A p-*norm* Flow Optimization Problem in Dense Wireless Sensor Networks

Mehdi Kalantari, Masoumeh Haghpanahi, Mark Shayman
Department of Electrical and Computer Engineering, University of Maryland
{mehkalan, masoumeh, shayman}@eng.umd.edu

***Abstract*— In a network with a high density of wireless nodes, we model flow of information by a continuous vector field known as the information flow vector field. We use a mathematical model that translates a communication network composed of a large but finite number of sensors into a continuum of nodes on which information flow is formulated by a vector field. The magnitude of this vector field is the intensity of the communication activity, and its orientation is the direction in which the traffic is forwarded. The information flow vector field satisfies a set of Neumann boundary conditions and a partial differential equation (PDE) involving the divergence of information, but the divergence constraint and Neumann boundary conditions do not specify the information flow vector field uniquely, and leave us freedom to optimize certain measures within their feasible set. Therefore, we introduce a p-*norm* flow optimization problem in which we minimize the p-*norm* of information flow vector field over the area of the network. This problem is a convex optimization problem, and we use sequential quadratic programming (SQP) to solve it. SQP is known for numerical stability and fast convergence to the optimal solution in convex optimization problems. By using standard SQP on p-*norm* flow optimization, we prove that the solution of each iteration of SQP is uniquely specified by an elliptic PDE with generalized Neumann boundary conditions. The p-*norm* flow optimization shows interesting properties for different values of p. For example, if p is close to one, the information routes resemble the geometric shortest paths of the sources and sinks, and for $p = 2$, the information flow shows an analogy to electrostatics. For infinitely large values of p, the problem minimizes the maximum magnitude of the information vector field over the network, and hence it achieves maximum load balancing.**

I. Introduction

Wireless sensor networks have been a subject of interest of many researchers in recent years. Fast growth of microelectronics and microprocessing devices have resulted in devices with very small physical dimensions and a small per sensor cost. The small cost of the devices enables networks with several hundred to several thousand sensors distributed in a geographical area. There are many applications for such networks including military, environment monitoring, transportation systems, surveillance, agriculture and home applications. Generally, sensors use radio frequency channels for communicating, and it is desired to collect the data acquired by all sensors in one or a few specific destinations in the network for processing. Such stations are known as *sinks* or *fusion centers*. For communicating with the traffic sinks, the sensors relay the packets of each other in a multi-hop way.

This work was partially supported by AFOSR under grant F496200210217 and NSF under grants CNS-0519554 and CCF-0729129.

As the number of wireless nodes grow, careful analysis of the network behavior becomes very hard by using conventional methods that employ a discrete model in space. In discrete models, different network properties such as transport rate of information are only evaluated at the locations of the wireless nodes. A main shortcoming of discrete models is that often the flow models become computationally prohibitive when the number of wireless nodes grows very large.

We use a continuous space model to formulate flow of information in a wireless network with a large number of nodes. For this purpose, we use a vector field model that represents flow of information at every point of a network. We introduced this idea in our former work [1,2], with a model inspired by electrostatics and using analogy of propagation of an electric field in a dielectric media with information transport in a network. In those works we used a quadratic cost function, and showed that solution to the optimization problem is found by solving a set of PDEs analogous to Maxwell's equations in electrostatics. Our work was followed by Toumpis and Tassiulas [3,4], where the authors showed that minimizing the quadratic cost function results in optimal deployment of sensors by minimizing the number of sensor nodes required to transport all the information to the intended sinks. In a recent work [5], we have given an optimization for the case that there are multiple commodity flows in the network, and we need to make a joint optimization on multiple flows. In this methodology an *information flow vector field* models transportation of traffic in a wireless network. The vector field has two components at every location of the network: a magnitude that represents the density of communication at that location and an orientation that gives the direction to which the traffic is forwarded. The model is most suitable in situations where the sensors in the network are distributed with a high enough density so that the routes from individual sensors to the traffic sinks can be chosen to well approximate the flux lines of the vector field passing through the sensors. The first property of the vector field is that it satisfies a partial differential equation (PDE), which impose the constraint that the traffic of all sources should be routed to their destinations. This constraint is incorporated into our model through the *divergence* of the vector field and represents flow conservation in conventional networks. Additionally, the vector field satisfies a set of *Neumann* boundary conditions, which state that the vector field has a zero component in the direction normal to the boundary of the network.

We present a general method for flow optimization in a wireless sensor network. In this method, we minimize a p-*norm* of the information flow vector field subject to the basic flow constraints (i.e., flow conservation and boundary constraints). We have called this problem the p-*norm flow optimization*. In this problem we always have $p > 1$. The p-*norm* optimization problem has several interesting properties for different values of p. For example, when p is close to 1, the routes tend to pass through shortest geometric paths from a source to a sink, and as a result the average transport delay is less compared to the case with higher values of p. When p is close to 1, the optimization does not make the best use of network resources, and for example, it may leave a lot of resources unused while some areas of the network that lie on shortest paths are overloaded by the network traffic. By increasing the value of p from 1 the optimization tries to spread the traffic in the network and use the resources more evenly. While this increases delay, it helps load balance the traffic compared to the case where p is smaller. The limiting case of $p \to \infty$ is a $minmax$ problem, where the maximum magnitude of the information flow vector field is minimized. Load balancing can be beneficial in numerous ways such as increasing the transport capacity of the network by using standard techniques such as spatial multiplexing and uniform use of network resources such as the nodes' batteries. While the case $p \to \infty$ achieves maximal load balancing, it may use longer paths for some portions of the traffic. In practical applications, p can be selected based on a trade-off between delay and balancing the traffic over the network.

In order to solve the p-*norm* flow optimization, we first show that this problem is a convex optimization problem. Then, we use Sequential Quadratic Programming (SQP) to solve the p-*norm* flow optimization on a general network geometry. SQP uses iterations, and in each iteration, it finds the quadratic approximation of the optimization problem around an operating point and solves the resulting quadratic optimization problem. Then it adds the solution to the operating point and finds a new operating point and repeats iterations. SQP is known for numerical stability and fast convergence to the optimal solution in convex optimization problems. We prove that the solution to each iteration of SQP is uniquely specified by an *elliptic* PDE with generalized Neumann boundary conditions. This elliptic PDE has a standard canonical form.

There are several lines of works that have studied dense wireless sensor networks. Non-quadratic cost functions in the flow problem were considered in [4], where the authors considered a general form of cost function, and optimizing the cost function requires solving nonlinear PDEs. Such PDEs are hard to solve for a generic cost function. Additionally, properties of the solutions for general cost functions are unknown. In comparison to the cost functions in [4], the p-*norm* flow optimization is a less general form of the cost function, but we present a concrete method to solve the resulting PDEs and give an in depth insight into the properties of the solution for different values of p. Load balancing problem in dense wireless sensor networks was studied in [6]. In this work, the authors consider a $minmax$ optimization problem and develop lower bounds for the objective of the achievable minimum. In [7], the performance of load balancing methods in the presence of multipath routes were compared with single path approaches, and it was shown that under certain models, the performance of the multipath approach does not give significant improvement unless the number of paths between a source and a destination is large enough. In [8], the analogy between information paths in a dense ad hoc network and light paths in optics was studied, and it was shown that the routes bend when the spatial density of nodes change in the network. This analogy was studied in further detail in [9], where the approach was developed for wireless networks with energy and bandwidth constraints. The asymptotic capacity of wireless networks in the number of nodes was studied in [10].

Our approach is distinguished from the body of existing work in two aspects: **(i)** We introduce p-*norm* as a family of optimization problems for a wireless sensor network in a general network geometry that is densely covered by sensors, where varying p results in different properties in the information flow. **(ii)** We use SQP to solve the p-*norm* problem and show that the solution to each iteration of SQP is uniquely defined by an elliptic PDE.

The remainder of this paper is organized as follows: First we introduce the basic definitions and notations in Section II. In Section III we give a brief background on formulating information flow as a vector field and discuss the solutions in the case where we minimize a quadratic cost function. In Section IV we introduce the p-*norm* flow optimization problem and use SQP to solve this problem. We show how the solution to each iteration of SQP is found by solving an elliptic PDE. An illustrative numerical example is given in Section V, and we conclude the paper in Section VI.

II. Notations and definitions

In this section, we give a brief summary of our notations and definitions. To avoid confusion, we only introduce notations that we will use frequently. There are other symbols and notations that we will introduce where they are first used.

Throughout this paper, boldface characters represent vector quantities. We use $\mathbf{z} = (x, y)$ to represents location in $\mathbb{R}^2$. Unless we state otherwise, we use Cartesian coordinate system to express vector quantities, where $\hat{\mathbf{i}}$ and $\hat{\mathbf{j}}$ represent the unit vectors along x and y axes, respectively. For a generic vector quantity we use the notation $\mathbf{v} = (v_x, v_y)$ to represent the vector, where v_x and v_y are the x and y components of $\mathbf{v}$ in a Cartesian coordinate system. Whenever we need to manipulate a vector as a matrix, we consider the vector as a 2×1 matrix: $\mathbf{v} = [v_x \quad v_y]^T$.

In order to make PDE expressions simpler, we use the nabla operator: $\nabla = \frac{\partial}{\partial \mathbf{x}}\hat{\mathbf{i}} + \frac{\partial}{\partial \mathbf{y}}\hat{\mathbf{j}}$; for a vector quantity $\mathbf{F} = (F_x, F_y)$, the density of sources of the vector field is found by the divergence of $\mathbf{F}$, which is defined as

$$\nabla \cdot \mathbf{F} = \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y}$$

Similarly, $\nabla \times \mathbf{F}$ represents the density of rotation of $\mathbf{F}$, which is expressed as:

$$\nabla \times \mathbf{F} = (-\frac{\partial F_x}{\partial y} + \frac{\partial F_y}{\partial x})\hat{\mathbf{k}}$$

in which $\hat{\mathbf{k}} = \hat{\mathbf{i}} \times \hat{\mathbf{j}}$.

For a scalar function $u(\mathbf{z})$, we use ∇u to represent the gradient of u. Moreover, we use the Laplacian operator, ∇^2, which appears in the form of Poisson equation in information flow in the case of quadratic optimization:

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

The network is located in a closed, connected, and bounded set $A \in \mathbb{R}^2$. The information sources are distributed with a spatial density function $\mu(\mathbf{z}) \in A$, which means that at location $\mathbf{z} = (x, y) \in A$, $\mu_s(\mathbf{z})$ bps/m^2 of information is generated. The information of the sources is needed to be transported to a set of distributed sinks. The sinks are distributed according to a density function $\mu_d(\mathbf{z})$ bps/m^2. The total information generated by the sources is equal to the total information collected by the sinks: $\int_A \mu_s(\mathbf{z})dxdy = \int_A \mu_d(\mathbf{z})dxdy$. Based on the above notions, we define the generalized density of information sources $\rho(\mathbf{z}) = \mu_s(\mathbf{z}) - \mu_d(\mathbf{z})$. Since all the information of sources is transported to the sinks, we have $\int_A \rho(\mathbf{z})dxdy = 0$.

The *p-norm* of an $n \times 1$ vector $\mathbf{X} = (x_1, x_2, \cdots, x_N)$ is:

$$|\mathbf{X}|_p = \left(\sum_{i=1}^{N} |x_i|^p\right)^{\frac{1}{p}}$$

Similarly, the *p-norm* of function $f(\mathbf{z})$ where $f : A \mapsto \mathbb{R}$ is:

$$|f|_p = \left(\int_A |f(\mathbf{z})|^p dxdy\right)^{\frac{1}{p}}$$

For a scalar function $u(\mathbf{z})$, where $u : A \mapsto \mathbb{R}$, the PDE

$$c_{11}\frac{\partial^2 u}{\partial x^2} + 2c_{12}\frac{\partial^2 u}{\partial x \partial y} + c_{22}\frac{\partial^2 u}{\partial y^2} + b_1\frac{\partial u}{\partial x} + b_2\frac{\partial u}{\partial y} = g$$

is the canonical form of a general second order linear PDE on A. In the above PDE c_{11}, c_{12}, c_{22}, b_1, b_2, and g are $A \mapsto \mathbb{R}$ functions. The above PDE is an *elliptic* PDE if the following matrix is positive semidefinite for every point $\mathbf{z} \in A$:

$$C = \begin{bmatrix} c_{11} & c_{12} \\ c_{12} & c_{22} \end{bmatrix}$$

III. Background: Information Flow as a Vector Field

In this section, we give a brief overview of our previous work on routing in wireless networks by using vector fields. We briefly state our results here; more complete presentations of these results can be found in [1,2,11].

In the first step, we define a vector quantity that models flow of information. Let $\mathbf{D}(\mathbf{z}) = (D_x, D_y)$ denote this vector field whose direction represents the direction of flow of information at point $\mathbf{z}$, and its magnitude represents the density of information rate passing per unit length of a line segment perpendicular to the direction of $\mathbf{D}(\mathbf{z})$. In other words, if we consider a line segment with a small length $\Delta\ell$ at $\mathbf{z}$ and perpendicular to the direction of $\mathbf{D}(\mathbf{z})$, the information crosses that line segment with rate $|\mathbf{D}(\mathbf{z})|\Delta\ell$. The above definition of $\mathbf{D}(\mathbf{z})$ implies that for a closed contour $C \in A$ we have:

$$\oint_C \mathbf{D}(\mathbf{z}) \cdot \mathbf{dn} = \int_{S(C)} \rho(\mathbf{z})dxdy \tag{1}$$

in which $\mathbf{dn}$ is a differential vector normal to the contour at each point of its boundary and pointing to the outside of the contour, the dot represents the inner product of vectors in two-dimensional space, and $S(C)$ is the area surrounded by the closed contour C. Equation (1) is analogous to Gauss' law in electrostatics theory, and it has a simple interpretation in our formulation: the rate at which information exits a contour is the net sum of the sources inside that contour. The following is known as the *Divergence Theorem* in vector calculus:

$$\oint_C \mathbf{D} \cdot \mathbf{dn} = \int_{S(C)} (\frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y})dxdy \tag{2}$$

Equations (1) and (2) hold for an arbitrary contour C. This implies the following PDE form for information flow:

$$\nabla \cdot \mathbf{D}(\mathbf{z}) = \frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y} = \rho(\mathbf{z}) \tag{3}$$

It is important to note that (3) is a representation of the *flow conservation law* in continuous form. The boundary condition of the above PDE is a result of the fact that information is not intended to exit the boundary of the network or enter it from the outside. Hence:

$$D_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \tag{4}$$

in which $D_n(\mathbf{z})$ is the normal component of $\mathbf{D}(\mathbf{z})$ on the boundary of A, and ∂A represents the boundary of A. This condition is known as *Neumann boundary condition*.

An important note about PDE (3) with the boundary condition (4) is that these equations do not result in a unique value for $\mathbf{D}(\mathbf{z})$. Therefore, we have freedom to impose additional condition(s) to optimize $\mathbf{D}(\mathbf{z})$ such that the resulting vector field generates a desirable property. In [1,2] we introduced a quadratic cost function of the information flow vector field $\mathbf{D}$ in order to find a unique solution:

$$\begin{aligned} \text{Minimize} \quad & J(\mathbf{D}) = \int_A |\mathbf{D}(\mathbf{z})|^2 dxdy \\ \text{s.t.} \quad & \nabla \cdot \mathbf{D}(\mathbf{z}) = \rho(\mathbf{z}) \\ & D_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{5}$$

The above form of cost function results in spatial spreading of the communication load over the space available in the network. To some extent, it balances the communication load of the network in such a way that it avoids having a high load somewhere in the network while the resources are underutilized somewhere else. Moreover, it is shown in [3] that minimizing the cost function in (5) minimizes the number

of sensor nodes required to handle the total communication burden of the network.

Our former result shows that the cost function in (5) is optimal if and only if the curl of $\mathbf{D}(\mathbf{z})$ is zero:

$$\nabla \times \mathbf{D}(\mathbf{z}) = (-\frac{\partial D_x}{\partial y} + \frac{\partial D_y}{\partial x})\hat{\mathbf{k}} = \mathbf{0}$$

Since $\nabla \times \mathbf{D} = \mathbf{0}$, then $\mathbf{D}$ is conservative, and it can be written as the gradient of a scalar potential function: $\mathbf{D}(\mathbf{z}) = \nabla U(\mathbf{z})$. This potential function satisfies the *Poisson's* equation:

$$\nabla^2 U(\mathbf{z}) = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = \rho(\mathbf{z}) \tag{6}$$

The combination of the divergence property, the zero-curl property, and the boundary conditions uniquely specify $\mathbf{D}(\mathbf{z})$. The boundary conditions of $\mathbf{D}$ imply that the partial derivative of U is zero along the normal direction to the boundary of A.

IV. *p-norm* FLOW OPTIMIZATION PROBLEM

One of the main motivations for using a quadratic cost function is to disperse the traffic in the network. By spreading the communication load of the network over the space, we make the best use of the available resources in the network. This reduces the interference among the wireless nodes, and increases the overall network throughput. However, the quadratic cost function does not achieve maximum spatial spreading of the traffic over the network. In this section, we consider the following general form of optimization problem:

$$\begin{aligned} \text{Minimize} \quad & J(\mathbf{D}) = \int_A |\mathbf{D}(\mathbf{z})|^p dxdy \\ \text{s.t.} \quad & \nabla \cdot \mathbf{D}(\mathbf{z}) = \rho(\mathbf{z}) \\ & D_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{7}$$

in which $p > 1$ is a real number. Recall that the constraints of the above optimization problem ensure delivery of the traffic generated by the sources to the sinks. Note that increasing the value of p causes the optimization problem to increase the amount of spatial spreading in the network. To illustrate this fact consider a point $\mathbf{z}$ belonging to an infinitesimal area dS. Since dS is infinitely small, the value of $|\mathbf{D}(\mathbf{z})|$ over it is almost constant, and hence the contribution to the cost function due to dS is $|\mathbf{D}(\mathbf{z})|^p |dS|$. Now if we double the value of $|\mathbf{D}(\mathbf{z})|$, then the above value of contribution increases by a factor of 2^p. This simple observation suggests that by using larger values of p, the optimization problem tends towards solutions that make more spatial spreading, which in turn results in a better spatial diversity of the network.

An observation about the optimization (7) is that this problem is equivalent to the case in which we raise the cost function to power $1/p$:

$$\begin{aligned} \text{Minimize} \quad & \tilde{J}(\mathbf{D}) = \left(\int_A |\mathbf{D}(\mathbf{z})|^p dxdy \right)^{\frac{1}{p}} \\ \text{s.t.} \quad & \nabla \cdot \mathbf{D}(\mathbf{z}) = \rho(\mathbf{z}) \\ & D_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{8}$$

Note that the cost function in the recent optimization problem is the *p-norm* of the information flow vector field $\mathbf{D}$. Based on this observation, we call the optimization (7) the *p-norm* flow optimization problem.

To further illustrate the fact that increasing p results in more spatial spreading, we study the case when $p \to \infty$ in which *p-norm* gives the maximum absolute value of $|\mathbf{D}(\mathbf{z})|$ over the network, and the optimization problem becomes equivalent to:

$$\begin{aligned} \text{Minimize} \quad & \tilde{J}(\mathbf{D}) = (\max_{\mathbf{z} \in A} |\mathbf{D}(\mathbf{z})|) \\ \text{s.t.} \quad & \nabla \cdot \mathbf{D}(\mathbf{z}) = \rho(\mathbf{z}) \\ & D_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned}$$

which is a $minmax$ problem and minimizes the maximum value of $|\mathbf{D}(\mathbf{z})|$ over the network area A. Apparently the maximal spatial spreading is achieved in this case, and therefore the corresponding $\mathbf{D}(\mathbf{z})$ is a very attractive case for the purpose of balancing the traffic load over the network.

Our first observation is that the *p-norm* flow optimization is a convex optimization problem. Recall that $\mathbf{D}(\mathbf{z}) = (D_x(\mathbf{z}), D_y(\mathbf{z}))$, in which $D_x(\mathbf{z})$ and $D_y(\mathbf{z})$ represent the x and y components of $\mathbf{D}(\mathbf{z})$, respectively. Hence, if we define $g(D_x, D_y) = |\mathbf{D}(\mathbf{z})|^p$, the second derivatives of g is:

$$H(g) = \begin{bmatrix} \frac{\partial^2 g}{\partial D_x^2} & \frac{\partial^2 g}{\partial D_x \partial D_y} \\ \frac{\partial^2 g}{\partial D_x \partial D_y} & \frac{\partial^2 g}{\partial D_y^2} \end{bmatrix} = p(D_x^2 + D_y^2)^{\frac{p-4}{2}} \begin{bmatrix} (p-1)D_x^2 + D_y^2 & (p-2)D_x D_y \\ (p-2)D_x D_y & D_x^2 + (p-1)D_y^2 \end{bmatrix} \tag{9}$$

in which $H(g)$ is the Hessian of g. Performing the eigenvalue decomposition on $H(g)$ we find:

$$H(g) = p(D_x^2 + D_y^2)^{\frac{p-4}{2}} \begin{bmatrix} D_x & -D_y \\ D_y & D_x \end{bmatrix} \begin{bmatrix} p-1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} D_x & D_y \\ -D_y & D_x \end{bmatrix} \tag{10}$$

Note that the eigenvalues of $H(g)$ are $\upsilon_1 = p(p-1)(D_x^2 + D_y^2)^{p/2-1}$, and $\upsilon_2 = p(D_x^2 + D_y^2)^{p/2-1}$. Therefore, both eigenvalues are nonnegative, and hence, the matrix $H(g)$ is positive semidefinite, which implies that g is convex in (D_x, D_y). We observe that the cost function J in *p-norm* flow optimization (7) is the integral of g over the network area, and the constraints are linear in (D_x, D_y).

A. An Illustrative Example

We study a simple example to illustrate the performance of *p-norm* flow optimization on a simple network geometry shown in Fig. 1. In this example, a source located on the left hand side of the network generates information at a rate of θ bps. The information of the source needs to be transmitted to a destination on the right hand side of the network as depicted in Fig. 1. There are two geometric paths between the source and the destination. Both paths have equal widths of W, but different lengths denoted by L_1 and L_2, respectively.

For transporting information from the source to the destination, we assume θ_1 bps is transmitted through path 1, and θ_2

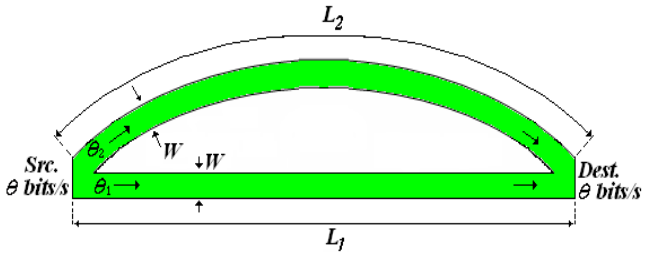

Fig. 1. Illustrating effect of the *p-norm* flow optimization. The amount of θ bits/sec traffic of a source is divided between to geometric paths. The paths have different lengths but equal widths. For $p \to 1$ the optimization chooses the shortest path. For $p = 2$, it splits the traffic such that $\theta_1 L_1 = \theta_2 L_2$. For $p \to \infty$, the optimization performs load balancing by setting $\theta_1 = \theta_2$.

is transmitted through path 2. The information is transported over the two paths by sensors that densely cover the area of both paths and relay packets from the source to the destination. The basic definition of $\mathbf{D}$ gives the following magnitude for the information flow vector field on the two paths:

$$|\mathbf{D_1}| = \frac{\theta_1}{W}, \quad |\mathbf{D_2}| = \frac{\theta_2}{W}$$

Note that the simple network geometry in this example implies that the value of $|\mathbf{D}|$ remains constant along each path but different for the two paths. The value of the cost function in this example is:

$$\begin{aligned} J(\mathbf{D}) &= \int_A |\mathbf{D}(\mathbf{z})|^p dxdy \\ &= |\mathbf{D_1}|^p W L_1 + |\mathbf{D_2}|^p W L_2 \\ &= \theta_1^p W^{1-p} L_1 + \theta_2^p W^{1-p} L_2 \end{aligned}$$

We assume that the second path is longer than the first path; therefore, $L_2 = kL_1$, where $k > 1$. Hence:

$$J(\mathbf{D}) = W^{1-p} L_1 (\theta_1^p + k\theta_2^p)$$

Flow conservation in this example translates into the constraint $\theta_1 + \theta_2 = \theta$. Forming the Lagrangian function, the optimality condition is found to be:

$$\theta_1 = k^{\frac{1}{p-1}} \theta_2$$

which results in the following optimal values for θ_1 and θ_2:

$$\theta_1 = \frac{k^{\frac{1}{p-1}}}{1+k^{\frac{1}{p-1}}}\theta, \quad \theta_2 = \frac{1}{1+k^{\frac{1}{p-1}}}\theta$$

Among values of p, there are three cases of special interest: $p \to 1$, $p = 2$, and $p \to \infty$. For $p \to 1$, we have $k^{\frac{1}{p-1}} \to \infty$; hence $\theta_1 = \theta$, and $\theta_2 = 0$. The *p-norm* optimization problem in this case chooses the shortest path from the source to the destination. While the shortest paths of $p \simeq 1$ result in minimum delay, they do not use all the network resources and as shown in this case, one of the paths between the source and the destination is left completely unused.

The next case of interest is $p = 2$. For this case we find $\theta_1 = k\theta_2$, or equivalently, $\theta_1 L_1 = \theta_2 L_2$. This implies that the optimization uses both paths, and the amount of information that is transported on each path is inversely proportional to the length of that path. In an earlier work, we have shown that in the case of $p = 2$, the mathematical model of information flow vector field has a one-to-one analogy with electrostatics [1,2]. Additionally the special case of $p = 2$ leads to optimal deployment of sensor nodes and requires the minimum number of sensors to transport the traffic of source to the destination [3]. Notice that $p = 2$ uses both paths, but it does not achieve a complete load balanced solution.

The last case of special interest is the asymptotic case of $p \to \infty$. In this case $k^{\frac{1}{p-1}} \to 1$; hence, $\theta_1 = \theta_2 = \theta/2$. In this case the optimization problem performs maximum load balancing, and the problem minimizes the maximum value of $|\mathbf{D}|$ over the network area.

As illustrated in this example, the *p-norm* flow optimization problem represents a family of optimization problems in which the properties of the resulting routes vary significantly by varying the value of p from one extreme to the other. In practical situations, the value of p depends on the specific requirements of each application and the trade-offs between delay and load balancing.

In the next step, we use the convexity of the *p-norm* flow optimization problem to introduce a set of algebraic PDEs that lead us to the optimal solution.

B. Solving the p-norm flow optimization problem

In order to solve the *p-norm* flow optimization problem, we use the SQP iterations. Such iterations are well known for their numerical stability and good convergence speed [12,13]. Additionally, since the *p-norm* flow optimization is a convex optimization problem, the SQP iterations are guarantied to converge to the global optimum [12]. Each iteration of SQP method can be summarized in the following two steps:

Quadratic approximation: Having a value for the information flow vector field as an operating point at the ith iteration, denoted by $\mathbf{D^{(i)}}$, which satisfies the set of constraints given by the divergence property and Neumann's boundary conditions, we find the quadratic approximation of the optimization problem near $\mathbf{D^{(i)}}$. For this purpose, we assume that the operating point $\mathbf{D^{(i)}}$ is perturbed by a small variation denoted by $\mathbf{e}$, and hence, we have $\mathbf{D} = \mathbf{D^{(i)}} + \mathbf{e}$. Then we find the second order approximation of the cost function $J(\mathbf{D})$ with respect to $\mathbf{e}$.

Optimization: In this step we find $\mathbf{e}$ that minimizes quadratic cost function of the previous step. Convexity of the *p-norm* flow optimization problem implies that the quadratic cost is convex in $\mathbf{e}$. By using the method of Lagrange multipliers and the geometric interpretation of the divergence property, we show that the optimal $\mathbf{e}$ is found by solving an elliptic PDE with generalized Neumann boundary conditions. After solving this PDE, we add the optimal value of $\mathbf{e}$ to $\mathbf{D^{(i)}}$ and find a new operating point.

We repeat the iterations above until a certain stop criterion is satisfied. We now give a mathematical derivation of SQP method in *p-norm* optimization detail. Assume at the ith iteration $\mathbf{D^{(i)}} = (D_x^{(i)}, D_y^{(i)})$ is known and satisfies

$$\begin{aligned} &\nabla \cdot \mathbf{D^{(i)}}(\mathbf{z}) = \rho(\mathbf{z}) \\ &D_n^{(i)}(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{11}$$

in which $D_n^{(i)}(\mathbf{z})$ represents the normal component of the vector field $\mathbf{D^{(i)}}$ along the boundary at a point $\mathbf{z} \in \partial\mathbf{A}$.

Then we define $\mathbf{e} = (e_x, e_y)$ as the variation of the information flow vector field near $\mathbf{D^{(i)}}$:

$$\mathbf{D}(\mathbf{z}) = \mathbf{D^{(i)}}(\mathbf{z}) + \mathbf{e}(\mathbf{z}) \tag{12}$$

Since $\mathbf{D}$ must satisfy the divergence property and the Neumann boundary conditions of the *p-norm* flow optimization problem, we have the following constraints on $\mathbf{e}$:

$$\begin{aligned} &\nabla \cdot \mathbf{e}(\mathbf{z}) = 0 \\ &e_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{13}$$

where $e_n(\mathbf{z})$ denotes the normal component of $\mathbf{e}$ at $\mathbf{z} \in \partial A$.

To find the second order approximation of $J(\mathbf{D})$ near $\mathbf{D^{(i)}}$, we use the following:

$$\begin{aligned} J(\mathbf{D}) &= \int_A |\mathbf{D}|^p dxdy = \int_A |\mathbf{D^{(i)}} + \mathbf{e}|^p dxdy \qquad (14) \\ &\approx \int_A |\mathbf{D^{(i)}}|^p dxdy \\ &+ \int_A \left(\mathbf{e}^T \nabla g(\mathbf{D^{(i)}}) + \frac{1}{2}\mathbf{e}^T H(g(\mathbf{D^{(i)}}))\mathbf{e} \right) dxdy \end{aligned}$$

in which $\nabla g(\mathbf{D^{(i)}})$ and $H(g(\mathbf{D^{(i)}}))$ represent the gradient and the Hessian of $g(\mathbf{D}) = |\mathbf{D}|^p$ with respect to $\mathbf{D}$ and evaluated at $\mathbf{D^{(i)}}$, respectively. The expression in (14) is the first 3 terms of the Taylor's series expansion of the function g.

In the next step we find the optimal solution of the quadratic approximation of the cost function by optimizing over $\mathbf{e}$ subject to the constraints given by (13). The first term in (14) does not depend on $\mathbf{e}$ and can be dropped from the optimization cost function. Hence, we have the following optimization problem over $\mathbf{e}$ in each iteration of SQP:

$$\begin{aligned} \text{Minimize} \quad & J_q^{(i)}(\mathbf{e}) = \int_A \left(\mathbf{e}^T R + \frac{1}{2}\mathbf{e}^T Q \mathbf{e} \right) dxdy \\ \text{s.t.} \quad & \nabla \cdot \mathbf{e}(\mathbf{z}) = 0 \\ & e_n(\mathbf{z}) = 0, \ \mathbf{z} \in \partial A \end{aligned} \tag{15}$$

in which $R = \nabla g(\mathbf{D^{(i)}})$ is a 2×1 matrix representing the gradient of g, and $Q = H(g(\mathbf{D^{(i)}}))$ is a 2×2 matrix representing Hessian of g, and $J_q^{(i)}(\mathbf{e})$ is the cost function in the ith iteration of SQP. Note that both Q and R are functions of $\mathbf{z}$. Recall that convexity of g in $\mathbf{D}$ implies that Q is a positive semidefinite matrix, which was shown (10).

We now introduce a scheme that converts the quadratic optimization problem into an elliptic PDE. For this purpose, we state the following lemma:

Lemma 1: For the optimal solution of the quadratic optimization problem (15) there exists a scalar function $\lambda(\mathbf{z}) : A \mapsto \mathbb{R}$ that satisfies:

$$\nabla \lambda = Q\mathbf{e} + R \tag{16}$$

Proof: To prove the lemma, we use the following identities:
Identity 1: If v is a scalar function and $\mathbf{F}$ is a vector field, then:

$$\nabla \cdot (v\mathbf{F}) = v \nabla \cdot \mathbf{F} + \mathbf{F} \cdot \nabla v \tag{17}$$

Identity 2: If A is a region in the plane with boundary ∂A, and F is a vector field on A, then

$$\int_A \nabla \cdot \mathbf{F} dxdy = \oint_{\partial A} \mathbf{F} \cdot \mathbf{dn} \tag{18}$$

in which $\mathbf{dn}$ is the differential vector normal to the boundary of A pointing outward. This identity is the divergence theorem, which we used earlier in a slightly different form in (2).

We use the Lagrangian method to prove Lemma 1. Note that for every point $\mathbf{z} \in A$, we have $\nabla \cdot \mathbf{e}(\mathbf{z}) = 0$. This implies that for each $\mathbf{z} \in A$ we need to define the Lagrange multiplier $\lambda(\mathbf{z})$. This results in the following Lagrangian function:

$$L(\mathbf{e}) = \int_A \left(\mathbf{e}^T R + \frac{1}{2}\mathbf{e}^T Q \mathbf{e} \right) dxdy + \int_A \lambda \nabla \cdot \mathbf{e} dxdy \tag{19}$$

Using Identity 1 for $\mathbf{F} = \mathbf{e}$ and $v = \lambda$ gives:

$$\begin{aligned} \nabla \cdot (\lambda \mathbf{e}) = \lambda \nabla \cdot \mathbf{e} + \mathbf{e} \cdot \nabla \lambda \Rightarrow \\ \lambda \nabla \cdot \mathbf{e} = \nabla \cdot (\lambda \mathbf{e}) - \mathbf{e} \cdot \nabla \lambda \end{aligned} \tag{20}$$

Substituting this expression for $\lambda \nabla \cdot \mathbf{e}$ in (19) gives:

$$\begin{aligned} L(\mathbf{e}) &= \int_A \left(\mathbf{e}^T R + \frac{1}{2}\mathbf{e}^T Q \mathbf{e} \right) dxdy \qquad (21) \\ &+ \int_A \nabla \cdot (\lambda \mathbf{e}) dxdy - \int_A \mathbf{e} \cdot \nabla \lambda dxdy \end{aligned}$$

Next we use Identity 2 for $\mathbf{F} = \lambda \mathbf{e}$. This identity implies:

$$\int_A \nabla \cdot (\lambda \mathbf{e}) dxdy = \oint_{\partial A} \lambda \mathbf{e} \cdot \mathbf{dn} \tag{22}$$

On the other hand, the boundary conditions of $\mathbf{e}$ specified in (13) require that the normal component of this vector field on the boundary of A is zero. Therefore, we have $\mathbf{e} \cdot \mathbf{dn} = 0$ for every point $\mathbf{z} \in \partial A$, which implies that the right hand side of (22) is zero. Hence:

$$\int_A \nabla \cdot (\lambda \mathbf{e}) dxdy = 0 \tag{23}$$

Substituting the surface integral above in (21) leads us to:

$$L(\mathbf{e}) = \int_A \left(\mathbf{e}^T (R - \nabla \lambda) + \frac{1}{2}\mathbf{e}^T Q \mathbf{e} \right) dxdy \tag{24}$$

Note that we have used $\mathbf{e} \cdot \nabla \lambda = \mathbf{e}^T \nabla \lambda$. The rest of the proof is based on calculus of variation. In the Lagrangian function (24), we replace $\mathbf{e}$ by $\tilde{\mathbf{e}} = \mathbf{e} + \mathbf{e}'$, where $\mathbf{e}' : A \mapsto \mathbb{R}^2$ is a small variation vector field. The Lagrangian function for $\tilde{\mathbf{e}}$ is:

$$\begin{aligned} L(\tilde{\mathbf{e}}) = \int_A \left(\tilde{\mathbf{e}}^T (R - \nabla \lambda) + \frac{1}{2}\tilde{\mathbf{e}}^T Q \tilde{\mathbf{e}} \right) dxdy \\ = \int_A \left((\mathbf{e} + \mathbf{e}')^T (R - \nabla \lambda) + \frac{1}{2}(\mathbf{e} + \mathbf{e}')^T Q (\mathbf{e} + \mathbf{e}') \right) dxdy \end{aligned}$$

Now we use the fact that $\mathbf{e}'$ is infinitely small; hence, we can ignore the second order terms. By expanding the integrand and

removing second order terms we find:

$$\begin{aligned} L(\tilde{\mathbf{e}}) &= \int_A \left(\mathbf{e}^T (R - \nabla\lambda) + \frac{1}{2}\mathbf{e}^T Q \mathbf{e} \right) dxdy \\ &+ \int_A \mathbf{e}'^T (R - \nabla\lambda + Q\mathbf{e})\, dxdy \\ &= L(\mathbf{e}) + \int_A \mathbf{e}'^T (R - \nabla\lambda + Q\mathbf{e})\, dxdy \end{aligned} \quad (25)$$

Hence the variation of Lagrangian is:

$$\Delta L := L(\tilde{\mathbf{e}}) - L(\mathbf{e}) = \int_A \mathbf{e}'^T (R - \nabla\lambda + Q\mathbf{e})\, dxdy \quad (26)$$

Optimality of Lagrangian implies that the above variation is zero for every $\mathbf{e}'$. Therefore, we can choose $\mathbf{e}' = \epsilon(R - \nabla\lambda + Q\mathbf{e})$, where ϵ is an infinitely small positive number. This choice of $\mathbf{e}'$ implies that $(R - \nabla\lambda + Q\mathbf{e}) = 0$, and hence $\nabla\lambda = Q\mathbf{e} + R$. **QED.**[1]

Lemma 1 gives the main key to converting the optimization problem (15) into a partial differential equation. For this purpose, we use Lemma 1 to write $\mathbf{e}$ in terms of the Lagrange multiplier λ:

$$\mathbf{e} = Q^{-1}(\nabla\lambda - R) \quad (27)$$

We will discuss later that in all cases of interest to us, Q is nonsingular; however, it can be ill-conditioned in special situations. We will discuss later in this section how to handle such cases. Next, we use $\nabla \cdot \mathbf{e}(\mathbf{z}) = 0$ which is the first optimization constraint in (15). This implies that:

$$\nabla \cdot \mathbf{e} = \nabla \cdot (Q^{-1}(\nabla\lambda - R)) = 0 \quad (28)$$

Equivalently:

$$\nabla \cdot (C\nabla\lambda) = \varrho \quad (29)$$

in which:

$$\begin{aligned} C(\mathbf{z}) &= Q^{-1}(\mathbf{z}) := \begin{bmatrix} c_{11}(\mathbf{z}) & c_{12}(\mathbf{z}) \\ c_{12}(\mathbf{z}) & c_{22}(\mathbf{z}) \end{bmatrix} \\ \varrho(\mathbf{z}) &= \nabla \cdot (Q^{-1}(\mathbf{z})R(\mathbf{z})) \\ &= \frac{\partial(r_1 c_{11} + r_2 c_{12})}{\partial x} + \frac{\partial(r_1 c_{12} + r_2 c_{22})}{\partial y} \end{aligned}$$

where r_1 and r_2 denote the entries of the column vector $R = [r_1\ r_2]^T$. Since Q is a positive semidefinite matrix, $C = Q^{-1}$ is also positive definite.

It is straightforward to verify that the second order PDE (29) is elliptic. To illustrate this we substitute the value of $\nabla\lambda = [\partial\lambda/\partial x \quad \partial\lambda/\partial y]^T$ in this PDE:

$$\nabla \cdot (C\nabla\lambda) = \varrho \Rightarrow$$
$$\frac{\partial(c_{11}\partial\lambda/\partial x + c_{12}\partial\lambda/\partial y)}{\partial x} + \frac{\partial(c_{12}\partial\lambda/\partial x + c_{22}\partial\lambda/\partial y)}{\partial y} = \varrho$$

[1]Lemma 1 can be generalized as follows: consider the optimization: $J(\mathbf{D}) = \int_A f(\mathbf{e})dxdy$, subject to: $\nabla \cdot \mathbf{e}(\mathbf{z}) = \varrho(\mathbf{z})$, and $e_n(\mathbf{z}) = 0$ where $\mathbf{z} \in \partial A$, and $f : \mathbb{R}^2 \mapsto \mathbb{R}$ is a differentiable function. Then the optimiality conditions are:

$$\frac{\partial f}{\partial e_x} = \frac{\partial \lambda}{\partial x} \quad \text{and} \quad \frac{\partial f}{\partial e_y} = \frac{\partial \lambda}{\partial y}$$

The proof is similar to the proof of Lemma 1.

which implies:

$$\begin{aligned} & c_{11}\frac{\partial^2\lambda}{\partial x^2} + 2c_{12}\frac{\partial^2\lambda}{\partial x\partial y} + c_{22}\frac{\partial^2\lambda}{\partial y^2} \\ + & \left(\frac{\partial c_{11}}{\partial x} + \frac{\partial c_{12}}{\partial y}\right)\frac{\partial\lambda}{\partial x} + \left(\frac{\partial c_{12}}{\partial x} + \frac{\partial c_{22}}{\partial y}\right)\frac{\partial\lambda}{\partial y} \\ = & \ \varrho \end{aligned} \quad (30)$$

Equivalently:

$$c_{11}\frac{\partial^2\lambda}{\partial x^2} + 2c_{12}\frac{\partial^2\lambda}{\partial x\partial y} + c_{22}\frac{\partial^2\lambda}{\partial y^2} + b_1\frac{\partial\lambda}{\partial x} + b_2\frac{\partial\lambda}{\partial y} = \varrho \quad (31)$$

where $b_1 = (\frac{\partial c_{11}}{\partial x} + \frac{\partial c_{12}}{\partial y})$, and $b_2 = (\frac{\partial c_{12}}{\partial x} + \frac{\partial c_{22}}{\partial y})$. Since C is positive semidefinite on A, the PDE in (31) is elliptic.

The final step is to determine the boundary conditions for λ in the elliptic PDE (31). For this purpose, we translate the Neumann boundary conditions of $\mathbf{e}$ into the boundary conditions for λ. Note that we have:

$$\mathbf{e} = C(\nabla\lambda - R)$$

Hence for a point $\mathbf{z} \in \partial A$ with outward unit normal vector $\hat{\mathbf{n}}(\mathbf{z}) = [n_x(\mathbf{z}) \quad n_y(\mathbf{z})]^T$, we have:

$$\begin{aligned} & e_n(\mathbf{z}) = 0 \Rightarrow \hat{\mathbf{n}}^T\mathbf{e} = 0 \\ & \Rightarrow \hat{\mathbf{n}}^T C(\nabla\lambda - R) = 0 \\ & \Rightarrow \hat{\mathbf{n}}^T (C\nabla\lambda) = \hat{\mathbf{n}}^T CR \\ & \Rightarrow (n_x c_{11} + n_y c_{12})\tfrac{\partial\lambda}{\partial x} + (n_x c_{12} + n_y c_{22})\tfrac{\partial\lambda}{\partial y} = \hat{\mathbf{n}}^T CR \end{aligned}$$

which is known as the *generalized Neumann boundary condition*. The existence and uniqueness of the solution of the elliptic PDEs with generalized Neumann boundary condition is a known fact in the context of PDEs [14].

The argument of this section can be summarized in the following theorem:

Theorem 1: The solution to each SQP iteration is specified by $\mathbf{e} = C(\nabla\lambda - R)$, where λ is uniquely specified by the following elliptic PDE

$$c_{11}\frac{\partial^2\lambda}{\partial x^2} + 2c_{12}\frac{\partial^2\lambda}{\partial x\partial y} + c_{22}\frac{\partial^2\lambda}{\partial y^2} + b_1\frac{\partial\lambda}{\partial x} + b_2\frac{\partial\lambda}{\partial y} = \varrho \quad (32)$$

with boundary condition

$$\begin{aligned} & (n_x c_{11} + n_y c_{12})\tfrac{\partial\lambda}{\partial x} + (n_x c_{12} + n_y c_{22})\tfrac{\partial\lambda}{\partial y} \\ & = \hat{\mathbf{n}}^T CR, \quad \mathbf{z} \in \partial A \end{aligned} \quad (33)$$

An important remark is that the matrix Q is positive semidefinite, and it may happen that in some areas of the network, this matrix is close to singular. This may cause problems in the elliptic PDE (29), which involves $C = Q^{-1}$. Note that the eigenvalues of $Q = H(g)$ are: $\upsilon_1 = p(p-1)(D_x^2 + D_y^2)^{p/2-1}$, and $\upsilon_2 = p(D_x^2 + D_y^2)^{p/2-1}$, and hence, C will be ill-conditioned when $p \to 1$, or $(D_x^2 + D_y^2) \to 0$. Since for all cases where $p > 1$ the traffic is spread over the network to some extent and in practice, $|\mathbf{D}|$ is nonzero for all of such cases; however, the value of $|\mathbf{D}|$ can be very small in some areas of the network, which causes numerical instability in SQP. A method to avoid numerical instability in such cases is to replace a close to zero eigenvalue by a very small positive fixed number ϵ whenever Q is close to singular. The

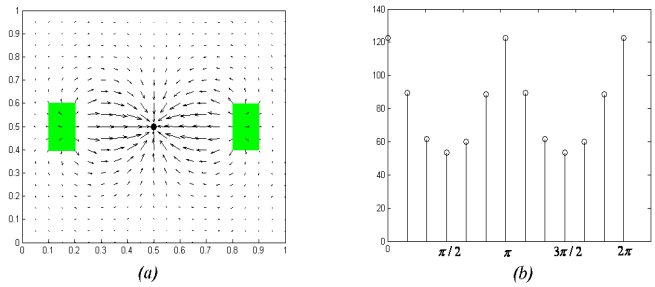

Fig. 2. **(a)** The information flow vector field for $p = 2$ used as the initial value of $\mathbf{D}$ in SQP iterations. The sources of information are uniformly spread on the two highlighted rectangles and the sink is at the center of the square. **(b)** Spatial distribution of traffic in the proximity of the sink. The bars show the values of $|\mathbf{D}|$ on 12 evenly spaced directions toward the sink and at distance $r = 0.2$ of the sink. The difference between the maximum and the minimum values of $|\mathbf{D}|$ on the 12 directions is 76.88 bps/m.

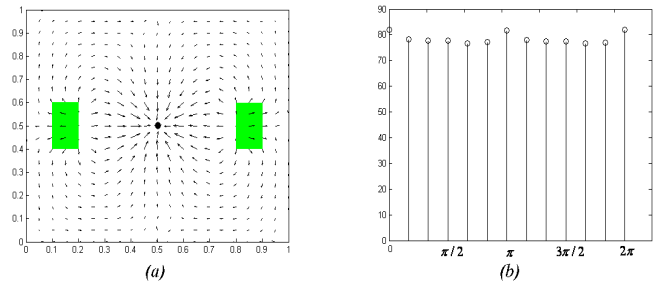

Fig. 3. **(a)** Information flow vector field after five SQP iterations for $p = 8$. **(b)** Spatial distribution of traffic in the proximity of the sink. The bars show values of $|\mathbf{D}|$ on 12 evenly spaced directions toward the sink and at distance $r = 0.2$ of the sink. The difference between the maximum and the minimum values of $|\mathbf{D}|$ on the 12 directions is reduced 5.08 bps/m.

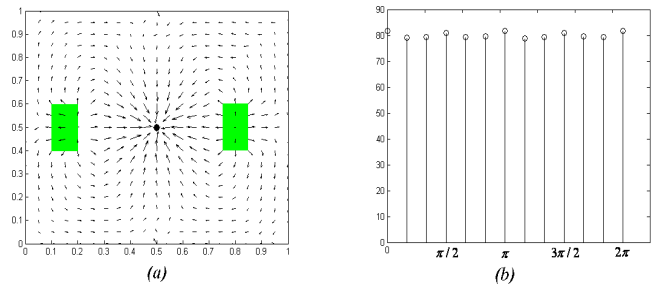

Fig. 4. **(a)** Information flow vector field after six SQP iterations for $p = 16$. **(b)** Spatial distribution of traffic in the proximity of the sink. The bars show values of $|\mathbf{D}|$ on 12 evenly spaced directions toward the sink and at distance $r = 0.2$ of the sink. Using $p = 16$, which is a relatively large value of p, the traffic approaching the sink in different directions is nearly balanced. The difference between the maximum and the minimum values of $|\mathbf{D}|$ on the 12 directions is reduced to 2.084 bps/m, which is a steep decrease compared to unbalanced cases such as $p = 2$.

eigenvalues of Q are lower bounded by ϵ after this correction. Such a correction is standard in SQP when the Hessian matrix is ill-conditioned [12]. In practice the value of ϵ may be very small and for example using a value of $\epsilon = 10^{-8}$ when Q is ill-conditioned has shown good stability in all numerical examples that we have studied.

Our final remark is on solving elliptic PDEs in practical applications. Fortunately, the canonic form of the elliptic PDE (32) with the boundary conditions (33) enables us to use the existing powerful numerical PDE solvers. The above elliptic PDE appears in different branches of science and engineering such as electromagnetics, heat exchange, fluid dynamics, and diffusion. Due to the broad spectrum of the applications, the numerical elliptic PDE solvers are very well studied and have reached a high level of accuracy and maturity [14,15]. The solvers use different techniques such as Finite Element Method [15] in order to solve the PDE with a high accuracy in a reasonable time. For example, the Matlab PDE toolbox [16] is capable of solving an elliptic PDE on a square with 64×64 grid point in less than one second.[2] We need a few SQP iterations to find the solution of the *p-norm* flow optimization. In all numerical examples that we studied, the SQP shows a fast convergence to the global optima with number of SQP iterations strictly less than 10.

V. A Numerical Example

In this section we present numerical examples that illustrate the performance of *p-norm* flow optimization with various values of p. Each iteration of SQP requires solving an elliptic PDE for which we have used the PDE toolbox of Matlab[3].

We use a network on a 1×1 square densely covered by sensors. The network in this scenario is shown in Fig 2. The traffic in this network is generated uniformly inside two equal size rectangles specified by $0.1 \leq x \leq 0.2, \quad 0.4 \leq y \leq 0.6$, and $0.8 \leq x \leq 0.9, \quad 0.4 \leq y \leq 0.6$, respectively. The sources inside the rectangles generate a total of 100 units of traffic. The traffic of the network is intended to be transported to a sink located at $(0.5, 0.5)$.

First, we study the problem with $p = 2$. In this case the problem is already quadratic, and we do not need SQP to find the optimal $\mathbf{D}$. To find $\mathbf{D}$, we solve the Poisson's equation given by the PDE (6) for the potential function U with Neumann boundary conditions $\hat{\mathbf{n}} \cdot \nabla U = 0$. Note that the Poisson's equation has a simple elliptic form. By taking the gradient of U we compute $\mathbf{D}$. The resulting value of $\mathbf{D}$ is shown in Fig. 2-a. The relative size of each arrow reflects the magnitude of $\mathbf{D}$, and its orientation is the direction of $\mathbf{D}$ at the corresponding point.

In order to show how the traffic approaches the sink, we have plotted the value of $|\mathbf{D}|$ on different directions in the proximity of the sink. The plot in Fig. 2-b shows 12 samples of $|\mathbf{D}|$ on the perimeter of a small circle with radius $r = 0.2$ centered at the sink. The horizontal axis of Fig. 2-b shows the angle between 0 and 2π at which each sample on the small circle was taken. As can be seen in this figure, the samples are very uneven, and indeed the difference between the largest and the smallest samples is 76.68 bps/m. An interesting fact about this plot is that the sum of the samples is proportional to the total rate of the traffic being sent to the sink since all the traffic needs to pass the small circle to reach the sink.

In the next step we use SQP to solve the *p-norm* optimization for $p = 8$. We use the value of $\mathbf{D}$ in the previous case

[2]The run time was measured on a generic PC with enough memory and an Intel Pentium IV processor.

[3]We have used *assempde* command to solve the elliptic PDEs. This command solves conic PDEs with canonical form of $\nabla \cdot (C\nabla u) + au = f$ with Dirichlet, Neumann, or mixed boundary conditions. [16]

as the initial information flow vector field for SQP iterations, $\mathbf{D}^{(0)}$. The experiments show that after 5 iterations, the SQP converges with a good accuracy ($|\mathbf{D}^{(i)} - \mathbf{D}^{(i-1)}| \leq 0.1$ bps/m). The resulting value of $\mathbf{D}$ is shown in Fig. 3-a. As can be seen, the distribution of the traffic changes in the proximity of the sink and the traffic approaches more evenly from the different directions of the sink. This fact is better visualized in Fig. 3-b, where similar to before, we have plotted 12 evenly spaced samples of $|\mathbf{D}|$ on the perimeter of a small circle with radius $r = 0.2$, centered at the sink. In this case, the difference between the largest and the smallest samples reduces to 5.084 bps/m, which shows a steep decrease compared to the case with $p = 2$.

Similarly, for $p = 16$, the values of $\mathbf{D}$, and the 12 samples of $|\mathbf{D}|$ in proximity of the sink are shown in Fig. 4-a and Fig. 4-b, respectively. In this case, the SQP converges after 6 iterations. As can be seen in Fig. 4-b, the pattern of traffic is very uniform in the proximity of sink and the amount of traffic approaching the sink from different directions is almost equal. In this case, the difference between the largest and the smallest samples reduces to 2.084 bps/m. Since $p = 16$ is a relatively large value of p, the problem has already achieved an almost load balanced solution. Balancing traffic helps the network resources to be utilized in a more even way. Also when the traffic is balanced, we can improve the capacity of the sink (e.g., by spatial multiplexing or space diversity). Since the sink is the major bottleneck of the network, increasing its capacity improves the overall capacity of the network. For example, the sink can use multiple directional antennas to differentiate the signal received on each direction (e.g., 12 identical directional antennas in the case of this example). In such a design, the sink performs a spatial multiplexing on the traffic approaching it in different directions, and therefore, achieving the maximum capacity is only possible when the amount of the traffic received through different directional antennas is balanced.

VI. Conclusion

In this paper, we presented a family of optimization problems in which the objective is to minimize *p-norm* of the vector quantity that models flow of information. We use a mathematical model that translates a communication network composed of a large but finite number of sensors into a continuum of nodes on which information flow is formulated by a vector field. The magnitude of this vector field is the intensity of the communication activity, and its orientation is the direction in which the traffic is forwarded. The information flow vector field satisfies a set of Neumann boundary conditions and a PDE involving the divergence of information, but the divergence constraint and Neumann boundary conditions do not specify the information flow vector field uniquely, and leave us freedom to optimize certain measures within their feasible set.

In order to solve the *p-norm* flow optimization problem, we used an SQP method, which is known for fast convergence and good numerical stability in general optimization problems. We showed that each iteration of SQP for *p-norm* flow optimization leads us to solving an elliptic PDE with generalized Neumann boundary conditions. Solving the elliptic PDE is easy since this PDE has a canonical form similar to the elliptic PDEs appearing in other branches of science and engineering such as electromagnetics, heat exchange, fluid dynamics, and there are many powerful tools to solve such PDEs with good accuracy and in reasonable time. We showed that the solution of *p-norm* optimization problem has different properties for different values of p. For example, if $p \simeq 1$, the optimization gives routes that have a tendency to transport the traffic of sources to the sink on the shortest geometrical paths. On the other hand, $p \to \infty$ minimizes the maximum magnitude of the information flow vector field over the network, and therefore, it achieves load balancing. While the solutions for the extreme of $p \simeq 1$ minimize delay, the larger values of p make a more uniform use of the network resources by spreading out the traffic over all possible paths between the sources to the sinks. In practice, the value of p depends on the requirements and the trade-offs between delay and load balancing.

References

[1] M. Kalantari and M.A. Shayman. Energy efficient routing in sensor networks. In *Conference on Information Sciences and Systems, 2004*.

[2] M. Kalantari and M.A. Shayman. Routing in wireless ad hoc networks by analogy to electrostatic theory. In *IEEE International Communications Conference, 2004*.

[3] S. Toumpis and L. Tassiulas. Efficient node deployment in massively dense sensor networks as an electrostatics problem. In *IEEE Infocom, 2005*.

[4] S. Toumpis and L. Tassiulas. Optimal deployment of large wireless sensor networks. *IEEE Transactions on Information Theory*, 52(7):2935–2953, 2006.

[5] M. Kalantari and M.A. Shayman. Routing in multi-commodity sensor networks based on partial differential equations. Princeton University, March 2006.

[6] Esa Hyytia and Jorma Virtamo. On traffic load distribution and load balancing in densewirelessmultihop networks. *EURASIP Journal on Wireless Communications and Networking*, 2007, Article ID 1692.

[7] Y. Ganjali and A. Keshavarzian. Load balancing in ad hoc networks: single-path routing vs. multi-path routing. In *IEEE Infocom 2004*.

[8] P. Jacquet. Geometry of information propagation in massively dense ad hoc networks. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 04)*.

[9] S. Toumpis R. Catanuto and G. Morabito. Optic,mal: Optical / optimal routing in massively dense wireless networks. In *IEEE Infocom 2007*.

[10] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

[11] M. Kalantari and M.A. Shayman. Design optimization of multi-sink sensor networks by analogy to electrostatic theory. In *Wireless Communication and Networking Conference, 2006*.

[12] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag New York, Inc., 1999.

[13] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2003.

[14] L. C. Evans. *Partial Differential Equations*. American Mathematical Society, 1998.

[15] G. Haase C. Douglas and U. Langer. *A Tutorial on Elliptic PDE Solvers and Their Parallelization*. Society for Industrial and Applied Mathematics (SIAM), 2003.

[16] The MathWorks. Matlab PDE Toolbox. http://www.mathworks.com/products/pde/.