

# Solving the BP in sensor networks

The BP is the problem

$$\begin{aligned} & \text{minimize} && \|x\|_1 \\ & \text{subject to} && A_{\text{BP}}x = b_{\text{BP}}, \end{aligned} \tag{1}$$

where the variable is  $x \in \mathbb{R}^{n_{\text{BP}}}$ . We can recast it to a linear program in standard form:

$$\begin{aligned} & \text{minimize} && c^\top x \\ & \text{subject to} && Ax = b \\ & && x \geq 0, \end{aligned} \tag{2}$$

where  $x, c \in \mathbb{R}^n$ ,  $b \in \mathbb{R}^m$ ,  $A \in \mathbb{R}^{m \times n}$ , and

$$A = [A_{\text{BP}} \quad -A_{\text{BP}}] \quad b = b_{\text{BP}}, \quad \text{and} \quad c = \mathbf{1}_{2n_{\text{BP}}}.$$

The passage from (1) to (2) is based on the fact that any  $x \in \mathbb{R}$  can be written as  $x = x^+ - x^-$ , where  $x^+ = \max\{x, 0\}$  and  $x^- = \max\{-x, 0\}$ . We also have  $|x| = x^+ + x^-$ . It follows that  $x \in \mathbb{R}^{n_{\text{BP}}}$  in (1) can be written as  $x = u - v$ , where  $u, v \geq 0$ . Thus,  $x = [u \quad v]^\top$  and  $n = 2n_{\text{BP}}$ .

Our goal is then to solve (2). We assume  $A$  is partitioned in rows into  $P$  blocks and each block  $A_p$  has  $m_p$  rows ( $m_1 + \dots + m_P = m$ ).

$$A = \begin{bmatrix} \text{---} A_1 \text{---} \\ \vdots \\ \text{---} A_P \text{---} \end{bmatrix}$$

Rewriting (2),

$$\begin{aligned} & \text{minimize} && c^\top x \\ & \text{subject to} && A_1x = b_1 \\ & && \vdots \\ & && A_Px = b_P \\ & && x \geq 0. \end{aligned}$$

We now clone  $x$  into  $x_1, \dots, x_P$  and make the consistency equalities be according to a given network  $\mathcal{V} \times \mathcal{E} = \{1, 2, \dots, P\} \times \{\dots, (i, j), \dots\}$  with  $P$  nodes, each of which stores a block of the matrix  $A$ .

$$\begin{aligned} & \text{minimize} && \frac{1}{P} \sum_{p=1}^P c^\top x_p \\ & \text{subject to} && A_1x_1 = b_1 \\ & && \vdots \\ & && A_Px_P = b_P \\ & && x_i = x_j, (i, j) \in \mathcal{E} \\ & && x_p \geq 0, p = 1, \dots, P \end{aligned} \tag{3}$$

The next step is to calculate the dual problem of (3). For that, we will only dualize the constraints  $x_i = x_j$ , associating to each one of these equalities the dual variable  $\lambda_{ij}$ . We assume  $i < j$  for  $(i, j) \in \mathcal{E}$ . Also, we compute the dual in the augmented Lagrangian sense. The dual of (3) is unconstrained and is given by

$$\underset{\lambda}{\text{maximize}} \quad L(\lambda), \quad (4)$$

where the dual function is

$$L(\lambda) = \inf_{\substack{A_p x_p = b_p \\ x_p \geq 0 \\ p=1, \dots, P}} L(x, \lambda) \quad (5)$$

and

$$L(x, \lambda) = \frac{1}{P} \sum_{p=1}^P c^\top x_p + \sum_{(i,j) \in \mathcal{E}} \lambda_{ij}^\top (x_i - x_j) + \sum_{(i,j) \in \mathcal{E}} \frac{\rho}{2} \|x_i - x_j\|^2. \quad (6)$$

It can be proved that the dual function  $L(\lambda)$  is concave and has a Lipschitz continuous gradient with constant  $\frac{1}{\rho}$ . We solve the dual problem using Nesterov's algorithm:

---

**Algorithm 1** Nesterov's algorithm

---

**Initialization** choose  $\lambda^{(0)} = \eta^{(0)}$  and set  $k = 1$

- 1: **repeat**
  - 2:    $\lambda^{(k)} = \eta^{(k-1)} + \rho \nabla_{\lambda} L(\eta^{(k-1)})$
  - 3:    $\eta^{(k)} = \lambda^{(k)} + \frac{k-1}{k+2} (\lambda^{(k)} - \lambda^{(k-1)})$
  - 4:    $k \leftarrow k + 1$
  - 5: **until** some stopping criterion is met
- 

The above algorithm is edge-wise separable, in the sense that node  $i$  for  $(i, j) \in \mathcal{E}$  can update  $\lambda_{ij}$ . This is possible because the gradient

$$\nabla L(\lambda, \mu) = \begin{bmatrix} \vdots \\ x_i(\lambda, \mu) - x_j(\lambda, \mu) \\ \vdots \end{bmatrix}, \quad (7)$$

where  $x(\lambda) := (x_1(\lambda), \dots, x_P(\lambda))$  solves the optimization problem in (5), is ‘‘almost’’ separable. We now address the problem of computing this gradient.

**Computing the gradient of  $L(\lambda)$ .** As seen in (7), to compute  $\nabla L(\lambda)$ , we first need to find a solution of the optimization problem in (5). We will find it in an iterative way using Nesterov's (projected) algorithm again. Let  $\lambda$  be fixed and denote  $g_\lambda(x) = L(x, \lambda)$ . Our problem becomes

$$\begin{aligned} &\text{minimize} && g_\lambda(x_1, \dots, x_P) \\ &\text{subject to} && A_p x_p = b_p \\ & && x_p \geq 0 \\ & && p = 1, \dots, P. \end{aligned}$$

---

**Algorithm 2** Nesterov's projected algorithm. Finding  $x_p$  in an ‘‘almost separable’’ way.

---

**Require:** All nodes know  $\rho$  and the total number of nodes  $P$

**Initialization** choose  $x^{(0)} = y^{(0)}$ ,  $L_{\text{in}} = \rho P$ , and set  $t = 1$

- 1: **repeat**
  - 2:    $x_p^{(t)} = \left[ y_p^{(t-1)} - \frac{1}{L_{\text{in}}} \nabla_{x_p} g(y_p^{(t-1)}) \right]_{A_p, b_p}^+$
  - 3:    $y_p^{(t)} = x_p^{(t)} + \frac{t-1}{t+2} (x_p^{(t)} - x_p^{(t-1)})$
  - 4:    $t \leftarrow t + 1$
  - 5: **until** some stopping criterion is met
-

In the above algorithm,  $[\cdot]_{A_p, b_p}^+$  denotes the projection onto the set  $\{z : A_p z = b_p, z \geq 0\}$ , i.e.,

$$[p]_{A_p, b_p}^+ = \begin{array}{ll} \text{minimize} & \frac{1}{2} \|z - p\|^2 \\ \text{subject to} & A_p z = b_p \\ & z \geq 0. \end{array}$$

The following ensures that we can apply Nesterov's projected algorithm on  $g_\lambda$ .

**Lemma 1.** *There holds:*

1.  $g_\lambda$  is convex
2.  $g_\lambda$  is continuously differentiable and the gradient of  $g_\lambda$  at the point  $x = (x_1, \dots, x_P)$  is

$$\begin{aligned} \nabla g_\lambda(x_1, \dots, x_P) &= \begin{bmatrix} \nabla_{x_1} g_\lambda(x_1, \dots, x_P) \\ \vdots \\ \nabla_{x_p} g_\lambda(x_1, \dots, x_P) \\ \vdots \\ \nabla_{x_P} g_\lambda(x_1, \dots, x_P) \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{P}c + \gamma_1 + \rho D_1 x_1 - \rho \sum_{j \in \mathcal{N}_1} x_j \\ \vdots \\ \frac{1}{P}c + \gamma_p + \rho D_p x_p - \rho \sum_{j \in \mathcal{N}_p} x_j \\ \vdots \\ \frac{1}{P}c + \gamma_P + \rho D_P x_P - \rho \sum_{j \in \mathcal{N}_P} x_j \end{bmatrix}, \end{aligned} \quad (8)$$

where  $\mathcal{N}_p$  is the set of neighbors of node  $p$ ,  $D_p$  the number of neighbors of node  $p$  ( $D_p = |\mathcal{N}_p|$ ) and  $\gamma_p = \sum_{j \in \mathcal{N}_p} \text{sign}(j - p) \lambda_{pj}$

3.  $g_\lambda$  has a Lipschitz gradient with constant  $L_{in} = \rho P$ .

*Proof.* 1. We start by rewriting  $g_\lambda$  as

$$g_\lambda(x_1, \dots, x_P) = \sum_{p=1}^P \left[ \frac{1}{P} c^\top x_p + \gamma_p^\top x_p \right] + \sum_{(i,j) \in \mathcal{E}} \frac{\rho}{2} \|x_i - x_j\|^2. \quad (9)$$

The convexity follows directly from convex analysis arguments, namely that the sum of convex functions is convex and the pre-composition with affine functions preserves convexity. In particular, the first of (9) is a linear function of  $x_p$  and each term in the second sum of (9) is a pre-composition of an affine function of  $x$  with a convex quadratic function. Thus,  $g_\lambda$  is a convex function.

2. The gradient of the first sum of (9) w.r.t.  $x_p$  is  $(1/P)c + \gamma_p$ . To calculate the gradient of the second sum w.r.t.  $x_p$ , we first rewrite this sum as

$$\begin{aligned} \sum_{(i,j) \in \mathcal{E}} \frac{\rho}{2} \|x_i - x_j\|^2 &= \sum_{(p,j) \in \mathcal{E}} \frac{\rho}{2} \|x_p - x_j\|^2 + \sum_{(j,p) \in \mathcal{E}} \frac{\rho}{2} \|x_j - x_p\|^2 + \sum_{\substack{(i,j) \in \mathcal{E} \\ i \neq j}} \frac{\rho}{2} \|x_i - x_j\|^2 \\ &= \sum_{\substack{j \in \mathcal{N}_p \\ j > p}} \frac{\rho}{2} \|x_p - x_j\|^2 + \sum_{\substack{j \in \mathcal{N}_p \\ j < p}} \frac{\rho}{2} \|x_j - x_p\|^2 + \sum_{\substack{(i,j) \in \mathcal{E} \\ i \neq j}} \frac{\rho}{2} \|x_i - x_j\|^2 \\ &= \sum_{j \in \mathcal{N}_p} \frac{\rho}{2} \|x_p - x_j\|^2 + \sum_{\substack{(i,j) \in \mathcal{E} \\ i \neq j}} \frac{\rho}{2} \|x_i - x_j\|^2. \end{aligned}$$

Then, its gradient w.r.t.  $x_p$  is  $\sum_{j \in \mathcal{N}_p} \rho(x_p - x_j) = \rho D_p x_p - \rho \sum_{j \in \mathcal{N}_p} x_j$ .

3. In order to prove that  $g_\lambda$  has a Lipschitz gradient with constant  $L_{\text{in}} = \rho P$ , we will prove that  $\nabla^2 g_{\lambda, \mu}(x_1, \dots, x_P) \preceq L_{\text{in}} I_{nP}$  for any  $x = (x_1, \dots, x_P)$ . The matrix  $I_{nP}$  is the identity matrix in  $\mathbb{R}^{nP \times nP}$ . We partition the hessian matrix  $\nabla^2 g_{\lambda, \mu}(x_1, \dots, x_P)$  into  $P^2$  blocks of size  $n \times n$  and denote each block by  $\nabla_{x_i} \nabla_{x_j} g_{\lambda, \mu}(x_1, \dots, x_P)$ , i.e.,

$$\nabla^2 g_\lambda = \begin{bmatrix} \nabla_{x_1}^2 g_\lambda & \nabla_{x_1} \nabla_{x_2} g_\lambda & \cdots & \nabla_{x_1} \nabla_{x_P} g_\lambda \\ \nabla_{x_2} \nabla_{x_1} g_\lambda & \nabla_{x_2}^2 g_\lambda & \cdots & \nabla_{x_2} \nabla_{x_P} g_\lambda \\ \vdots & \vdots & \ddots & \vdots \\ \nabla_{x_P} \nabla_{x_1} g_\lambda & \nabla_{x_P} \nabla_{x_2} g_\lambda & \cdots & \nabla_{x_P}^2 g_\lambda \end{bmatrix}. \quad (10)$$

Let us now calculate  $\nabla_{x_i} \nabla_{x_j} g_\lambda$  for all pairs  $(x_i, x_j)$ :

- If  $j = i$ , we have

$$\begin{aligned} \nabla_{x_i}^2 g_\lambda &= \nabla_{x_i} \left( \frac{1}{P} c + \gamma_p + \rho D_i x_i - \rho \sum_{l \in \mathcal{N}_i} x_l \right) \\ &= \rho D_i I_n. \end{aligned}$$

- If  $j \neq i$ , we have two cases

- If  $j \in \mathcal{N}_i$ , then

$$\nabla_{x_i} \nabla_{x_j} g_\lambda = -\rho I_n.$$

- If  $j \notin \mathcal{N}_i$ , we have

$$\nabla_{x_i} \nabla_{x_j} g_\lambda = 0.$$

Thus,

$$\nabla^2 g_\lambda = \rho \begin{bmatrix} D_1 I_n & -I_n & \cdots & 0 \\ -I_n & D_2 I_n & \cdots & -I_n \\ \vdots & \vdots & \ddots & \vdots \\ 0 & -I_n & \cdots & D_P I_n \end{bmatrix} = \rho(\mathcal{L} \otimes I_n),$$

where the  $(i, j)$ th block of  $W_1$  is  $I_n$  whenever  $(i, j) \in \mathcal{E}$ . The matrix  $\mathcal{L}$  is the Laplacian of the network. It is known (PUT REF) that all eigenvalues of  $\mathcal{L}$  are bounded by  $P$ . Therefore, by the properties of the Kronecker product<sup>1</sup>, we have that  $\|\nabla^2 g_\lambda(x)\| \leq \rho P$ , for all  $x$ . □

**Solving the projection problem.** We now address the problem of projecting point onto the set  $\{z : A_p z = b_p, z \geq 0\}$ , i.e., solving

$$\begin{aligned} [p]_{A_p, b_p}^+ &= \text{minimize} && \frac{1}{2} \|z - p\|^2 \\ &\text{subject to} && A_p z = b_p \\ &&& z \geq 0. \end{aligned} \quad (11)$$

We first start by recasting (11) as an unconstrained problem (PUT REF).

The KKT conditions for (11) are

$$\begin{cases} x \in \arg \min_z \frac{1}{2} \|z - p\|^2 - \lambda^\top (Az - b) - \mu^\top z \\ Ax = b \\ x \geq 0 \\ \mu \geq 0 \\ \mu^\top x = 0 \end{cases} \Leftrightarrow \begin{cases} x - \mu = p + A^\top \lambda \\ Ax = b \\ x \geq 0 \\ \mu \geq 0 \\ \mu^\top x = 0 \end{cases}.$$

<sup>1</sup>If  $\lambda_i(A)$  and  $\lambda_j(B)$  are respectively the  $i$ th and  $j$ th eigenvalues of the matrices  $A$  and  $B$ , then all the eigenvalues of  $A \otimes B$  are given by all the pairs  $\lambda_i(A)\lambda_j(B)$ .

The equations  $x - \mu = p + A^\top \lambda$ ,  $x \geq 0$ ,  $\mu \geq 0$  and  $\mu^\top x = 0$  imply that  $x = [p + A^\top \lambda]^+$ , where  $[z]^+ = \max\{z, 0\}$ . Therefore the KKT conditions are equivalent to

$$\begin{cases} x = [p + A^\top \lambda]^+ \\ A[p + A^\top \lambda]^+ = b \end{cases} \quad (12)$$

This also means that once we find the Lagrange multiplier  $\lambda$ , we can recover  $x$  using the first equation of (12).

We now focus on solving the second equation of (12). Solving this equation is equivalent to solving the following unconstrained optimization problem:

$$\underset{\lambda}{\text{minimize}} \quad \Phi(\lambda) \quad (13)$$

where  $\Phi(\lambda) = \eta(A^\top \lambda + p) - b^\top \lambda$ , and

$$\eta(x) = \frac{1}{2} \|x\|^2 - \min_{z \geq 0} \frac{1}{2} \|z - x\|^2. \quad (14)$$

To see why, let us first compute the gradient of  $\eta(x)$  at the point  $x$ . Let  $z^*(x)$  solve the optimization problem in (14). It is straightforward to see that  $z^*(x) = x^+$ . Therefore, by Danskin's theorem,  $\nabla \eta(x) = x - x + x^+ = x^+$ . Consequently,

$$\nabla \Phi(\lambda) = A[A^\top \lambda + p]^+ - b.$$

Thus, setting  $\nabla \Phi(\lambda) = 0$ , we find the minimizer of (13). Here we can finally conclude that solving the second equation of (12) is equivalent to solving (13).