Assignment 3
MATH 3423 - Numerical Methods 2
Acadia University

Daniel Lemire, Ph.D.

Winter 2002
Due March 18th 2002

1 Requirements

You can do this assignment in teams of two or alone. 5% of the mark will be based on presentation. All required plots should be computer-generated. A title page is required. All computer code must be properly commented and should be computationally efficient. **For this one assignment, you are required to use Matlab.**

2 Linear Splines

2.1 Practice problems

1. Write a Matlab routine $interp(x1,y1,z1,x2,y2,z2,x3,y3,z3,x,y)$ doing linear interpolation over the three data points $(x1,y1,z1),...,(x3,y3,z3)$ at the point x,y. The formula was given in the solution of the previous assignment

$$z = \frac{\begin{array}{c}((x1-x)y2+(x-x2)y1+(x2-x1)y)z_3+\\((x-x1)y3+(x3-x)y1+(x1-x3)y)z_2+\\((x2-x)y3+(x-x3)y2+(x3-x2)y)z_1\end{array}}{(x2-x1)y3+(x1-x3)y2+(x3-x2)y1}$$

   Make sure that your routine works no matter where the data points are! (No division by zero allowed!!!) See below for strategies to handle pathological cases.

2. Given an array of *xarray* values and an array of *yarray* values and two values $(xtarget,ytarget)$, write 4 routines: $matchlowerleft$, $matchupperleft$, $matchlowerright$, and $matchupperright$. The routines must find, respectively, the closest point described by *xarray* and *yarray* so that its x and y values are lower than the target, y value greater but x value smaller,... You must handle pathological cases where there is no solution.

3. Generalize your routine from part 2 for the case where the x values are considered to be *modulo* some fix number: $matchmodulolowerleft$, $matchmodulouppperleft$, $matchmodulolowerright$, and $matchmodulouppperright$. The last parameter in these functions is the modulo.

2.2 Pratical Problem

The problem submitted to us by Richard Karsten is a somewhat open problem. I don't know what the best way of solving it is. Richard doesn't and you probably don't. We will try to find the best way. Feel free to share your ideas, since what we want to do most is learn. Credit will be granted for shared ideas. Obviously, you are still somewhat "competing" to get the best solution since we have to select the best code at the end.

Your function takes as its input a range of N latitudes, N longitudes, N data points, and some target latitudes and longitudes. One way to look at the problem (admittedly not the most efficient!) is to pick up each target point and search 4 neighbours. This can be done relatively efficiently (meaning $O(N)$) since you can simply go through all data

points and search for the closest data point with latitude $\leq$ and longitude $\leq$, for the closest data point with latitude $\geq$ and longitude $\leq$, and so on. At that point we have some problems to handle.

1. You must assume that the data is periodic along the y axis (longitude). It implies that longitude should always be manipulated with "modulo". Hence the distance function must be "smart". Given x and y, one distance function modulo n is given by $d(x,y) = \min\{|x-y|, |x-y-n|, |x-y+n|\}$. You have a similar distance function for your problem (in 2D).

2. Latitude-wise, there might not be any data point above, or below the current target. Hence, the search might fail. One way to solve this problem is to "reflect" the data points. That is, if you don't find a neighbour with latitude $\leq$ and longitude $\leq$, then use the value of the neighbour with latitude $\geq$ and longitude $\leq$. (More sophisticated routines could be made up.)

3. Your target might be such that the 4 neighbours are not necessarily distinct. For example, your target might right on a data point and so all 4 neighbours are the same.

How do you compute the interpolated value given the 4 closest points n_0, n_1, n_2, n_3? Let y_0, y_1, y_2, y_3 be the values for each neighbour and let x be the position of our target and t its (unknown) value.

Only one distinct neighbour n_0 interpolate to the value of this neighbour : $t = y_0$

Only two distinct neighbour n_0,n_1 interpolate using linear interpolation given a distance function d: $t = \frac{d(n_0,x)y_0 + d(n_1,x)y_1}{d(n_0,n_1)}$.

Three distinct neighbour n_0,n_1, n_2 Use your result from question 1 in the previous section. You must interpolate using a linear polynomial.

Four distinct neighbour n_0,n_1, n_2, n_3 In general, we can't interpolate 4 points a linear polynomial. At most, we can handle 3 points. Therefore, we want to reject one of the points if possible. You could try to reject the farthest neighbour. If that doesn't work (two or more neighbours are a the farthest distance), try interpolating using more than one triangle and taking the average.

Finally, you must not spread "land". This means that when interpolating, if only one of your neighbor is land, you should consider rejecting the value. If more than one neighboor is land, then make the interpolated value land. That's one algorithm you might use.