448 Project Notes

Slide 1: Title

* Title

Slide 2: Background

* Paper written by Alexander Stotsky and Attila Forgo
* Created a new computationally efficient recursive spline interpolation algorithm
* Designed for engine development and real time control of Volvo cars. They need a very accurate spline to ensure controllability of the car.
* Tested there model using crankshaft angular data. Being able to predict how the crankshaft will move with respect to time.
* Supposed we have a set of measured data labelled {alpha\_0 to alpha\_k} measured at {t\_0 to t\_k}
* We are going to create a window of size w
* So now we’re just looking at the data {alpha\_0 to alpha\_w-1}
* We can choose to start to start at t = 0 or we can choose to start at any point on the curve.
* The next step is to build our polynomial through the given points in the window
* We are then going to slide our window so now we’re looking at the data {alpha\_1 to alpha\_w}. We will continually do this to model the next point on the curve.
* The sliding of the window is where the recursion comes from. We can use the previous step to model the next step

Slide 3: Background on equations

* First equation is our interpolation polynomial. Alpha-hat is our estimate of the measured signal. In this case the crank shaft signal.
* Second is our equation is our squared error. We want to minimize this at every step
* Accomplish this by taking the derivative with respect to c and setting it equal to 0.
* That’s how we get these sets of equations. We will then use these to solve for our coefficients
* To solve for our coefficients, first we need to compute all the sums in the equation
* If we have a large window size these equations can be very computationally heavy and difficult to compute even for a computer. So they created a recursive algorithm to compute these sums

Slide 4: Recursion

* Our equations on the left models the sums on the left hand side of the system of equations.
* Where m = 1-2n where n Is our order. We define the sum of order m at step k as: (where k determines what set of values are in our window)
* The equations on the right model the right hand side of the equation
* We can see how with a large window there’s going to be a lot of computations

Slide 5: Recursion Cont

* Using the equations from the last slide they are able to come up with these recursive equations to make the computation much easier.

Slide 6: Solving for coefficients

* We want to solve for our coefficients using the matrix equation Ac = b
* Where c is our coefficients and b is the right hand side of our original linear system

Slide 7: Their Results

* Both these pictures represent the comparison of two engines as a function of time. The engines actual speed is the dotted lines while the solid line is the calculated engine speed using their interpolation method
* The only parameter to be optimized is the window size w. If the derivative is moving slowly you can use a large window size. But if it’s changing quickly then you must use a smaller window size to be able to pick it up better. However since the method accumulates data as the window shifts a small window size can result in a big error.
* They were able to reduce their error by repeated initialization of the algorithm.

Slide 8: My Results

* I studied sin(x) with 20 points from 0 to 1.
* First I used w= 9 and w=4