Mixed Integer Programming applied to Image Partitioning

May 17, 2014

Given a bunch of image points $(t_1, f(t_1))$, $(t_2, f(t_2))$, ..., $(t_n, f(t_n))$ on a 2−dimension graph, we want to approximate these points using a piece-wise linear function $f^\star$:

$$f^\star(t_i) = \begin{cases} a_1 t_i + b_1 & \text{if } t_i \in \Omega_1 \\ a_2 t_i + b_2 & \text{if } t_i \in \Omega_2 \end{cases} \qquad \forall\, t_i,$$

where $\Omega_1 = \{t_i :\ \mu(t_i) = 0\}$, $\Omega_2 = \{t_i :\ \mu(t_i) = 1\}$, and $\Omega = \Omega_1 \cup \Omega_2$ is the whole discrete points space, and μ is the indicator function that divide the the space Ω into two.

If we already decide Ω_1 and Ω_2, then the optimazation problem becomes the following min-square error problem:

$$\min_{a_1,a_2,b_1,b_2} \sum_{t_i \in \Omega} (f(t_i) - f^\star(t_i))^2,$$

or more specificly:

$$\min_{a_1,b_1} \sum_{t_i \in \Omega_1} (f(t_i) - (a_1 t_i + b_1))^2 + \min_{a_2,b_2} \sum_{t_i \in \Omega_2} (f(t_i) - (a_2 t_i + b_2))^2,$$

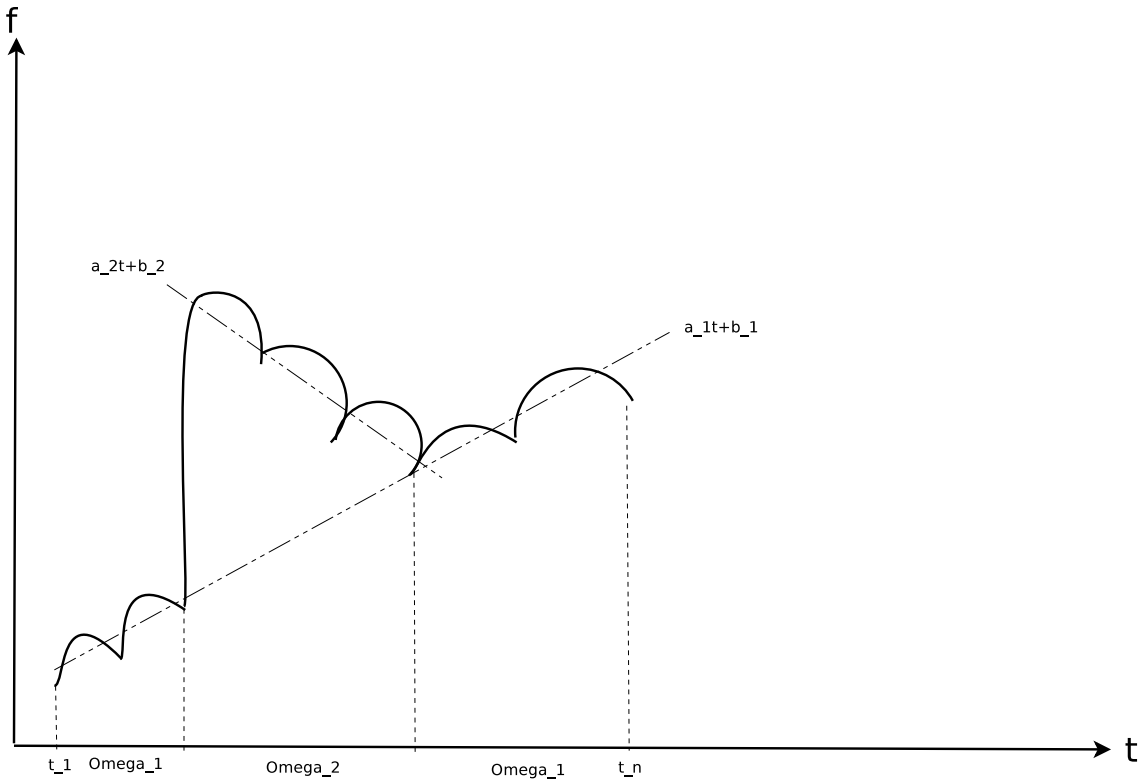
which is not hard to solve.

Then the question now becomes how to decide the function μ, or equvilently, how to decide the partition Ω_1 and Ω_2, in order to best approximate the image points. Hence, this problem is eventually a bi-level optimization problem:

$$\min_{\mu} \{ \min_{a_1,b_1} \sum_{t_i \in \Omega_1} (f(t_i) - (a_1 t_i + b_1))^2 + \min_{a_2,b_2} \sum_{t_i \in \Omega_2} (f(t_i) - (a_2 t_i + b_2))^2 \}.$$

Note that here every partition of Ω is feasible, so that there are 2^n of possibilities, given there are n such image points.

The following graph gives a brief idea of what's going on here.



Simplified model 1:

Suppose now we change the assumption a little bit, we are given the slopes of these two linear functions, say a_1, a_2, but the intercepts remain unkonwn. Hence, we can denote $f^\star(t_i) = a_1 t_i + b_i$ as type 1 if using slope a_1, and $f^\star(t_i) = a_2 t_i + b_i$ as type 2. Now, let

$$\bar{x}_{ij} = \begin{cases} 1 & \text{if } t_i \in \text{type } j \\ 0 & \text{otherwise} \end{cases} \qquad \forall\, t_i, i = 1, 2, \ldots, N, \textit{and } j = 1, 2.$$

Using the above binary variable $\bar{x}_{ij}$, we can write $f^\star(t_i) = \sum_{j=1}^{2} a_j \bar{x}_{ij} t_i + b_i$, where $b_i \in R$ is a real variable. Now, we can model the bilevel programming as a one-level quadratic programming as follows:

$$\min_{\bar{x}_{i1}, \bar{x}_{i2}, b_i} \sum_{i=1}^{N} (f^\star(t_i) - f(t_i))^2 + M,$$

where M is a penalty function on the number of partitions, to fight against the case of noisy nodes. For example,

$$M = \sum_{i=1}^{N-1} (b_{i+1} - b_i)^2 + \sum_{j=1}^{2} \sum_{i=1}^{N-1} (\bar{x}_{i+1,j} - \bar{x}_{ij}).$$

Thus, under these assumptions, we now have a quadrative problem which has $2N$ of binary variabls $\bar{x}_{i1}$ and N of real variables b_i.

$$\min_{\bar{x}_{i1}, \bar{x}_{i2}, b_i} \sum_{i=1}^{N} (f^\star(t_i) - f(t_i))^2 + \sum_{i=1}^{N-1} (b_{i+1} - b_i)^2 + \sum_{j=1}^{2} \sum_{i=1}^{N-1} (\bar{x}_{i+1,j} - \bar{x}_{ij}).$$

such that $\sum_{j=1}^{2} \sum_{i=1}^{N-1} \bar{x}_{ij} = N$, and where $\bar{x}_{i1} \in \{0, 1\}$, $b_i \in R$ are variables.

Using Xpress, we managed to run examples of 100 points, and results are given within 1 sec.

Simplified model 2:

Suppose this time, we are given the number of partitions, say $k + 1$, let

$$\bar{s}_i = \begin{cases} 1 & \text{if } i \text{ is the jump point between two partitions,} \\ 0 & \text{otherwise.} \end{cases} \qquad \forall\, i = 1, 2, \ldots, N.$$

Then the constraint $\sum_{i=1}^{N} s_i = k$ should be satisfied, and inside each k partitions, we can do a linear regression independtly, with both slope and

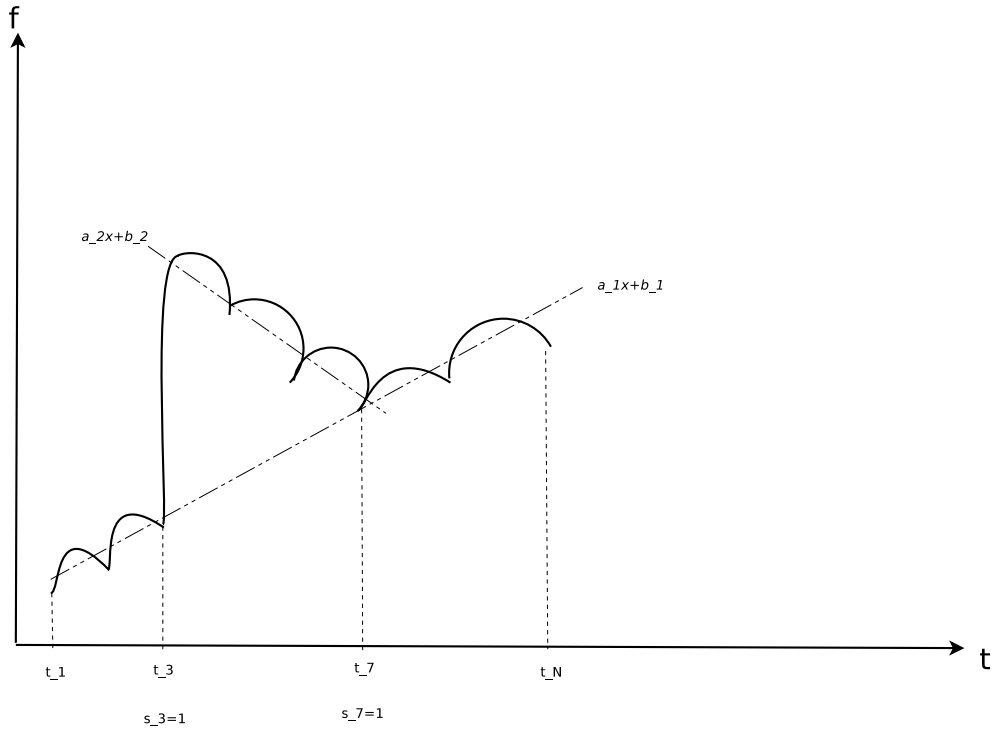
intercept unknown. So, we have now a one-level problem, with N of binary variables s, and $2N$ real variables of a_i and b_i, for $i = 1, 2, \ldots, N$.

$$\min_{s_i,a_i,b_i} \sum_{i=1}^{i_1}(a_1t_i+b_1-f(t_i))^2+\sum_{i=i_1+1}^{i_2}(a_2t_i+b_2-f(t_i))^2+\ldots+\sum_{i=i_k+1}^{N}(a_1t_i+b_1-f(t_i))^2,$$

where i_1 is the first indice for which $S_{i_1} = 1$, and so on. For example, in the graph below, we have $i_1 = 3$ and $i_2 = 7$, and point t_1 up to t_3 form the first partition, and t_4 up to t_7 form the second partition, and so on. And inside each partition, we do a linear approximation independently.

In other words, i_j is a function of s, for $j = 1, 2, \ldots, k$, which makes this model unsolvable. Note that inside each partition, is just a min-square-error problem.

The following graph gives a brief idea of what's going on here.



Model 2:

Suppose this time, we upper bound the number of partitions, say k, and introduce a binary variable $x_{i,l}$,

$$x_{i,l} = \begin{cases} 1 & \text{if } i \text{ is partition } l, \\ 0 & \text{otherwise.} \end{cases} \qquad \forall\, i = 1, 2, \ldots, N, l = 1, 2, \ldots, k$$

Since every point must lie in one partition, we have

$$\sum_{l=1}^{k} x_{i,l} = 1,\ \forall i = 1, 2, \ldots, N.$$

Also, the partition in which point $i+1$ lies should be no less than that of point i,

$$\sum_{l=1}^{k} l * x_{i,l} \leq \sum_{l=1}^{k} l * x_{i+1,l},\ \forall i = 1, 2, \ldots, N-1.$$

Inside each partition, we do a linear regression. We also want to add penalty for the number of partitions, say S,

$$S = \sum_{i=1}^{N-1} (\sum_{l=1}^{k} l * x_{i+1,l} - \sum_{l=1}^{k} l * x_{i,l}) + 1$$

Let a_l, b_l be the linear regression coefficient of partition l, for $l = 1, 2, .., k$. Upon above settings, our model becomes:

$$\begin{aligned} \underset{x_{i,l}, a_l, b_l}{\text{minimize}} \quad & \lambda * S + \sum_{l=1}^{k} \sum_{i=1}^{N} x_{i,l} * |a_l t_i + b_l - f(t_i)| \\ \text{subject to} \quad & \sum_{l=1}^{k} x_{i,l} = 1,\ \forall i = 1, 2, \ldots, N, \\ & \sum_{l=1}^{k} l * x_{i,l} \leq \sum_{l=1}^{k} l * x_{i+1,l},\ \forall i = 1, 2, \ldots, N-1, \end{aligned} \tag{1}$$

where λ is the penalty coefficient on the number of partitions S.

One thing to notice is that the min-square-error problem can be translated into a 1-norm problem, by noticing that

$$\min \sum (x - y)^2$$

is equivalent to the system

$$\min \sum t$$

subject to $t \geq x - y$ and $t \geq y - x$.

Also, a quadratic term $x * y$ with x binary and y bounded by $L \leq y \leq U$, can be turned into a linear one z, by the following transformation:

$$\begin{aligned} &Lx \leq z \leq Ux, \\ &y - U(1-x) \leq z \leq y - L(1-x). \end{aligned}$$

Thus, by letting $c_{i,l} = a_l t_i + b_l - f(t_i)$, our model becomes:

$$\begin{aligned} \underset{x_{i,l},a_l,b_l}{\text{minimize}} \quad & \lambda * S + \sum_{l=1}^{k} \sum_{i=1}^{N} x_{i,l} * c_{i,l} \\ \text{subject to} \quad & \sum_{l=1}^{k} x_{i,l} = 1, \quad \forall i = 1, 2, \ldots, N, \\ & \sum_{l=1}^{k} l * x_{i,l} \leq \sum_{l=1}^{k} l * x_{i+1,l}, \quad \forall i = 1, 2, \ldots, N-1, \\ & c_{il} \geq a_l t_i + b_l - f(t_i), \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k, \\ & c_{il} \geq f(t_i) - a_l t_i - b_l, \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k. \end{aligned}$$

Then, by noticing that $x_{i,l}$ is binary, $c_{i,l} \geq 0$, and we assume $c_{i,l}$ has an upper bound M, $\forall i = 1, 2, \ldots, N, l =, 1, 2, \ldots, k$, we then replace the quadratic term $x_{i,l} * c_{i,l}$ with $d_{i,l}$ And our model becomes:

$$\begin{aligned} \underset{x_{i,l},a_l,b_l}{\text{minimize}} \quad & \lambda * S + \sum_{l=1}^{k} \sum_{i=1}^{N} d_{i,l} \\ \text{subject to} \quad & \sum_{l=1}^{k} x_{i,l} = 1, \ \forall i = 1, 2, \ldots, N, \\ & \sum_{l=1}^{k} l * x_{i,l} \leq \sum_{l=1}^{k} l * x_{i+1,l}, \quad \forall i = 1, 2, \ldots, N, \\ & c_{il} \geq a_l t_i + b_l - f(t_i), \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k, \\ & c_{il} \geq f(t_i) - a_l t_i - b_l, \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k, \\ & 0 \leq d_{i,l} \leq M * c_{i,l}, \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k, \\ & c_{i,l} - M * (1 - x_{i,l}) \leq d_{i,l} \leq c_{i,l}, \quad \forall i = 1, 2, \ldots, N, \ \forall l = 1, 2, \ldots, k. \end{aligned}$$

And we notice that this model is linear, thus can be solved efficiently.