tested (natural log scale), and their GCV scores compared. This is recommended only when fitting a single projection (if you are fitting more than one projection then we recommend that you use one fixed value of λ in order to keep things as simple as possible). To begin, try -10 to 10 in intervals of 2. Most data sets will not warrant λ values smaller than -14. One can get a feeling for the complexity of an estimate by the "effective number of parameters" provided in the output (see below).

Statistical inference using the bootstrap—We have included routines to carry out statistical tests and compute standard errors. The power and reliability of these quantities are unknown, and so the method should be used with caution.

The first option calculates standard errors for the predicted values Y_{hat} and for the coefficients of the directions $\mathbf{a}_1$, $\mathbf{a}_2$, ... $\mathbf{a}_m$. The output also lists the number of bootstrap replicates in which the estimated coefficient was of different sign than in the original estimate calculated from the actual data. If this number, when divided by the total number of bootstrap replicates, is less than 0.05,

then the coefficient may be deemed statistically significant (i.e., nonzero) at approximately the 5% level.

The second option allows testing the significance of an added direction. For example, if you chose 2 directions (cross sections) in the options above, then you may test whether fitting 2 directions results in a significantly better fit to the data than fitting only 1. If you chose to search for only 1 direction, then this option will test whether a regression based on 1 direction yields a significantly better fit to the data than would be expected if no natural selection were present at all. In other words, this second option, with 1 projection, is a test of whether or not the fitness surface is merely a flat plane. The number of bootstrap replicates should always be as large as possible, and should not be less than 100. This means that you should expect the analysis to take a great deal of computer time. Note that while the bootstrap approach is better than completely foregoing statistical testing, its validity in projection pursuit regression has not been thoroughly checked using simulation, nor has its statistical power been evaluated. Hence we recommend that the results be interpreted with caution. This routine will also give unpredictable results if the distribution of Y-values for a given value of $\mathbf{z}$ does not follow the distribution assumed (i.e., using the Poisson option when Y for each $\mathbf{z}$ is not in fact Poisson).

Other parameter settings—Default values are listed in the resulting file pp.in. The number of backfit iterations, the number of iterations in the simplex search, and the maximum number of iterations for the spline fit are all set to 5. Iterations for the spline fit are explained in the univariate program GLMS. They are needed to fit splines when the data are binary (0 or 1) or poisson (0, 1, 2, ...). Five iterations is usually adequate, but 7 or more may be needed if λ is small (e.g., less than 10). To increase the number of iterations, edit the input file pp.in and rerun as shown above. Other default parameter settings are: size of search regions for coarse and fine simplex (eps and eps2), convergence tolerance for spline fit (stop), a parameter to control starting values for spline fit (job), and a parameter to control the volume and detail of output (iprint).

F) Interpretation of output

This section explains the output provided by the minimal setting. The output begins with a list the data file name and several of the most important parameters (these are explained in the previous section).

WSS —The next part of the output lists the weighted residual sum of squares (WSS) attained at each backfit, for each projection (cross section). Projection pursuit regression seeks to minimize this quantity. As the program proceeds the WSS should converge, such that subsequent backfit iterations do not greatly improve the fit. If convergence is not observed, then the number of backfit iterations (and possibly the number of iterations in the simplex search) should be increased above the default settings in the input file pp.in. If only one projection is sought, then the program stops after 2 backfits, the extra ones are not needed.

Results from separate runs in which different random number seeds are used will rarely yield identical values for WSS. This may occur for either of two reasons. First, a given run of the

program may only find a local minimum WSS rather than the global minimum. To circumvent this problem it is best to elevate the number of random directions sought in the search for the best directions. Second, even though separate runs of the program may be converging on the same single solution to minimize WSS the program stops before the minimum is attained, as determined by practical limits to the number of backfit iterations and other parameter settings. It is best to choose parameter settings that yield solutions after a reasonable but not exorbitant amount of computer time and in which the difference in WSS values from separate runs is tolerably low.

Note that as the program proceeds the WSS may sometimes increase with successive backfit iterations rather than decrease, but this is expected. The program is indeed converging on an estimate of f minimizing WSS, subject to the additive model described above. However, the additive assumption is not met until the later backfit iterations and until that time the WSS may appear to increase.

Iterations for spline fit—The number of iterations used in the spline fit (with non-normal data) is given to the right of the WSS. The default maximum number of iterations is set to 5. If the maximum number is actually used (as in the example data set) then it is likely that the spline fit itself has not yet converged. In this case you may wish to increase the maximum number of iterations for the spline fit to 7 (e.g.) to see if the results are greatly affected. To increase the number of iterations it will be necessary to edit the input file pp.in.

Projections—The next section of output gives the projections, each one listed in a separate column. A column contains the coefficients describing the direction of the corresponding cross section of the surface. The magnitude of the coefficient for any variable indicates its contribution to variation in fitness in that direction. The variables are listed in the same order in which they appear in the data file (i.e., variable 1 = z_1, etc.). The coefficients are scaled such that for each projection the sum of squared coefficients equals 1.

Coefficients are given without any attention paid to sign. A projection $\mathbf{a}_1$ and the projection produced by multiplying each element of $\mathbf{a}_1$ by −1 are considered to be the same. Thus

Variable	Projection 1
1	.21844
2	.55300
3	−.80404

and

Variable	Projection 1
1	−.21844
2	−.55300
3	.80404

are identical solutions.

When bootstrap replicates are carried out in order to estimate standard errors for the coefficients, then the signs of different replicates are altered in order to conform with one another. For example, if the above two projections were obtained from different bootstrap replicates, then the sign of one of them would be changed before standard errors were computed.

If more than one projection is provided, then they will usually be listed in order of importance. The quantity GCV(i) is provided to ascertain this. This quantity is the GCV score calculated using the corresponding projection alone, and is a measure of how well each projection fits the data on its own (smaller GCV score = better fit; the meaning of a GCV score is explained again below, and in Schluter 1988). However, if both of two directions fit the data nearly equally well, then the order of projections may sometimes be reversed. In this case the following two solutions should be considered identical:

Variable	Projection1	Projection2
1	.21844	−.34311
2	.55300	−.90354
3	−.80404	.25669

and

Variable	Projection1	Projection2
1	−.34311	.21844
2	−.90354	.55300
3	.25669	−.80404

Results from separate runs in which different random number seeds are used will rarely yield identical projections. If this happens, first check whether WSS is nevertheless similar in the different runs—if not, see the section above on WSS for the remedy. If WSS is similar between runs but the projections are not, then there may be two reasons. First, there may be more than one minimum WSS (rare). Second, and more likely, is that the variables in the data set are highly correlated with one another. When variables are highly correlated, many different linear combinations of them are able to predict the *Y*-values nearly equally well. This is the same problem as "multicollinearity" that plagues multiple regression analysis. If variables are highly correlated, then it will not be possible to reliably determine which of them most strongly determine fitness. However, not all is lost: One may nevertheless use the predicted *Y*-values to estimate the fitness surface, as these may not vary greatly from run to run even when the trait coefficients vary substantially.

Number of parameters—The number of parameters is a useful measure of the effective complexity of the estimated surface (the quantity does not refer to the actual number of parameters in the equation for the cubic spline, which is large and unvarying). It can be interpreted with reference to a polynomial with similar degree. In the case of a single projection, 2 effective parameters (the minimum number) indicates a function that is approximately linear. Six effective parameters (as in demo.out) refers to a function with roughly the same degree of

complexity as a 5th-order polynomial. In the case of more than one projection, the effective number of parameters should be divided by the number of projections to give an idea of the complexity of each univariate function making up the estimate of the surface. For example, if two projections are fit to a set of data, then 10 effective parameters means that functions fit to each of the two cross sections of the surface have about the same complexity as a 4th-order polynomial. This quantity sometimes serves as a useful warning, particularly when data sets are small (e.g., $N<5000$: if your estimate of f using two projections has on the order of 18 effective projections, then something has likely gone wrong). A technical explanation of the effective number of parameters is given in the Appendix of Schluter and Nychka (1994).

GCV ***score***—This is a measure of prediction errors associated with a particular value of the smoothing parameter λ . It is explained in more detail in Schluter (1988) and in the information accompanying our earlier univariate program GLMS. The GCV score provides an objective criterion for choosing the best value of λ —that is, to minimize GCV. The program computes the overall GCV score for each of the λ values requested, so that a minimum may be identified. In some cases no minimum will be found (i.e., the minimum occurs at a very small value of $\ln(\lambda)$, and is accompanied by a large number of effective parameters (see above)). The cause of this problem is not entirely clear, but it is usually associated with small data sets. In this case one must choose an appropriate (larger) value of λ subjectively, in order to produce a more reasonable estimate of f. We have also noticed that the "best" λ is often too small when we test a range of λ values while searching for two or more projections. (This is why we usually recommend that a range of λ's be tested only when a single projections is fit to the data. Once the best λ is found using 1 projection, the same (fixed) value can then be used when search for extra projections.) Another problem that sometimes arises is that more than one minimum GCV may be found when λ is varied. The most conservative solution in this case is to choose the largest of the λ values corresponding to a GCV minimum (alternatively, both estimates of the surface may be presented). These problems are discussed in more detail in the GLMS package.

Predicted values for Y—These are provided as output only when a single value of λ is chosen. The following variables are listed in separate columns: Y is the observed survival or reproductive success of individuals. Y_{hat} is the corresponding predicted value (probability of survival, mean number of mates, etc.). *pX1* is each individual's measurement along the first cross section of the surface. It is computed as a linear combination of the original z-variables using the coefficients of the first projection $\mathbf{a}_1$. f_1 is the "ridge function" corresponding to the first projection. It contains the predicted values of Y along the first cross section of the surface. In the case of a single projection, and normally-distributed residuals, f_1 and Y_{hat} will be identical. With a single projection and survival (0 or 1) data f_1 will be the predicted value of Y on the logit scale (i.e., $f_1 = \ln(Y_{hat}/(1- Y_{hat}))$. In the case of Poisson data (e.g., number of mates), $f_1 = \ln(Y_{hat})$.

If two or more projections are requested, the output also includes individual measurements (pX) along the corresponding directions and the associated predicted values.

G) Program compilation notes

I compiled the source code using the GNU Fortran 77 compiler g77 for Windows included in the distribution package gcc-2.95. g77 is just a front-end for gcc, the GNU C/C++ compiler. The package consisting of gcc and g77 is available free at http://www.xraylith.wisc.edu/~khan/software/gnu-win32/gcc.html. I used the Mingw32 version of the CRTDLL runtime libraries, gcc-2.95.2-crtdll.exe. More recent versions of the compiler are available at http://gcc.gnu.org/.

If you wish to edit and recompile the source code, first install gcc on your computer by following the instructions accompanying that program. After you install, and have modified your PATH statement in AUTOEXEC.BAT to include the gcc command directory, compile the source code in glms.f using the command:

```
g77 -fugly-assumed pp13.f -o pp.exe
```

You may wish to increase the size of the arrays, to allow analysis of larger data sets, or more variables, or more projections. To change the number of cases, alter the parameter statements in the source code files pp.f: change ALL instances of "nmax=10000" to "nmax=20000" or whatever the desired new settings. When editing the source code files, it is a good idea to scan the text for every instance of the word "parameter" to ensure that you have modified them all consistently.

The program surface.exe is a stand-alone program that should not be linked with other code when being compiled.

```
g77 -fugly-assumed surface11.f -o surface.exe
```

H) Acknowledgments

Thanks to Patrick Phillips (University of Oregon, Eugene), who kindly spent many hours testing the program and helped to uncover some subtle bugs.

I) References

Schluter, D. 1988. Estimating the form of natural selection on a quantitative trait. Evolution 42:849-861.

Schluter, D. and D. Nychka. 1994. Exploring fitness surfaces. American Naturalist 143: 597-616.