Solving a Network of Sensors Problem using Gradient Descent

Guntur Ravindra
Senior Member of the ACM

November 9, 2018

1 Problem Formulation

We have multiple sensors in a constellation of sensors each represented by a feature vector $(X_j, Y_j, \alpha_j, \beta_j)$. There are multiple target points T_i that these sensors can sense. The target point is represented by a high dimensional feature vector, but for the current discussion we limit it to two dimensions that we refer to as (x_i, y_i). Our goal is to find the best values of (α_j, β_j) so that the estimated value $(\widetilde{x_i}, \widetilde{y_i})$ is as close to the true value for each target T_i.

The sensors are independent of each other when they sense the targets T_i. Nevertheless, multiple sensors can sense the same target and when they do, we would like the target feature vector to be the same across all the sensors. In addition we have a data collection phase during which each sensor is capable of recording the true value of the feature vector (x_i, y_i) in addition to a random variable described later in (1).

2 Conditions and constraints

- We have function $D_{ij} = f\left((x_i, y_i), (X_j, Y_j)\right)$ where $D_{i,j} \in \Re$.
- Then we have a relationship of the form

$$r_{ijk}, \theta_{ijk} = \alpha_j \log\left(D_{ij}\right) + \beta_j \tag{1}$$

Note how the value of $D_{i,j}$ is independent of the index k and is also independent of θ_{ijk}. Here θ_{ijk} is an angular direction detected by the j^{th} sensor for the i^{th} target T_i, and r_{ijk} is a random sensed value. This means for different values of $r_{i,j,k}$ we could have the same value of $D_{i,j}$ according to (1).

- Further the constants $\alpha_j$and β_j are dependent only on the sensors. We need to find the optimal value of α_j and and β_j for each of the j sensors such that for every r_{ijk}, the calculated value for $(X_{ijk}, Y_{ijk}) \Rightarrow (\widetilde{x_i}, \widetilde{y_i})$ for the target point T_i is as close as possible to the true feature vector (x_i, y_i). This means we have a constraint optimization function of the form

$$\arg\min_{\alpha_j, \beta_j} \sum_j \frac{1}{N_j} \sum_k \sum_i \left(X_{ijk} - x_i\right)^2 + \left(Y_{ijk} - y_i\right)^2 \tag{2}$$

where (X_{ijk}, Y_{ijk}) is a calculated feature vector for T_i determined using (1). Note that (1) does not have explicit reference to (X_{ijk}, Y_{ijk}) but we can determine D_{ij}. The sensors are directional, in the sense they can determine the direction for D_{ij} defined by θ_{ijk}.

- We define the function $f\left((x_i, y_i), (X_{ijk}, Y_{ijk})\right)$ to be the $L2$ norm. Hence we can find the actual (X_{ijk}, Y_{ijk}) from (1) by using the relationship

$$X_{ijk} = X_j + 10^{\frac{r_{ijk}-\beta_j}{\alpha_j}} \cos(\theta_{ijk}) \quad Y_{ijk} = Y_j + 10^{\frac{r_{ijk}-\beta_j}{\alpha_j}} \sin(\theta_{ijk}) \tag{3}$$

Hence the constraint function (2) is dependent on (1) through the relationship (3).

- Further, we have additional constraints on the range of values for α_j, β_j, given by

$$\begin{aligned} \alpha_j &\leq B_\alpha \\ \beta_j &\leq B_\beta \\ \alpha_j &> A_\alpha \\ \beta_j &> A_\beta \\ \left(\alpha_j^2 + \beta_j^2\right) &\leq C \end{aligned}$$

3 Solving using Gradient Descent

Because we have constraints on the estimated variables, we assimilate these constraints into the main objective function using Lagrangean relaxation. As a result we have lagrangean multipliers referred to by the symbol λ. The multipliers are given by the relationship (4)

$$\begin{aligned} &\lambda_{1,j}(-\alpha_j + A_\alpha) \leq 0 \;\; \lambda_{2,j}(\alpha_j + B_\alpha) \leq 0 \\ &\lambda_{3,j}(-\beta_j + A_\beta) \leq 0 \;\; \lambda_{4,j}(\beta_j + B_\beta) \leq 0 \\ &\lambda_{5,j}\left(\alpha_j^2 + \beta_j^2\right) \leq C \end{aligned} \tag{4}$$

The partial derivative of the Lagrangean with respect to α_j and β_j is as below

$$\frac{1}{N_j}\sum_k\sum_i \left[2 \times (X_{ijk} - X_i)\frac{\partial X_{ijk}}{\partial \alpha_j} + 2 \times (Y_{ijk} - Y_i)\frac{\partial Y_{ijk}}{\partial \alpha_j}\right] + \frac{\partial}{\partial \alpha_j}$$
$$\left(-\lambda_{1,j}(-\alpha_j + A_\alpha) - \lambda_{2,j}(\alpha_j + B_\alpha) - \lambda_{3,j}(-\beta_j + A_\beta) - \lambda_{4,j}(\beta_j + B_\beta) - \lambda_{5,j}\left(\alpha_j^2 + \beta_j^2 - C\right)\right)$$

$$\frac{1}{N_j}\sum_k\sum_i \left[2 \times (X_{ijk} - X_i)\frac{\partial X_{ijk}}{\partial \beta_j} + 2 \times (Y_{ijk} - Y_i)\frac{\partial Y_{ijk}}{\partial \beta_j}\right] + \frac{\partial}{\partial \beta_j}$$
$$\left(-\lambda_{1,j}(-\alpha_j + A_\alpha) - \lambda_{2,j}(\alpha_j + B_\alpha) - \lambda_{3,j}(-\beta_j + A_\beta) - \lambda_{4,j}(\beta_j + B_\beta) - \lambda_{5,j}\left(\alpha_j^2 + \beta_j^2 - C\right)\right)$$

Because we wish to minimize the error represented by (2), we equate the partial derivatives to zero as below and then solve them.

$$\begin{aligned} \frac{1}{N_j}\sum_k\sum_i \left[2 \times A_{ijk}\frac{\partial X_{ijk}}{\partial \alpha_j} + 2 \times B_{ijk}\frac{\partial Y_{ijk}}{\partial \alpha_j}\right] + \lambda_{1,j} - \lambda_{2,j} - 2\lambda_{5,j}\alpha_j = 0 \\ \implies \Delta\alpha_j = P + \lambda_{1,j} - \lambda_{2,j} - 2\lambda_{5,j}\alpha_j = 0 \end{aligned} \tag{5}$$

$$\begin{aligned} \frac{1}{N_j}\sum_k\sum_i \left[2 \times A_{ijk}\frac{\partial X_{ijk}}{\partial \beta_j} + 2 \times B_{ijk}\frac{\partial Y_{ijk}}{\partial \beta_j}\right] + \lambda_{3,j} - \lambda_{4,j} - 2\lambda_{5,j}\beta_j = 0 \\ \implies \Delta\beta_j = Q + \lambda_{3,j} - \lambda_{4,j} - 2\lambda_{5,j}\beta_j = 0 \end{aligned} \tag{6}$$

The relationship between the different different values of the lagrangean multipliers in terms of the differentails is given as below and the terms P and Q are the partial derivative components with respect to α_j and β_j without the lagrangean multipliers:

$$\lambda_{1j} = \frac{2\alpha_j C\left(\alpha_j + B_\alpha\right) - P(\alpha_j^2 + \beta_j^2)\left(\alpha_j + B_\alpha\right)}{\left(\alpha_j^2 + \beta_j^2\right)\left[\left(1 + \alpha_j + A_\alpha\right)\left(1 + \alpha_j + B_\alpha\right) - 1\right]} \tag{7}$$

$$\lambda_{2j} = \frac{P(\alpha_j^2 + \beta_j^2)\left(\alpha_j + A_\alpha\right) - 2\alpha_j C\left(\alpha_j + A_\alpha\right)}{\left(\alpha_j^2 + \beta_j^2\right)\left[\left(1 + \alpha_j + A_\alpha\right)\left(1 + \alpha_j + B_\alpha\right) - 1\right]} \tag{8}$$

$$\lambda_{3j} = \frac{2\beta_j C\left(\beta_j + B_\beta\right) - Q(\alpha_j^2 + \beta_j^2)\left(\beta_j + B_\beta\right)}{\left(\alpha_j^2 + \beta_j^2\right)\left[\left(1 + \beta_j + A_\beta\right)\left(1 + \beta_j + B_\beta\right) - 1\right]} \tag{9}$$

$$\lambda_{4j} = \frac{Q(\alpha_j^2 + \beta_j^2)\left(\beta_j + A_\beta\right) - 2\beta_j C\left(\beta_j + A_\beta\right)}{\left(\alpha_j^2 + \beta_j^2\right)\left[\left(1 + \beta_j + A_\beta\right)\left(1 + \beta_j + B_\beta\right) - 1\right]} \tag{10}$$

$$\lambda_{5,j} = \frac{C}{\alpha_j^2 + \beta_j^2} \tag{11}$$

We cannot derive a closed form solution by substituting (7)-(11) in (5) and (6). Hence we used gradient descent to find an iterative solution to the system of equations. The update function for α_j, β_j is given by

$$\begin{aligned} \alpha_j^{t+1} &= \alpha_j^t - \tau \Delta \alpha_j \\ \beta_j^{t+1} &= \beta_j^t - \tau \Delta \beta_j \end{aligned}$$

By substituting the updated values of α_j, β_j in (7)-(11) we get the updated values of the lagrangean multipliers for the next iteration of gradient descent.

4 Implementation

Gradient descent iterations implemented in Python without using any external optimization packages.