

Re: Derivative of splinefit w.r.t. control points.

Source: <http://sci.tech-archive.net/Archive/sci.math.num-analysis/2005-09/msg00160.html>

- *From:* spellucci@xx (Peter Spellucci)
 - *Date:* Wed, 21 Sep 2005 17:59:04 +0000 (UTC)
-

In article <1dc2c\$433176d6\$82a1bc06\$7365@xxxxxxxxxxxxxxxxxxxx>, Maurice van de Rijzen <bertvansesamamstraat@xxxxxxxxxxxx> writes:

>Dear Peter,
>
>I already thought so.
>But my problem is that I use the spline-toolbox of Matlab and therefore
>I don't have direct acces to the equations.
>Furthermore is my problem bivariate the makes bit more difficult.
>I use a natural cubic spline.
>I hoped it would be possible to retrieve them in one way or the other
>from the spline-coefficients either on the polynomials or B-spline
>formulation.
>Do you know whether this would be possible?
>Thanks,
>Maurice
>

the natural cubic spline can be computed quite simply by a tridiagonal system in 1D and a block tridiagonal one for the tensor product 2D case. the good message: the spline coefficients are of course linearly dependent on the function values, (but not the node positions!) hence the simple forward difference quotient, with a stepsize of one say, gives already the correct derivative numerically, hence you can use the toolbox as a black box and compute those derivatives numerically. the coefficients are given by a strucutre and you must access a component of the structure in order to get direct access to the coefficients. then you build a correponding new structure for the derivatives of the spline with respect to the y's. however, in 2D, this might give quite a huge number of computations. but anyway, also done directly done analytically, this gives lots of work, but the speciality is that you have to solve systems with the same matrix and many different right hand sides. for example , for the moments of the spline in 1D the system is tridiagonal
$$A(h(2), \dots, h(n)) * (m(2), \dots, m(n-1))' = ((y(3)-y(2))/h(2) - (y(2)-y(1))/h(2), \dots, (y(n)-y(n-1))/h(n) - (y(n-1)-y(n-2))/h(n-1))'$$
where the data are $(x(1), y(1)), \dots, (x(n), y(n))$, $m(i) = s''(x(i))/6$
 $h(i) = x(i) - x(i-1)$ $i=2, \dots, n$ and
 $A = \text{tridiag}(h(i), 2*(h(i)+h(i+1)), h(i+1); i=2, \dots, n-1)$
 $m(1) = m(n) = 0$ (the natural end condition). if you differentiate this with respect

Re: Derivative of splinefit w.r.t. control points.

to $y(3)$ say, then you get on the right hand side only two entries not=0 and the same coefficient matrix. hence using this property directly would save a lot of work.

hth
peter

>> In article <26396943.1127226707247.JavaMail.jakarta@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>,
>> Maurice <bertvansesamstraat@xxxxxxxxxxxx> writes:
>> >Dear All,
>> >I'm working on an optimization problem using splines.
>> >Now I've the following question.
>> >Say that I've a cubic natural spline $f(x)$.
>> >This implies that the variables of the function are the values at the control points, $(y(0)..y(N))$.
>> >Is it possible to determine the derivative of the fit in a certain evaluation point analytically.
>> > $df/dy(0) | x=x_eval$.
>> >Thanks in advance,
>> >Maurice
>>
>> yes of course. go through the complete algorithm to compute the spline and
>> differentiate each equation , using chain rule, with respect to the y's.
>> then you will get the derivative of the splines' coefficients with respect to
>> the data and can evaluate the derivative of the spline with respect to the data
>> by evaluating the representation with the new coefficients instead of the
>> original ones.
>> hth
>> peter
>>
>> .

• **References:**

◆ **[Re: Derivative of splinefit w.r.t. control points.](#)**

◇ *From:* Peter Spellucci

• Prev by Date: **[Re: bounds on Raleigh quotient](#)**

• Next by Date: **[Re: Derivative of splinefit w.r.t. control points.](#)**

• Previous by thread: **[Re: Derivative of splinefit w.r.t. control points.](#)**

• Next by thread: **[Re: Derivative of splinefit w.r.t. control points.](#)**

• Index(es):

◆ **[Date](#)**

◆ **[Thread](#)**