

Re: Help solving a bounded linear-least squares problem.

Source: <http://sci.tech-archive.net/Archive/sci.math.num-analysis/2006-12/msg00015.html>

- *From:* "svlad" <madsvlad@xxxxxxxx>
 - *Date:* 29 Nov 2006 09:52:11 -0800
-

Peter Spellucci wrote:

In article <1164762109.180850.262750@xx>, "svlad" <madsvlad@xxxxxxxx> writes:

>First off, thank you so much for you time and patience. It is
>appreciated.....

>

>> >

>> > x p21

>> > p31 x |

>> > \ |

>> > \ |

>> > \ |

>> > \o-----x p11

>> >

>> this looks as if you had a set of
>> different directions : the same directions
>> for all the points? and the stepsize: different
>> for the different directions?
>> but below you apply different t's for the
>> different coordinates????

>

>What the ASCII diagram above is supposed to (sorry for the confusion)
>depict is how a single point (m=1) might be affected by three variables
>(n=3). Each variable corresponds to an equation for linear
>interpolation along a line. The lines associated with each variable
>have the same starting point, but obviously different endpoints.

>

>Now....each of the m points has its own set of n line equations for the
>n variables i.e. the same n variables move each point differently. The
>goal here is find a set of values for the n variables which minimizes
>the distance between each point and its corresponding target position.

>

>Each of the n columns of the X matrix corresponds to the vector of the
>line defined by one of the n variables. There are actually (m*3) rows
>in the matrix to account for the fact that I have points in R^3. This
>is why I believe that

Re: Help solving a bounded linear-least squares problem.

>
>y = z + X * t
>
>is the correct formulation of this problem.
>
>>>My goal is to find values for the variables t that minimizes the
>>>distance between the point and its target position. By defining di as
>>>the delta of (pi1 - p0) the problem can be stated in matrix-vector
>>>form:
>>>
>>>| d11 d12 . . . d1n | z = [p10 p20 . . . pm0]'
>>>| d21 d22 . . . d2n |
>>>X = | . . . | y = [tp1 tp2 . . . tpm]'
>>>| . . . |
>>>| dm1 dm2 . . . dmn |
>>>
>>>X is a mxn matrix containing the n delta values for the m points.
>>
>> in the rows!
>>
>>>z is a m-vector containing the original position of each point
>
>> should this be a matrix?? or what do you mean by
>> position???
>
>Sorry....I was taking a notational short-cut. My points are in R^3 each
>point has three line equations (x, y, z):
>X is a (m*3) x n matrix containing the n delta values for the m points
>(dx, dy, dz).
>z is a (m*3) vector containing the original position of the each point
>in R^3.
>y is a (m*3) vector containing the target positions for each point in
>R^3.
>t is still a n-vector containing the variable to be optimized.
>
>>>y is a m-vector containing the target positions for each point.
>>>t is a n-vector containing the variables to be optimized.
>>>
>>>The problem can now be stated in matrix-vector terms as:
>>>
>>>y = z + (X * t)
>>>
>>>Since I need to minimize the distance I have:
>>>
>>>d2 = (y - z + X * t)^2
>>>= (y - z)'*(y - z) + (y - z)' * X*t + t'*X'*X*t
>> one again the "2" is missing here
>
>A typo I make almost everytime :(
>
>> there is a confusion:

Re: Help solving a bounded linear-least squares problem.

Re: Help solving a bounded linear-least squares problem.

>> let us assume that your problem formulation is correct (I don't believe this)
>> so we have
>>
> $f(t) = (y - z + X*t)^2 = \min$ subject to $t(i)$ in $[0,1]$, $i=1, \dots, n$
>> and y, z are in m -dimensional space, X is m times n
>> then the Kuhn Tucker conditions are
>>
>> $2*X'(y - z - X*t) - \lambda_1 + \lambda_2 = 0$
>> $\lambda_1(i)*t(i) = 0$
>> $\lambda_2(i)*(1 - t(i)) = 0$
>> $\lambda_1(i), \lambda_2(i) \geq 0, 0 \leq t(i) \leq 1$
>>
>> you can solve this using e.g. SOR with projection on the box.
>> since the multipliers λ_1 and λ_2 cannot be nonzero simultaneously,
>> you can read off from the sign of the gradient what to do: move the $t(i)$
>> or fix it at the bound
>
> Is it also valid to use the magnitude of the gradient in the manner of
> the BVLS active set strategy to determine which variable most wants to
> become active? If that is true and I solve the variables in the order
> of greatest gradient magnitude to smallest gradient magnitude, could I
> then implement the mutex constraints by clamping a variable to its
> lower bound (0) if one of its mutex variables is not at its lower
> bound? That would be cheap and easy.
>
> Thanks again!
>

Now I got it. o.k. but the input to BVLS must be
matrix X
right hand side $y - z$
variable t in the $[0,1]$ box,
not the gradient !

In a "DOH!" moment, that just occurred to me last night.....a very
silly mistake on my part :(I should've looked a closer at what BVLS
was really doing.

Thank you for your assistance!

.