Network Optimization: Notes and Exercises

Michael J. Neely
University of Southern California
http://www-bcf.usc.edu/~mjneely

Abstract

These notes provide a tutorial treatment of topics of Pareto optimality, Lagrange multipliers, and computational algorithms for multiobjective optimization, with emphasis on applications to data networks. Problems with two objectives are considered first, called bicriteria optimization problems (treated in Sections I and II). The main concepts of bicriteria optimization naturally extend to problems with more than two objectives, called multicriteria optimization problems. Multicriteria problems can be more complex than bicriteria problems, and often cannot be solved without the aid of a computer. Efficient computational methods exist for problems that have a convex structure. Convexity is formally defined in Section V. Section V describes a general class of multicriteria problems with a convex structure, called convex programs. A drift-plus-penalty algorithm is developed in Section VI as a computational procedure for solving convex programs. The drift-plus-penalty algorithm extends as an online control technique for optimizing time averages of system objectives, even when the underlying system does not have a convex structure. Section VII focuses on application of drift-plus-penalty theory to multi-hop networks, including problems of network utility maximization and power-aware routing. Exercises are provided to reinforce the theory and the applications.

HOW TO REFERENCE THESE NOTES

Sections I, V present optimization material that is well known and/or easily derived from basic definitions (see also [1], [2]). Sections VI, VII present more recent material on drift-plus-penalty theory for convex programs and data networks. Readers who want to cite the drift-plus-penalty material in these notes should cite the published works [3] and [4].

I. BICRITERIA OPTIMIZATION

Consider a system that has a collection $\mathcal{M}$ of different operating modes, where $\mathcal{M}$ is an abstract (possibly infinite) set that contains at least one element. Each operating mode $m \in \mathcal{M}$ determines a two-dimensional vector $(x(m), y(m))$, where $x(m)$ and $y(m)$ represent distinct system objectives of interest. Suppose it is desirable to keep both objectives $x(m)$ and $y(m)$ as small as possible. We want to find a mode $m \in \mathcal{M}$ that “minimizes both” $x(m)$ and $y(m)$. Of course, it may not be possible to simultaneously minimize both objectives. This tension motivates the study of bicriteria optimization.

Example I.1. Consider the problem of finding the best route to use for sending a single message over a network. Let $\mathcal{M}$ represent the set of all possible routes. Suppose each link $(i, j)$ has a link distance $d_{ij}$ and a link energy expenditure $e_{ij}$. For each route $m \in \mathcal{M}$, let $x(m)$ be the total distance of the route, and let $y(m)$ be the total energy used. It is desirable to keep both objectives $x(m)$ and $y(m)$ small.

Example I.2. Consider the problem of transmitting over a single wireless link. Let $p$ be a variable that represents the amount of power used, and suppose this variable must be chosen over an interval $[0, p_{\text{max}}]$ for some positive maximum power level $p_{\text{max}}$. The power used determines the transmission rate $\mu(p) = \log(1 + p)$. The goal is to operate the system while minimizing power and maximizing transmission rate. Define set $\mathcal{M}$ as the interval $[0, p_{\text{max}}]$. For each $p \in \mathcal{M}$, define $x(p) = p$ as the power used and $y(p) = -\mu(p)$ as $-1$ times the transmission rate achieved (so that minimizing $y(p)$ is the same as maximizing $\mu(p)$). We want to choose $p \in \mathcal{M}$ to keep both objectives $x(p)$ and $y(p)$ small.

Example I.3. Consider a wireless device that transmits to three different users over orthogonal links. The device must choose a power vector $(p_1, p_2, p_3) \in \mathbb{R}^3$ that satisfies the following constraints:

$$p_1 + p_2 + p_3 \leq p_{\text{max}} \quad (1)$$

$$p_i > 0 \quad \forall i \in \{1, 2, 3\} \quad (2)$$

where $p_{\text{max}}$ is a positive real number that constrains the sum power usage. For each $i \in \{1, 2, 3\}$, let $\mu_i(p_i) = \log(1 + \gamma_i p_i)$ be the transmission rate achieved over link $i$ as a function of the power variable $p_i$, where $\gamma_i$ is some known attenuation coefficient for link $i$. Define $\mathcal{M}$ as the set of all $(p_1, p_2, p_3) \in \mathbb{R}^3$ that satisfy the constraints (1) and (2). Define $x(p_1, p_2, p_3) = p_1 + p_2 + p_3$ as the sum power used. Define $y(p_1, p_2, p_3) = -[\mu_1(p_1) + \mu_2(p_2) + \mu_3(p_3)]$ as $-1$ times the sum rate over all three links. The goal is to choose $(p_1, p_2, p_3) \in \mathcal{M}$ to keep both $x(p_1, p_2, p_3)$ and $y(p_1, p_2, p_3)$ small.

Example I.4. Consider the same 3-link wireless system as Example I.3. However, suppose we do not care about power expenditure. Rather, we care about:

• Maximizing the sum rate $\mu_1(p_1) + \mu_2(p_2) + \mu_3(p_3)$. 
• Maximizing the proportionally fair utility metric \( \log(\mu_1(p_1)) + \log(\mu_2(p_2)) + \log(\mu_3(p_3)) \). This is a commonly used notion of fairness for rate allocation over multiple users.\(^1\)

Again let \( \mathcal{M} \) be the set of all vectors \((p_1, p_2, p_3) \in \mathbb{R}^3 \) that satisfy (1)-(2). Define \( x(p_1, p_2, p_3) = -[\mu_1(p_1) + \mu_2(p_2) + \mu_3(p_3)] \) as \(-1\) times the sum rate, and define \( y(p_1, p_2, p_3) = -[\log(\mu_1(p_1)) + \log(\mu_2(p_2)) + \log(\mu_3(p_3))] \) as \(-1\) times the proportionally fair utility metric. The goal is to choose \((p_1, p_2, p_3) \in \mathcal{M} \) to minimize both \( x(p_1, p_2, p_3) \) and \( y(p_1, p_2, p_3) \).

Examples\(^1\)[2][4] illustrate how a bicriteria optimization problem that seeks to maximize one objective while minimizing another, or that seeks to maximize both objectives, can be transformed into a bicriteria minimization problem by multiplying the appropriate objectives by \(-1\). Hence, without loss of generality, it suffices to assume the system controller wants both components of the vector of objectives \((x, y)\) to be small.

### A. Pareto optimality

Define \( \mathcal{A} \) as the set of all \((x, y)\) vectors in \( \mathbb{R}^2 \) that are achievable via system modes \( m \in \mathcal{M} \):

\[
\mathcal{A} = \{(x(m), y(m)) \in \mathbb{R}^2 | m \in \mathcal{M}\}
\]

Every \((x, y)\) pair in \( \mathcal{A} \) is a feasible operating point. Once the set \( \mathcal{A} \) is known, system optimality can be understood in terms of selecting a desirable 2-dimensional vector \((x, y)\) in the set \( \mathcal{A} \). With this approach, the study of optimality does not require knowledge of the physical tasks the system must perform for each mode of operation in \( \mathcal{M} \). This is useful because it allows many different types of problems to be treated with a common mathematical framework.

The set \( \mathcal{A} \) can have an arbitrary structure. It can be finite, infinite, closed, open, neither closed nor open, and so on. Assume the system controller wants to find an operating point \((x, y) \in \mathcal{A}\) for which both \(x\) and \(y\) are small.

**Definition I.1.** A vector \((x, y) \in \mathcal{A}\) dominates (or is “preferred over”) another vector \((w, z) \in \mathcal{A}\), written \((x, y) \prec (w, z)\), if the following two inequalities hold

- \(x \leq w\)
- \(y \leq z\)

and if at least one of the inequalities is strict (so that either \(x < w\) or \(y < z\)).

**Definition I.2.** A vector \((x^*, y^*) \in \mathcal{A}\) is Pareto optimal if there is no vector \((x, y) \in \mathcal{A}\) that satisfies \((x, y) \prec (x^*, y^*)\).

A set can have many Pareto optimal points. An example set \( \mathcal{A} \) and its Pareto optimal points are shown in Fig. 1. For each vector \((a, b) \in \mathbb{R}^2\), define \( \mathcal{S}(a, b) \) as the set of all points \((x, y)\) that satisfy \(x \leq a\) and \(y \leq b\):

\[
\mathcal{S}(a, b) = \{(x, y) \in \mathbb{R}^2 | x \leq a, y \leq b\}
\]

Pictorially, the set \( \mathcal{S}(a, b) \) is an infinite square in the 2-dimensional plane with upper-right vertex at \((a, b)\) (see Fig. 1). If \((a, b)\) is a point in \( \mathcal{A}\), any other vector in \( \mathcal{A}\) that dominates \((a, b)\) must lie in the set \( \mathcal{S}(a, b)\). If there are no points in \( \mathcal{A} \cap \mathcal{S}(a, b)\) other than \((a, b)\) itself, then \((a, b)\) is Pareto optimal.

### B. Degenerate cases and the compact assumption

In some cases the set \( \mathcal{A} \) will have no Pareto optimal points. For example, suppose \( \mathcal{A} \) is the entire set \( \mathbb{R}^2\). If we choose any point \((x, y) \in \mathbb{R}^2\), there is always another point \((x - 1, y) \in \mathbb{R}^2\) that dominates. Further, it can be shown that if \( \mathcal{A} \) is an open subset of \( \mathbb{R}^2\), then it has no Pareto optimal points (see Exercise VIII-A.5). To avoid these degenerate situations, it is often useful to impose the further condition that the set \( \mathcal{A} \) is both closed and bounded. A closed and bounded subset of \( \mathbb{R}^N\) is called a compact set. If \( \mathcal{A} \) is a finite set then it is necessarily compact.

It can be shown that if \( \mathcal{A} \) is a nonempty compact set, then:

1) It has Pareto optimal points.
2) For every point \((a, b) \in \mathcal{A}\) that is not Pareto optimal, there is a Pareto optimal point that dominates \((a, b)\).

See Exercise VIII-A.11 for a proof of the above two claims. Therefore, when \( \mathcal{A} \) is compact, we can restrict attention to choosing an operating point \((x, y)\) that is Pareto optimal.

\(^1\)See [3] for a development of proportionally fair utility and its relation to the \(\log(\mu)\) function. The constraints \(^2\) avoid the singularity of the \(\log(\mu)\) function at 0, so that \(\log(\mu_1(p_1)) + \log(\mu_2(p_2)) + \log(\mu_3(p_3))\) is indeed a real number whenever \((p_1, p_2, p_3)\) satisfies (1)-(2). An alternative is to use constraints \(p_i \geq 0\) (which allow zero power in some channels), but to modify the utility function from \(\log(\mu)\) to \(1/b \log(1+b\mu)\) for some constant \(b > 0\).
II. OPTIMIZATION WITH ONE CONSTRAINT

Let $A \subseteq \mathbb{R}^2$ be a set of all feasible $(x,y)$ operating points. Assume the system controller wants to make both components of the vector $(x,y)$ small. One way to approach this problem is to minimize $y$ subject to the constraint $x \leq c$, where $c$ is a given real number. To this end, fix a constant $c \in \mathbb{R}$ and consider the following constrained optimization problem:

\begin{align*}
\text{Minimize:} & \quad y \\
\text{Subject to:} & \quad x \leq c \\
& \quad (x,y) \in A
\end{align*}

The variables $x$ and $y$ are the optimization variables in the above problem, while the constant $c$ is assumed to be a given and fixed parameter. The above problem is feasible if there exists an $(x,y) \in \mathbb{R}^2$ that satisfies both constraints (4)-(5).

**Definition II.1.** A point $(x^*, y^*)$ is a solution to the optimization problem (3)-(5) if the following two conditions hold:

- $(x^*, y^*)$ satisfies both constraints (4)-(5).
- $y^* \leq y$ for all points $(x,y)$ that satisfy (4)-(5).

It is possible for the problem (3)-(5) to have more than one optimal solution. It is also possible to have no optimal solution, even if the problem is feasible. This happens when there is an infinite sequence of points $\{(x_n, y_n)\}_{n=1}^\infty$ that satisfy the constraints (4)-(5) with strictly decreasing values of $y_n$, but for which the limiting value of $y_n$ cannot be achieved (see Exercise [VIII-B.1]). This can only happen if the set $A$ is not compact. On the other hand, it can be shown that if $A$ is a compact set, then the problem (3)-(5) has an optimal solution whenever it is feasible.

A. The tradeoff function

The problem (3)-(5) uses a parameter $c$ in the inequality constraint (4). If the problem (3)-(5) is feasible for some given parameter $c$, then it is also feasible for every parameter $c'$ that satisfies $c' \geq c$. Thus, the set of all values $c$ for which the problem is feasible forms an interval of the real number line of the form either $[c_{min}, \infty)$ or $[c_{min}, \infty)$. Call this set the feasibility interval. The value $c_{min}$ is the infimum of the set of all real numbers in the feasibility interval. For each $c$ in the feasibility interval, define $\psi(c)$ as the infimum value of the objective function in problem (3)-(5) with parameter $c$. In particular, if $(x^*, y^*)$ is an optimal solution to (3)-(5) with parameter $c$, then $\psi(c) = y^*$. If $A$ is a compact set, it can be shown that the feasibility interval has the form $[c_{min}, \infty)$ and that problem (3)-(5) has an optimal solution for all $c \in [c_{min}, \infty)$.

The function $\psi(c)$ is called the tradeoff function. The tradeoff function establishes the tradeoffs associated with choosing larger or smaller values of the constraint $c$. Intuitively, it is clear that increasing the value of $c$ imposes less stringent constraints on the problem, which allows for improved values of $\psi(c)$. This is formalized in the next lemma.

**Lemma II.1.** The tradeoff function $\psi(c)$ is non-increasing over the feasibility interval.

**Proof.** For simplicity assume $A$ is compact. Consider two values $c_1$ and $c_2$ in the interval $[c_{min}, \infty)$, and assume $c_1 \leq c_2$. We want to show that $\psi(c_1) \geq \psi(c_2)$. Let $(x^*_1, y^*_1)$ and $(x^*_2, y^*_2)$ be optimal solutions of (3)-(5) corresponding to parameters $c_1$ and $c_2$. If $x^*_1 \leq x^*_2$, then $\psi(c_1) \geq \psi(c_2)$. Otherwise, $x^*_1 > x^*_2$ and $y^*_1 < y^*_2$. Since $A$ is compact, there exist $x^*_3, y^*_3 \in A$ such that $x^*_1 \geq x^*_3 \geq x^*_2$ and $y^*_1 \leq y^*_3 \leq y^*_2$. Then $(x^*_3, y^*_3)$ is an optimal solution of (3)-(5) with parameter $c_2$, and $\psi(c_2) \geq \psi(c_1)$, as desired.
Fig. 2. The set \( A \) from Fig. 1 with its (non-increasing) tradeoff function \( \psi(c) \) drawn in green. Note that \( \psi(c) \) is discontinuous at points \( c_1, c_2, c_3 \).

\[ c = c_1 \text{ and } c = c_2, \text{ respectively. Then:} \]

\[ y_1^* = \psi(c_1) \]
\[ y_2^* = \psi(c_2) \]

By definition of \((x_2^*, y_2^*)\) being optimal for the problem with parameter \( c = c_2 \), we know that for any vector \((x, y) \in A\) that satisfies \( x \leq c_2 \), we have:

\[ y_2^* \leq y \] (6)

On the other hand, we know \((x_1^*, y_1^*)\) is a point in \( A \) that satisfies \( x_1^* \leq c_1 \leq c_2 \), so (6) gives:

\[ y_2^* \leq y_1^* \]

Substituting \( y_1^* = \psi(c_1) \) and \( y_2^* = \psi(c_2) \) gives the result.

Note that the tradeoff function \( \psi(c) \) is not necessarily continuous (see Fig. 2). It can be shown that it is continuous when the set \( A \) is compact and has a convexity property \( ^2 \). Convexity is defined in Section IV.

The tradeoff curve is defined as the set of all points \( (c, \psi(c)) \) for \( c \) in the feasibility interval. Exercise VIII-A.8 shows that every Pareto optimal point \((x^{(p)}, y^{(p)})\) of \( A \) is a point on the tradeoff curve, so that \( \psi(x^{(p)}) = y^{(p)} \).

B. Lagrange multipliers for optimization over \((x, y) \in A\)

The constrained optimization problem (3)-(5) may be difficult to solve because of the inequality constraint (4). Consider the following related problem, defined in terms of a real number \( \mu \geq 0 \):

Minimize: \[ y + \mu x \] (7)

Subject to: \((x, y) \in A\) (8)

The problem (7)-(8) is called the unconstrained optimization problem because it has no inequality constraint. Of course, it still has the set constraint (5). The constant \( \mu \) is called a Lagrange multiplier. It acts as a weight that determines the relative importance of making the \( x \) component small when minimizing the objective function (7). Note that if \((x^*, y^*)\) is a solution to the unconstrained optimization problem (7)-(8) for a particular value \( \mu \), then:

\[ y^* + \mu x^* \leq y + \mu x \text{ for all } (x, y) \in A \] (9)

In particular, all points of the set \( A \) are on or above the line consisting of points \((x, y)\) that satisfy \( y + \mu x = y^* + \mu x^* \). This line has slope \(-\mu\) and touches the set \( A \) at the point \((x^*, y^*)\) (see Fig. 3).

\(^2\)In particular, \( \psi(c) \) is both continuous and convex over \( c \in [c_{min}, \infty) \) whenever \( A \) is compact and convex. Definitions of convex set and convex function are provided in Section IV.
The next theorem shows that if a point \((x^*, y^*)\) solves the unconstrained problem \((7)-(8)\) for a particular parameter \(\mu \geq 0\), then it must also solve the constrained problem \((3)-(5)\) for a particular choice of the \(c\) value, namely, \(c = x^*\).

**Theorem II.1.** If \((x^*, y^*)\) solves the unconstrained problem \((7)-(8)\), then:

a) If \(\mu \geq 0\), then \((x^*, y^*)\) solves the following optimization problem (where \((x, y)\) are the optimization variables and \(x^*\) is treated as a given constant):

Minimize:

\[ y \]  

Subject to:

\[ x \leq x^* \]  
\[ (x, y) \in A \]  
(10)  
(11)  
(12)

b) If \(\mu \geq 0\), then \((x^*, y^*)\) is a point on the tradeoff curve \((c, \psi(c))\). Specifically, \(\psi(x^*) = y^*\).

c) If \(\mu > 0\), then \((x^*, y^*)\) is Pareto optimal in \(A\).

**Proof.** To prove part (a), suppose \((x^*, y^*)\) solves the unconstrained problem \((7)-(8)\). Then \((x^*, y^*)\) also satisfies the constraints of problem \((10)-(12)\). Indeed, the constraint \((11)\) is trivially satisfied by the vector \((x^*, y^*)\) because the first variable of this vector is less than or equal to \(x^*\) (that is, \(x^* \leq x^*\) is a trivially true inequality). Further, vector \((x^*, y^*)\) also satisfies \((12)\) because this constraint is the same as \((8)\). Next, we want to show that \((x^*, y^*)\) is a solution to \((10)-(12)\). Let \((x, y)\) be any other vector that satisfies \((11)-(12)\). It suffices to show that \(y^* \leq y\). Since \((x, y) \in A\) we have from \((9)\):

\[ y^* + \mu x^* \leq y + \mu x \]

\[ y^* + \mu x^* \leq y + \mu x^* \]

where the final inequality follows from \((11)\) together with the fact that \(\mu \geq 0\). Simplifying the above inequality gives \(y^* \leq y\). This proves part (a). Since \(y^*\) is the optimal objective function value in the problem \((10)-(12)\), it follows that \(y^* = \psi(x^*)\), which proves part (b). The proof of part (c) is left as an exercise (see Exercises VIII-A.9 and VIII-A.10).

In summary, the above theorem implies that solving the unconstrained problem \((7)-(8)\) for various values of \(\mu \geq 0\) generates points \((x^*, y^*)\) on the tradeoff curve, and generates Pareto optimal points whenever \(\mu > 0\).

The theorem can be illustrated by the following intuitive example. Suppose we want to find a point \((x, y) \in A\) to minimize \(y\) subject to \(x \leq 5.3\). Imagine that we have a computer program that takes an input parameter \(\mu \geq 0\) and outputs a particular solution \((x_p, y_p)\) to the unconstrained problem of minimizing \(y + \mu x\) over \((x, y) \in A\). Suppose we run the program with input parameter \(\mu = 1\), and the program outputs \((x_p, y_p) = (7.3, 19.4)\). The above theorem implies that the vector \((7.3, 19.4)\) is the solution to the problem of finding \((x, y) \in A\) to minimize \(y\) subject to \(x \leq 7.3\). Unfortunately, this is not the desired problem, and the resulting vector \((7.3, 19.4)\) does not satisfy the desired constraint \(x \leq 5.3\). We need to work harder to satisfy the desired constraint. So we decide to increase the Lagrange multiplier from 1 to 5 and try again. Suppose that plugging \(\mu = 5\) into the program generates output \((x_p, y_p) = (4.1, 28.3)\). Then, \((4.1, 28.3)\) is the solution to the problem of finding \((x, y) \in A\) to minimize \(y\) subject to \(x \leq 4.1\). While the vector \((4.1, 28.3)\) satisfies our desired constraint \(x \leq 5.3\), it might be overly conservative. We can look for solutions that reduce the \(y\) value below 28.3 by decreasing the Lagrange multiplier \(\mu\) to some value lower than 5 but higher than 1. It is reasonable to try again with \(\mu = 3\). If we are very lucky, the resulting output will
have an \( x \)-value exactly equal to 5.3, in which case we are assured this is an exact solution to the desired problem. Otherwise, we can increase or decrease the Lagrange multiplier \( \mu \) accordingly and keep going.

The above paragraph describes an intuitive procedure for changing the Lagrange multiplier \( \mu \) to push towards a desired constraint. It was implicitly assumed that increasing \( \mu \) places more importance on constraint minimization and hence produces solutions \((x,y)\) with \( x \)-values that are either the same or smaller. This turns out to be true, and a precise statement and proof of this fact are worked out in Exercise VII-B.5. There are some remaining ambiguities in the above discussion. Specifically,

- How should we choose the initial value of \( \mu \)?
- By how much should we increase or decrease \( \mu \) at each step?
- Does the above procedure always “converge” to an answer, in the sense that the \( x \)-value gets closer and closer to the desired constraint? If so, how many steps do we need to get close to the desired answer?

Of course, the answer to the third question is “no”: The above procedure does not always converge to an answer \((x,y)\in\mathcal{A}\) that meets the desired constraint with equality. Indeed, there may not even exist a point in the set \( \mathcal{A} \) with an \( x \)-value that is close to the desired constraint. For example, suppose \( \mathcal{A} \) consists only of points \((x,y)\) that take integer values, while the desired constraint is \( x \leq 5.3 \). Furthermore, even if there is a point \((x,y)\in\mathcal{A}\) that meets the desired constraint with equality, it might be impossible to find this point via the above procedure. It is not always possible to find all points \((x,y)\) that satisfy a stronger form of continuity, called \emph{Lipschitz continuity}, in order to ensure that exponential convergence of \( \mu \) to \( \mu^* \) translates to exponential convergence of \((x^*(\mu),y^*(\mu))\) to the optimal solution \((x^*(\mu^*),y^*(\mu^*))\).

On the positive side, there are many constrained optimization problems for which Theorem II.1 can be used to find either exact analytical solutions or numerical approximations. Consider the following two special cases:

1) \textbf{Analytical solutions:} Suppose we can analytically compute a solution \((x^*(\mu),y^*(\mu))\) to the unconstrained problem (7)-(8) for each \( \mu \geq 0 \). Then, for each \( \mu \in [c_{\min},\infty) \), we obtain \( \mu_c \) by finding a value \( \mu_c \geq 0 \) that satisfies the equation \( x^*(\mu_c) = c \) (assuming this equation can be satisfied). Such a value \( \mu_c \) yields \( \psi(c) = y^*(\mu_c) \). For example, suppose the desired constraint is \( x \leq 5.3 \) (so \( c = 5.3 \)), and suppose we analytically determine that \( x^*(\mu) = 16 - 3\mu \). Solving the equation \( 16 - 3\mu = 5.3 \) gives \( \mu_c = 10.7/3 \). The optimum point is then \((x^*(\mu_c),y^*(\mu_c)) = (5.3, y^*(10.7/3)) \).

2) \textbf{Bisection search:} Suppose our computation of (7)-(8) generates points \((x^*(\mu),y^*(\mu))\) for which \( x^*(\mu) \) is continuous in \( \mu \). If we can bracket the desired constraint \( c \) by non-negative values \( \mu_{\min} \) and \( \mu_{\max} \), such that \( x^*(\mu_{\min}) \leq c \leq x^*(\mu_{\max}) \), then a simple bisection procedure can be used to quickly find a value \( \mu \) that satisfies \( x^*(\mu) \approx c \), so that \( y^*(\mu) \approx \psi(c) \). Bisection reduces the size of the search interval \([\mu_{\min},\mu_{\max}]\) by a factor of two at each step, so convergence is \emph{exponentially fast}.

\section*{C. Lagrange multipliers for optimization over \((x_1,\ldots,x_N)\in\mathcal{X})\}

Let \( N \) be a positive integer. Let \( x = (x_1,\ldots,x_N) \) be a vector in \( \mathbb{R}^N \). Consider the following constrained optimization problem:

\begin{align}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g(x) \leq c \quad (13) \\
& \quad x \in \mathcal{X} \quad (14) \\
& \quad x \in \mathcal{X} \quad (15)
\end{align}

where:

- \( \mathcal{X} \) is a general subset of \( \mathbb{R}^N \).
- \( f(x) \) and \( g(x) \) are real-valued functions defined over \( \mathcal{X} \).
- \( c \) is a given real number.

The tradeoff function \( \psi(c) \) associated with problem (13)-(15) is defined as the infimum objective function value over all \( x \in \mathcal{X} \) that satisfy the constraint \( g(x) \leq c \). In particular, if \( x^* \) is a solution to problem (13)-(15) with parameter \( c \), then \( \psi(c) = f(x^*) \). The problem (13)-(15) is similar to the problem (3)-(5). In fact, it can be viewed as a special case of the problem (3)-(5) if we define \( \mathcal{A} \) as the set of all vectors \((g,f)\in\mathbb{R}^2\) such that \((g,f) = (g(x),f(x))\) for some \( x = (x_1,\ldots,x_N) \in \mathcal{X} \). Thus, a Lagrange multiplier approach is also effective for the problem (13)-(15). Fix a Lagrange multiplier \( \mu \geq 0 \) and consider the

\[\text{Lipschitz continuity, in order to ensure that exponential convergence of } \mu \text{ to } \mu^* \text{ translates to exponential convergence of } (x^*(\mu),y^*(\mu)) \text{ to the optimal solution } (x^*(\mu^*),y^*(\mu^*))\]
unconstrained problem:

Minimize: \( f(x) + \mu g(x) \) \hfill (16)

Subject to: \( x \in \mathcal{X} \) \hfill (17)

As before, choosing a large value of \( \mu \) for the problem (16)-(17) places more emphasis on keeping \( g(x) \) small. If \( x^* = (x_1^*, \ldots, x_N^*) \) is an optimal solution to the unconstrained problem (16)-(17), then

\[
 f(x^*) + \mu g(x^*) \leq f(x) + \mu g(x) \quad \text{for all } x \in \mathcal{X}
\] \hfill (18)

**Theorem II.2.** Suppose \( \mu \geq 0 \). If \( x^* = (x_1^*, \ldots, x_N^*) \) is an optimal solution to the unconstrained problem (16)-(17), then it is also an optimal solution to the following problem:

Minimize: \( f(x) \) \hfill (19)

Subject to: \( g(x) \leq g(x^*) \) \hfill (20)

\( x \in \mathcal{X} \) \hfill (21)

**Proof.** Suppose \( x^* = (x_1^*, \ldots, x_N^*) \) solves (16)-(17). Then this vector also satisfies all constraints of problem (19)-(21). Now suppose \( x = (x_1, \ldots, x_N) \) is another vector that satisfies the constraints (20)-(21). We want to show that \( f(x^*) \leq f(x) \). Since \( x \in \mathcal{X} \), we have from (18):

\[
 f(x^*) + \mu g(x^*) \leq f(x) + \mu g(x) \leq f(x) + \mu g(x^*)
\]

where the final inequality holds because \( \mu \geq 0 \) and because \( x \) satisfies (20). Canceling common terms in the above inequality proves \( f(x^*) \leq f(x) \).

The above theorem extends easily to the case of multiple constraints (see Theorem III.1).

**Example II.1.** Minimize the function \( \sum_{i=1}^{N} a_i x_i \) subject to \( \sum_{i=1}^{N} b_i x_i^2 \leq 4 \) and \( (x_1, \ldots, x_N) \in \mathbb{R}^N \), where \( (a_1, \ldots, a_N) \) and \( (b_1, \ldots, b_N) \) are given real numbers such that \( b_i > 0 \) for all \( i \).

Solution: Fix \( \mu \geq 0 \). We minimize \( \sum_{i=1}^{N} a_i x_i + \mu \sum_{i=1}^{N} b_i x_i^2 \) over \( (x_1, \ldots, x_N) \in \mathbb{R}^N \). This is a separable minimization of \( a_i x_i + \mu b_i x_i^2 \) for each variable \( x_i \in \mathbb{R} \). When \( \mu > 0 \), the result is \( x_i = -a_i/(2\mu b_i) \) for \( i \in \{1, \ldots, N\} \). Choosing \( \mu \) to satisfy the constraint with equality gives \( 4 = \sum_{j=1}^{N} b_j x_j^2 = \sum_{j=1}^{N} b_j (-a_j/(2\mu b_j))^2 \). Thus:

\[
 \mu^* = \frac{1}{4} \sqrt{\sum_{j=1}^{N} a_j^2/b_j}
\]

and \( x_i^* = -a_i/(2\mu^* b_i) \) for all \( i \in \{1, \ldots, N\} \). This is optimal by Theorem II.2.

D. Critical points for unconstrained optimization

If a real-valued function \( f(x) \) of a multi-dimensional vector \( x = (x_1, \ldots, x_N) \) is differentiable at a point \( x = (x_1, \ldots, x_N) \in \mathbb{R}^N \), its gradient is the vector of partial derivatives:

\[
 \nabla f(x) = \left[ \frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \ldots, \frac{\partial f(x)}{\partial x_N} \right]
\]

The problem (16)-(17) seeks to find a global minimum of the function \( f(x) + \mu g(x) \) over all vectors \( x = (x_1, \ldots, x_N) \) in the set \( \mathcal{X} \). Recall from basic calculus that, if a global minimum exists, it must occur at a critical point. Specifically, a point \( x^* = (x_1^*, \ldots, x_N^*) \) is a critical point for this problem if \( x^* \in \mathcal{X} \) and if \( x^* \) satisfies at least one of the following three criteria:

- \( x^* \) is on the boundary of \( \mathcal{X} \).
- \( \nabla f(x^*) + \mu \nabla g(x^*) \) does not exist as a finite vector in \( \mathbb{R}^N \).
- \( \nabla f(x^*) + \mu \nabla g(x^*) = 0 \) (where the “0” on the right-hand-side represents the all-zero vector).

The condition \( \nabla f(x^*) + \mu \nabla g(x^*) = 0 \) is called the stationary equation. While this condition arises in the search for a global minimum of \( f(x) + \mu g(x) \), it can also find local minima or local maxima. It turns out that such critical points are often important. In particular, for some non-convex problems, a search for solutions that solve the stationary equation can reveal points \( (g(x), f(x)) \) that lie on the tradeoff curve \( (c, \psi(c)) \), even when it is impossible to find such points via a global minimization (see Appendix A for a development of this idea).

\(^4\) A boundary point of a set \( \mathcal{X} \subseteq \mathbb{R}^N \) is a point \( x \in \mathbb{R}^N \) that is arbitrarily close to points in \( \mathcal{X} \) and also arbitrarily close to points not in \( \mathcal{X} \). The set \( \mathbb{R}^N \) has no boundary points. A set is closed if and only if it contains all of its boundary points. A set is open if and only if it contains none of its boundary points. A point in \( \mathcal{X} \) is an interior point if and only if it is not a boundary point.
Example II.2. Define $X$ as the set of real numbers in the interval $[0, 1]$. Define $f(x) = x^2$ and $g(x) = -x$. Given $\mu \geq 0$, we want to minimize $f(x) + \mu g(x)$ over $x \in [0, 1]$. Find the critical points. Then find the optimal $x^*$.

Solution: The boundary points of $[0, 1]$ are $x = 0$ and $x = 1$. The function $f(x) + \mu g(x)$ is differentiable for all $x$. The stationary equation is $f'(x) + \mu g'(x) = 0$. This produces a critical point $x = \mu/2$. However, this point is only valid if $0 \leq \mu \leq 2$ (since if $\mu > 2$ then $x = \mu/2 > 1$, which is out of the desired interval $[0, 1]$).

Thus, for $\mu \in [0, 2]$ we test the critical points $x \in \{0, 1, \mu/2\}$:

<table>
<thead>
<tr>
<th>$x$</th>
<th>$x^2 - \mu x$ for $\mu \in [0, 2]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>$1 - \mu$</td>
</tr>
<tr>
<td>$\mu/2$</td>
<td>$-\mu^2/4$</td>
</tr>
</tbody>
</table>

It can be shown that $-\mu^2/4 \leq 1 - \mu$ for all $\mu \in \mathbb{R}$. Thus, $x^* = \mu/2$ whenever $\mu \in [0, 2]$. If $\mu > 2$ then the critical points occur at $x = 0$ and $x = 1$ (with $x^2 - \mu x$ values of $0$ and $1 - \mu$, respectively). Since $1 - \mu < 0$ whenever $\mu > 2$, it follows that $x^* = 1$ whenever $\mu > 2$. In summary:

$$x^* = \begin{cases} \mu/2 & \text{if } \mu \in [0, 2] \\ 1 & \text{if } \mu > 2 \end{cases}$$ (22)

A simpler method for obtaining (22) uses convexity theory together with Lemma IV.5 (given in a later section). See also Exercise VIII-D.19.

Example II.3. Define $X$ as the set of real numbers in the interval $[0, 1]$. Define $f(x) = |x - 1/2|$ and $g(x) = x^2$. Given $\mu = 2$, we want to minimize $f(x) + 2g(x)$ over $x \in [0, 1]$. Find the critical points. Then find the optimal $x^*$.

Solution: The boundary points of $[0, 1]$ are $x = 0$ and $x = 1$. The point where $f(x) + 2g(x)$ is not differentiable is $x = 1/2$. If $x \in (1/2, 1]$ then $f(x) + 2g(x) = x - 1/2 + 2x^2$ and $f''(x) + 2g''(x) = 0$ only when $x = -1/4$, which is not in the interval $[0, 1]$. If $x \in [0, 1/2)$ then $f(x) + 2g(x) = 1/2 - x + 2x^2$ and $f''(x) + 2g''(x) = 0$ when $x = 1/4$. Thus, the critical points to test are $x \in \{0, 1/4\}$.

| $x$ | $|x - 1/2| + 2x^2$ |
|-----|------------------|
| 0   | 1/2              |
| 1/2 | 5/2              |
| 1/4 | 3/8              |

Thus, the optimal point is $x^* = 1/4$, which achieves the minimum of $f(x^*) + 2g(x^*) = 3/8$.

E. Rate allocation example

Consider three devices that send data over a common link of capacity $C$ bits/second. Let $(r_1, r_2, r_3)$ be the vector of data rates selected for the three devices. Define $X$ as the set of all non-negative rate vectors in $\mathbb{R}^3$:

$$X = \{(r_1, r_2, r_3) \in \mathbb{R}^3 | r_1 \geq 0, r_2 \geq 0, r_3 \geq 0\}$$

Consider the following weighted proportionally fair rate allocation problem:

Maximize: $\log(r_1) + 2\log(r_2) + 3\log(r_3)$

Subject to:

$$r_1 + r_2 + r_3 \leq C$$

$$(r_1, r_2, r_3) \in X$$

The weights imply that higher indexed devices have higher priority in the rate maximization. To solve, turn this into a minimization problem as follows:

Minimize: $(-1)[\log(r_1) + 2\log(r_2) + 3\log(r_3)]$

Subject to:

$$r_1 + r_2 + r_3 \leq C$$

$$(r_1, r_2, r_3) \in X$$

The corresponding unconstrained problem (which uses a Lagrange multiplier $\mu \geq 0$) is:

Minimize: $-[\log(r_1) + 2\log(r_2) + 3\log(r_3)] + \mu [r_1 + r_2 + r_3]$

Subject to:

$$(r_1, r_2, r_3) \in X$$
Since \( \mathcal{X} \) is the set of all \((r_1, r_2, r_3)\) that satisfy \( r_1 \geq 0, r_2 \geq 0, r_3 \geq 0 \), the above problem is separable and can be solved by separately minimizing over each \( r_i \):

- Choose \( r_1 \geq 0 \) to minimize \(-\log(r_1) + \mu r_1\). Thus, \( r_1 = 1/\mu \) (assuming \( \mu > 0 \)).
- Choose \( r_2 \geq 0 \) to minimize \(-2\log(r_2) + \mu r_2\). Thus, \( r_2 = 2/\mu \) (assuming \( \mu > 0 \)).
- Choose \( r_3 \geq 0 \) to minimize \(-3\log(r_3) + \mu r_3\). Thus, \( r_3 = 3/\mu \) (assuming \( \mu > 0 \)).

Since we have a solution parameterized by \( \mu \), we can choose \( \mu > 0 \) to meet the desired constraint with equality:

\[
C = r_1 + r_2 + r_3 = 6/\mu
\]

Thus \( \mu = 6/C \), which is indeed non-negative. Thus, Theorem II.2 ensures that this rate allocation is optimal for the original constrained optimization problem:

\[
(r_1^*, r_2^*, r_3^*) = (C/6, C/3, C/2)
\]

E. Routing example

Consider sending data of rate \( r \) over a choice of three parallel queues, each with a processing rate of \( C \). The data is split into separate streams of rates \( x_1, x_2, x_3 \), where data of rate \( x_i \) is sent into queue \( i \) (see Fig. 4). The first two queues have no additional traffic, while the third queue serves an additional traffic stream of rate \( 1 \). Suppose the average number of packets in queue \( i \) is equal to \( f_i/(C - f_i) \), where \( f_i \) is the total arrival rate to the queue \( i \).

We want to choose \( x_1, x_2, x_3 \) to minimize total average number of packets in the system. That is, we want to solve:

Minimize:

\[
\frac{x_1}{C-x_1} + \frac{x_2}{C-x_2} + \frac{x_3+1}{C-(x_3+1)}
\]  

Subject to:

\[
\begin{align*}
1 & \leq x_1 + x_2 + x_3 \\
x_1 & \in [0, C), x_2 \in [0, C), x_3 \in [0, C - 1]
\end{align*}
\]

Assume that \( C > 1 \) and \( r + 1 < 3C \), so that the above problem is feasible. Since the objective function (23) is increasing in the \( x_i \) values, it is clear that an optimal solution should satisfy constraint (24) with equality (otherwise, the solution could be improved by reducing one or more values of \( x_i \)). Thus, the solution to (23)-(25) is the same as the solution to a modified problem that replaces constraint (24) with the equality constraint \( r = x_1 + x_2 + x_3 \). This equality constraint is more natural for the problem. However, we have used the inequality constraint (24) because it conforms to the inequality structure of Theorem II.2 (a related theorem that specifically deals with equality constraints is given in Theorem II.3 of Section II-H).

To solve the problem (23)-(25), define \( \mathcal{X} \) as the set of all \((x_1, x_2, x_3)\) that satisfy (25). Next, note that the constraint (24) has an inverted inequality. This can be put in the standard form:

\[-(x_1 + x_2 + x_3) \leq -r\]

Now fix a Lagrange multiplier \( \mu > 0 \) and consider the unconstrained problem:

Minimize:

\[
\frac{x_1}{C-x_1} + \frac{x_2}{C-x_2} + \frac{x_3+1}{C-(x_3+1)} - \mu(x_1 + x_2 + x_3)
\]

Subject to:

\[
\begin{align*}
x_1 & \in [0, C), x_2 \in [0, C), x_3 \in [0, C - 1]
\end{align*}
\]

The approach is to obtain a solution \((x_1^*, x_2^*, x_3^*)\) as a function of \( \mu \), and then to choose \( \mu \) to meet the inequality constraint (24) with equality: \( x_1^* + x_2^* + x_3^* = r \). If this can be done, then Theorem II.2 ensures the result is optimal for the original constrained problem (23)-(25).

The variables \( x_1, x_2, x_3 \) can be optimized separately in the unconstrained problem (26)-(27):

\footnote{This is an M/M/1 approximation for the average number of packets in each queue.}
• Choose $x_1 \in [0, C)$ to minimize $x_1/(C - x_1) - \mu x_1$.
• Choose $x_2 \in [0, C)$ to minimize $x_2/(C - x_2) - \mu x_2$.
• Choose $x_3 \in [0, C - 1)$ to minimize $(x_3 + 1)/(C - (x_3 + 1)) - \mu x_3$.

First look for critical points that correspond to derivatives of zero:
\[
\frac{d}{dx}\left[\frac{x_1}{C - x_1} - \mu x_1\right] = 0 \implies \frac{C}{(C - x_1)^2} = \mu \implies x_1 = C - \sqrt{C/\mu}
\]
\[
\frac{d}{dx}\left[\frac{x_2}{C - x_2} - \mu x_2\right] = 0 \implies \frac{C}{(C - x_2)^2} = \mu \implies x_2 = C - \sqrt{C/\mu}
\]
\[
\frac{d}{dx}\left[\frac{x_3 + 1}{C - (x_3 + 1)} - \mu x_3\right] = 0 \implies \frac{C}{(C - (x_3 + 1))^2} = \mu \implies x_3 = C - 1 - \sqrt{C/\mu}
\]

The intuition behind the above solution is that the derivatives of the individual link cost functions should be equalized to a common value of $\mu$. However, one must also ensure that the $x_1, x_2, x_3$ values are non-negative. It can be shown that the true solution to the individual minimization problems is found by simply projecting the above values onto the non-negative real numbers:
\[
x_1^* = \left[ C - \sqrt{C/\mu} \right]^+ \quad (x_1^* > 0 \text{ whenever } \mu > 1/C)
\]
\[
x_2^* = \left[ C - \sqrt{C/\mu} \right]^+ \quad (x_2^* > 0 \text{ whenever } \mu > 1/C)
\]
\[
x_3^* = \left[ C - 1 - \sqrt{C/\mu} \right]^+ \quad (x_3^* > 0 \text{ whenever } \mu > C/(C - 1)^2)
\]

Thus, there are two different regimes: The first regime is when $1/C < \mu \leq C/(C - 1)^2$ and has $x_1^* = x_2^* > 0$ and $x_3^* = 0$. The second regime is when $\mu > C/(C - 1)^2$ and has $x_1^* = x_2^* > 0$, $x_3^* > 0$. The transition between these two regimes occurs when $\mu = C/(C - 1)^2$.

In the first regime, choosing $\mu$ to satisfy the desired constraint with equality ensures that $x_1^* + x_2^* + 0 = r$, and so $x_1^* = x_2^* = r/2$. This implies that $r/2 = C - \sqrt{C/\mu}$. The transition point $\mu = C/(C - 1)^2$ gives rise to the transition rate $r$ that satisfies $r/2 = C - \sqrt{C/(C - 1)^2}$ (so that the transition rate is $r = 2$). This transition rate is intuitive: It is exactly when the derivative of the cost function of paths 1 and 2 (evaluated when $x_1 = x_2 = 1$) is equal to the derivative of the cost function of path 3 (when $x_3 = 0$):
\[
\frac{d}{dx_1}\left[\frac{x_1}{C - x_1}\right]_{x_1=1} = \frac{d}{dx_2}\left[\frac{x_2}{C - x_2}\right]_{x_2=1} = \frac{d}{dx_3}\left[\frac{x_3 + 1}{C - (x_3 + 1)}\right]_{x_3=0}
\]

Thus, when $0 < r \leq 2$, the optimal solution is $x_1^* = x_2^* = r/2$, $x_3^* = 0$.

On the other hand, when $2 < r < 3C - 1$, the optimal solution has $r = x_1^* + x_2^* + x_3^* = 2(C - \sqrt{C/\mu}) + (C - 1 - \sqrt{C/\mu})$. This means that:
\[
\mu^* = C\left(\frac{3}{3C - 1 - r}\right)^2
\]

and the optimal solution is:
\[
x_1^* = x_2^* = C - \sqrt{C/\mu^*} = (r + 1)/3
\]
\[
x_3^* = C - 1 - \sqrt{C/\mu^*} = (r - 2)/3
\]

This solution is intuitive because it equalizes the total input rate on each path, and hence also equalizes the individual path derivatives:
\[
\frac{d}{dx_1}\left[\frac{x_1}{C - x_1}\right]_{x_1=x_1^*} = \frac{d}{dx_2}\left[\frac{x_2}{C - x_2}\right]_{x_2=x_2^*} = \frac{d}{dx_3}\left[\frac{x_3 + 1}{C - (x_3 + 1)}\right]_{x_3=x_3^*}
\]

In summary, the solution to (23)-(25) is:
\[
(x_1^*, x_2^*, x_3^*) = \begin{cases} 
(\frac{r + 1}{3}, \frac{r + 1}{3}, \frac{r + 2}{3}) & \text{if } 0 < r \leq 2 \\
(\frac{r + 1}{2}, \frac{r + 1}{2}, \frac{r - 2}{2}) & \text{if } 2 < r < 3C - 1
\end{cases}
\]

G. Power allocation example

Consider a collection of $N$ orthogonal channels. A wireless transmitter can send simultaneously over all channels using a power vector $p = (p_1, \ldots, p_N)$. For each channel $i$, let $f_i(p_i)$ be the transmission rate over the channel. The goal is to maximize the sum transmission rate subject to a sum power constraint of $p_{\text{max}}$ (for some given value $p_{\text{max}} > 0$):

Maximize: \[
\sum_{i=1}^{N} f_i(p_i)
\] (28)

Subject to: \[
\sum_{i=1}^{N} p_i \leq p_{\text{max}}
\] (29)

\[p_i \geq 0 \quad \forall i \in \{1, \ldots, N\}\] (30)
For this example, assume that each function \( f_i(p) \) is increasing over the interval \( p \in [0, p_{max}] \). Further assume the function is differentiable and has a decreasing derivative, so that if \( p_1, p_2 \) are in the interval \([0, p_{max}]\) and if \( p_1 < p_2 \), then \( f'(p_1) > f'(p_2) \). Such a function \( f_i(p) \) has a diminishing returns property with each incremental increase in \( p \), and can be shown to be a strictly concave function (a formal definition of strictly concave is given in Section IV). An example is:

\[
 f_i(p) = \log(1 + \gamma_i p)
\]

where \( \gamma_i \) is a positive attenuation parameter for each channel \( i \in \{1, \ldots, N\} \).

Converting to a minimization problem gives:

\[
\begin{align*}
\text{Minimize:} & \quad -\sum_{i=1}^{N} f_i(p_i) \\
\text{Subject to:} & \quad \sum_{i=1}^{N} p_i \leq p_{max} \quad (32) \\
& \quad p_i \geq 0 \quad \forall i \in \{1, \ldots, N\} \quad (33)
\end{align*}
\]

The set \( \mathcal{X} \) is considered to be the set of all \((p_1, \ldots, p_N)\) that satisfy (33). The corresponding unconstrained problem, with Lagrange multiplier \( \mu \geq 0 \), is:

\[
\begin{align*}
\text{Minimize:} & \quad -\sum_{i=1}^{N} f_i(p_i) + \mu \sum_{i=1}^{N} p_i \\
\text{Subject to:} & \quad p_i \geq 0 \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\]

This problem separates into \( N \) different minimization problems: For each \( i \in \{1, \ldots, N\} \), solve the following:

\[
\begin{align*}
\text{Maximize:} & \quad f_i(p_i) - \mu p_i \\
\text{Subject to:} & \quad p_i \geq 0
\end{align*}
\]

where the minimization has been changed to a maximization for simplicity. Each separate problem is a simple maximization of a function of one variable over the interval \( p_i \in [0, \infty) \). The optimal \( p_i \) is either a critical point (being either the endpoint \( p_i = 0 \) or a point with zero derivative), or is achieved at \( p_i = \infty \). Because the functions \( f_i(p) \) are assumed to have decreasing derivatives, it can be shown that a solution to (34)-(35) is as follows:

- If \( f_i'(0) \leq \mu \) then \( p_i = 0 \).
- Else, if \( f_i'(z) = \mu \) for some \( z > 0 \) then \( p_i = z \).
- Else, if \( f_i'(\infty) \geq \mu \) then \( p_i = \infty \).

Assume the channels are rank ordered so that:

\[
 f_1'(0) \geq f_2'(0) \geq f_3'(0) \geq \cdots \geq f_N'(0)
\]

(36)

Assume that \( \mu \) is large enough so that \( f_i'(\infty) < \mu \) for all \( i \), and small enough so that \( f_i'(0) > \mu \). Define \( K \) as the largest integer such that \( f_i'(0) > \mu \) for all \( i \in \{1, \ldots, K\} \). Then an optimal solution to (34)-(35) has:

- \( f_i'(p_i) = \mu \) for \( i \in \{1, \ldots, K\} \).
- \( p_i = 0 \) for \( i \in \{K + 1, \ldots, N\} \).

The value of \( \mu \) is shifted appropriately until the above solution satisfies the power constraint \( \sum_{i=1}^{N} p_i = p_{max} \) with equality. By Theorem II.2 that value \( \mu \) yields a power vector \( (p_1^*, \ldots, p_N^*) \) that is an optimal solution to the original constrained optimization problem. An illustration of the solution is given in Fig. 5. The arrows in the figure show how the \((p_1^*, \ldots, p_N^*)\) values move to the right as \( \mu \) is pushed down.

For a specific example, when the \( f_i(p) = \log(1 + \gamma_i p) \) for all \( i \), and when \( \mu > 0 \), the problem (34)-(35) becomes:

\[
\begin{align*}
\text{Maximize:} & \quad \log(1 + \gamma_i p_i) - \mu p_i \\
\text{Subject to:} & \quad p_i \geq 0
\end{align*}
\]

The solution is:

\[
p_i = \left[ \frac{1}{\mu} - \frac{1}{\gamma_i} \right]^+ \quad (37)
\]

where \([x]^+ = \max[x, 0]\). That is, \( p_i > 0 \) if and only if \( 1/\mu > 1/\gamma_i \). In this case, the parameter \( 1/\mu \) should be increased, starting from 0, until:

\[
\sum_{i=1}^{N} \left[ \frac{1}{\mu} - \frac{1}{\gamma_i} \right]^+ = p_{max} \quad (38)
\]

When such a value \( \mu \) is found, it follows from Theorem II.2 that the resulting powers \( p_i \) given by (37) are optimal. The rank ordering (36) implies:

\[
\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_N
\]
Fig. 5. An illustration of the derivative requirement for optimality in the problem (28)-(30). As the value $\mu$ is pushed down, the $p_1$ and $p_2$ values increase along their respective curves. Currently $p_3 = 0$. Pushing $\mu$ below the $f_3'(0)$ threshold activates the third curve with $p_3 > 0$.

Intuitively, this means that better channels come first in the rank ordering. For some integer $K \in \{1, \ldots, N\}$ the optimal solution has $p_i > 0$ for $i \in \{1, \ldots, K\}$ and $p_i = 0$ for $i > K$. One way to solve this is to consider all potential values of $K$, starting with $K = N$:

- Assume $K = N$. Then $p_i > 0$ for all $i$, and so $1/\mu \geq 1/\gamma_i$ for all $i \in \{1, \ldots, N\}$. The equation (38) becomes:

$$\sum_{i=1}^{N} \left( \frac{1}{\mu} - \frac{1}{\gamma_i} \right) = p_{\max}$$

and so:

$$\frac{1}{\mu} = \frac{p_{\max} + \sum_{i=1}^{N} 1/\gamma_i}{N}$$

If this $1/\mu$ value indeed satisfies $1/\mu \geq 1/\gamma_i$ for all $i \in \{1, \ldots, N\}$, we are done. Else go to the next step.

- Assume $K = N - 1$, so that $p_i > 0$ for $i \in \{1, \ldots, N - 1\}$ and $p_N = 0$. Then $1/\mu \geq 1/\gamma_i$ for all $i \in \{1, \ldots, N - 1\}$ and $1/\mu < 1/\gamma_N$. The equation (38) becomes:

$$\sum_{i=1}^{N-1} \left( \frac{1}{\mu} - \frac{1}{\gamma_i} \right) = p_{\max}$$

and so:

$$\frac{1}{\mu} = \frac{p_{\max} + \sum_{i=1}^{N-1} 1/\gamma_i}{N - 1}$$

If this $1/\mu$ value indeed satisfies $1/\mu \geq 1/\gamma_i$ for all $i \in \{1, \ldots, N - 1\}$, we are done. Else go to the next step.

- and so on.

The above procedure involves at most $N$ steps. One can speed up the procedure by performing a bisection like search, rather than a sequential search, which is helpful when $N$ is large.

H. Equality constraints

Consider a problem where the inequality constraint is replaced with an equality constraint:

Minimize: $f(x)$ \hspace{1cm} (39)

Subject to: $g(x) = c$ \hspace{1cm} (40)

$x \in X$ \hspace{1cm} (41)

where $x = (x_1, \ldots, x_N)$, $X \subset \mathbb{R}^N$, and $f(x)$ and $g(x)$ are real-valued functions over $X$. The Lagrange multiplier approach considers the unconstrained problem defined by a parameter $\lambda \in \mathbb{R}$:

Minimize: $f(x) + \lambda g(x)$ \hspace{1cm} (42)

Subject to: $x \in X$ \hspace{1cm} (43)
The only difference is that for equality constraints, the Lagrange multiplier \( \lambda \) can possibly be a negative value.

**Theorem II.3.** If \( x^* \) is a solution to (42)-(43), then \( x^* \) is also a solution to:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g(x) = g(x^*)
\end{align*}
\]

\( x \in \mathcal{X} \)

**Proof.** The proof is almost identical to that of Theorem II.2 and is left as an exercise (see Exercise VIII-B.2). \( \square \)

As before, in the special case when \( f(x) \) and \( g(x) \) are differentiable over the interior of \( \mathcal{X} \), solutions to the stationary equation \( \nabla f(x) + \lambda \nabla g(x) = 0 \) are often useful even if they do not correspond to a global minimum of \( f(x) + \lambda g(x) \) over \( x \in \mathcal{X} \) (see Appendix A).

### III. LAGRANGE MULTIPLIERS FOR MULTIPLE CONSTRAINTS

Lagrange multiplier theory extends naturally to the case of multiple constraints. Fix \( N \) and \( K \) as positive integers. Let \( \mathcal{X} \) be a subset of \( \mathbb{R}^N \). Let \( f, g_1, \ldots, g_K \) be real-valued functions over \( x \in \mathcal{X} \). Let \( c = (c_1, \ldots, c_K) \) be a given vector in \( \mathbb{R}^N \). The constrained problem is:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g_k(x) \leq c_k, \quad \forall k \in \{1, \ldots, K\} \\
& \quad x \in \mathcal{X}
\end{align*}
\]

The **unconstrained problem** uses Lagrange multipliers \( \mu_1, \ldots, \mu_K \), where \( \mu_k \geq 0 \) for all \( k \in \{1, \ldots, K\} \):

\[
\begin{align*}
\text{Minimize:} & \quad f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \\
\text{Subject to:} & \quad x \in \mathcal{X}
\end{align*}
\]

As in the one-constraint case, there is a connection between the constrained and unconstrained problems.

**Theorem III.1.** *(Lagrange multipliers for multiple constraints)* If \( x^* \) is a solution to the unconstrained problem (44)-(46) corresponding to a nonnegative Lagrange multiplier vector \( \mu = (\mu_1, \ldots, \mu_K) \), then it also solves the constrained problem (44)-(46) whenever \( (c_1, \ldots, c_K) \) is such that \( g_k(x^*) \leq c_k \) for all \( k \in \{1, \ldots, K\} \) and \( g_k(x^*) = c_k \) whenever \( \mu_k > 0 \).

**Proof.** Suppose \( x^* \) solves the unconstrained problem and \( (c_1, \ldots, c_K) \) is such that \( g_k(x^*) \leq c_k \) for all \( k \), and \( g_k(x^*) = c_k \) whenever \( \mu_k > 0 \). Then \( x^* \) clearly satisfies all constraints of the constrained problem, and:

\[
\mu_k g_k(x^*) = \mu_k c_k, \quad \forall k \in \{1, \ldots, K\} \tag{49}
\]

Indeed, the equality (49) can be verified by considering cases \( \mu_k = 0 \) and \( \mu_k > 0 \). Suppose \( x \) is another vector that satisfies all constraints of the constrained problem (so \( x \in \mathcal{X} \) and \( g_k(x) \leq c_k \) for all \( k \)). We want to show that \( f(x^*) \leq f(x) \). Since \( x^* \) solves the unconstrained problem and \( x \in \mathcal{X} \), we have:

\[
\begin{align*}
f(x^*) + \sum_{k=1}^{K} \mu_k g_k(x^*) & \overset{(a)}{\leq} f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \\
& \overset{(b)}{=} f(x) + \sum_{k=1}^{K} \mu_k c_k \\
& \overset{(c)}{=} f(x) + \sum_{k=1}^{K} \mu_k g_k(x^*)
\end{align*}
\]

where (a) holds by definition of \( x^* \) being optimal for the the unconstrained problem, (b) holds because \( \mu_k \geq 0 \) and \( g_k(x) \leq c_k \) for all \( k \), and (c) holds by (49). Canceling the common term on both sides proves that \( f(x^*) \leq f(x) \). \( \square \)

Unfortunately, the above theorem for multiple constraints is more difficult to apply than the corresponding theorem for one constraint. Why? Let us suppose there is indeed a nonnegative vector \( \mu^* = (\mu_1^*, \ldots, \mu_K^*) \) such that the optimal solution to the unconstrained problem for this vector \( \mu^* \) is a point \( x^* \) such that \( g_k(x^*) = c_k \) for all \( k \in \{1, \ldots, K\} \) \(^6\). In that case, the above theorem ensures this point \( x^* \) also solves the desired constrained optimization problem. However, how can we find this Lagrange multiplier vector \( (\mu_1^*, \ldots, \mu_K^*) \)? Even if we knew a bound \( \mu_{max} \) on the maximum component, so that \( 0 \leq \mu_k^* \leq \mu_{max} \) for all \( k \in \{1, \ldots, K\} \), we would still need to search over the entire \( K \)-dimensional hypercube \([0, \mu_{max}]^K\).

\(^6\)In fact, problems with many constraints often do not have solutions that meet all constraints with equality. Typically, an optimal solution \( x^* \) satisfies some constraints with equality (such as \( g_1(x^*) = c_1 \)) and satisfies others with strict inequality (such as \( g_2(x^*) = c_2 - 1/2 \)).
particular vector \((\mu_1, \ldots, \mu_K)\) in this hypercube, and the resulting solution to the unconstrained problem is a vector \(x^*\). How do we change our Lagrange multiplier vector to improve on this solution in the next try?

For example, suppose \(K = 2\) and we want to solve:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g_1(x) \leq 5 \\
& \quad g_2(x) \leq 8 \\
& \quad x \in \mathcal{X}
\end{align*}
\]

Let's start with a guess Lagrange multiplier vector \((\mu_1, \mu_2) = (1, 1)\). Suppose the resulting solution to the unconstrained problem of minimizing \(f(x) + \mu_1 g_1(x) + \mu_2 g_2(x)\) over \(x \in \mathcal{X}\) is a vector \(x^*_\mu\) that satisfies \(g_1(x^*_\mu) = 6\) and \(g_2(x^*_\mu) = 9\). How should we change \(\mu\)? We need to bring both constraints down. So, should we increase \(\mu_1\) and \(\mu_2\)? Will such an increase surely reduce both constraints? Not necessarily (see Exercise VIII-B.6). In contrast, problems with only one constraint have only one Lagrange multiplier, and it is easy to determine whether we should increase or decrease \(\mu\) to improve a solution (recall the discussion after Theorem II.1 in Section II-B). A simple bisection search can be done in the one-constraint case, whereas that does not seem possible with multiple constraints.

Fortunately, there are systematic ways of adaptively changing the Lagrange multiplier vector \((\mu_1, \ldots, \mu_K)\) to converge to a desired vector. However, those methods require further structure on the problem and further analytical development. A drift-plus-penalty method is developed in Section VI for this purpose. For intuition, it is useful to understand more deeply how changes in the Lagrange multipliers impact multi-dimensional constraints. Such intuition is given in the following theorem.

**Theorem III.2.** (Changing Lagrange multipliers) Fix \(N\) and \(K\) as positive integers. Let \(\mathcal{X}\) be a subset of \(\mathbb{R}^N\). Let \(f, g_1, \ldots, g_K\) be real-valued functions over \(x \in \mathcal{X}\). Define \(g : \mathbb{R}^N \to \mathbb{R}^K\) as the vector-valued function \(g(x) = (g_1(x), \ldots, g_K(x))\). Let \(\mu = (\mu_1, \ldots, \mu_K)\) be a given vector of real numbers, and let \(x_\mu\) be a solution to the following problem:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) + \mu^T g(x) \\
\text{Subject to:} & \quad x \in \mathcal{X}
\end{align*}
\]

where \(\mu^T g(x) = \sum_{k=1}^{K} \mu_k g_k(x)\). Let \(\lambda = (\lambda_1, \ldots, \lambda_K)\) be another given vector, and let \(x_\lambda\) be a solution to the following:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) + \lambda^T g(x) \\
\text{Subject to:} & \quad x \in \mathcal{X}
\end{align*}
\]

Then the vectors \((\lambda - \mu)\) and \((g(x_\lambda) - g(x_\mu))\) have a nonpositive inner product:

\[
(\lambda - \mu)^T (g(x_\lambda) - g(x_\mu)) \leq 0
\]

**Proof.** The proof uses only the definition of \(x_\mu\) and \(x_\lambda\) being optimal for their corresponding problems (for example, you can start with the inequality \(f(x_\mu) + \mu^T g(x_\mu) \leq f(x_\lambda) + \mu^T g(x_\lambda)\)). Details are worked out as Exercise VIII-B.6.

Start with a nonnegative vector \(\mu\). The above theorem shows that we can control the direction of change in the constraint vector \((g_1(x_\mu), \ldots, g_K(x_\mu))\) by choosing a new Lagrange multiplier vector \(\lambda\) such that the difference vector \(\lambda - \mu\) is opposite to the direction of desired change. Thus, to push a vector \(g(x_\mu) = (g_1(x_\mu), \ldots, g_K(x_\mu))\) closer to the constant vector \(c = (c_1, \ldots, c_K)\), it makes sense to choose \(\lambda = \mu + \delta(g(x_\mu) - c)\), where \(\delta > 0\) is some step size. Of course, \(\lambda\) must be nonnegative to be a valid Lagrange multiplier vector. Hence, one can consider the heuristic that takes a max with 0:

\[
\lambda_k = \max[\mu_k + \delta(g_k(x_\mu) - c_k), 0], \quad \forall k \in \{1, \ldots, K\}
\]

This heuristic for changing the Lagrange multipliers can in fact be analyzed in certain cases II.2, and is called the **dual subgradient algorithm** (with fixed step-size \(\delta > 0\)). It is similar to the drift-plus-penalty algorithm described in Section VI.

**IV. Convexity**

**A. Convex sets**

Let \(\mathcal{X}\) be a subset of \(\mathbb{R}^N\).

**Definition IV.1.** A set \(\mathcal{X} \subseteq \mathbb{R}^N\) is convex if for any two points \(x\) and \(y\) in \(\mathcal{X}\), the line segment between those points is also in \(\mathcal{X}\). That is, for any \(\theta \in [0, 1]\), we have \(\theta x + (1 - \theta)y \in \mathcal{X}\).

By convention, the empty set is considered to be convex. Likewise, a set with only one element is convex. It can be shown that the intersection of two convex sets is still convex. Indeed, let \(\mathcal{A}\) and \(\mathcal{B}\) be convex sets in \(\mathbb{R}^N\). Let \(x\) and \(y\) be two points in \(\mathcal{A} \cap \mathcal{B}\). Since both points \(x\) and \(y\) are in \(\mathcal{A}\), the line segment between them must also be in \(\mathcal{A}\) (since \(\mathcal{A}\) is convex). Similarly, the line segment must be in \(\mathcal{B}\). So the line segment is in \(\mathcal{A} \cap \mathcal{B}\). By the same argument, it follows that the intersection of an arbitrary (possibly uncountably infinite) number of convex sets is convex.
For a vector \( x = (x_1, \ldots, x_N) \in \mathbb{R}^N \), define the norm \( ||x|| = \sqrt{\sum_{i=1}^{N} x_i^2} \). The set \( \{ x \in \mathbb{R}^N \text{ such that } ||x|| = 1 \} \) is not convex because it contains the point \((1,0,0,\ldots,0)\) and \((-1,0,0,\ldots,0)\), but does not contain \( \frac{1}{2}(1,0,0,\ldots,0) + \frac{1}{2}(-1,0,0,\ldots,0) = (0,0,0,\ldots,0) \).

**Example IV.1.** Let \( a \in \mathbb{R}^N \). Define \( A = \{ x \in \mathbb{R}^N \text{ such that } ||x-a|| \leq 1 \} \). The set \( A \) is convex.

**Proof.** Let \( x \) and \( y \) be points in \( A \) and let \( \theta \in [0,1] \). We want to show that \( \theta x + (1-\theta)y \in A \). We have:

\[
||\theta x + (1-\theta)y - a|| = ||\theta(x-a) + (1-\theta)(y-a)|| \\
\leq ||\theta(x-a)|| + ||(1-\theta)(y-a)|| \\
= \theta||x-a|| + (1-\theta)||y-a|| \\
\leq \theta + (1-\theta) \\
= 1
\]

where (51) is the triangle inequality, and (52) holds because \( x \) and \( y \) are both in \( A \).

\( \square \)

**B. Convex sets contain their convex combinations**

Let \( \mathcal{X} \) be a subset of \( \mathbb{R}^N \). A convex combination of points in \( \mathcal{X} \) is a vector \( x \) of the form:

\[
x = \sum_{i=1}^{k} \theta_i x_i
\]

where \( k \) is a positive integer, \( \{x_1, \ldots, x_k\} \) are vectors in \( \mathcal{X} \), and \( \theta_1, \ldots, \theta_k \) are non-negative numbers that sum to 1. If \( \mathcal{X} \) is a convex set, then it contains all convex combinations of two of its points (by definition of convex). That is, if \( \mathcal{X} \) is convex and \( x_1, x_2 \) are in \( \mathcal{X} \) then (by definition of convex):

\[
\theta_1 x_1 + \theta_2 x_2 \in \mathcal{X}
\]

whenever \( \theta_1, \theta_2 \) are non-negative and satisfy \( \theta_1 + \theta_2 = 1 \). By induction, it can be shown that if \( \mathcal{X} \) is a convex set, then it contains all convex combinations of its points (for any positive integer \( k \)). That is, if \( \mathcal{X} \) is convex, if \( x_1, \ldots, x_k \) are points in \( \mathcal{X} \), and if \( \theta_1, \ldots, \theta_k \) are non-negative numbers that sum to 1, then:

\[
\sum_{i=1}^{k} \theta_i x_i \in \mathcal{X}
\]

The value \( \sum_{i=1}^{k} \theta_i x_i \) can be viewed as an expectation \( \mathbb{E}[X] \) of a random vector \( X \) that takes values in the \( k \)-element set \( \{x_1, \ldots, x_k\} \) with probabilities \( \theta_1, \ldots, \theta_k \). Thus, if \( \mathcal{X} \) is a convex set and \( X \) is a random vector that takes one of a finite number of values in \( \mathcal{X} \), the expression (53) means that \( \mathbb{E}[X] \in \mathcal{X} \). This holds true more generally for random vectors \( X \) that can take a possibly infinite number of outcomes, where the expectation \( \mathbb{E}[X] \) is defined either in terms of a summation over a probability mass function or an integral over a distribution function. This is formalized in the following lemma.

**Lemma IV.1.** Let \( X \) be a random vector that takes values in a set \( \mathcal{X} \subseteq \mathbb{R}^N \). If \( \mathcal{X} \) is convex and if \( \mathbb{E}[X] \) is finite, then \( \mathbb{E}[X] \in \mathcal{X} \).

In the special case when the set \( \mathcal{X} \) is closed and the expectation \( \mathbb{E}[X] \) can be approached arbitrarily closely by a convex combination of a finite number of points in \( \mathcal{X} \), then Lemma [IV.1] holds by (53) together with the fact that closed sets contain their boundary points. The proof for general convex sets \( \mathcal{X} \) is nontrivial and is omitted for brevity.\(^7\) Lemma [IV.1] is used to prove an inequality called Jensen’s inequality in Exercise [VIII-D.14].

\(^7\)Lemma [IV.1] holds for any convex set \( \mathcal{X} \), regardless of whether or not it is closed and/or bounded. The proof of this fact uses the hyperplane separation theorem for \( \mathbb{R}^N \) together with induction on the dimensionality of the problem. In particular, if \( \mathbb{E}[X] \) is finite but is not in \( \mathcal{X} \), then there is a \((N-1)\)-dimensional hyperplane that passes through \( \mathbb{E}[X] \) and contains \( \mathcal{X} \) in its upper half. In particular, there is a nonzero vector \( \gamma \) such that \( \gamma^T x \geq \gamma^T \mathbb{E}[X] \) for all \( x \in \mathcal{X} \), and hence \( \gamma^T X \geq \gamma^T \mathbb{E}[X] \) for all realizations of the random variable \( X \). It follows that, with probability 1, the random vector \( X \) lies on the (smaller dimensional) hyperplane for which it is known (by the induction hypothesis) that the expectation cannot leave the convex set. The assumption that \( \mathbb{E}[X] \) is finite is important. For example, one can define \( \mathcal{X} = \mathbb{R} \) (note that the set \( \mathbb{R} \) is convex) and choose any random variable \( X \) with an infinite mean. Then \( X \in \mathbb{R} \) always, but \( \mathbb{E}[X] = \infty \) \( \notin \mathbb{R} \).
C. Convex functions

Let $\mathcal{X} \subseteq \mathbb{R}^N$ be a convex set. Let $f : \mathcal{X} \to \mathbb{R}$ be a real-valued function defined over $x \in \mathcal{X}$.

**Definition IV.2.** A real-valued function $f(x)$ defined over the set $\mathcal{X}$ is a convex function if the set $\mathcal{X}$ is convex and if for all $x$ and $y$ in $\mathcal{X}$ and all $\theta \in [0,1]$ we have:

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$

The function is said to be strictly convex if it is convex and if the above inequality holds with strict inequality whenever $\theta \in (0,1)$ and $x \neq y$.

**Definition IV.3.** A real-valued function $f(x)$ defined over the set $\mathcal{X}$ is a concave function if the set $\mathcal{X}$ is convex and if for all $x$ and $y$ in $\mathcal{X}$ and all $\theta \in [0,1]$ we have:

$$f(\theta x + (1-\theta)y) \geq \theta f(x) + (1-\theta)f(y)$$

The function is said to be strictly concave if it is concave and if the above inequality holds with strict inequality whenever $\theta \in (0,1)$ and $x \neq y$.

It follows that a function $f(x)$ defined over a convex set $\mathcal{X}$ is a concave function if and only if $-f(x)$ is a convex function. Likewise, $f(x)$ is strictly concave if and only if $-f(x)$ is strictly convex. The set $\mathcal{X}$ must be convex for the definitions of a convex function and concave function to make sense. Otherwise, the expression $f(\theta x + (1-\theta)y)$ may not be defined. The following facts can be proven directly from the definition of convex:

- Let $c$ be a non-negative real number. If $f(x)$ is a convex function, then $cf(x)$ is also a convex function. Likewise, if $g(x)$ is a concave function, then $cg(x)$ is concave.
- The sum of two convex functions is convex, and the sum of two concave functions is concave.
- The sum of a convex function and a strictly convex function is strictly convex.

Suppose $f(x)$ is a convex function over $x \in \mathbb{R}$, and define $\bar{f}(x_1, \ldots, x_N) = f(x_1)$. It can be shown that $\bar{f}(x_1, \ldots, x_N)$ is a convex function over $(x_1, \ldots, x_N) \in \mathbb{R}^N$. However, $\bar{f}(x_1, \ldots, x_N)$ is not strictly convex over $\mathbb{R}^N$, regardless of whether or not $f(x_1)$ is strictly convex over $\mathbb{R}$ (see Exercise VIII-D.3). A function of the type $f(x) = b + c_1 x_1 + c_2 x_2 + \cdots + c_N x_N$, where $b, c_1, \ldots, c_N$ are given real numbers, is called an affine function. It can be shown that an affine function defined over a convex set $\mathcal{X} \subseteq \mathbb{R}^N$ is both a convex function and a concave function (but neither strictly convex nor strictly concave, see Exercise VIII-D.4). In particular, a constant function is both convex and concave.

**Lemma IV.2.** Suppose $f_1(x_1), \ldots, f_N(x_N)$ are convex functions from $\mathbb{R}$ to $\mathbb{R}$. Let $x = (x_1, \ldots, x_N)$ and define $f(x) = \sum_{i=1}^N f_i(x_i)$.

a) The function $f(x)$ is convex over $\mathbb{R}^N$.

b) The function $f(x)$ is strictly convex if and only if all functions $f_i(x_i)$ are strictly convex over $\mathbb{R}$.

**Proof.** To prove part (a), let $x = (x_1, \ldots, x_N)$ and $y = (y_1, \ldots, y_N)$ be vectors in $\mathbb{R}^N$ and let $\theta \in [0,1]$. We have:

$$f(\theta x + (1-\theta)y) \leq \sum_{i=1}^N f_i(\theta x_i + (1-\theta)y_i) \leq \sum_{i=1}^N [\theta f_i(x_i) + (1-\theta)f_i(y_i)] = \theta f(x) + (1-\theta)f(y)$$

where the inequality holds because each function $f_i(x_i)$ is convex. This proves part (a). The proof of part (b) is an exercise (see Exercise VIII-D.5). □

**Lemma IV.3.** (Convex inequality constraints) Let $x = (x_1, \ldots, x_N)$ and let $g_1(x), \ldots, g_K(x)$ be convex functions over a convex set $\mathcal{X} \subseteq \mathbb{R}^N$. Let $c_1, \ldots, c_K$ be a collection of real numbers. Define $\mathcal{A}$ as the set of all $x \in \mathcal{X}$ that satisfy all of the following constraints:

$$g_k(x) \leq c_k \text{ for all } k \in \{1, \ldots, K\}$$

Then the set $\mathcal{A}$ is convex.

**Proof.** Exercise (see Exercise VIII-D.6). □

**Lemma IV.4.** (Differentiable functions of one variable) Suppose $f(x)$ is a differentiable function over $x \in \mathbb{R}$.

a) If $f'(x)$ is nondecreasing, then $f(x)$ is convex.

b) If $f(x)$ is twice differentiable and satisfies $f''(x) \geq 0$ for all $x \in \mathbb{R}$, then $f(x)$ is convex.
c) If \( f(x) \) is twice differentiable and satisfies \( f''(x) > 0 \) for all \( x \in \mathbb{R} \), then \( f(x) \) is strictly convex.

*Proof.* To prove part (a), suppose \( f(x) \) is nondecreasing. Let \( x \) and \( y \) be real numbers such that \( x < y \). Let \( \theta \in (0, 1) \) and define \( m = \theta x + (1-\theta)y \). Suppose that \( f(m) > \theta f(x) + (1-\theta)f(y) \) (we reach a contradiction, see Fig. 6). By the mean value theorem, there is a point \( x_1 \in (x, m) \) such that \( f'(x_1) = \frac{f(m) - f(x)}{m - x} \). Likewise, there is a point \( x_2 \in (m, y) \) such that \( f'(x_2) = \frac{f(y) - f(m)}{y - m} \). Notice that \( x_1 < x_2 \). By geometry, it can be seen that the first slope is strictly greater than the second (see Fig. 6), and so \( f'(x_1) > f'(x_2) \), contradicting the fact that \( f(x) \) is nondecreasing. This proves part (a). Part (b) follows from (a) by noting that \( f''(x) \geq 0 \) implies that \( f(x) \) is nondecreasing. Part (c) is similar and is omitted for brevity. \( \square \)

![Fig. 6. An illustration of the two slopes \( f'(x_1) \) and \( f'(x_2) \) associated with the proof of Lemma IV.4](image)

It follows from Lemmas IV.2 and IV.4 that the following functions are convex over \( \mathbb{R}^3 \):

\[
\begin{align*}
f(x_1, x_2, x_3) &= e^{5x_1 + x_2^2} - x_3 \\
g(x_1, x_2, x_3) &= 17 + x_1^8 + (4.2)e^{x_1} - 6x_2 + 7x_3
\end{align*}
\]

Suppose \( x = (x_1, \ldots, x_N) \) and \( f(x) \) is a twice differentiable function over \( x \in \mathbb{R}^N \). It can be shown that if \( \nabla^2 f(x) \) is a positive semidefinite matrix for all \( x \in \mathbb{R}^N \), then \( f(x) \) is convex over \( \mathbb{R}^N \) (see Exercise VIII-D.20). If \( \nabla^2 f(x) \) is positive definite for all \( x \in \mathbb{R}^N \), then \( f(x) \) is strictly convex. The following lemma is useful for minimizing convex and differentiable functions of one variable over an interval.

**Lemma IV.5.** (Minimizing single-variable convex functions over an interval) Let \( f : \mathbb{R} \to \mathbb{R} \) be convex. Suppose \( y^* \in \mathbb{R} \) minimizes \( f(x) \) over \( x \in \mathbb{R} \). Let \( a \) and \( b \) be real numbers such that \( a < b \) and consider the problem:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad x \in [a, b]
\end{align*}
\]

The solution to this problem is \( x^* = [y^*]_a^b \), where \( [y^*]_a^b \) denotes the projection of \( y^* \) onto the interval \( [a, b] \):

\[
[y^*]_a^b = \begin{cases} 
  a & \text{if } y^* < a \\
  y^* & \text{if } a \leq y^* \leq b \\
  b & \text{if } b < y^*
\end{cases}
\]

*Proof.* See Exercise VIII-D.18 \( \square \)

A similar result holds when minimizing a convex single-variable function over an interval of the form \( [a, \infty) \) or \( (-\infty, b] \). For example, the minimum of the convex function \( f(x) = (x-5)^2 \) over \( x \in \mathbb{R} \) is equal to \( y^* = 5 \). Thus:

- The minimum of \( f(x) \) over the interval \( [0, 1] \) is \( x^* = [5]_0^1 = 1 \).
- The minimum of \( f(x) \) over the interval \( [3, 8] \) is \( x^* = [5]^3_8 = 5 \).
- The minimum of \( f(x) \) over the interval \( [6, 2, \infty) \) is \( x^* = [5]^6_\infty = 6.2 \).

Caveat 1: The result of Lemma IV.5 does not hold for multi-dimensional problems. For example, if \( (x^*, y^*) \) minimizes the convex function \( f(x, y) \) over \( (x, y) \in \mathbb{R}^2 \), then the projection of \( (x^*, y^*) \) onto the hypercube \( [0, 1]^2 \) does not necessarily minimize \( f(x, y) \) over the hypercube.

Caveat 2: Be careful to use Lemma IV.5 only for the region in which \( f(x) \) is defined. For example, consider minimizing the convex function \( f(x) = -\log(x) - 5x \) over \( x \in [1/2, 1] \). Differentiating gives \( f'(x) = -1/x - 5 \). Setting the expression \(-1/x - 5\) to zero gives \( x = -1/5 \). So, is the minimizer \( x^* = [-1/5]_{1/2} = 1/2 \)? No. Clearly the minimum of the decreasing function \( f(x) \) over the interval \([1/2, 1]\) occurs at \( x^* = 1 \). The reason the projection operation failed is that the derivative of \( f(x) \) is only defined for \( x > 0 \), it does not make sense to use this derivative information for \( x \leq 0 \).
D. Jensen’s inequality

Let $\mathcal{X}$ be a convex subset of $\mathbb{R}^N$ and let $f(x)$ be a convex function defined over $x \in \mathcal{X}$. By the definition of convexity, we know that for any two vectors $x_1$ and $x_2$ in $\mathcal{X}$ and any probabilities $\theta_1$ and $\theta_2$ that are non-negative and sum to 1, we have:

$$f(\theta_1 x_1 + \theta_2 x_2) \leq \theta_1 f(x_1) + \theta_2 f(x_2)$$

Now consider three vectors $x_1, x_2, x_3$ in $\mathcal{X}$, and three probabilities $\theta_1, \theta_2, \theta_3$ that are non-negative and sum to 1. At least one of these probabilities must be positive. Without loss of generality assume $\theta_3 > 0$. Now define $\tilde{\theta} = (\theta_2 + \theta_3)$ and note that $\tilde{\theta} > 0$. Define:

$$\tilde{x} = \frac{\theta_2}{\tilde{\theta}} x_2 + \frac{\theta_3}{\tilde{\theta}} x_3$$

Since $\mathcal{X}$ is convex, we know $\tilde{x} \in \mathcal{X}$. Then:

$$f(\theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3) = f\left(\theta_1 x_1 + \tilde{\theta} \left[\frac{\theta_2}{\tilde{\theta}} x_2 + \frac{\theta_3}{\tilde{\theta}} x_3\right]\right)$$

$$= f(\theta_1 x_1 + \tilde{\theta} \tilde{x})$$

$$\leq \theta_1 f(x_1) + \tilde{\theta} f(\tilde{x})$$

$$\leq \theta_1 f(x_1) + \tilde{\theta} \left[\frac{\theta_2}{\tilde{\theta}} f(x_2) + \frac{\theta_3}{\tilde{\theta}} f(x_3)\right]$$

$$= \theta_1 f(x_1) + \theta_2 f(x_2) + \theta_3 f(x_3)$$

where the first two inequalities hold because $f(x)$ is convex.

Similarly (by induction), it can be shown that if $x_1, \ldots, x_k$ is any finite sequence of points in $\mathcal{X}$ and if $\theta_1, \ldots, \theta_k$ are any non-negative values that sum to 1, we have:

$$f \left( \sum_{i=1}^{k} \theta_i x_i \right) \leq \sum_{i=1}^{k} \theta_i f(x_i)$$

The above inequality is called Jensen’s inequality. A special case of Jensen’s inequality holds for time averages: Let $\{x(t)\}_{t=0}^{\infty}$ be an infinite sequence of vectors in $\mathcal{X}$. For slots $t \in \{1, 2, 3, \ldots\}$, define the time average:

$$\bar{x}(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} x(\tau)$$

Then:

$$f(\bar{x}(t)) \leq \frac{1}{t} \sum_{\tau=0}^{t-1} f(x(\tau))$$

The inequality \cite{57} is used in the development of the drift-plus-penalty algorithm for convex programs.

A more general form of Jensen’s inequality is as follows: Let $\mathcal{X}$ be a convex set and let $f(x)$ be a convex function over $\mathcal{X}$. Let $X$ be a random vector that takes values in the set $\mathcal{X}$ and that has finite mean $\mathbb{E}[X]$. Then $\mathbb{E}[X] \in \mathcal{X}$ and $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$.

In the special case when the random vector takes a finite number of possibilities $x_1, \ldots, x_k$ with probabilities $\theta_1, \ldots, \theta_k$, then the equation $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$ reduces to \cite{56}. However, the equation $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$ holds more generally in cases when $X$ can take a countably or uncountably infinite number of values (see Exercise VIII-D.14).

E. Convex hulls

Let $\mathcal{X}$ be a subset of $\mathbb{R}^N$. The convex hull of $\mathcal{X}$, written $\text{Conv}(\mathcal{X})$, is the set of all convex combinations of points in $\mathcal{X}$ (including all points in $\mathcal{X}$ themselves). Thus, if $x \in \text{Conv}(\mathcal{X})$, then $x = \sum_{i=1}^{k} \theta_i x_i$ for some positive integer $k$, some non-negative values $\theta_1, \ldots, \theta_k$ that sum to 1, and for some vectors $x_1, \ldots, x_k$ that satisfy $x_i \in \mathcal{X}$ for all $i \in \{1, \ldots, k\}$.

It can be shown that $\text{Conv}(\mathcal{X})$ is always a convex set. In general, it holds that $\mathcal{X} \subseteq \text{Conv}(\mathcal{X})$. If the set $\mathcal{X}$ itself is convex, then $\mathcal{X} = \text{Conv}(\mathcal{X})$. If two sets $\mathcal{X}$ and $\mathcal{Y}$ satisfy $\mathcal{X} \subseteq \mathcal{Y}$, then $\text{Conv}(\mathcal{X}) \subseteq \text{Conv}(\mathcal{Y})$. It can be shown that if $\mathcal{X}$ is a compact set, then $\text{Conv}(\mathcal{X})$ is also a compact set.

F. Hyperplane separation

A theory of hyperplane separation for convex sets, which shows when solutions of constrained optimization problems are also solutions of unconstrained problems with Lagrange multiplier weights, is given in Appendix B.

It can be shown that $\text{Conv}(\mathcal{X})$ is the “smallest” convex set that contains $\mathcal{X}$, in the sense that if $A$ is a convex set that contains $\mathcal{X}$, then $\text{Conv}(\mathcal{X}) \subseteq A$. This property is sometimes used as an equivalent definition of $\text{Conv}(\mathcal{X})$. Since the intersection of an arbitrary number of convex sets is convex, the intersection of all convex sets that contain $\mathcal{X}$ must be the “smallest” convex set that contains $\mathcal{X}$, and so this intersection is $\text{Conv}(\mathcal{X})$. 

V. CONVEX PROGRAMS

Let \( N \) be a positive integer. A convex program is an optimization problem that seeks to find a vector \( x = (x_1, \ldots, x_N) \in \mathbb{R}^N \) that minimizes a convex function \( f(x) \) subject to a collection of convex constraints. Specifically, it is a problem of the form:

\[
\text{Minimize:} \quad f(x) \tag{58}
\]
\[
\text{Subject to:} \quad g_k(x) \leq c_k \quad \forall k \in \{1, \ldots, K\} \tag{59}
\]
\[
x \in \mathcal{X} \tag{60}
\]

where \( c_1, \ldots, c_K \) are given real numbers, \( \mathcal{X} \) is a convex subset of \( \mathbb{R}^N \), and \( f(x), g_1(x), \ldots, g_K(x) \) are continuous and convex functions from \( \mathcal{X} \) to \( \mathbb{R} \). It can be shown that a convex function defined over a convex set \( \mathcal{X} \) is continuous at every interior point of \( \mathcal{X} \). Thus, the assumption that the functions \( f(x), g_1(x), \ldots, g_K(x) \) are both convex and continuous ensures that continuity holds at all points of \( \mathcal{X} \), including points on the boundary.\(^9\) The convexity assumptions allow convex programs to be solved more easily than general constrained optimization problems. The convex program is called a linear program in the special case when \( \mathcal{X} = \mathbb{R}^N \) and the functions \( f(x), g_1(x), \ldots, g_K(x) \) are affine.

The problem is feasible if there exists an \( x \in \mathbb{R}^N \) that satisfies all of the constraints (59)-(60). Lemma IV.3 ensures that the set of all vectors \( x \in \mathbb{R}^N \) that satisfy the constraints (59)-(60) is a convex set. Specifically, a constraint of the form \( g(x) \leq c \) is called a convex constraint whenever \( c \in \mathbb{R} \) and \( g(x) \) is a convex function over the set \( \mathcal{X} \) of interest. That is because the set of all \( x \in \mathcal{X} \) that satisfy this constraint forms a convex set. Since the intersection of convex sets is convex, imposing more and more convex constraints can shrink the feasible set but maintains its convexity. It can be shown that if the problem is feasible and if \( \mathcal{X} \) is a compact set, then there always exists an optimal solution \( x^* \).

Without loss of generality, one can assume all constants \( c_k \) in the above convex program are zero. This is because a constraint of the form \( g_k(x) \leq c_k \) is equivalent to \( \tilde{g}_k(x) \leq 0 \), where \( \tilde{g}_k(x) \) is defined by \( \tilde{g}_k(x) = g_k(x) - c_k \). Note that \( \tilde{g}_k(x) \) is convex if and only if \( g_k(x) \) is convex.

A. Equivalent forms and a network flow example

The structure (58)-(60) is the standard form for a convex program. Standard form is useful for proving results about general convex programs, and for developing and implementing algorithms that produce exact or approximate solutions. However, standard form is not always the most natural way of writing a convex program.

For example, consider a network that supports communication of a collection of \( N \) different traffic streams. Each traffic stream flows over its own path of links. The paths can overlap, so that some links support multiple traffic streams. Let \( L \) be the number of links, and let \( C_l \) be the capacity of each link \( l \in \{1, \ldots, L\} \). Let \( x = (x_1, \ldots, x_N) \) be a vector of flow rates for each stream. Given the link capacities, the problem is to find a vector of flow rates the network can support that maximizes a concave utility function \( \phi(x) = \sum_{i=1}^N \log(1 + x_i) \), which represents a measure of network fairness. This network utility maximization problem is easily described as follows:

\[
\text{Maximize:} \quad \sum_{i=1}^N \log(1 + x_i) \tag{61}
\]
\[
\text{Subject to:} \quad \sum_{i \in \mathcal{N}(l)} x_i \leq C_l \quad \forall l \in \{1, \ldots, L\} \tag{62}
\]
\[
x_i \geq 0 \quad \forall i \in \{1, \ldots, N\} \tag{63}
\]

where \( \mathcal{N}(l) \) is the set of streams in the set \( \{1, \ldots, N\} \) that use link \( l \) (defined for each link \( l \in \mathcal{L} \)).

While the above optimization problem is not in standard form, it is correct to call it a convex optimization problem. This is because the problem can easily be put in standard form by changing the maximization to a minimization, bringing all non-constant terms of the inequality constraints to the left-hand-side, and/or by defining a convex set \( \mathcal{X} \) consisting of the intersection of one or more of the constraints:

\[
\text{Minimize:} \quad -\sum_{i=1}^N \log(1 + x_i)
\]
\[
\text{Subject to:} \quad \sum_{i \in \mathcal{N}(l)} x_i \leq C_l \quad \forall l \in \{1, \ldots, L\}
\]
\[
x \in \mathcal{X}
\]

where \( \mathcal{X} \) is defined as the set of all vectors \( x = (x_1, \ldots, x_N) \) that satisfy the constraints (63). To formally see that the above is now in standard form, note that \( \mathcal{X} \) is a convex set. Further, we can define functions \( f(x) \) and \( g_l(x) \) by:

\[
f(x) = -\sum_{i=1}^N \log(1 + x_i)
\]
\[
g_l(x) = \sum_{i \in \mathcal{N}(l)} x_i \quad \forall l \in \{1, \ldots, L\}
\]

\(^9\)All convex functions \( f(x) \) defined over \( \mathbb{R}^N \) are continuous because the set \( \mathbb{R}^N \) has no boundary points. An example function \( f(x) \) defined over \([0, 1]\) that is convex but not continuous is \( f(x) = 0 \) for \( x \in [0, 1) \) and \( f(1) = 1 \). Of course, the point of discontinuity occurs on the boundary of \([0, 1]\).
and note that these are convex and continuous functions over $x \in X$.

The following structures are not in standard form, but are accepted ways of writing convex programs. That is because they are often more natural to write than the corresponding standard form, and they can easily be put into standard form by trivial rearrangements.

- Maximizing a concave function $\phi(x)$. This is equivalent to minimizing the convex function $-\phi(x)$.
- Enforcing a constraint $g(x) \leq r(x)$ (where $g(x)$ is convex and $r(x)$ is concave). This is equivalent to the convex constraint $g(x) - r(x) \leq 0$.
- Enforcing a constraint $g(x) \geq r(x)$ (where $g(x)$ is concave and $r(x)$ is convex). This is equivalent to the convex constraint $r(x) - g(x) \leq 0$.
- Enforcing a linear equality constraint $\sum_{i=1}^{N} a_i x_i = c$. This is equivalent to the following two linear (and hence convex) inequality constraints:

$$
\sum_{i=1}^{N} a_i x_i - c \leq 0
$$

$$
c - \sum_{i=1}^{N} a_i x_i \leq 0
$$

- Enforcing an interval constraint $x_i \in [a, b]$. This clearly imposes a convex constraint, and is equivalent to the following two linear inequality constraints:

$$
-x + a \leq 0
$$

$$
x - b \leq 0
$$

---

**Example V.1.** Consider the network shown in Fig. 7. There are three traffic flows. Let $r_1, r_2, r_3$ be the rate of each flow (in units of bits/second). Each link can support a maximum flow rate (in units of bits/second), called the link capacity. The capacities of each link are shown in the figure. The traffic flows use paths shown in the figure. Note that the second flow is split into two subflows with rates $x$ and $y$. The goal is to find a vector of flow rates $(r_1, r_2, r_3)$, together with rates $x$ and $y$, that can be supported over the network and that maximize the utility function $\phi(r_1, r_2, r_3) = \log(1+r_1) + \log(1+r_2) + \log(1+r_3)$.

Define $\mathcal{X}$ as the set of all $(r_1, r_2, r_3, x, y) \in \mathbb{R}^N$ such that $r_i \geq 0$ for all $i \in \{1, 2, 3\}$ and $x \geq 0$ and $y \geq 0$. The resulting convex program is:

Maximize: $\sum_{i=1}^{3} \log(1 + r_i)$

Subject to:

- $r_1 \leq 3$
- $r_1 + x \leq 4$
- $r_1 \leq 4$
- $x \leq 2$
- $x \leq 4$
- $y \leq 4$
- $y + x + r_3 \leq 4$
- $x + y = r_2$
- $(r_1, r_2, r_3, x, y) \in \mathcal{X}$
The above problem includes some redundant constraints. For example, the constraint \( r_1 \leq 4 \) is implied by the constraint \( r_1 \leq 3 \). Similarly, the constraint \( r_1 \leq 4 \) is implied by the constraints \( r_1 + x \leq 4 \) and \( x \geq 0 \). Removing this redundancy gives:

\[
\begin{align*}
\text{Maximize:} & \quad \sum_{i=1}^{3} \log(1 + r_i) \\
\text{Subject to:} & \quad r_1 \leq 3 \\
& \quad r_1 + x \leq 4 \\
& \quad x \leq 2 \\
& \quad y + x + r_3 \leq 4 \\
& \quad x + y = r_2 \\
& \quad (r_1, r_2, r_3, x, y) \in \mathcal{X} 
\end{align*}
\]

There are other ways of representing this same problem. For example, one can define a set \( \tilde{X} \) as the union of \( \mathcal{X} \) with the set of all \((r_1, r_2, r_3, x, y)\) such that \( x + y = r_2 \). Then the last two constraints above can be replaced by \((r_1, r_2, r_3, x, y) \in \tilde{X}\).

**Example V.2.** (Unhelpful representations) Some ways of representing a problem are correct but unhelpful. For example, the problem (64)-(70) can be equivalently written by introducing utility variables \( u_i = \log(1 + r_i) \):

\[
\begin{align*}
\text{Maximize:} & \quad u_1 + u_2 + u_3 \\
\text{Subject to:} & \quad r_1 \leq 3 \\
& \quad r_1 + x \leq 4 \\
& \quad x \leq 2 \\
& \quad y + x + r_3 \leq 4 \\
& \quad x + y = r_2 \\
& \quad (r_1, r_2, r_3, x, y) \in \mathcal{X} \\
& \quad u_i = \log(1 + r_i) \quad \forall i \in \{1, 2, 3\} \\
& \quad u_i \geq 0 \quad \forall i \in \{1, 2, 3\}
\end{align*}
\]

However, the above representation is not a convex program because the constraints (71) are not convex (recall that equality constraints are convex only when both sides are affine functions of the optimization variables). One can fix the problem, while still keeping the \( u_i \) variables, by changing the non-convex constraints (71) to the convex constraints \( u_i \leq \log(1 + r_i) \) for all \( i \in \{1, 2, 3\} \). This does not change the set of solutions because an optimal solution must meet the inequality constraints \( u_i \leq \log(1 + r_i) \) with equality. Indeed, any candidate solution that satisfies all constraints but has \( u_i < \log(1 + r_i) \) for some \( i \in \{1, 2, 3\} \) can be strictly improved, without violating the constraints, by increasing \( u_i \) to \( \log(1 + r_i) \).

**B. Linear programs**

When \( \mathcal{X} = \mathbb{R}^N \) and the functions \( f(x), g_1(x), \ldots, g_K(x) \) are affine, the convex program (58)-(60) has the form:

\[
\begin{align*}
\text{Minimize:} & \quad c_0 + \sum_{i=1}^{N} c_i x_i \\
\text{Subject to:} & \quad \sum_{i=1}^{N} a_{ik} x_i \leq b_k \quad \forall k \in \{1, \ldots, K\} \\
& \quad (x_1, \ldots, x_N) \in \mathbb{R}^N
\end{align*}
\]

for some given real numbers \( c_0, c_1, \ldots, c_N, a_{ik} \) for \( i \in \{1, \ldots, N\} \) and \( k \in \{1, \ldots, K\} \), and \( b_k \) for \( k \in \{1, \ldots, K\} \). Of course, \( c_0 \) does nothing but shift up the value of the objective function by a constant, and so the solution to the above problem is the same as the solution to a modified problem where \( c_0 \) is removed:

\[
\begin{align*}
\text{Minimize:} & \quad \sum_{i=1}^{N} c_i x_i \\
\text{Subject to:} & \quad \sum_{i=1}^{N} a_{ik} x_i \leq b_k \quad \forall k \in \{1, \ldots, K\} \\
& \quad (x_1, \ldots, x_N) \in \mathbb{R}^N
\end{align*}
\]

It is often easier to represent the above problem in matrix form: Let \( c = (c_1, \ldots, c_N) \) be a column vector, let \( b = (b_1, \ldots, b_K) \) be a column vector, and let \( A = (a_{ik}) \) be a \( K \times N \) matrix. Let the variables be represented by a column vector \( x = (x_1, \ldots, x_N) \). The problem is then:

\[
\begin{align*}
\text{Minimize:} & \quad c^T x \\
\text{Subject to:} & \quad Ax \leq b \\
& \quad x \in \mathbb{R}^N
\end{align*}
\]

where \( c^T x \) is the inner product of \( c \) and \( x \), and the inequality \( Ax \leq b \) is taken row-by-row.
VI. THE DRIFT-PLUS-PENALTY ALGORITHM FOR CONVEX PROGRAMS

A. Convex programs over compact sets

Let $x = (x_1, \ldots, x_N)$ represent a vector in $\mathbb{R}^N$. Consider the following convex program:

\begin{align*}
\text{Minimize:} & \quad f(x) & (72) \\
\text{Subject to:} & \quad g_k(x) \leq c_k \forall k \in \{1, \ldots, K\} & (73) \\
\end{align*}

where $\mathcal{X}$ is a convex and compact subset of $\mathbb{R}^N$. Recall that a subset of $\mathbb{R}^N$ is said to be compact if it is closed and bounded.

- Functions $f(x), g_1(x), \ldots, g_K(x)$ are continuous and convex functions over $x \in \mathcal{X}$.
- $c_k$ values are given real numbers (possibly 0).

The only significant difference between the above problem and a general convex program is that the set $\mathcal{X}$ here is assumed to be both convex and compact, rather than just convex. The compactness assumption ensures that an optimal solution exists whenever the problem is feasible (that is, whenever it is possible to satisfy all constraints (73)-(74)). Compactness is also useful because it restricts the search for an optimal solution to a bounded region of $\mathbb{R}^N$.

Assume the constraints are feasible. Define $x^*$ as an optimal solution to the above problem. Define $f^* = f(x^*)$ as the optimal objective function value. A vector $\tilde{x} \in \mathcal{X}$ is called an $\epsilon$-approximation of the solution if:

\begin{align*}
\epsilon & \equiv f(\tilde{x}) - f^* \leq \epsilon \\
\epsilon & \equiv g_k(\tilde{x}) - c_k \leq \epsilon \forall k \in \{1, \ldots, K\}
\end{align*}

We say that $\tilde{x}$ is an $O(\epsilon)$-approximation if the “$\epsilon$” values in the above inequalities are replaced by a constant multiple of $\epsilon$.

The following subsections develop an algorithm that produces a feasible solution to the convex program, for any desired value $\epsilon > 0$. The convergence time of the algorithm is $O(1/\epsilon^3)$ in general cases, and is $O(1/\epsilon^2)$ under the mild assumption that a Lagrange multiplier vector exists (see Section VI-G). A modified algorithm with a delayed start mechanism has $O(1/\epsilon)$ convergence time under additional assumptions [6].

B. Virtual queues

The drift-plus-penalty algorithm is a method for choosing values $x(t) = (x_1(t), \ldots, x_N(t)) \in \mathcal{X}$ over a sequence of time slots $t \in \{0, 1, 2, \ldots\}$ so that the time average $\tau(t) = (\tau_1(t), \ldots, \tau_N(t))$ converges to a close approximation of the solution to a particular optimization problem. The algorithm is developed in [3] for more general stochastic problems, and has close connections to optimization for queueing networks. These notes consider the drift-plus-penalty algorithm in the special case of the (non-stochastic) convex program (72)-(74).

Recall that the time average $\tau(t)$ is defined for all $t \in \{1, 2, 3, \ldots\}$ by:

$$
\tau(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} x(\tau)
$$

For each constraint $k \in \{1, \ldots, K\}$, define a virtual queue $Q_k(t)$ with update equation:

$$
Q_k(t + 1) = \max[Q_k(t) + g_k(x(t)) - c_k, 0]
$$

with initial condition $Q_k(0) = 0$ for all $k \in \{1, \ldots, K\}$. The value $g_k(x(t))$ acts as a virtual arrival to the queue, and the value $c_k$ acts as a virtual service rate (per slot). In a physical queueing system the arrivals and service rates are always non-negative. However, in this virtual queue, these values $g_k(x(t))$ and $c_k$ might be negative. If the queue $Q_k(t)$ is “stable,” then that the long term departure rate is equal to the long term arrival rate, then the time average of the arrival process $g_k(x(t))$ must be less than or equal to the service rate $c_k$. By Jensen’s inequality, this implies that the limiting value of $g_k(\tau(t))$ is also less than or equal to $c_k$. Thus, stabilizing the virtual queue ensures that the desired inequality constraint is satisfied. This observation is formalized in the following lemma.

Lemma VI.1. (Virtual queues) Under the queue update (75) we have for every slot $t > 0$ and for all $k \in \{1, \ldots, K\}$:

$$
g_k(\tau(t)) \leq c_k + \frac{Q_k(t)}{t}
$$

Proof. Fix $k \in \{1, \ldots, K\}$. The equation (75) for a given slot $\tau$ implies:

$$
Q_k(\tau + 1) \geq Q_k(\tau) + g_k(x(\tau)) - c_k
$$

There is no loss of generality in assuming that the $c_k$ values are all zero, since the $g_k(x)$ functions can simply be modified to $\tilde{g}_k(x) = g_k(x) - c_k$. 
Rearranging terms gives:
\[ Q_k(t + 1) - Q_k(t) \geq g_k(x(t)) - c_k \]
Summing the above inequality over \( \tau \in \{0, 1, \ldots, t - 1\} \) (for some slot \( t > 0 \)) gives:
\[ Q_k(t) - Q_k(0) \geq \sum_{\tau=0}^{t-1} g_k(x(\tau)) - c_k t \]
Dividing by \( t \) and using the fact that \( Q_k(0) = 0 \) gives:
\[ \frac{Q_k(t)}{t} \geq \frac{1}{t} \sum_{\tau=0}^{t-1} g_k(x(\tau)) - c_k \]
Using Jensen’s inequality \(^{57}\) gives:
\[ \frac{Q_k(t)}{t} \geq g_k(\mathcal{X}(t)) - c_k \]

The value \( Q_k(t)/t \) can be viewed as a bound on the constraint violation of constraint \( k \) up to slot \( t \). The above lemma implies that if we control the system to ensure that \( \lim_{t \to \infty} Q_k(t)/t = 0 \) for all \( k \in \{1, \ldots, K\} \), then all desired constraints are asymptotically satisfied\(^{11}\):
\[ \lim_{t \to \infty} g_k(\mathcal{X}(t)) \leq c_k \]
A queue that satisfies \( Q_k(t)/t \to 0 \) is called rate stable. Thus, the goal is to make all queues rate stable. More importantly, if we can control the system to maintain a finite worst-case queue size \( Q_{max} \), then the constraint violations decay like \( Q_{max}/t \) as time progresses.

### C. Lyapunov optimization

Define \( L(t) = \frac{1}{2} \sum_{k=1}^{K} Q_k(t)^2 \) as the sum of squares of all virtual queues (divided by 2 for convenience). This is called a Lyapunov function. The value \( L(t) \) is a scalar measure of the current queue backlogs. To ensure rate stability of all queues, it is desirable to make decisions that push \( L(t) \) down as much as possible from one slot to the next. Define \( \Delta(t) = L(t+1) - L(t) \) as the Lyapunov drift, being the difference in \( L(t) \) over one slot. The drift-plus-penalty algorithm chooses \( x(t) \in \mathcal{X} \) every slot \( t \) to minimize a bound on the following drift-plus-penalty expression \(^{58}\):
\[ \Delta(t) + V f(x(t)) \]
where \( V \) is a non-negative parameter that affects the amount to which we consider minimization of the penalty term \( f(x(t)) \). Intuitively, we want to make \( \Delta(t) \) small to ensure low queue backlogs. On the other hand, we also want to make \( f(x(t)) \) small to ensure a small value of the objective function. These two goals are managed by minimizing the weighted sum \( \Delta(t) + V f(x(t)) \). It will be shown that the parameter \( V \) affects a performance tradeoff between distance to the optimal objective function value and the convergence time required to satisfy the desired constraints.

The first step is to compute a bound on \( \Delta(t) \). We have for each queue \( k \in \{1, \ldots, K\} \):
\[ Q_k(t + 1)^2 = \max[Q_k(t) + g_k(x(t)) - c_k, 0]^2 \]
\[ \leq (Q_k(t) + g_k(x(t)) - c_k)^2 \]
\[ = Q_k(t)^2 + (g_k(x(t)) - c_k)^2 + 2Q_k(t)(g_k(x(t)) - c_k) \]
Therefore:
\[ \frac{1}{2} [Q_k(t + 1)^2 - Q_k(t)^2] \leq \frac{1}{2} (g_k(x(t)) - c_k)^2 + Q_k(t)(g_k(x(t)) - c_k) \]
Summing the above over \( k \in \{1, \ldots, K\} \) gives:
\[ \Delta(t) \leq \frac{1}{2} \sum_{k=1}^{K} (g_k(x(t)) - c_k)^2 + \sum_{k=1}^{K} Q_k(t)(g_k(x(t)) - c_k) \]
Define \( B \) as an upper bound on the worst-case value of \( \frac{1}{2} \sum_{k=1}^{K} (g_k(x(t)) - c_k)^2 \). This value \( B \) is finite because the set \( \mathcal{X} \) is assumed to be compact and the functions \( g_k(x) \) are continuous. Then:
\[ \Delta(t) \leq B + \sum_{k=1}^{K} Q_k(t)(g_k(x(t)) - c_k) \]

\(^{11}\) More formally, the result is that \( \lim \sup_{t \to \infty} g_k(\mathcal{X}(t)) \leq c_k \).
Adding \( V f(x(t)) \) to both sides gives the following important drift-plus-penalty inequality:

\[
\Delta(t) + V f(x(t)) \leq B + V f(x(t)) + \sum_{k=1}^{K} Q_k(t)[g_k(x(t)) - c_k]
\]  

(76)

The drift-plus-penalty algorithm is designed to operate as follows: Every slot \( t \), all queues \( Q_k(t) \) are observed. Then, the controller makes a greedy decision by selecting \( x(t) \in \mathcal{X} \) to minimize the right-hand-side of (76).

**Drift-plus-penalty algorithm (DPP):** Every slot \( t \in \{0, 1, 2, \ldots \} \), observe \( Q_1(t), \ldots, Q_K(t) \) and perform the following:

- Choose \( x(t) \in \mathcal{X} \) to minimize:
  \[
  V f(x(t)) + \sum_{k=1}^{K} Q_k(t) g_k(x(t))
  \]
  (77)

- Update the virtual queues \( Q_k(t) \) via (75), that is, for all \( k \in \{1, \ldots, K\} \):
  
  \[
  Q_k(t+1) = \max[Q_k(t) + g_k(x(t)) - c_k, 0]
  \]
  (78)

- Update the time average vector \( \pi(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} x(\tau) \) by:
  \[
  \pi(t+1) = \frac{1}{t+1} \sum_{\tau=0}^{t} x(\tau) = \pi(t) \left( \frac{t}{t+1} \right) + x(t) \left( \frac{1}{t+1} \right)
  \]

It is important to emphasize that, on slot \( t \), the values \( Q_1(t), \ldots, Q_K(t) \) are treated as known constants that act as weights in the expression (77). Given these weights for slot \( t \), the expression is minimized by searching over all \( x(t) \in \mathcal{X} \). The minimizer \( x(t) \) is used in the queue update equation to compute the new weights \( Q_1(t+1), \ldots, Q_K(t+1) \) for the next slot.

The relationship between the above algorithm and the dual subgradient (DPP) algorithm (50) is as follows. Assume \( V > 0 \) and define scaled virtual queues \( \tilde{Q}_k(t) = Q_k(t)/V \). The DPP algorithm can be written in terms of \( \tilde{Q}_k(t) \) by dividing (77)-(78) by \( V \):

- Choose \( x(t) \in \mathcal{X} \) to minimize:
  \[
  f(x(t)) + \sum_{k=1}^{K} \tilde{Q}_k(t) g_k(x(t))
  \]
  (79)

- Update \( \tilde{Q}_k(t) \) for each \( k \in \{1, \ldots, K\} \) by:
  
  \[
  \tilde{Q}_k(t+1) = \max[\tilde{Q}_k(t) + (1/V)(g_k(x(t)) - c_k), 0]
  \]
  (80)

With the above notation, the variables \( \tilde{Q}_1(t), \ldots, \tilde{Q}_K(t) \) can be viewed as Lagrange multiplier estimates \( (\mu_1, \ldots, \mu_K) \) at each step \( t \), and the update equation (79) is the same as the dual subgradient update (50) with a step size \( \delta = 1/V \). The main difference is that the dual subgradient algorithm does not traditionally take a time average and requires certain strict convexity properties to converge. The time averaging operation in the DPP algorithm allows optimization of time averages for more general problems, including general convex programs as well as nonconvex and stochastic problems [3].

**D. Example convex program**

Consider the following example convex program, stated in terms of optimization variables \( x \) and \( y \):

\[
\text{Minimize:} \quad e^x + y^2
\]

\[
\text{Subject to:} \quad x + y \geq 4
\]

\[
\quad x + 3y \geq 6
\]

\[
\quad x \in [0, 5], \quad y \in [0, 5]
\]

(80) (81) (82) (83)

While this problem can easily be solved by hand, it is instructive to show the steps of the drift-plus-penalty algorithm. The problem is equivalent to the following problem that inverts the inequality constraints (81)-(81):

\[
\text{Minimize:} \quad e^x + y^2
\]

\[
\text{Subject to:} \quad -x - y \leq -4
\]

\[
\quad -x - 3y \leq -6
\]

\[
\quad x \in [0, 5], \quad y \in [0, 5]
\]

(84) (85) (86) (87)

Let \( \mathcal{X} \) be the set of all \( (x, y) \) that satisfy constraints (87) (this set is a square and is indeed a convex and compact set). There are two additional constraints (85)-(86), each will receive its own virtual queue. Notice that all non-constant terms of
the constraints (85)-(86) have been shifted to the left-hand-side, as required. The drift-plus-penalty algorithm reduces to the following:

**Virtual queues:** Define $Q_1(0) = Q_2(0) = 0$. Variables $x(t), y(t)$ are used in the following queue update equations every slot $t \in \{0, 1, 2, \ldots\}$:

\[
Q_1(t+1) = \max[Q_1(t) - x(t) - y(t) + 4, 0] \quad (88)
\]
\[
Q_2(t+1) = \max[Q_2(t) - x(t) - 3y(t) + 6, 0] \quad (89)
\]

**Variable selection:** Every slot $t \in \{0, 1, 2, \ldots\}$, observe $Q_1(t), Q_2(t)$ and choose $(x(t), y(t)) \in \mathcal{X}$ to minimize:

\[
V(e^{x(t)} + y(t))^2 + Q_1(t)(-x(t) - y(t)) + Q_2(t)(-x(t) - 3y(t))
\]

This is a correct answer, but not complete. That is because a complete answer should exploit separable optimization in the variable selection whenever possible. By rearranging terms, the variable selection decision corresponds to the following:

Minimize: \((V e^{x(t)} - (Q_1(t) + Q_2(t))x(t)) + (V y(t)^2 - (Q_1(t) + 3Q_2(t))y(t))\)

Subject to: \((x(t) \in [0, 5], y(t) \in [0, 5])\)

It is apparent that $x(t)$ and $y(t)$ can be optimized separately:

- **$x(t)$ selection:** Choose $x(t) \in [0, 5]$ to minimize $V e^{x(t)} - (Q_1(t) + Q_2(t))x(t)$. Thus:

\[
x(t) = \left[ \log \left( \frac{Q_1(t) + Q_2(t)}{V} \right) \right]^5_0 \quad (90)
\]

where $[a]^5_0$ represents a projection of the real number $a$ onto the interval $[0, 5]$.

- **$y(t)$ selection:** Choose $y(t) \in [0, 5]$ to minimize $V y(t)^2 - (Q_1(t) + 3Q_2(t))y(t)$. Thus:

\[
y(t) = \left[ \frac{Q_1(t) + 3Q_2(t)}{2V} \right]^5_0 \quad (91)
\]

**Deterministic queue bounds:** Exercise VIII-F.5 shows that, in this example, there are constants $\beta_1, \beta_2, C_1, C_2$ such that $Q_1(t) \leq \beta_1 V + C_1$ and $Q_2(t) \leq \beta_2 V + C_2$ for all $t$. Substituting this deterministic queue bound into Lemma VI.1 implies that for all $t \in \{1, 2, 3, \ldots\}$:

\[
-\pi(t) - \bar{y}(t) \leq -4 + (\beta_1 V + C_1) / t
\]
\[
-\pi(t) - 3\bar{y}(t) \leq -6 + (\beta_2 V + C_2) / t
\]

Thus, the vector $(\pi(t), \bar{y}(t))$ is very close to satisfying the desired constraints when $t$ is large relative to $V$.

**Discussion:** Choosing the set $\mathcal{X}$ as the square corresponding to constraints (87) was convenient as it led to a simple separable optimization for $x(t)$ and $y(t)$. To produce the simplest algorithm for multi-dimensional problems, a rule of thumb is to set $\mathcal{X}$ as a multi-dimensional hyper-rectangle so that each variable is restricted to a particular interval. However, this is not necessary for the algorithm, as shown in the next subsection.

E. **Choosing a different set $\mathcal{X}$**

Consider the same problem (84)-(87). However, now define $\mathcal{X}$ as the set of all $(x, y)$ that satisfy the constraints (86)-(87). This is the intersection of the compact square $[0, 5] \times [0, 5]$ with the convex set defined by constraint $-x - 3y \leq -6$. The resulting set is still convex and compact, but is no longer a square. The only remaining constraint in the problem is (85) (being the constraint $-x - y \leq -4$). Thus, the drift-plus-penalty algorithm uses only one virtual queue:

**Virtual queue:** Define the virtual queue $Q(t)$ with update equation:

\[
Q(t+1) = \max[Q(t) - x(t) - y(t) + 4, 0]
\]

**Variable selection:** Every slot $t \in \{0, 1, 2, \ldots\}$, observe $Q(t)$ and choose $(x(t), y(t)) \in \mathcal{X}$ to minimize:

\[
V(e^{x(t)} + y(t))^2 + Q(t)(-x(t) - y(t))
\]

This reduces to the following problem every slot $t$:

Minimize: \((V e^{x(t)} - Q(t)x(t)) + (V y(t)^2 - Q(t)y(t))\)

Subject to: \((x(t) \in [0, 5], y(t) \in [0, 5], x(t) + 3y(t) \geq 6)\)

While the objective function in the above minimization is still a separable sum of terms involving $x(t)$ and $y(t)$, the constraint $x(t) + 3y(t) \geq 6$ couples the $(x(t), y(t))$ selection, so that these variables cannot be chosen separately. One can obtain a (non-separable) solution to the above problem by using a Lagrange multiplier on the constraint $x(t) + 3y(t) \geq 6$.

The performance theorem of the next subsection ensures that time averages $(\pi(t), \bar{y}(t))$ from both the algorithm in this subsection and the algorithm in the previous subsection approach the same optimal solution to problem (80)-(83) as $V \to \infty$. 
F. Performance theorem

Define $Q(t) = (Q_1(t), \ldots, Q_K(t))$, and define $||Q(t)|| = \sqrt{\sum_{i=1}^{K} Q_i(t)^2}$. The following theorem is a special case of results for more general stochastic problems in [3].

**Theorem VI.1.** If the convex program (72), (74) is feasible, then the drift-plus-penalty algorithm ensures:

(a) $f(\pi(t)) \leq f^* + B/V$ for all slots $t \in \{1, 2, 3, \ldots\}$.

(b) $\lim_{t \to \infty} \frac{Q_k(t)}{t} = 0$ for all $k \in \{1, \ldots, K\}$.

(c) $\limsup_{t \to \infty} g_k(\pi(t)) \leq c_k$ for all $k \in \{1, \ldots, K\}$ [2]

(d) $||Q(t)||/t \leq O(\sqrt{V/t})$.

**Proof.** (Theorem VI.1 part (a)) Since the drift-plus-penalty algorithm chooses $x(\tau) \in X$ to minimize the right-hand-side of (76) on each slot $\tau$, and the optimal solution $x^*$ is also a point in $X$, we have:

$$\Delta(\tau) + V f(x(\tau)) \leq B + V f(x^*) + \sum_{k=1}^{K} Q_k(\tau)(g_k(x^*) - c_k)$$

where the final inequality holds because $f(x^*) = f^*$ and $g_k(x^*) \leq c_k$ for all $k$. Summing the above over $\tau \in \{0, \ldots, t - 1\}$ gives:

$$L(t) - L(0) + V \sum_{\tau=0}^{t-1} f(x(\tau)) \leq Bt + V f^* t$$

Dividing by $Vt$ and using the fact that $L(0) = 0$ gives:

$$\frac{L(t)}{Vt} + \frac{1}{t} \sum_{\tau=0}^{t-1} f(x(\tau)) \leq \frac{B}{V} + f^*$$

(94)

Using $L(t) \geq 0$ and Jensen’s inequality (57) gives:

$$f(\pi(t)) \leq \frac{B}{V} + f^*$$

\[\square\]

**Proof.** (Theorem VI.1 parts (b)-(d)) From (94) we have for all $t \in \{1, 2, 3, \ldots\}$:

$$\frac{L(t)}{Vt} + f_{\min} \leq \frac{B}{V} + f^*$$

where $f_{\min} = \inf_{x \in X} f(x)$. The value $f_{\min}$ is finite because $X$ is compact and $f(x)$ is continuous. Rearranging terms gives:

$$L(t) \leq Bt + (f^* - f_{\min})Vt$$

Substituting the definition $L(t) = \frac{1}{2}||Q(t)||^2$ gives:

$$||Q(t)||^2 \leq 2[B + (f^* - f_{\min})V]t$$

Thus:

$$\frac{||Q(t)||}{t} \leq \sqrt{\frac{2[B + (f^* - f_{\min})V]}{t}}$$

This proves part (d). Part (d) immediately proves part (b). Part (b) with Lemma VI.1 proves part (c).

\[\square\]

The theorem shows that $V$ can be chosen as large as desired to ensure $f(\pi(t))$ is arbitrarily close to the optimal value $f^*$, with a corresponding tradeoff in the time required for the constraints to be close to being satisfied. Choosing $V = 1/\epsilon$ yields an $O(\epsilon)$-approximation. The time required for the constraints to be satisfied within $O(\epsilon)$ is the time $t$ such that $||Q(t)||/t \leq O(\epsilon)$.

---

12The “$\limsup$” can be replaced by “$\lim$” whenever the regular limit exists. The value $\limsup_{t \to \infty} h(t)$ exists (possibly being $\infty$ or $-\infty$) for any real-valued function $h(t)$, and is the largest limiting value over any subsequence of times $t_k$ that increase to infinity and for which the regular limit of $h(t_k)$ exists. For example, $\limsup_{t \to \infty} \cos(t)$ does not exist, but $\limsup_{t \to \infty} \cos(t_k) = 1$. Indeed, note that no subsequence of times $t_k$ can have a limiting value of $\cos(t_k)$ that is larger than 1, but one can define the particular subsequence $t_k = 2\pi k$ (for positive integers $k$) that satisfies $\lim_{t \to \infty} \cos(t_k) = 1$. If $h(t)$ is a real valued function and if $c$ is a real number, the statement “$\limsup_{t \to \infty} h(t) \leq c$” is equivalent to the statement that “for all $\epsilon > 0$, there is a time $T_{\epsilon}$ such that $h(t) \leq c + \epsilon$ whenever $t \geq T_{\epsilon}$.” It is also equivalent to the statement that “$\lim_{t \to \infty} \max[h(t), c] = c$.” If two functions $h_1(t)$ and $h_2(t)$ satisfy $h_1(t) \leq h_2(t)$ for all $t \geq 0$, it can be shown that $\limsup_{t \to \infty} h_1(t) \leq \limsup_{t \to \infty} h_2(t)$. The “$\limsup$” can be replaced by “$\lim$” whenever the regular limit exists. Note that a subsequence $t_k$ may converge to infinity, but the subsequence $t_k$ itself need not converge to infinity. If $t_k$ is a subsequence that converges to infinity, then $\limsup_{k \to \infty} (t_k) = \infty$. If $t_k$ is a subsequence that does not converge to infinity, then $\limsup_{k \to \infty} (t_k)$ may or may not exist.
From part (d) of the above theorem, this is ensured if $\sqrt{V/t} \leq \epsilon$. If $V = 1/\epsilon$, it follows that we need time $t \geq 1/\epsilon^3$. However, the bound in part (d) is overly conservative. An improved convergence time bound of $O(1/\epsilon^2)$ is given in the next subsection under an assumption that a Lagrange multiplier vector exists.

The above drift-plus-penalty theorem is developed in a more general stochastic context in [3] (see also [2]). Applications to distributed convex programming are in [8], where an improved $O(1/\epsilon^2)$ convergence time is shown under a Slater condition. A similar $O(1/\epsilon^2)$ convergence time is shown in [9]. In the (deterministic) convex program context, the drift-plus-penalty algorithm is similar to the dual subgradient algorithm [1][2]. However, the dual subgradient algorithm traditionally does not take time averages and requires more stringent strict convexity assumptions for its mathematical analysis.

G. Improved convergence time analysis

Theorem VI.1 only requires the convex program (72)-(74) to be feasible. The following theorem provides a tighter $O(1/\epsilon^2)$ convergence time analysis under the additional assumption that a Lagrange multiplier vector exists. Specifically, assume there exists a vector $\mu = (\mu_1, \ldots, \mu_K)$ such that $\mu_k \geq 0$ for all $k \in \{1, \ldots, K\}$, and:

$$f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \geq f(x^*) + \sum_{k=1}^{K} \mu_k c_k \ orall x \in \mathcal{X}$$ (95)

This Lagrange multiplier assumption is equivalent to the existence of a non-vertical hyperplane that is tangent to a certain convex set, as discussed in detail in Appendix B. The following theorem is from [4].

**Theorem VI.2.** (Improved convergence time [3]) Suppose the convex program (72)-(74) is feasible and that a Lagrange multiplier vector $\mu = (\mu_1, \ldots, \mu_K)$ exists (so that (95) holds). Then the drift-plus-penalty algorithm with $V \geq 0$ and initial conditions $Q_k(0) = 0$ for all $k \in \{1, \ldots, K\}$ ensures:

(a) $f(\bar{x}(t)) \leq f^* + \frac{B}{\sqrt{V}}$ for all slots $t \in \{1, 2, 3, \ldots\}$, where $B$ is the constant used in (76).

(b) $\|Q(t)\| \leq \frac{V\|\mu\| + \sqrt{V^2\|\mu\|^2 + 2B^2}}{\epsilon}$ for all $t \in \{1, 2, 3, \ldots\}$.

(c) Define $V = 1/\epsilon$. Then for any integer $t \geq 1/\epsilon^2$ we have:

$$f(\bar{x}(t)) \leq f^* + O(\epsilon)$$

$$g_k(\bar{x}(t)) \leq c_k + O(\epsilon) \ \forall k \in \{1, \ldots, K\}$$

$$\bar{x}(t) \in \mathcal{X}$$

Hence, the drift-plus-penalty algorithm produces an $O(\epsilon)$ approximation to the solution with a convergence time of $O(1/\epsilon^2)$.

**Proof.** Part (a) is already known from Theorem VI.1. Part (b) follows immediately from parts (a) and (b) together with Lemma VI.1. Part (c) is proven in Appendix C. \qed

Additional convergence time improvements, faster than $O(1/\epsilon^2)$, are shown in the following references:

- Work in [6] shows that under certain linear-type assumptions, the convergence time of DPP can be further pushed to $O(1/\epsilon)$ by using delayed time averages that start the averaging after some later time $T > 0$.
- Work in [10] shows that an algorithm equivalent to DPP has $O(1/\epsilon)$ convergence time under certain strong convexity assumptions.
- Work in [11] shows the DPP algorithm has $O(1/\epsilon)$ convergence time under strong convexity assumptions that are slightly less stringent than [10]. Work [11] also considers DPP with delayed time averages: Under strong convexity assumptions that are more stringent than [10], the algorithm is shown to have an improved convergence time of $O(1/\epsilon^2/3)$. Further improvements to $O(\log(1/\epsilon))$ are shown in [11] under even more stringent assumptions.
- Work in [12] develops a modification of DPP that achieves $O(1/\epsilon)$ convergence time for general convex programs (without requiring linearity or strong convexity).
- Alternative algorithms for $O(1/\epsilon)$ convergence time under various assumptions are considered in [13][14][15].

The algorithms in the above bulleted list are related to the basic dual subgradient algorithm. Such algorithms typically require a large number of iterations, but make simple decisions at each iteration. There are other classes of algorithms (such as interior point and Newton-based algorithms) that use a small number of iterations, but make more complex decisions at each iteration. The reader is referred to [2][1] and related literature for more details on these.

---

**A function $f(x)$ is strongly convex** if $f(x) - \alpha \|x\|^2$ is convex in $x$ for some constant $\alpha > 0$. A strongly convex program has a strongly convex objective function. This structure is restrictive because it requires all variables that appear in the constraints to also appear in the objective function. For example, this holds in network utility maximization problems with fixed path routing, but not with multi-path routing.
H. Choosing the set $\mathcal{X}$ for general problems

Consider the convex program (72)-(74), which seeks to minimize $f(x)$ subject to $g_k(x) \leq c_k$ for all $k \in \{1, \ldots, K\}$ and $x \in \mathcal{X}$ (where $\mathcal{X}$ is a convex set and $f(x)$, $g_k(x)$ are convex functions). Now define $\tilde{\mathcal{X}}$ as the intersection of $\mathcal{X}$ and the set of all $x \in \mathbb{R}^N$ that satisfy $g_K(x) \leq c_K$. Since the intersection of convex sets is convex, the set $\tilde{\mathcal{X}}$ is convex. The original problem (72)-(74) is equivalent to the following:

Minimize: $f(x)$
Subject to: $g_k(x) \leq c_k \quad \forall k \in \{1, \ldots, K-1\}$
$x \in \tilde{\mathcal{X}}$

The only difference is that the above convex program is written by shifting the last constraint $g_K(x) \leq c_K$ into the set constraint $x \in \tilde{\mathcal{X}}$. While this does not change the underlying problem, it gives rise to a different implementation of the drift-plus-penalty algorithm that uses one fewer virtual queue and that seeks to find $x(t) \in \tilde{\mathcal{X}}$ to minimize:

$$V f(x(t)) + \sum_{k=1}^{K-1} Q_k(t) g_k(x(t))$$

The advantage of using one less virtual queue is that the constant $B$ in Theorem [VI.1] is now smaller. Intuitively, the algorithm provides tighter performance at the cost of doing a search for $x(t)$ over the more complex set $\tilde{\mathcal{X}}$, rather than the larger (but typically simpler) set $\mathcal{X}$. Indeed, in the extreme case, one can place all constraints $g_k(x) \leq c_k$ into an abstract set $\mathcal{X}'$. That is, we can define $\mathcal{X}'$ as the intersection:

$$\mathcal{X}' = \mathcal{X} \cap \{x \in \mathbb{R}^N | g_k(x) \leq c_k \}$$

The resulting drift-plus-penalty algorithm then uses no virtual queues, and converges to the correct answer immediately on slot 0. Of course, this is because it is forced to choose $x(0) \in \mathcal{X}'$ as the minimizer of $f(x)$, which is identical to choosing $x(0)$ as the optimal solution to the original convex program!

One can write a single convex program in multiple ways. In practice, one should choose the abstract set $\mathcal{X}$, and the corresponding constraints $g_k(x) \leq c_k$ that are not incorporated into $\mathcal{X}$, in such a way that searching over $x \in \mathcal{X}$ to minimize $V f(x) + \sum_{k=1}^{K} Q_k(t) g_k(x)$ is easy. A rule of thumb is to choose $\mathcal{X}$ as a hyper-rectangle whenever possible, so that it has the form:

$$\mathcal{X} = \{x \in \mathbb{R}^N | x_{i,min} \leq x_i \leq x_{i,max} \quad \forall i \in \{1, \ldots, N\} \}$$

for some real numbers $x_{i,min}$ and $x_{i,max}$. This is especially useful when the functions $f(x)$ and $g_k(x)$ are separable sums of the individual $x_i$ variables. For example, suppose that:

$$f(x) = \sum_{i=1}^{N} f_i(x_i)$$
$$g_k(x) = \sum_{i=1}^{N} g_{ki}(x_i) \quad \forall k \in \{1, \ldots, K\}$$

for some convex functions $f_i(x_i)$ and $g_{ki}(x_i)$. In this case, the drift-plus-penalty algorithm of choosing $x(t) \in \mathcal{X}$ to minimize $V f(x(t)) + \sum_{k=1}^{K} Q_k(t) g_k(x(t))$ is reduced to separately choosing each variable $x_i(t)$, for each $i \in \{1, \ldots, N\}$, to solve:

Minimize: $V f_i(x_i(t)) + \sum_{k=1}^{K} Q_k(t) g_{ki}(x_i(t))$
Subject to: $x_i(t) \in [x_{i,min}, x_{i,max}]$

This is a minimization of a convex function of one variable over an interval. The minimum occurs at a critical point: either a boundary point $x_{i,min}$ or $x_{i,max}$, a point where the derivative does not exist, or a point of zero derivative. Often, the optimal choice of $x_i(t)$ can be solved in closed form. This is the case for linear programs, where the functions $f(x)$ and $g_k(x)$ are affine (see Exercise [VIII-F.1]). This separable property is particularly useful for distributed implementation in systems where different devices choose different $x_i(t)$ values. Separability can often be designed by creating local estimation variables to facilitate distributed implementation (see [8] and Exercises [VIII-F.15], [VIII-F.17]).

I. A small network example

Consider the following example exercise: A network with two flows must be optimized for maximum utility subject to power constraints (see Fig. [8]). The flow rates are $x_1, x_2$ and are chosen over intervals $x_1 \in [0, x_{max}]$ and $x_2 \in [0, x_{max}]$, where $x_{max}$ is some given positive number that bounds the maximum flow rate. The total power (summed over all links) is
\(e^{f_1} + e^{f_2} + e^{f_3} + e^{f_4} + e^{f_5} - 5\), where \(f_l\) is the total flow rate over link \(l\), defined for each \(l \in \{1, \ldots, 5\}\). We want to maximize the utility function \(\log(1 + x_1) + \log(1 + x_2)\) subject to a total power constraint of \(p_{tot}\) (where \(p_{tot}\) is a given positive constant):

\[
e^{f_1} + e^{f_2} + e^{f_3} + e^{f_4} + e^{f_5} - 5 \leq p_{tot}
\]

The convex optimization problem is to choose input rates \(x_1, x_2\) and flow rates \(a, b, c\) to solve:

**Maximize:** \(\log(1 + x_1) + \log(1 + x_2)\)  \hspace{1cm} (96)

**Subject to:**

\[
a + b + c \geq x_1
\]

(97)

\[
h(x_1, x_2, a, b, c) \leq p_{tot}
\]

(98)

\[
a, b, c, x_1, x_2 \in [0, x_{max}]
\]

(99)

a) Define the convex function \(h(x_1, x_2, a, b, c)\) that defines total power used in the network.

b) Let \(\mathcal{X}\) be the set described by constraint (99). State all virtual queues for this problem.

c) Using answers from parts (a)-(b), state the drift-plus-penalty algorithm for this problem. Make sure to exactly specify all choices of the decision variables, and to exploit separable structure whenever possible. Explicitly solve in closed form for any variables that are completely separable.

**Solution:**

a) We have \(h(x_1, x_2, a, b, c) = 2e^a + e^b + e^c + e^{c+x_2} - 5\). This is indeed a convex function of \((x_1, x_2, a, b, c)\).

b) Since \(\mathcal{X}\) is the set of all \((x_1, x_2, a, b, c)\) that satisfy (99), there are only two remaining constraints, namely, constraints (97)-(98). We could label their virtual queues as \(Q_1(t)\) and \(Q_2(t)\). Alternatively, we could simply label the virtual queues as \(Q(t)\) and \(Z(t)\). Using the latter notation gives:

\[
Q(t+1) = \max[Q(t) + x_1(t) - a(t) - b(t) - c(t), 0]
\]

\[
Z(t+1) = \max[Z(t) + 2e^a(t) + e^b(t) + e^c(t) + e^{c+x_2(t)} - 5 - p_{tot}, 0]
\]

Notice that for virtual queue \(Q(t)\), we remembered to modify the constraint (97) to a less-than-or-equal-to constraint, and to shift all non-constant terms to the left-hand-side (so the constraint is equivalent to \(x_1 - (a + b + c) \leq 0\)).

c) Every slot \(t\), observe the virtual queues and choose \((x_1(t), x_2(t), a(t), b(t), c(t)) \in \mathcal{X}\) to minimize:

\[
-V \log(1 + x_1(t)) - V \log(1 + x_2(t)) + Q(t)(x_1(t) - a(t) - b(t) - c(t)) + Z(t)(2e^a(t) + e^b(t) + e^c(t) + e^{c+x_2(t)} - 5 - p_{tot})
\]

This reduces to choosing:

- \(x_1(t) \in [0, x_{max}]\) to maximize \(V \log(1 + x_1(t)) - Q(t)x_1(t)\). Thus, \(x_1(t) = [V/Q(t) - 1]_0^{x_{max}}\).

- \((x_2(t), c(t)) \in [0, x_{max}] \times [0, x_{max}]\) to minimize \(-V \log(1 + x_2(t)) + Z(t)e^{c(t)+x_2(t)} + Z(t)e^{c(t)} - Q(t)c(t)\). Thus, \(x_2(t)\) and \(c(t)\) must be chosen together.

- Choose \(a(t) \in [0, x_{max}]\) to minimize \(-a(t)Q(t) + 2Z(t)e^{a(t)}\). Thus, choose \(a(t) = \left[\log \left(\frac{Q(t)}{2Z(t)}\right)\right]_0^{x_{max}}\).

- Choose \(b(t) \in [0, x_{max}]\) to minimize \(-b(t)Q(t) + Z(t)e^{b(t)}\). Thus, \(b(t) = \left[\log \left(\frac{Q(t)}{Z(t)}\right)\right]_0^{x_{max}}\).

**J. General time averages (without convexity)**

Theorem [VI.1] and Lemma [VI.1] only use convexity of \(f(x)\) and \(g_k(x)\) when applying Jensen’s inequality at the every end to push a time average inside a convex function while preserving a desired inequality. No convexity assumptions are needed if one wants to optimize the time average of a function, rather than a function of a time average. Indeed, suppose \(\mathcal{Y}\) is a closed and
bounded (possibly non-convex) subset of $\mathbb{R}^{K+1}$ (possibly non-convex). Consider the following time average problem: Every time slot $t \in \{0, 1, 2, \ldots\}$, choose a vector $y(t) = (y_0(t), y_1(t), \ldots, y_K(t)) \in \mathcal{Y}$ so that the resulting time averages solve:

\[
\begin{align*}
\text{Minimize:} & \quad \lim_{t \to \infty} \bar{y}_0(t) \\
\text{Subject to:} & \quad \lim_{t \to \infty} \bar{y}_k(t) \leq c_k \quad \forall k \in \{1, \ldots, K\} \\
& \quad x(t) \in \mathcal{X} \quad \forall t \in \{0, 1, 2, \ldots\}
\end{align*}
\]

The general algorithm for this is as follows [3]: For each $k \in \{1, \ldots, K\}$, define virtual queues $Q_k(t)$ by:

\[Q_k(t + 1) = \max\{Q_k(t) + y_k(t) - c_k, 0\}\] (100)

Every slot $t$, observe $Q_1(t), \ldots, Q_K(t)$ and choose $y(t) \in \mathcal{Y}$ to minimize:

\[V y_0(t) + \sum_{k=1}^{K} Q_k(t) y_k(t)\]

Then update the queues via (100). See Exercise VIII-F.12 for a simple example.

K. Equality constraints

Again let $\mathcal{X}$ be a general (possibly non-convex) set and suppose $x(t) \in \mathcal{X}$ for all $t \in \{0, 1, 2, \ldots\}$. Define $y_0(t) = f(x(t))$, $y_k(t) = g_k(x(t))$, and $w_m(t) = h_m(x(t))$ for some bounded functions $f(x)$, $g_k(x)$, $h_m(x)$ over $x \in \mathcal{X}$ for all $k \in \{1, \ldots, K\}$ and $m \in \{1, \ldots, M\}$ (where $K$ and $M$ are some non-negative integers). Consider the problem:

\[
\begin{align*}
\text{Minimize:} & \quad \lim_{t \to \infty} \bar{y}_0(t) \\
\text{Subject to:} & \quad \lim_{t \to \infty} \bar{y}_k(t) \leq 0 \quad \forall k \in \{1, \ldots, K\} \\
& \quad \lim_{t \to \infty} \bar{w}_m(t) = 0 \quad \forall m \in \{1, \ldots, M\} \\
& \quad x(t) \in \mathcal{X} \quad \forall t \in \{0, 1, 2, \ldots\}
\end{align*}
\]

The drift-plus-penalty algorithm in this context uses queues $Q_k(t)$ and $Z_m(t)$ with updates [3]:

\[
\begin{align*}
Q_k(t + 1) &= \max\{Q_k(t) + y_k(t), 0\} \\
Z_m(t + 1) &= Z_m(t) + w_m(t)
\end{align*}
\]

Every slot $t \in \{0, 1, 2, \ldots\}$, the algorithm observes $Q_1(t), \ldots, Q_K(t)$ and $Z_1(t), \ldots, Z_M(t)$ and chooses $x(t) \in \mathcal{X}$ to minimize:

\[V y_0(t) + \sum_{k=1}^{K} Q_k(t) y_k(t) + \sum_{m=1}^{M} Z_m(t) w_m(t)\]

That is, choose $x(t) \in \mathcal{X}$ to minimize the following:

\[V f(x(t)) + \sum_{k=1}^{K} Q_k(t) g_k(x(t)) + \sum_{m=1}^{M} Z_m(t) h_m(x(t))\]

If the $f(x)$ and $g_k(x)$ functions are continuous and convex, the $h_m(x)$ functions are affine, and the set $\mathcal{X}$ is convex, then Jensen’s inequality ensures this procedure provides an $O(\epsilon)$-approximation to the convex program of minimizing $f(x)$ subject to $g_k(x) \leq 0$ for all $k \in \{1, \ldots, K\}$ and $h_m(x) = 0$ for all $m \in \{1, \ldots, M\}$ (where $\epsilon = 1/V$).

L. Stochastic problems

See [3] for a development of the drift-plus-penalty algorithm in more general scenarios, including stochastic scenarios.

VII. Network optimization via drift-plus-penalty

This section applies the drift-plus-penalty algorithm to general network optimization problems. Recall that the drift-plus-penalty algorithm is as follows: To solve the convex program of finding $x \in \mathcal{X}$ to minimize $f(x)$ subject to $g_k(x) \leq c_k$ for all $k \in \{1, \ldots, K\}$, first define virtual queues:

\[Q_k(t + 1) = \max\{Q_k(t) + g_k(x(t)) - c_k, 0\} \quad \forall k \in \{1, \ldots, K\}\] (101)

Assume that $Q_k(0) = 0$ for all $k \in \{1, \ldots, K\}$. Every slot $t \in \{0, 1, 2, \ldots\}$, observe the virtual queues $Q_k(t)$ and choose $x(t) \in \mathcal{X}$ to minimize the expression:

\[V f(x(t)) + \sum_{k=1}^{K} Q_k(t) g_k(x(t))\]

Then update the virtual queues for slot $t + 1$ via (101) using the vector $x(t)$ that was chosen on slot $t$. 
A. Flow-based optimization

Consider a network with $L$ links and $N$ traffic flows. Let $C_l$ be the capacity of link $l \in \{1, \ldots, L\}$, taking units of bits/slot (where a time slot is selected as a convenient unit of time). Assume each flow uses a pre-determined path that consists of a subset of links. Define $P(i)$ as the subset of links used by flow $i$, defined for each $i \in \{1, \ldots, N\}$. Let $\mathcal{N}(l)$ denote the set of flows in $\{1, \ldots, N\}$ that use link $l$. Let $x = (x_1, \ldots, x_N)$ be the vector of flow rates, so that $x_i$ is the traffic rate of flow $i$ (in bits/slot). We want to choose $x \in \mathbb{R}^N$ to solve the following network utility maximization problem:

\begin{align}
\text{Maximize:} & \quad \sum_{i=1}^N \phi_i(x_i) \\
\text{Subject to:} & \quad \sum_{i \in \mathcal{N}(l)} x_i \leq C_l \quad \forall l \in \{1, \ldots, L\} \\
& \quad x_i \in [0, x_{\max}] \quad \forall i \in \{1, \ldots, N\}
\end{align}

where $x_{\max}$ is some maximum flow rate, and $\phi_i(x)$ are concave functions over $x \in [0, x_{\max}]$ for each $i \in \{1, \ldots, N\}$. This problem is similar to (61)-(63) with the exception that a general concave utility function is used (not necessarily $\phi_i(x) = \log(1 + x)$) and the constraint $x_i \geq 0$ is changed to $x_i \in [0, x_{\max}]$. This change is important to ensure the optimization variables $x$ are chosen in a compact set $\mathcal{X}$. In this case, the set $\mathcal{X}$ is the set of all $x \in \mathbb{R}^N$ that satisfy (104).

Virtual queues: For each constraint $l \in \{1, \ldots, L\}$, define a virtual queue $Q_l(t)$ with dynamics:

$$Q_l(t + 1) = \max \left[ Q_l(t) + \sum_{i \in \mathcal{N}(l)} x_i(t) - C_l, 0 \right]$$

Drift-plus-penalty algorithm: Every slot $t$, the network controller observes $Q_1(t), \ldots, Q_L(t)$ and chooses $x(t) \in \mathcal{X}$ to minimize:

$$-V \sum_{i=1}^N \phi_i(x_i(t)) + \sum_{l=1}^L Q_l(t) \left[ \sum_{i \in \mathcal{N}(l)} x_i(t) \right]$$

Separable implementation: The algorithm reduces to the following: Each node $i \in \{1, \ldots, N\}$ chooses $x_i(t) \in [0, x_{\max}]$ to maximize:

$$V \phi_i(x_i(t)) - \left[ \sum_{i \in \mathcal{P}(i)} Q_l(t) \right] x_i(t)$$

Define:

$$W_i(t) = \sum_{l \in \mathcal{P}(i)} Q_l(t)$$

Then $x_i(t)$ is chosen in $[0, x_{\max}]$ to maximize $V \phi_i(x_i(t)) - W_i(t) x_i(t)$.

Special case: In the special case when $\phi_i(x) = (\theta_i/b) \log(1 + bx)$ for all $i$, the decision at flow $i$ is:

$$x_i^*(t) = \left[ \frac{V \theta_i/W_i(t) - 1}{b} \right] x_{\max}$$

Under this algorithm, it is not difficult to show that:

$$Q_l(t) \leq V \theta_{\max}/b + x_{\max} \quad \forall t \in \{0, 1, 2, \ldots\}$$

where $\theta_{\max} = \max_{i \in \{1, \ldots, N\}} \theta_i$, provided that $Q_l(0)$ satisfies this inequality\footnote{More generally, it can be shown that $Q_l(t) \leq O(V)$ for all $t$ whenever the utility functions have a bounded maximum right-derivative.}. Since deviation from optimal utility is like $O(1/V)$, the deviation from optimality can be pushed arbitrarily small by increasing the $V$ parameter, with a corresponding $O(V)$ tradeoff in queue size.

Interpretation: The virtual queueing equation (105) looks like an actual queue for link $l$ that receives arrivals $\sum_{i \in \mathcal{N}(l)} x_i(t)$ on slot $t$, and that has a service rate $C_l$. This is an approximation to the actual network queueing dynamics because it assumes all new arrivals are placed immediately and simultaneously on all links of the path. Of course, the actual arrivals would traverse the path one link at a time. Nevertheless, the approximation is useful and is typically a good approximation of network performance.

The weight $W_i(t)$ is the sum of the current queue values along the path $\mathcal{P}(i)$. It is difficult to obtain the exact value of $W_i(t)$. The value $W_i(t)$ can be approximated by having each packet observe the queue contents in each queue of its path, and store the accumulating sum in a header field. The resulting sum is passed as a delayed feedback message to the source of each flow $i$. This provides an approximation of $W_i(t)$ that is typically within an additive constant $C$ from the true value, called a $C$-additive approximation. It can be shown that the drift-plus-penalty algorithm still works optimally with such approximations.
(see Exercise VIII-F.11). However, the queue bounds increase with $C$, and the resulting value of $V$ typically needs to increase to achieve the same performance as the case $C = 0$.

The above algorithm is similar to the algorithms of [16][17], which use an alternative dual subgradient method for its derivation, and to the “Fast-TCP” implementation in [18] that bases decisions on a path price. The dual subgradient algorithm in that context uses a “step size” parameter $\delta$, and requires updates to be performed every $\delta$ units of time (optimality is approached only when $\delta \to 0$). In contrast, the drift-plus-penalty algorithm is implemented over fixed size slots that do not need to shrink down to zero. This allows a fixed amount of time for network decisions to be made and implemented, which is important in practical situations when decisions cannot be made arbitrarily rapidly. However, the resulting decisions are identical to those of the dual subgradient algorithm under the change of variables $V = 1/\delta$ (so $V$ can be viewed abstractly as an inverse step size parameter). There is another advantage of using a small slot size, even with the drift-plus-penalty algorithm. Indeed, if one assumes that the maximum per-slot arrivals and service are proportional to the slot size, it is not difficult to show that shrinking the slot size can maintain the same network utility with proportionally smaller queue sizes (see Exercise VIII-F.14). In practice, one should use a slot size that is as small as possible, but no smaller than that which is physically practical.

The traditional dual subgradient algorithm analysis does not involve a time average, and hence requires more stringent strict convexity assumptions on the objective function. In particular, it does not support additional routing variables for extended problems of multi-path routing, since those variables appear in the constraints but not in the objective function. The time averaging analysis in the drift-plus-penalty algorithm does not require strict convexity, and hence it can handle any convex programs, including those with multi-path routing (see Exercise VIII-F.6). Further, the time averages often have a direct physical meaning, such as average throughput or power.

It can be shown that using link weights that are a constant multiple of $Q_1(t)$ will still give a correct operation. That is, instead of using $W_i(t)$, one can use $W_i(t) = \sum_{i \in P(i)} \gamma_i Q_i(t)$ for some arbitrary (but fixed) values $\gamma_i > 0$. This is because each constraint $l$ in (103) is equivalent to the corresponding constraint when both sides are multiplied by $\gamma_i$ (see Exercise VIII-F.10). This observation is of practical importance because it implies that any weight that is proportional to $Q_i(t)$ (such as the observed average delay on link $l$) will work as a useful proxy for $Q_i(t)$. Implementations such as TCP-Vegas and TCP-Reno can use weights of this form [17][18]. Indeed, packets can be marked in proportion to the queue size and/or delay experienced over each link, or a “round trip time” can be used to estimate the sum of delays. Such values can be used as proxies for $W_i(t)$.

A more accurate proxy is the sum weight over a “half-trip-time” (using only links from source to destination) rather than a round trip time. Nevertheless, variations of TCP that control flow rates as a function of such weights can often be understood as approximate implementations of the (optimal) drift-plus-penalty algorithm (and hence, also the optimal dual subgradient algorithm in the special case when strict convexity holds). Another practical consideration is that transport layers often use a window-based admission structure, so that admission of a precise amount of data $x_i(t)$ can only be approximated.

### B. Power constraints

Consider the same network scenario as the previous section. Thus, there are $N$ traffic flows, each using a fixed path through the network. However, for each $l \in \{1, \ldots, L\}$, assume the link transmission rate is a function of a power allocation variable: $C_l = \mu_l(p_l)$, where $\mu_l$ is a concave increasing function over $p_l \geq 0$. Suppose the network has $K$ nodes, and let $Out(k)$ be the set of links that transmit out of node $k$. The goal is to maximize utility subject to the additional constraints that each node $k \in \{1, \ldots, K\}$ must maintain an average power constraint. That is, we choose flow rates $x_1, \ldots, x_N$ and powers $p_1, \ldots, p_L$ to solve:

Maximize:  
$$\sum_{i=1}^{N} \phi_i(x_i)$$

Subject to:  
$$\sum_{i \in N(l)} x_i \leq \mu_l(p_l) \quad \forall l \in \{1, \ldots, L\}$$  

$$\sum_{i \in Out(k)} p_i \leq p_k^{\text{max}} \quad \forall k \in \{1, \ldots, K\}$$  

$$x_i \in [0, x_{\max}] \quad \forall i \in \{1, \ldots, N\}$$  

$$p_i \in [0, p_{\max}] \quad \forall \in \{1, \ldots, L\}$$

for some given values $x_{\max}$, $p_{\max}$, and some desired average power constraints $p_k^{\text{avg}}$.

**Solution:** To solve this, define $X$ as the set of all $(x_1, \ldots, x_N, p_1, \ldots, p_L)$ vectors that satisfy (111)-(112).

---

15For example, the drift-plus-penalty analysis assumes arrivals $x_i(t)$ are chosen as any real number in the interval $[0, x_{\max}]$, whereas a practical system often must admit data in packetized units. One way to address this is to use auxiliary variables [3], which maintain packetized admissions without loss of optimality. Another way is to maintain virtual queues with virtual admissions equal to the real numbers $x_i(t)$, but admit actual data as packets $\hat{x}_i(t)$ so that $\sum_{t=0}^{t} x_i(t) \leq \sum_{t=0}^{t} \hat{x}_i(t) \leq \sum_{t=0}^{t} x_i(t) + x_{\max}$ for all $t$, where $x_{\max}$ is the bit size of the largest packet. This idea can also be useful for matching idealized admission rates to actual transport layer admissions in a system that uses a window-based packet admission structure.
Virtual queues: For each constraint \( l \in \{1, \ldots, L\} \) in (109), define a virtual queue \( Q_l(t) \) with update:

\[
Q_l(t+1) = \max \left[ Q_l(t) + \sum_{i \in N(l)} x_i(t) - \mu_l(p_l(t)), 0 \right]
\]  (113)

For each constraint \( k \in \{1, \ldots, K\} \) in (110), define a virtual queue \( Z_k(t) \) with update:

\[
Z_k(t+1) = \max \left[ Z_k(t) + \sum_{l \in \text{Out}(k)} p_l(t) - p_l^{av}, 0 \right]
\]  (114)

Drift-plus-penalty algorithm: Every slot \( t \), observe the queues \( Q_1(t), \ldots, Q_L(t) \) and \( Z_1(t), \ldots, Z_K(t) \) and choose \( x_i(t) \) and \( p_l(t) \) variables to minimize:

\[
-V \sum_{i=1}^N \phi_i(x_i(t)) + \sum_{l=1}^L Q_l(t) \left( \sum_{i \in N(l)} x_i(t) - \mu_l(p_l(t)) \right) + \sum_{k=1}^K Z_k(t) \left( \sum_{l \in \text{Out}(k)} p_l(t) \right)
\]

This reduces to the following separable algorithm:

- (Flow control) Each flow \( i \) chooses \( x_i(t) \in [0, x_{max}] \) to maximize:

\[
V \phi_i(x_i(t)) - W_i(t)x_i(t)
\]

where \( W_i(t) \) is defined in (106).

- (Power allocation) Each node \( k \) chooses powers \( p_l(t) \in [0, p_{max}] \) for all \( l \in \text{Out}(k) \) to minimize:

\[
-Q_l(t)\mu_l(p_l(t)) + Z_k(t)p_l(t)
\]

- (Queue updates) Each link \( l \) updates \( Q_l(t) \) via (113). Each node \( k \) updates \( Z_k(t) \) via (114).

C. Backpressure routing

The previous examples of flow allocation assume flows take fixed paths. The resulting convex programs were written to optimize flow rates subject to link capacity constraints. This does not include an optimization over all possible network routes. In general, a network might send data from the same traffic session over multiple paths. Since the number of paths between two nodes in a network is typically exponential in the size of the network, it is not obvious how to optimize over all possibilities. Nevertheless, this can be done according to a simple convex program that involves a number of variables and constraints that is polynomial in the network size. The key is to write flow conservation equations at each node. This is a node based approach, rather than a link based approach.

Consider a network with \( N \) nodes. The nodes are connected by directional links with capacities \( C_{ab} \), where \( (a,b) \) denotes a link from node \( a \) to node \( b \). Define \( C_{ab} = 0 \) if there is no link. Let \( L \) be the set of all links \( (a,b) \). Suppose there are \( M \) different traffic flows, with flow rates \( x_1, \ldots, x_M \). For each \( m \in \{1, \ldots, M\} \), define \( \text{source}(m) \) and \( \text{dest}(m) \) as the source and destination of flow \( m \). Different flows are said to have the same commodity if they have the same destination. That is, for each node \( c \in \{1, \ldots, N\} \), we say that commodity \( c \) data is data that is destined for node \( c \). Let \( f_{ij}^{(c)} \) be a variable that represents the amount of commodity \( c \) data that is sent over link \( (i,j) \). The goal is to choose routing variables \( f_{ij}^{(c)} \) that represent a feasible way of delivering commodity data. This holds when the \( f_{ij}^{(c)} \) variables satisfy the following flow conservation, link capacity, nonnegativity, and flow efficiency constraints:

\[
\sum_{m \in \mathcal{A}(n,c)} x_m + \sum_{a=1}^N f_{an}^{(c)} = \sum_{b=1}^N f_{nb}^{(c)} \quad \forall n \in \{1, \ldots, N\}, \forall c \neq n
\]  (115)

\[
\sum_{c=1}^N f_{ab}^{(c)} \leq C_{ab} \quad \forall (a,b) \in L
\]  (116)

\[
f_{aa}^{(c)} = 0, f_{ca}^{(c)} = 0 \quad \forall a,c \in \{1, \ldots, N\}
\]  (117)

\[
f_{ab}^{(c)} \geq 0 \quad \forall a,b,c \in \{1, \ldots, N\}
\]  (118)

where \( \mathcal{A}(n,c) \) is defined as the set of all flows \( m \in \{1, \ldots, M\} \) such that \( \text{source}(m) = n \) and \( \text{dest}(m) = c \). Constraints (115) are the flow conservation constraints and ensure that the total commodity \( c \) flow into a node that is not the destination is equal to the total commodity \( c \) flow out. Constraints (116) are the link capacity constraints and ensure that the sum flow rate over a given link \( (a,b) \) does not exceed the link capacity \( C_{ab} \). Constraints (117) are the flow efficiency constraints and ensure that the network does not use a link \( (a,a) \), and does not reinject data that has already arrived to its destination back into the network.
For each flow $m \in \{ 1, \ldots, M \}$ define $x_m^{\text{max}}$ as a positive value that bounds the maximum flow rate, so that $x_m \in [0, x_m^{\text{max}}]$. Let $\phi_m(x)$ be a concave and increasing function over $x \in [0, x_m^{\text{max}}]$. The resulting convex program is:

Maximize:  
\[
\sum_{m=1}^{M} \phi_m(x_m) \tag{119}
\]

Subject to:  
\[
\begin{align*}
\sum_{m \in A(n,c)} x_m + \sum_{n=1}^{n} f_{an}^{(c)} = \sum_{b=1}^{N} f_{nb}^{(c)} \quad \forall n \in \{1, \ldots, N\}, \forall c \neq n \\
x_m \in [0, x_m^{\text{max}}] \quad \forall m \in \{1, \ldots, M\} \tag{120} \\
\text{Constraints (116)-(118)} \tag{121}
\end{align*}
\]

It can be shown that the above convex program is unchanged if the equality constraint (120) is replaced by an inequality $\leq$, meaning that the flow rate in is less than or equal to the flow rate out. This is because node $n$ can generate fake bits that transmit out more than what comes in.

**Queues:** Define $\mathcal{X}$ as the set of all $(x_m)$ and $(f_{ab}^{(c)})$ variables that satisfy (120)-(122). To treat the constraints (120), for each pair $(n, c)$ such that $n \neq c$ define virtual queue $Q_v^{(c)}(t)$ with update equation:

\[
Q_v^{(c)}(t+1) = \max \left[ Q_v^{(c)}(t) + \sum_{m \in A(n,c)} x_m(t) + \sum_{a=1}^{N} f_{an}^{(c)}(t) - \sum_{b=1}^{N} f_{nb}^{(c)}(t), 0 \right]
\]

It can be shown that the algorithm works just as well with the following modified queueing equation, which is more physically practical for multihop networks:

\[
Q_v^{(c)}(t+1) = \max \left[ Q_v^{(c)}(t) - \sum_{b=1}^{N} f_{nb}^{(c)}(t), 0 \right] + \sum_{m \in A(n,c)} x_m(t) + \sum_{a=1}^{N} f_{an}^{(c)}(t) \tag{123}
\]

This equation ensures that the exogenous arrivals on slot $t$, and the endogenous arrivals from other nodes, cannot be transmitted out on the same slot in which they arrive.

**Drift-plus-penalty decisions:** Every slot $t$, the network controller observes the queues and chooses variables $x_m(t)$ and $f_{ab}^{(c)}(t)$ in the set $\mathcal{X}$ to minimize:

\[
-V \sum_{m=1}^{M} \phi_m(x_m) + \sum_{(n, c)} Q_v^{(c)}(t) \left[ \sum_{m \in A(n,c)} x_m(t) + \sum_{a=1}^{N} f_{an}^{(c)}(t) - \sum_{b=1}^{N} f_{nb}^{(c)}(t) \right]
\]

By simple rearrangements and switching the sums, the above expression becomes:

\[
\sum_{m=1}^{M} \left[ -V \phi_m(x_m) + x_m(t)Q_v^{(\text{dest}(m))}(t) \right] + \sum_{(a,b) \in \mathcal{L}} \sum_{c=1}^{N} f_{ab}^{(c)}(t) \left[ Q_v^{(c)}(t) - Q_v^{(c)}(t) \right]
\]

The value $Q_v^{(c)}(t) - Q_v^{(c)}(t)$ is called the differential backlog of commodity $c$ between nodes $a$ and $b$. If this value is large, there is a pressure gradient associated with commodity $c$ data on the $(a, b)$ link. Define:

\[
W_{ab}^{(c)}(t) = Q_v^{(c)}(t) - Q_v^{(c)}(t)
\]

The above expression is minimized by the following separable decisions:

- (Flow control) Each flow $m \in \{ 1, \ldots, M \}$ chooses $x_m(t) \in [0, x_m^{\text{max}}]$ to maximize:

\[
-V \phi_m(x_m) - x_m(t)Q_v^{(\text{dest}(m))}(t)
\]

This is a simple decision about flow allocation at the source that only requires knowledge of the queue backlog in the source queue. Unlike the link-based algorithm, summing weights along a path is not required.

- (Commodity selection) Each link $(a, b)$ observes the differential backlog $W_{ab}^{(c)}(t)$ for all of its commodities $c \in \{1, \ldots, N\}$.

It then chooses the single commodity $c_{ab}^{*}(t)$ that maximizes $W_{ab}^{(c)}(t)$ (breaking ties arbitrarily). This is a simple decision that is made in a distributed way at each node.

- (Transmission) Each link $(a, b)$ does the following: If $\max_{c \in \{1, \ldots, N\}} W_{ab}^{(c)}(t) > 0$, then choose $f_{ab}^{(c_{ab}^{*}(t))(t)} = C_{ab}$ and $f_{ab}^{(c)}(t) = 0$ for all $c \neq c_{ab}^{*}(t)$. That is, the link sends the single commodity $c_{ab}^{*}(t)$ with the largest differential backlog over the link, using the full link capacity $C_{ab}$. If there is not enough data of commodity $c_{ab}^{*}(t)$ to fill up the link capacity, then fake bits are transmitted.

16The work in [19] starts with [123], defines a Lyapunov function, and shows that the resulting drift-plus-penalty expression still satisfies an inequality of the form [70].

17Intuitively, this does not limit optimality because such a situation occurs only when a queue is not in danger of instability.
The first backpressure algorithm was developed in [20] in the special case \( V = 0 \), so there was no utility optimization, no flow control, and the flow rates \( x_m \) were given constants that were assumed to be supportable over the network. This was done using a “pure drift” approach of minimizing the Lyapunov drift \( \Delta(t) \). Treatment of joint flow control and backpressure was done in [19] [21] using a \( V \) parameter to minimize the drift-plus-penalty expression \( \Delta(t) - V \sum_{m=1}^{M} \phi_m(x_m(t)) \).

A simple single-commodity backpressure example is given in Exercise VII-F.13.

Experimental improvements: Note that the above algorithm achieves optimal network utility (over all possible routing algorithms) in a distributed way and without knowing a routing table. The reason is that the backlog gradients build up, much like pressure gradients when water flows through a system of pipes, and these gradients eventually push the data in optimal directions. However, this basic algorithm can introduce large network delay, particularly when data wanders around circuitous paths before gradients build up. Two standard improvements have been observed to experimentally reduce delay dramatically:

- Using augmented weights \( W^{(c)}_{ab}(t) = Q^{(c)}_a(t) - Q^{(c)}_b(t) + \theta(G^{(c)}_a - G^{(c)}_b) \), where \( G^{(c)}_a \) is an estimate of the distance between nodes \( a \) and \( c \), and \( \theta \) is some non-negative constant. It can be shown mathematically that any non-negative values \( \theta \) and \( G^{(c)}_a \) can be used without affecting throughput utility of the algorithm (provided that \( V \) is sufficiently large). However, it is observed experimentally that choosing \( G^{(c)}_a \) according to distance estimates provides a significant delay reduction [3] [7] [22] [23].

- Using Last-in-First-Out (LIFO) implementation instead of First-in-First-Out (FIFO) implementation [24] [25]. This can dramatically reduce delay of 98\% of the data, at the cost of incurring a large (possibly infinite) delay of the remaining 2\% of the data.\(^{18}\) Intuitively, the reason is that the network has a transient phase where backpressure gradients are built out of data itself. Under LIFO, the early data that create these gradients tend to stay in the same nodes for a long (possibly infinite) time, while the majority of the data that arrives after this transient phase speedily traverses the network using the backpressure gradients as a form of routing table.

D. Flow-based multi-path routing

As an alternative to backpressure routing, one can use the flow-based approach of Section VII-A without considering variables for all of the exponentially many paths, by taking advantage of the fact that finding a shortest path in a directed graph can be done in polynomial time (using, for example, a Dijkstra or Bellman-Ford algorithm). This section develops such an approach. The analysis of this section uses the time-average optimization framework of Section VI-J, and was originally developed in a more general stochastic context in Chapter 4.1 of [3].

As in Section VII-A, consider a network with \( L \) links (with link capacities \( C_1, \ldots, C_L \)) and \( N \) traffic flows. Every slot \( t \in \{0, 1, 2, \ldots\} \), each traffic flow \( i \in \{1, \ldots, N\} \) chooses an amount of data \( x_i(t) \in [0, x_{\text{max}}] \) and a route matrix \( (1_{il}(t)) \). The value \( x_i(t) \) represents the amount of data (in units such as bits or packets) injected into the network layer from flow \( i \) on slot \( t \). The route matrix \( (1_{il}(t)) \) is a matrix that specifies the particular path used for this newly injected data, so that \( 1_{il}(t) \) is an indicator function that is 1 if the path uses link \( l \), and is 0 else. Thus, every slot, each session can choose a new path for its data. Let \( P_i \) be the set of valid paths through the network from the source of flow \( i \) to its destination. The set \( P_i \) might contain only one or two path options, or might contain all possible paths from source to destination. For each link \( l \in \{1, \ldots, L\} \), define \( y_i(t) \) as the amount of data injected into the network layer on slot \( t \) that will eventually use link \( l \):

\[
y_i(t) = \sum_{i=1}^{N} 1_{il}(t)x_i(t)
\]

Define time averages for all slots \( t > 0 \):

\[
\bar{x}_i(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} x_i(\tau) \quad \forall i \in \{1, \ldots, N\}
\]

\[
\bar{y}_l(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} y_l(\tau) \quad \forall l \in \{1, \ldots, L\}
\]

Each link capacity constraint can be specified in a time average sense:

\[
\lim_{t \to \infty} \bar{y}_l(t) \leq C_l \quad \forall l \in \{1, \ldots, L\}
\]

\(^{18}\)There is nothing fundamental about the number 0.98 in this context. For any \( \delta > 0 \), the \( V \) parameter can be chosen sufficiently large to ensure the rate of packets with large delay is at most \( \delta \).
For each \( i \in \{1, \ldots, N\} \), let \( \phi_i(x_i) \) be a concave and non-decreasing utility function defined over \( [0, x_{\text{max}}] \). The time-average optimization problem is:

\[
\begin{align*}
\text{Maximize:} & \quad \lim_{t \to \infty} \sum_{i=1}^{N} \phi_i(x_i(t)) \\
\text{Subject to:} & \quad \lim_{t \to \infty} \mathcal{B}_i(t) \leq C_i \quad \forall i \in \{1, \ldots, L\} \\
& \quad x_i(t) \in [0, x_{\text{max}}] \quad \forall i \in \{1, \ldots, N\} \\
& \quad (1_{il}(t)) \in \mathcal{P}_i \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\] (124)

(125)

(126)

(127)

The above problem is almost in the form of the general time average optimization problem of Section VI-J. However, the objective (124) seeks to maximize a concave function of a vector of time averages \((\mathcal{X}_1(t), \ldots, \mathcal{X}_N(t))\), rather than maximize the time average of a function \(y_0(t)\). Fortunately, a redundant constraint can be added to the problem to make it exactly fit the framework of Section VI-J without changing the underlying optimal solution. This is done using the auxiliary variable method of [3]. For each \( i \in \{1, \ldots, N\} \), let \( \gamma_i(t) \) be an auxiliary variable chosen in the set \([0, x_{\text{max}}]\) for each slot \( t \). Define:

\[
y_0(t) = \sum_{i=1}^{N} \phi_i(\gamma_i(t))
\] (128)

Consider the modified problem:

\[
\begin{align*}
\text{Maximize:} & \quad \lim_{t \to \infty} y_0(t) \\
\text{Subject to:} & \quad \lim_{t \to \infty} \mathcal{B}_i(t) \leq C_i \quad \forall i \in \{1, \ldots, L\} \\
& \quad \mathcal{X}_i(t) \leq \mathcal{X}_i \quad \forall i \in \{1, \ldots, N\} \\
& \quad x_i(t) \in [0, x_{\text{max}}] \quad \forall i \in \{1, \ldots, N\} \\
& \quad (1_{il}(t)) \in \mathcal{P}_i \quad \forall i \in \{1, \ldots, N\} \\
& \quad \gamma_i(t) \in [0, x_{\text{max}}] \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\] (129)

(130)

(131)

(132)

(133)

(134)

It can be shown that the optimal solution to this modified problem uses decisions \(x_i(t)\) and \((1_{il}(t))\) that are also optimal for the original problem (124)-(127)\] [3]. Moreover, this new problem (129)-(134) is in the exact form required for the time average optimization procedure of Section VI-J. Indeed, one can define \(w(t)\) as a concatenated vector \(w(t) = (y_0(t), y_1(t), \ldots, y_L(t), \gamma_1(t) - x_1(t), \ldots, \gamma_N(t) - x_N(t))\), and define \(\mathcal{W}\) as the set of all \(w(t)\) vectors possible under the constraints (132)-(134). The resulting set \(\mathcal{W}\) is non-convex, but this does not matter in the time averaging framework of Section VI-J. Intuitively, this is because the time averaging operation produces a mixture of points in \(\mathcal{W}\) that enables optimization over the convex hull of \(\mathcal{W}\).

Applying the drift-plus-penalty procedure of Section VI-J to the problem (129)-(134) gives the following algorithm: For each constraint in (130) define a virtual queue \(Q_i(t)\):

\[
Q_i(t + 1) = \max\{Q_i(t) + y_i(t) - C_i, 0\}
\]

\[
= \max \left[ Q_i(t) + \sum_{i=1}^{N} 1_{il}(t)x_i(t) - C_i, 0 \right] \quad \forall i \in \{1, \ldots, L\}
\] (135)

For each constraint in (131) define a virtual queue \(Z_i(t)\):

\[
Z_i(t + 1) = \max\{Z_i(t) + \gamma_i(t) - x_i(t), 0\} \quad \forall i \in \{1, \ldots, N\}
\] (136)

The virtual queues \(Q_i(t)\) can be interpreted the same way as in Section VII-A. They represent an approximate network layer queue for link \(l\), where arrivals are the newly admitted data on slot \(t\) (that eventually passes through link \(l\)) and service is the value \(C_l\). The virtual queues \(Z_i(t)\) also have a physical interpretation: They can be viewed as transport layer queues, where \(\gamma_i(t)\) is the amount of of data added to the transport layer queue on slot \(t\), and \(x_i(t)\) is the amount shifted from this queue to the network layer.

Every slot \(t \in \{0, 1, 2, \ldots\\}\), observe the virtual queues \((Q_1(t), \ldots, Q_L(t))\) and \((Z_1(t), \ldots, Z_N(t))\) and choose decision variables \(x_i(t)\), \(\gamma_i(t)\), and \((1_{il}(t))\) subject to (132)-(134) to minimize:

\[
-Vy_0(t) + \sum_{i=1}^{L} Q_i(t) \left[ \sum_{i=1}^{N} 1_{il}(t)x_i(t) \right] + \sum_{i=1}^{N} Z_i(t)(\gamma_i(t) - x_i(t))
\]

\[\text{19}\text{The key step in the proof uses Jensen’s inequality to conclude } \mathcal{B}_i(t) \leq \sum_{i=1}^{N} \phi_i(\mathcal{X}_i(t)) \text{ for all slots } t \geq 0. \text{This, together with (131) and the non-decreasing assumption on } \phi_i(\cdot), \text{ implies } \lim_{t \to \infty} \mathcal{B}_i(t) \leq \lim_{t \to \infty} \sum_{i=1}^{N} \phi_i(\mathcal{X}_i(t)).\]
Substituting the definition of $y_0(t)$ in (128) into the above expression and rearranging terms gives the following expression to be minimized:

$$
\sum_{i=1}^{L} [-V\phi_i(\gamma_i(t)) + Z_i(t)\gamma_i(t)] + \sum_{i=1}^{N} x_i(t) \left[ \sum_{l=1}^{L} 1_{il}(t)Q_l(t) - Z_l(t) \right]
$$

Notice that the expression $\sum_{l=1}^{L} 1_{il}(t)Q_l(t)$ is the sum of link weights $Q_l(t)$ over all links on the path chosen by flow $i$ for its data injected into the network layer at time slot $t$. Minimization of the above expression results in the following separable algorithm that is implemented every slot $t \in \{0, 1, 2, \ldots \}$:

- Choose auxiliary variables $\gamma_i(t)$ (transport layer decisions): Each flow $i$ observes $Z_i(t)$ and separately chooses $\gamma_i(t) \in [0, x_{\text{max}}]$ to minimize:
  $$
  - V\phi_i(\gamma_i(t)) + Z_i(t)\gamma_i(t)
  $$
  (137)

- Choose routing variables $1_{il}(t)$ (network layer decisions): Each flow $i$ observes the queues $Q_l(t)$ in the network, and chooses a path from source to destination in $P_i$ that minimizes $\sum_{l=1}^{L} 1_{il}(t)Q_l(t)$. This is equivalent to finding a shortest path using link weights equal to the virtual queue values $Q_l(t)$. If the set of paths $P_i$ for flow $i$ contains all paths from the flow $i$ source to the flow $i$ destination, the shortest path can be found via Bellman-Ford, Dijkstra, or any other shortest path finder. Let $W_i(t)$ represent the resulting sum weight along the shortest path for flow $i$.

- Choose flow control variables $x_i(t)$ (transport layer decisions): Each flow $i$ observes $Z_i(t)$ and also observes the weight $W_i(t)$ computed from the slot $t$ routing decision specified above. Then:
  $$
  x_i(t) = \begin{cases} 
  x_{\text{max}} & \text{if } W_i(t) \leq Z_i(t) \\
  0 & \text{otherwise}
  \end{cases}
  $$
  (138)

- Queue updates: Update $Q_l(t)$ and $Z_l(t)$ for $l \in \{1, \ldots, L\}$, $i \in \{1, \ldots, N\}$ via (135)-(136).

Overall, the above algorithm uses shortest-path routing with link weights equal to virtual queue values $Q_l(t)$. Each flow $i$ admits $x_i(t) = x_{\text{max}}$ units of data into the network layer whenever the sum link weight along the shortest path is sufficiently small. However, if the sum link weight is too large (due to congestion in the links), the algorithm switches the transport layer variables $x_i(t)$ to 0. This restricts new arrivals from flow $i$ until some of the queue backlog is reduced. The resulting algorithm produces flow rates that have total network utility within $O(1/V)$ of optimality. If the utility functions $\phi_i(x)$ have bounded right-derivatives over the interval $[0, x_{\text{max}}]$, then it can be shown that the $Q_l(t)$ and $Z_l(t)$ queues are deterministically bounded with worst case queue size that is proportional to $V$ (see [3] and Exercise VIII-F.18). Thus, the algorithm exhibits an $O(1/V)$ approximation to optimal utility with a corresponding $O(V)$ tradeoff in queue size.

The most difficult parts of the above algorithm are the determination of link weights $Q_l(t)$ across the network, and the implementation of the shortest-path solver with respect to these weights. As before, any link weights within an additive constant of the true weights $Q_l(t)$ can be used, since this results in a $C$-additive approximation. Furthermore, as $V$ gets large, it can be shown that the weights $Q_l(t)$ stay relatively close to a Lagrange multiplier vector associated with the problem [26][27], so that past estimates of these weights are accurate approximations to the current weights.

The above algorithm approximates the network layer queue via (135), whereas the backpressure approach of the previous subsection uses an actual queueing equation. Backpressure is also easier to implement in a distributed fashion, since its routing decisions are based on weights of neighboring nodes (with no need for a shortest-path computation). However, backpressure requires maintaining network layer queues $Q_n^c(t)$ for each node $n$ and each traffic flow $c$, whereas the above algorithm requires only one network layer queue $Q_l(t)$ for each link $l \in \{1, \ldots, L\}$. An early version of this shortest-path based algorithm was developed in [28] for the special case with no utility maximization. Specifically, [28] assumes data arrives to the network according to a fixed rate vector that is in the “network capacity region” (so that the network can support all arrival rates), and used a Lyapunov-based min-drift policy to achieve network stability. The above algorithm for joint stability and utility maximization was developed [3] using Lyapunov optimization and the drift-plus-penalty technique.

 VIII. Exercises

A. Pareto optimality exercises

**Exercise VIII-A.1.** (Pareto optimality over 5 points) Define $\mathcal{A} = \{(1, 4), (2.8, 4.1), (2.5, 2.9), (3.5, 1.5), (3, 1)\}$.  
  a) Plot these points, and find all the ones that are Pareto optimal.
  b) Find $c_{\text{min}}$, and plot the tradeoff function $\psi(c)$ for all $c \in [c_{\text{min}}, \infty)$.

**Exercise VIII-A.2.** (Lagrange multipliers for a finite set) Define $\mathcal{A} = \{(1, 4), (2.8, 4.1), (2.5, 2.9), (3.5, 1.5), (3, 1)\}$, as in Exercise VIII-A.1  
  a) Plot the points in $\mathcal{A}$ together with the line $y + 2x = b$ for $b = 1, b = 3, b = 5$. What is the largest value of $b$ for which all points of $\mathcal{A}$ are on or above the line? 
  b) Find an optimal solution of (7)-(8) for $\mu = 2$. Repeat for $\mu = 3$. 
c) Find an optimal solution of (7)-(8) for \( \mu = 0 \). Call your answer \((x^*, y^*)\). For what range of \( \mu \) values is \((x^*, y^*)\) a solution to (7)-(8)?

d) Are there any Pareto optimal points in \( A \) that do not have Lagrange multipliers? That is, are there any Pareto optimal points \((x, y)\) that are not solutions to (7)-(8) for any \( \mu \geq 0 \)?

![Fig. 9. A set \( A \) of 8 different operating points \((x, y)\) for Exercise VIII-A.3.](image)

Exercise VIII-A.3. Consider the set \( A \) of 8 different operating points \((x, y)\) in Fig. 9. We want to make both \( x \) and \( y \) small.

a) Specify all Pareto optimal points.

b) Suppose we run an algorithm to find a point \((x^*, y^*)\) in \( A \) that minimizes \( y + \mu x \). If there are ties, the algorithm breaks the ties arbitrarily in a way that is beyond our control. State a value of \( \mu > 0 \) that will surely find the point \((2.5, 3)\).

c) Suppose we want to make the \( x \) coordinate small, and also the \( y \) coordinate small. List all Pareto optimal points.

d) Suppose we want to make the \( x \) coordinate big and the \( y \) coordinate big. Define a new set \( \tilde{A} \) that turns this into a bicriteria minimization problem. Draw the set \( A \) and the set \( \tilde{A} \). Give all Pareto optimal points for \( \tilde{A} \).

e) Suppose we want to make the \( x \) coordinate big and the \( y \) coordinate small. What are the Pareto optimal points?

Exercise VIII-A.4. Consider a triangular region with vertices at \((x, y) = (1, 1)\), \((x, y) = (1, 2)\), and \((x, y) = (4, 1)\). Let \( A \) be the set of all points inside and on this triangle.

a) Suppose we want to make the \( x \) coordinate small, and also the \( y \) coordinate small. List all Pareto optimal points.

b) Suppose we want to make the \( x \) coordinate big and the \( y \) coordinate big. Define a new set \( \tilde{A} \) that turns this into a bicriteria minimization problem. Give all Pareto optimal points for \( \tilde{A} \).

c) Suppose we want to make the \( x \) coordinate big and the \( y \) coordinate small. What are the Pareto optimal points?

Exercise VIII-A.5. (Open sets have no Pareto optimal points) Define \( A \) as the open ball of radius 1 about the point \((5, 3)\):

\[ A = \{(x, y) \in \mathbb{R}^2 | \sqrt{(x-5)^2 + (y-3)^2} < 1\} \]

a) Let \((a, b)\) be a point in \( A \). Find a point \((p, q)\) in \( A \) that satisfies \((p, q) \prec (a, b)\). Your values of \( p \) and \( q \) should be functions of \( a \) and \( b \).

b) Explain why this means that \( A \) has no Pareto optimal points (the proof for general open sets is similar).

Exercise VIII-A.6. (No entry of a Pareto optimal point can be improved without making the other entry worse) Prove that a point \((x^*, y^*)\) in \( A \) is Pareto optimal if and only if every other distinct point \((x, y)\) in \( A \) satisfies the following two conditions:

- If \( x < x^* \) then \( y > y^* \).
- If \( y < y^* \) then \( x > x^* \).

Exercise VIII-A.7. (Pareto optimality in more than 2 dimensions) Generalize the definition of \( x \prec y \) to \( N \)-dimensional vectors \( x = (x_1, \ldots, x_N), y = (y_1, \ldots, y_N) \). Give a definition of Pareto optimality for sets \( A \subseteq \mathbb{R}^N \).

Exercise VIII-A.8. (Pareto optimal points are on the tradeoff curve) Suppose that \((x^*, y^*)\) is Pareto optimal in \( A \). This problem shows that \((x^*, y^*)\) must solve the following optimization problem (defined in terms of optimization variables \( x \) and \( y \)):

\[
\begin{align*}
\text{Minimize:} & \quad y \\
\text{Subject to:} & \quad x \leq x^* \\
& \quad (x, y) \in A
\end{align*}
\]

\[
\text{where } x^* \text{ is treated as a fixed parameter in the above optimization problem.}
\]

a) Show that \((x^*, y^*)\) satisfies the constraints (140)-(141).
b) Show that \( y^* \leq y \) whenever \((x, y)\) is a vector that satisfies (140)–(141). This proves \((x^*, y^*)\) is a solution to (139)–(141).

c) Argue that \( \psi(x^*) = y^* \), so that \((x^*, y^*)\) is a point on the tradeoff curve \((c, \psi(c))\).

**Exercise VIII-A.9.** (Why \( \mu > 0 \) is needed for Pareto optimality) Give a counterexample that shows the result of part (c) of Theorem II.7 does not necessarily hold when \( \mu = 0 \).

**Exercise VIII-A.10.** (Proof of part (c) of Theorem II.7) Prove part (c) of Theorem II.7.

**Exercise VIII-A.11.** (Existence of Pareto optimal points) Let \( A \) be a nonempty compact subset of \( \mathbb{R}^2 \). We want to show that \( A \) has a Pareto optimal point. A theorem of real analysis states that a continuous function defined over a compact set must have a (possibly non-unique) minimizer. In particular, if \( f(x, y) \) is a continuous function defined over the compact set \( A \), then there is a point \((x^*, y^*)\) \( \in A \) such that \( f(x^*, y^*) \leq f(x, y) \) for all \((x, y)\) \( \in A \).

a) Define \( f(x, y) = x + y \). Prove that its minimizer \((x^*, y^*)\) \( \in A \) is Pareto optimal in \( A \).

b) Consider any point \((a, b)\) \( \in A \). Show there exists a point \((x^*, y^*)\) that is Pareto optimal in \( A \) and that satisfies \( x^* \leq a, y^* \leq b \). Hint: The set \( S(a, b) \) is closed, and the intersection of a compact set and a closed set is compact.

### B. Optimization and Lagrange multiplier theory exercises

**Exercise VIII-B.1.** (Problems with no optimal solution) Consider the problem of minimizing \( y \) subject to the constraints \( x \leq 1 \) and \((x, y)\) \( \in A \).

a) Suppose \( A = \mathbb{R}^2 \). Show that there is no optimal solution. Is the set \( A \) compact?

b) Suppose \( A = \{(x, y) \in \mathbb{R}^2 | 0 \leq x \leq 2, 0 \leq y \leq 2 \} \). Show that there is no optimal solution. Is the set \( A \) compact?

**Exercise VIII-B.2.** (Lagrange multiplier for equality constraint) Prove Theorem II.3.

**Exercise VIII-B.3.** (If \( g(x^*) < c \) then \( \nabla f(x^*) = 0 \)) Suppose \( x^* \) is an interior point of \( X \) that is also a solution to (13)–(15). Suppose \( g(x^*) < c \). Assume that \( g(x) \) is continuous and \( f(x) \) is differentiable at the point \( x^* \).

a) Show that if \( \nabla f(x^*) \neq 0 \), then defining \( v = -\nabla f(x^*) \) ensures that there is a \( \delta_{\text{max}} > 0 \) such that \( x^* + \delta v \in X \) and \( f(x^* + \delta v) < f(x^*) \) for all \( \delta \in (0, \delta_{\text{max}}] \). Hint: Use Lemma VIII.1 in Appendix A.

b) Use continuity of \( g(x) \) to show that if \( \delta \) is chosen sufficiently small in the interval \((0, \delta_{\text{max}}]\), then \( g(x^* + v\delta) < c \).

c) Conclude from parts (a) and (b) that \( \nabla f(x^*) = 0 \).

**Exercise VIII-B.4.** (Separable minimization) Fix \( N \) as a positive integer, and fix positive constants \( a_1, \ldots, a_N \) and \( b_1, \ldots, b_N \). Find \( x \in \mathbb{R}^N \) to minimize:

\[
\left( \sum_{i=1}^{N} a_i x_i \right) - \left( \sum_{i=1}^{N} b_i x_i \right)
\]

**Exercise VIII-B.5.** (Increasing the Lagrange multiplier \( \mu \)) Let \( A \) be a nonempty subset of \( \mathbb{R}^2 \). Consider the unconstrained problem:

\[
\begin{align*}
\text{Minimize:} & \quad y + \mu x \\
\text{Subject to:} & \quad (x, y) \in A
\end{align*}
\]

where \( \mu \) is a given nonnegative real number. Intuitively, we understand that increasing the value of \( \mu \) places more emphasis on keeping the \( x \)-value small. This problem is designed to make that intuition precise. Fix real numbers \( \mu_1 \) and \( \mu_2 \) such that \( 0 \leq \mu_1 < \mu_2 \). Let \((x_1^*, y_1^*)\) be an optimal solution to the unconstrained problem for the case \( \mu = \mu_1 \). Let \((x_2^*, y_2^*)\) be an optimal solution to the unconstrained problem for the case \( \mu = \mu_2 \).

a) Prove that \( x_2^* \leq x_1^* \).

b) Give an example where \( x_2^* = x_1^* \).

**Exercise VIII-B.6.** (Changing multiple Lagrange multipliers) Fix \( N \) and \( K \) as positive integers. Let \( X \) be a subset of \( \mathbb{R}^N \). Let \( f(x), g_1(x), \ldots, g_K(x) \) be real-valued functions over \( x \in X \). Define \( g : \mathbb{R}^N \rightarrow \mathbb{R}^K \) as the vector-valued function \( g(x) = (g_1(x), \ldots, g_K(x)) \). Let \( \mu = (\mu_1, \ldots, \mu_K) \) be a given vector of real numbers. Consider the problem:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) + \mu^T g(x) \\
\text{Subject to:} & \quad x \in X
\end{align*}
\]

where \( \mu^T g(x) = \sum_{k=1}^{K} \mu_k g_k(x) \). Let \( x_\mu \) be a solution to the above minimization problem. Let \( \lambda = (\lambda_1, \ldots, \lambda_K) \) be another vector. Let \( x_\lambda \) be a solution to the modified minimization problem that arises by replacing the vector \( \mu \) with the vector \( \lambda \).

a) Show that \( (\lambda - \mu)^T (g(x_\lambda) - g(x_\mu)) \leq 0 \) (see hint in proof of Theorem III.2).

b) Use (a) to show that if \( \lambda_1 > \mu_1 \) but \( \lambda_k = \mu_k \) for \( k \geq 2 \), then \( g_1(x_\lambda) \leq g_1(x_\mu) \).

c) Give an example problem with \( N = 1, K = 2 \), and vectors \( \mu = (1, 0) \), \( \lambda = (2, 1) \) such that \( g_1(x_\lambda) > g_1(x_\mu) \). In this example, both Lagrange multipliers are increased, but the value of \( g_1 \) increases. Do the same for \( \mu = (1, 1) \), \( \lambda = (2, 2) \).
Exercise VIII-B.7. (Weak duality) Fix $N$ and $K$ as positive integers. Let $\mathcal{X} \subseteq \mathbb{R}^N$. Let $f, g_1, \ldots, g_K$ be real-valued functions over $x \in \mathcal{X}$. Consider the problem:

\begin{align}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g_k(x) \leq 0, \quad \forall k \in \{1, \ldots, K\} \\
& \quad x \in \mathcal{X}
\end{align}

a) Argue that this problem structure can be used to model all problems of the form (44)–(46).

b) Define $\mathbb{R}^N_+ = \{(x_1, \ldots, x_N) \in \mathbb{R}^N : x_k \geq 0 \text{ for all } k\}$. For nonnegative vectors $\mu = (\mu_1, \ldots, \mu_K)$, the dual function for the above problem is defined:

$$d(\mu) = \inf_{x \in \mathcal{X}} \left[ f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \right]$$

where $d(\mu)$ is allowed to take the value $-\infty$. Show that if $x^*$ is an optimal solution to (142)–(144), then $d(\mu) \leq f(x^*)$ for all nonnegative vectors $\mu = (\mu_1, \ldots, \mu_K)$. This is called weak duality: The dual function of a constrained optimization problem is less than or equal to its optimal objective value. Hint: The infimum of a function is less than or equal to the function evaluated at any particular point.

c) Define $d^* = \sup_{\mu \in \mathbb{R}^N_+} d(\mu)$. Show that $d^* \leq f(x^*)$. The difference $f(x^*) - d^*$ is called the duality gap. We say that strong duality holds if the duality gap is zero. Show that strong duality holds if there is a $\mu^* \in \mathbb{R}^N_+$ such that $d(\mu^*) = f(x^*)$.

Exercise VIII-B.8. (Strong duality) Consider the constrained optimization problem (142)–(144) of the Exercise VIII-B.7 (with optimal solution $x^*$) and define the dual function $d(\mu)$ in the same way. Further suppose the set $\mathcal{X}$ is convex and the functions $f, g_1, \ldots, g_K$ are convex.

a) Define $\mathcal{A} = \{(y_0, y_1, \ldots, y_K) \in \mathbb{R}^{K+1} : (y_0, y_1, \ldots, y_K) \geq (f(x), g_1(x), \ldots, g_K(x)) \text{ for some } x \in \mathcal{X}\}$, where the vector inequality is taken entrywise. Show that $\mathcal{A}$ is a convex set.

b) Show that $(f(x^*), 0, 0, \ldots, 0) \notin \mathcal{A}$ but $(f(x^*) - \delta, 0, 0, \ldots, 0) \notin \mathcal{A}$ for any $\delta > 0$. Hence, $(f(x^*), 0, 0, \ldots, 0)$ is a boundary point of $\mathcal{A}$.

c) The hyperplane separation theorem ensures there is a hyperplane that passes through the boundary point $(f(x^*), 0, 0, \ldots, 0)$ and contains the convex set $\mathcal{A}$ on one side. Thus, there is a nonzero vector $(a_0, a_1, \ldots, a_K)$ such that:

$$a_0 y_0 + a_1 y_1 + \ldots + a_K y_K \geq a_0 f(x^*) + 0 + 0 + \ldots + 0, \quad \forall(y_0, y_1, \ldots, y_K) \in \mathcal{A}$$

Prove that all $a_k$ values are nonnegative. Hint: Increasing any component of a vector in $\mathcal{A}$ produces another vector in $\mathcal{A}$.

d) If $a_0 \neq 0$, the hyperplane is said to be nonvertical. Assume the hyperplane is nonvertical and define $\mu^* = (\mu_1^*, \ldots, \mu_K^*)$ by $\mu_k^* = a_k/a_0$ for all $k \in \{1, \ldots, K\}$. Then $\mu^* \in \mathbb{R}^N_+$. Use part (c) to argue that:

$$f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \geq f(x^*), \quad \forall x \in \mathcal{X}$$

Conclude that $d(\mu^*) = f(x^*)$. Thus, if the hyperplane is nonvertical, then strong duality holds.

C. Lagrange multiplier application exercises

Remember to convert maximization problems into minimization problems, and constraints “≥” into constraints “≤.”

Exercise VIII-C.1. Fix $N$ as a positive integer and fix positive constants $a_1, \ldots, a_N$.

a) Find an optimal solution $x^*$ to the following problem:

\begin{align}
\text{Minimize:} & \quad \sum_{i=1}^{N} e^{a_i x_i} \\
\text{Subject to:} & \quad \sum_{i=1}^{N} x_i \geq -8 \\
& \quad (x_1, \ldots, x_N) \in \mathbb{R}^N
\end{align}

You should be able to get a closed form solution as a function of $a_1, \ldots, a_N$.

b) State $(x_1^*, x_2^*, x_3^*, x_4^*)$ for the special case $N = 4$ and $a_i = i$ for $i \in \{1, 2, 3, 4\}$.

Exercise VIII-C.2. Maximize the function $\log(x_1) + \log(x_2) + \log(x_3)$ subject to the constraints $x_1 + 2x_2 + 8x_3 \leq 1$ and $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$. Hint: It makes sense to define $\mathcal{X} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 : x_i \geq 0 \forall i \in \{1, 2, 3\}\}$.

Exercise VIII-C.3. Minimize $x_1^2 + x_2^2 + x_3^2$ subject to $x_1 + 2x_2 - 3x_3 \geq 8$ and $(x_1, x_2, x_3) \in \mathbb{R}^3$.

Exercise VIII-C.4. Maximize $x_1 + 2x_2 - 3x_3$ subject to $x_1^2 + x_2^2 + x_3^2 \leq 1$.

Exercise VIII-C.5. Minimize $x_1^2 + 5x_2^2 - x_3$ subject to $x_1 + x_2 - x_3 \geq 4$ and $(x_1, x_2, x_3) \in \mathbb{R}^3$. Hint: You should get a cubic equation for $\mu$ that has a positive real solution in the interval $\mu \in [6, 7]$, and which can be solved numerically.
Exercise VIII-C.6. (Problem with an equality constraint) Let \( a = (a_1, \ldots, a_N) \) and \( b = (b_1, \ldots, b_N) \) be nonzero vectors in \( \mathbb{R}^N \). Consider the problem of choosing \( x = (x_1, \ldots, x_N) \in \mathbb{R}^N \) to minimize \( \sum_{i=1}^N a_i x_i^2 \) subject to \( \sum_{i=1}^N b_i x_i = 1 \). Use a Lagrange multiplier approach to compute the optimal \( x^* = (x_1^*, \ldots, x_N^*) \) in terms of entries of the \( a \) and \( b \) vectors.

Exercise VIII-C.7. (Bicriteria shortest paths)

Consider the 6 node network of Fig. [10]. Each link \( (i, j) \) has a distance \( d_{ij} \) and an energy cost \( c_{ij} \) (given in Fig. [10]). We want to find paths to the destination 1 that have low total distance and low total cost. If we focus on a particular start node \( i \neq 1 \), this is a bicriteria minimization problem with feasible set \( A \) consisting of all possible cost-distance pairs \( (c_{tot}, d_{tot}) \) achievable over paths from node \( i \) to node 1. Let \( \mu \geq 0 \) be a real number (the value \( \mu \) will act as a Lagrange multiplier). We want to minimize \( d_{tot} + \mu c_{tot} \) over all paths \( P \) to the destination. That is, we want to find a path \( P \) that minimizes:

\[
\sum_{(i,j) \in P} [d_{ij} + \mu c_{ij}]
\]

This is solved by a shortest path algorithm with weights \( \tilde{d}_{ij} = d_{ij} + \mu c_{ij} \). In this problem, you can use either the Bellman-Ford or Dijkstra algorithms for finding shortest paths (either by hand or by computer).

a) Solve and draw the shortest path tree (using the weights \( \tilde{d}_{ij} \)) for the case \( \mu = 0 \). Considering the path from 6 to 1, what point \( (c_{tot}, d_{tot}) \) on the tradeoff curve does this produce?

b) Fix \( \mu = 1 \). Using the \( \tilde{d}_{ij} \) weights, solve and draw the shortest path tree. Is the tree different from the previous problem?

c) State the Pareto optimal point \( (c_{tot}, d_{tot}) \) achieved for your path in part (b) for:

- The path from 6 to 1.
- The path from 4 to 1.

d) Your answer in part (c) finds a path \( P \) that solves an optimization problem of the form:

\[
\begin{align*}
\text{Minimize:} & \quad \sum_{(i,j) \in P} d_{ij} \\
\text{Subject to:} & \quad \sum_{(i,j) \in P} c_{ij} \leq \theta \\
& \quad P \text{ is a path from 6 to 1}
\end{align*}
\]

What is the value of \( \theta \)?

e) What optimization problem (with the structure of part (d)) does your answer in part (c) solve, considering paths from 4 to 1?

Exercise VIII-C.8. (Continuing the previous problem) For the same system from Exercise VIII-C.7, solve again (either by hand, or by computer) the cases \( \mu = 2 \), \( \mu = 5 \) and \( \mu = 10 \):

a) For the path from 6 to 1 with \( \mu = 5 \): What Pareto optimal point \( (c_{tot}, d_{tot}) \) do you get?

b) For the path from 6 to 1 with \( \mu = 10 \): What Pareto optimal point \( (c_{tot}, d_{tot}) \) do you get? (Note: In this case there are two distinct trees that are valid solutions. These lead to two distinct Pareto optimal points that have the same value of \( d_{tot} + \mu c_{tot} \). You just need to find one of them).

c) For the path from 6 to 1 with \( \mu = 11 \): What Pareto optimal point \( (c_{tot}, d_{tot}) \) do you get?

d) For the path from 6 to 1: Including the cases \( \mu = 0 \) and \( \mu = 1 \) from Exercise VIII-C.7 you tested five different \( \mu \) values. This should give you 3 different points on the tradeoff curve (you will find that some of the \( \mu \) values lead to the same point). Plot the three points on a graph with x-axis cost \( c_{tot} \) and y-axis distance \( d_{tot} \).

Exercise VIII-C.9. (Another constrained shortest path problem)

We want to find routes from all nodes to node \( e \) for the graph in Fig. [11]. All links can go in either direction. Each link has a distance and an energy cost, as shown in the figure (the distances and costs are the same for either direction on the link). For \( i \in \{a, b, c, d\} \), let \( D_i(p) \) and \( C_i(p) \) be the distance and energy cost, respectively, associated with a particular path \( p \) from node \( i \) to destination \( e \). For a given node \( i \), we want to solve:
Fig. 11. The 5-node graph for Exercise VIII-C.9 with link distances and energy costs labeled.

Minimize: \( D_i(p) \)
Subject to: \( C_i(p) \leq \theta_i \)

where \( \theta_i \) is a given bound on the energy cost.

a) Use a Lagrange multiplier \( \mu = 2 \) to solve the unconstrained minimization of \( D_i(p) + \mu C_i(p) \) for each \( i \in \{a, b, c, d, e\} \).

Use either the Bellman-Ford or Dijkstra algorithms for computing a shortest path.

b) Draw the shortest path tree corresponding to your work in part (a) for the \( \mu = 2 \) case.

c) Consider the source node \( a \). What constrained optimization problem does your answer to part (a) help you solve for this source node?

Fig. 12. Routing over three parallel links for Exercises VIII-C.10 and VIII-C.11.

Exercise VIII-C.10. Traffic of rate \( r \) bits/second must be split and routed over three links with rates \( x_1, x_2, x_3 \), respectively (see Fig. 12). The power used on each link \( k \) depends on the rate \( x \) it supports according to the function \( p_k(x) = e^{kx} - 1 \) for \( k \in \{1, 2, 3\} \). That is, \( p_1(x_1) = e^{x_1} - 1 \), \( p_2(x_2) = e^{x_2} / 2 - 1 \), and \( p_3(x_3) = e^{x_3} / 3 - 1 \). We want to choose \( x_1, x_2, x_3 \) to solve:

Minimize: \( p_1(x_1) + p_2(x_2) + p_3(x_3) \)
Subject to: \( x_1 + x_2 + x_3 \geq r \)
\( x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \)

a) Fix a Lagrange multiplier \( \mu > 0 \) and solve the problem of minimizing
\( p_1(x_1) + p_2(x_2) + p_3(x_3) - \mu(x_1 + x_2 + x_3) \)
subject to \( x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \). Let \( x_1(\mu), x_2(\mu), x_3(\mu) \) be the result as a function of \( \mu \).

b) Find a \( \mu \) value that gives \( x_1(\mu) + x_2(\mu) + x_3(\mu) = 0.5 \), and hence solves the original constrained optimization for the case \( r = 0.5 \).

c) Solve the original constrained optimization for the case \( r = 0.9 \).

d) Find the smallest value \( r^* \) such that the optimal solution to the constrained optimization problem uses all three links whenever \( r > r^* \). Solve for the case \( r = r^* + 4 \).

Exercise VIII-C.11. Consider the problem of optimally splitting traffic of rate \( r \) over three parallel links. Let \( x_1, x_2, x_3 \) be the flow rates over the links (see Fig. 12). Link costs for \( x_i \geq 0 \) are given by the following convex functions:
\[
\begin{align*}
c_1(x_1) &= (x_1 + 1)^3 - 1 \\
c_2(x_2) &= 2(x_2 + 1)^3 - 2 \\
c_3(x_3) &= 3(x_3 + 1)^3 - 3
\end{align*}
\]
Given rate \( r \), we want to find \( x_1, x_2, x_3 \) to solve:

\[
\begin{align*}
\text{Minimize:} & \quad c_1(x_1) + c_2(x_2) + c_3(x_3) \\
\text{Subject to:} & \quad x_1 + x_2 + x_3 \geq r \\
& \quad x_1 \geq 0, x_2 \geq 0, x_3 \geq 0
\end{align*}
\]

Equivalently, you can replace the constraint \( x_1 + x_2 + x_3 \geq r \) with \( x_1 + x_2 + x_3 = r \).

a) Let \( (x_1^*, x_2^*, x_3^*) \) be the optimal solution. Find the threshold rate \( r_{\text{thresh}} \) for which \( x_3^* = 0 \) if \( r \leq r_{\text{thresh}} \) and \( x_3^* > 0 \) if \( r > r_{\text{thresh}} \).

b) Find the optimal solution \( (x_1^*, x_2^*, x_3^*) \) for \( r = r_{\text{thresh}} + 7 \).

Exercise VIII-C.12. (Lagrange multipliers for one link) Consider the problem of \( N \) users that send over one link of fixed capacity \( C \). Consider the utility function:

\[
\phi(x_1, \ldots, x_N) = \sum_{i=1}^{N} \frac{\theta_i}{b} \log(1 + bx_i)
\]

where \( \theta_i \) are given positive priority weights for users \( i \in \{1, \ldots, N\} \), and \( b \) is a given positive constant.\(^{20}\) We want to solve:

\[
\begin{align*}
\text{Maximize:} & \quad \phi(x_1, \ldots, x_N) \\
\text{Subject to:} & \quad \sum_{i=1}^{N} x_i \leq C \\
& \quad x_i \geq 0 \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\]

a) Fix \( \mu > 0 \). Maximize \( \phi(x_1, \ldots, x_N) - \mu \sum_{i=1}^{N} x_i \) over \( x_i \geq 0 \). Provide a general formula for each \( x_i \) in terms of \( \mu \).

b) Find \( (x_1^*, x_2^*, x_3^*) \) and \( \text{util}^* = \phi(x_1^*, x_2^*, x_3^*) \) for the case \( N = 3 \), \( b = 5 \), \( \theta_1 = 1 \), \( \theta_2 = \theta_3 = 2 \), \( C = 1 \).

c) Find \( (x_1^*, x_2^*, x_3^*) \) and \( \text{util}^* = \phi(x_1^*, x_2^*, x_3^*) \) for the case \( N = 3 \), \( b = 5 \), \( \theta_1 = 1 \), \( \theta_2 = \theta_3 = 5 \), \( C = 1 \).

---

\( ^{20} \)Using a large value of \( b \) allows approximation of the (weighted) proportional fairness utility \( \sum_{i=1}^{N} \theta_i \log(x_i) \).
D. Convexity theory exercises

Exercise VIII-D.1. Let $A$ be a finite subset of $\mathbb{R}^N$ that consists of $K$ points, where $K \geq 2$. Is $A$ a convex set?

Exercise VIII-D.2. Let $X$ be a convex subset of $\mathbb{R}^N$. Let $f_1(x), \ldots, f_K(x)$ be a collection of convex functions from $X$ to $\mathbb{R}$. Let $c_1, \ldots, c_K$ be non-negative numbers. Prove that $g(x) = \sum_{i=1}^{K} c_i f_i(x)$ is a convex function from $X$ to $\mathbb{R}$.

Exercise VIII-D.3. Define a function from $\mathbb{R}^3$ to $\mathbb{R}$ by $f(x_1, x_2, x_3) = x_1^2$. Show that $f(x_1, x_2, x_3)$ is convex but not strictly convex.

Exercise VIII-D.4. (Affine functions are both convex and concave) Define a function from $\mathbb{R}^N$ to $\mathbb{R}$ by $f(x) = a_0 + a_1 x_1 + \ldots + a_N x_N$, where $a_i \in \mathbb{R}$ for all $i \in \{0, 1, \ldots, N\}$. Prove that $f(x)$ is both convex and concave, but neither strictly convex nor strictly concave.

Exercise VIII-D.5. Suppose $f_1(x_1), \ldots, f_N(x_N)$ are functions from $\mathbb{R}$ to $\mathbb{R}$. Let $x = (x_1, \ldots, x_N)$ and define $f(x) = \sum_{i=1}^{N} f_i(x_i)$.
   a) Show that if $f(x)$ is strictly convex over $\mathbb{R}^N$, then each $f_i(x_i)$ function is strictly convex over $\mathbb{R}$.
   b) Suppose each $f_i(x_i)$ function is strictly convex over $\mathbb{R}$. Let $x = (x_1, \ldots, x_N)$ and $y = (y_1, \ldots, y_N)$ be points in $\mathbb{R}^N$ such that $x \neq y$. Fix $\theta \in (0, 1)$. Prove that $f(\theta x + (1-\theta) y) < \theta f(x) + (1-\theta) f(y)$. Hint: Find an entry $j$ such that $x_j \neq y_j$.

Exercise VIII-D.6. (Convex inequality constraints) Prove Lemma IV.3

Exercise VIII-D.7. (Convexity of norms) Let $a = (a_1, \ldots, a_N) \in \mathbb{R}^N$ and let $f(x) = \|x - a\|$. Prove that $f(x)$ is convex over $x \in \mathbb{R}^N$.

Exercise VIII-D.8. Consider the function $f(x) = x^3$.
   a) Is this function convex over $x \in \mathbb{R}$?
   b) Is this function convex over the interval $x \in [0, \infty)$?

Exercise VIII-D.9. Let $X \subseteq \mathbb{R}^N$ be a convex set and let $f(x), g(x)$ be convex functions over $x \in X$. Define $A = \{(u, v) \in \mathbb{R}^2 | (u, v) \geq (g(x), f(x))$ for some $x \in X\}$ (where inequality is taken entrywise). Prove that $A$ is a convex set.

Exercise VIII-D.10. (Convex functions are minimized at extreme points) Let $X$ be a (possibly non-convex) subset of $\mathbb{R}^N$, and let $f(x)$ be a concave function defined over $x \in \text{Conv}(X)$. Suppose $x^*$ minimizes $f(x)$ over all $x \in \text{Conv}(X)$.
   a) Use the definition of $\text{Conv}(X)$ to verify that $x^* = \sum_{i=1}^{k} \theta_i x_i$ for some positive integer $k$, some positive values $\theta_1, \ldots, \theta_k$ that sum to 1, and for some vectors $x_1, \ldots, x_k$ that satisfy $x_i \in X$ for all $i \in \{1, \ldots, k\}$.
   b) Use the fact that $x^*$ solves the minimization problem to conclude that $f(x^*) \leq f(x_i)$ for all $i \in \{1, \ldots, k\}$.
   c) Show that $f(x_i) = f(x^*)$ for all $i \in \{1, \ldots, k\}$. Thus, the minimum of the concave function $f(x)$ over $x \in \text{Conv}(X)$ can be achieved by a point in the set $X$ itself.
   d) Show that if $f(x)$ is strictly concave over $x \in \text{Conv}(X)$, then $x^* \in X$. Thus, all solutions that minimize the strictly concave function $f(x)$ over $x \in \text{Conv}(X)$ must be in the set $X$ itself.

Exercise VIII-D.11. Let $X$ be a finite set of 2-dimensional points: $X = \{(0, 0), (1, 2), (2, 1.5), (-1, 3), (5, 5)\}$.
   a) Plot $\text{Conv}(X)$.
   b) Find the minimum of $f(x, y) = 3x + 2y$ over $(x, y) \in \text{Conv}(X)$ (you can use the result of Exercise VIII-D.10).
   c) Repeat for minimizing $f(x, y) = -x + y$ over $(x, y) \in \text{Conv}(X)$.

Exercise VIII-D.12. Let $f(y)$ be a convex function from $\mathbb{R}$ to $\mathbb{R}$. Let $a_0, a_1, \ldots, a_N$ be a collection of $N + 1$ real numbers. Let $x = (x_1, \ldots, x_N)$. Prove that $g(x) = f\left(a_0 + \sum_{i=1}^{N} a_i x_i\right)$ is a convex function from $\mathbb{R}^N$ to $\mathbb{R}$.

Exercise VIII-D.13. Let $f(y)$ be a convex and nondecreasing function from $\mathbb{R}$ to $\mathbb{R}$, so that $f(y_1) \leq f(y_2)$ whenever $y_1 \leq y_2$. Let $X$ be a convex subset of $\mathbb{R}^N$ and let $g(x)$ be a convex function from $X$ to $\mathbb{R}$. Define $h(x) = f(g(x))$. Show that $h(x)$ is a convex function from $X$ to $\mathbb{R}$.

Exercise VIII-D.14. (Proof of Jensen’s inequality) Let $Z$ be a convex subset of $\mathbb{R}^M$, where $M$ is a positive integer. Let $Z$ be a random vector that takes values in $Z$ and has finite expectation $E[Z]$. Lemma IV.7 ensures that $E[Z] \in Z$. This fact is used to prove Jensen’s inequality. Let $X$ be a convex subset of $\mathbb{R}^N$ (where $N$ is a positive integer). Let $f(x)$ be a convex function from $X$ to $\mathbb{R}$. Let $X$ be a random vector that takes values in $X$. We want to show $f\left(E[X]\right) \leq E\left[f(X)\right]$.
   a) Define $Z = \{(x, y) \in \mathbb{R}^{N+1} | x \in X$ and $y \geq f(x)\}$. The set $Z$ is called the epigraph of the function $f(x)$ over $x \in X$. Show that $Z$ is a convex subset of $\mathbb{R}^{N+1}$.
   b) Argue that $(X, f(X)) \in Z$ for all realizations of the random vector $X$. Conclude that $f\left(E[X]\right) \leq E\left[f(X)\right]$.

Exercise VIII-D.15. (The difference between a convex set and a convex function) Define $f(x) = x^2$ for all $x \in \mathbb{R}$.
   a) Define $A = \{(x, f(x)) \in \mathbb{R}^2 | x \in \mathbb{R}\}$. Is $A$ a convex set?
b) Is \( f(x) \) a convex function over \( x \in \mathbb{R} \)?

c) Define \( B = \{(x,y) \in \mathbb{R}^2 | y \geq f(x)\} \). Is \( B \) a convex set?

**Exercise VIII-D.16.** (Convexity of the tradeoff function for a convex program) Let \( X \) be a convex and compact subset of \( \mathbb{R}^N \) and let \( f(x), g_1(x), \ldots, g_K(x) \) be convex functions over \( x \in X \). Define \( C \) as the subset of all vectors \( c = (c_1, \ldots, c_K) \in \mathbb{R}^K \) such that the problem of finding \( x \in X \) to minimize \( f(x) \) subject to \( g_k(x) \leq c_k \) for all \( k \in \{1, \ldots, K\} \) is feasible. For all \( c \in C \), define \( \psi(c) \) as the optimal value of the objective function \( f(x) \) in this convex program (with constraint constants \( c_1, \ldots, c_K \)).

\[ \psi(c) = \min_{x \in X} f(x) \text{ subject to } g_k(x) \leq c_k \text{ for all } k \in \{1, \ldots, K\} \]

\[ \psi(c) = \min_{x \in X} f(x) \]

\[ \text{for } c \in C \]

a) Show that \( C \) is a convex set.

b) Show that \( \psi(c) \) is a convex function over \( c \in C \).

**Exercise VIII-D.17.** (Comparisons via Jensen’s inequality) Let \( X \) be a convex subset of \( \mathbb{R}^N \) and let \( X \) be a random vector that takes values in \( X \). Use Jensen’s inequality to compare \( \mathbb{E} \|X\|^2 \), \( \mathbb{E} \|X\|^2 \), and \( \|\mathbb{E}X\|^2 \) using a chain of inequalities of the form \( a \leq b \leq c \). Explicitly state which convex functions are used to establish each inequality.

**Exercise VIII-D.18.** (Minimizing convex functions of one variable) This problem proves Lemma IV.2. Suppose \( f : \mathbb{R} \to \mathbb{R} \) is a convex function and \( y^* \) is a minimizer of \( f \) over \( x \in \mathbb{R} \). Consider any interval \([a, b]\). We want to show the projection \([y^*]_a^b \) minimizes \( f \) over \( x \in [a, b] \).

a) Suppose \( y^* \in [a, b] \). Prove the result in this case.

b) Suppose \( y^* > b \). Prove the result in this case. Hint: Suppose a minimum of \( f \) over \([a, b]\) occurs at a point \( x^* \) such that \( x^* < b \). Draw a picture to get intuition.

c) State the remaining case. Which of the previous two cases is this remaining case similar to?

**Exercise VIII-D.19.** (Using the above result) Fix \( \mu \geq 0 \). Use Lemma IV.2 (also stated in the above problem) to minimize \( f(x) = x^2 - \mu x \) over \( x \in [0, 1] \). Show the result is the same as given in (22).

**Exercise VIII-D.20.** (Positive semidefinite condition for convexity) Let \( f : \mathbb{R}^N \to \mathbb{R} \) be a twice differentiable function such that \( \nabla^2 f(x) \) is positive semidefinite for all \( x \in X \). We want to show \( f \) is convex.

a) Fix \( x, y \in \mathbb{R}^N \) and define \( h : [0, 1] \to \mathbb{R} \) by \( h(\theta) = f(\theta x + (1 - \theta)y) \). Show that \( h''(\theta) = (x - y)^T \nabla^2 f(\theta x + (1 - \theta)y)(x - y) \).

b) Conclude that \( h \) is convex and \( h(\theta) \leq \theta h(1) + (1 - \theta) h(0) \) for all \( \theta \in [0, 1] \). Conclude that \( f \) is convex.

**Exercise VIII-D.21.** (Convexity of \( x^2/y \)) Show that the function \( f : \mathbb{R} \times (0, \infty) \to \mathbb{R} \) given by \( f(x, y) = x^2/y \) is a convex function. Hint: Show the second derivative matrix \( \nabla^2 f \) is positive semidefinite for all \( x \in \mathbb{R}, y > 0 \).

**Exercise VIII-D.22.** (Convexity of linear combinations of convex sets) Let \( X \) and \( Y \) be convex subsets of \( \mathbb{R}^N \). Let \( a, b \) be given real numbers. Define \( C = \{w \in \mathbb{R}^N : w = ax + by \text{ for some } x \in X, y \in Y\} \). Show that \( C \) is a convex set.

**Exercise VIII-D.23.** (Hyperplane separation of disjoint convex sets) The hyperplane separation theorem implies that if \( X \) is a convex set and \( x^* \) is a point not in \( X \), then there is a hyperplane that passes through \( x^* \) that contains \( X \) on one side. That is, there is a nonzero vector \( a \) such that \( a^T x \geq a^T x^* \) for all \( x \in X \). We want to use this to show there is also a hyperplane that separates any two disjoint convex sets \( X \) and \( Y \).

a) Suppose \( X \) and \( Y \) are nonempty disjoint convex subsets of \( \mathbb{R}^N \). Show there is a nonzero vector \( a \in \mathbb{R}^N \) such that \( a^T x \geq a^T y \) for all \( x \in X, y \in Y \). Hint: Show \( \{x - y : x \in X, y \in Y\} \) is convex and does not contain the zero vector.

b) Define \( b = \inf_{x \in X} a^T x \). Show that \( b \) is finite and \( a^T x \geq b, a^T y \leq b \) for all \( x \in X, y \in Y \). Hence, the hyperplane \( \{x \in \mathbb{R}^N : a^T x = b\} \) separates \( X \) and \( Y \) so that all points of \( X \) are on or above, and all points of \( Y \) are on or below.

**E. Convex programs**

**Exercise VIII-E.1.** (Cost-aware multihop routing)

Consider the network of Fig. 14 with the capacities written on the links. There are two input streams with given rates \( \lambda_1 \) and \( \lambda_2 \). The rate \( \lambda_1 \) must be split over the top and bottom paths. Let \( x \) and \( y \) be the amount used for the top and bottom paths,
respectively, so that $x$ and $y$ are non-negative and satisfy $x + y = \lambda_1$. The problem is to route the data over the network so that the link capacities are not exceeded and total sum cost is minimized. Assume that the cost function for the link $(i, j)$ with total flow rate $r_{ij}$ is equal to $\Phi_{ij}(r_{ij}) = e^{|r_{ij}|} - 1$. Assume that $\lambda_1, \lambda_2$ are non-negative and satisfy $\lambda_1 + \lambda_2 \leq 6$ and $\lambda_2 \leq 3$.

a) Write down a convex program, with the corresponding constraints, to minimize the sum of costs. The decision variables are $x$ and $y$.

b) In the 2-d plane, plot the region of all non-negative $(\lambda_1, \lambda_2)$ vectors that satisfy $\lambda_1 + \lambda_2 \leq 6$ and $\lambda_2 \leq 3$. Argue that this region is exactly the set of all rate vectors $(\lambda_1, \lambda_2)$ that the network can support, called the network capacity region. Hint: First prove that if one of the constraints is violated, then it is impossible to support the traffic over the network. Next, prove that if both constraints are satisfied, then there is a way to split $\lambda_1$ into components $x$ and $y$ such that the total flow is supportable over the network.

c) Suppose we want both $\lambda_1$ and $\lambda_2$ to be large. However, we require $(\lambda_1, \lambda_2)$ to be in the network capacity region specified by part (b). State the set of all Pareto optimal vectors $(\lambda_1, \lambda_2)$ (where Pareto optimality here is in terms of desiring both coordinates to be large, rather than small). Over all Pareto optimal vectors $(\lambda_1, \lambda_2)$, which vector $(\lambda_1^*, \lambda_2^*)$ maximizes $2\log(\lambda_1) + \log(\lambda_2)$? Which maximizes $\log(\lambda_1) + \log(\lambda_2)$?

Exercise VIII-E.2. (Network fairness) Consider three flows:

- Flow 1 uses links 1 and 2.
- Flow 2 uses links 2 and 3.
- Flow 3 uses link 3.

Links 1 and 3 have capacity 1, and link 2 has capacity 0.8. Let $x_i$ be the rate of flow $i$. The optimization problem is:

\[
\begin{align*}
\text{Maximize: } & \phi_1(x_1) + \phi_2(x_2) + \phi_3(x_3) \\
\text{Subject to: } & x_1 \leq 1 \\
& x_1 + x_2 \leq 0.8 \\
& x_2 + x_3 \leq 1 \\
& x_i \geq 0 \text{ for all } i \in \{1, 2, 3\}
\end{align*}
\]

where $\phi_i(x_i)$ is a concave and non-decreasing utility function for flow $i$.

a) Argue that, for optimality, we can assume that $x_1 = 0.8 - x_2$, $x_3 = 1 - x_2$, and $x_2 \in [0, 0.8]$. This reduces the problem to a calculus problem of optimizing a function of a single variable $x_2$ over an interval.

b) Solve for the case $\phi_i(x) = \log(x)$ for all $i \in \{1, 2, 3\}$. This is proportional fairness.

c) Solve for the case $\phi_i(x) = x$ for all $i \in \{1, 2, 3\}$. This seeks to maximize the sum rate.

d) Compare and discuss the fairness for the utility functions in parts (b) and (c).

Exercise VIII-E.3. (Importance of nonnegativity constraints) Consider the same system as Exercise VIII-E.2. However, suppose that:

- We use $\phi_i(x) = x$ for all $i \in \{1, 3\}$ and $\phi_2(x) = 5x$. (so we seek to maximize $x_1 + 5x_2 + x_3$).
- We neglect the constraints.

Show that the maximum value of the objective function is infinity. That is, for any arbitrarily large number $M$, there are vectors $(x_1, x_2, x_3) \in \mathbb{R}^3$ that satisfy \eqref{146} \eqref{147} and yield $x_1 + 5x_2 + x_3 \geq M$.

Exercise VIII-E.4. (Linear program for server scheduling) Consider a network with three links that operate in discrete time $t \in \{0, 1, 2, \ldots\}$. Only two links can be activated per slot. Every slot $t$, the network controller must choose which two links to activate. An active link transmits one packet per slot. An idle link transmits no packets. Thus, the decision can be viewed as a selection of a transmission vector $b(t) = (b_1(t), b_2(t), b_3(t))$ in the set $B = \{(1, 1, 0), (1, 0, 1), (0, 1, 1)\}$. Let $p_1$ be the fraction of time that $b(t) = (1, 1, 0)$, let $p_2$ be the fraction of time that $b(t) = (1, 0, 1)$, and let $p_3$ be the fraction of time that $b(t) = (0, 1, 1)$. Let $b_i$ be the time average transmission rate on link $i$.

a) Write a linear program (with no optimization objective, that is, with the objective of minimizing the function 1) to find variables $p_1, p_2, p_3$ to ensure $b_i \geq \lambda_i$, where $\lambda_1, \lambda_2, \lambda_3$ are a given set of non-negative numbers. Be sure to include the constraints $p_i \geq 0$ for all $i$, and $p_1 + p_2 + p_3 = 1$.

b) Solve the problem by hand (by intuition and/or by trial-and-error) for the case $(\lambda_1, \lambda_2, \lambda_3) = (2/3, 2/3, 2/3)$. Then solve for the case $(\lambda_1, \lambda_2, \lambda_3) = (1, 1/2, 1/2)$.

c) Solve the problem by hand (by intuition and/or by trial-and-error) for the case $(\lambda_1, \lambda_2, \lambda_3) = (3/8, 7/8, 6/8)$.

d) Solve the problem by hand (by intuition and/or by trial-and-error) for the case $(\lambda_1, \lambda_2, \lambda_3) = (0.5, 0.6, 0.7)$.

e) Prove that it is impossible to solve the problem if $\lambda_1 + \lambda_2 + \lambda_3 > 2$.

Exercise VIII-E.5. (Feasibility of server scheduling) For the same system as Exercise VIII-E.4:

a) Prove that it is impossible to solve the linear program if there is an $i \in \{1, 2, 3\}$ such that $\lambda_i > 1$.

b) Prove that it is impossible to solve the linear program if $\lambda_1 + \lambda_2 + \lambda_3 > 2$. 
Exercise VIII-E.6. (Linear programs for generalized scheduling) Consider a generalized scheduling system with \( N \) links. Every timeslot \( t \in \{0, 1, 2, \ldots \} \) the system controller chooses a transmission rate vector \( b(t) = (b_1(t), \ldots, b_N(t)) \) subject to the constraint \( b(t) \in \mathcal{B} \), where \( \mathcal{B} \) is a finite set of transmission rate vectors. Specifically, assume that \( \mathcal{B} = \{r^{(1)}, r^{(2)}, \ldots, r^{(K)}\} \), where \( r^{(k)} \) is the \( k \)th vector in the set \( \mathcal{B} \) and has components \( r^{(k)} = (r_1^{(k)}, \ldots, r_N^{(k)}) \), for \( k = 1, \ldots, K \). Let \( p_k \) be the fraction of time that \( r^{(k)} \) is used. Write a linear program with no optimization objective (that is, with the objective of minimizing the function \( 1) \) to find values \( p_1, \ldots, p_K \) that ensure the time average transmission rates satisfy \( b_i \geq \lambda_i \) for \( i \in \{1, \ldots, N\} \), where \( \lambda_1, \ldots, \lambda_N \) are a given set of non-negative numbers.

Exercise VIII-E.7. (Quality-aware video over a network) Suppose a network must support \( N \) video streams. Let \( x_i \) be the rate of video stream \( i \in \{1, \ldots, N\} \). Suppose the rates can be chosen in an interval \( x_i \in [0, x_{\text{max}}] \), which affects the distortion of the video through a non-negative and convex rate-distortion function \( d_i(x_i) \) for each \( i \in \{1, \ldots, N\} \). The goal is to minimize \( \sum_{i=1}^{N} (d_i(x_i))^2 \) subject to the rates \( \{x_1, \ldots, x_N\} \) being supportable on the network. Let \( \mathcal{L} \) be the set of network links. Assume each video \( i \in \{1, \ldots, N\} \) takes a path \( \mathcal{P}_i \) that involves a collection of links, and the capacity available for serving video streams on a given link \( l \) is equal to \( C_l \).

(a) Write the resulting convex program that seeks to find the optimal \( \{x_1^*, \ldots, x_N^*\} \).

(b) Suppose the first two links \( l = 1 \) and \( l = 2 \) have capacities that depend on the power allocated to them. Let \( C_1(p_1) \) and \( C_2(p_2) \) be the capacities, assumed to be concave functions of the non-negative power variables \( p_1, p_2 \). Write a new convex program that incorporates the constraint \( p_1 + p_2 \leq 1 \), to find optimal values \( \{x_1^*, \ldots, x_N^*, p_1^*, p_2^*\} \).

F. Drift-plus-penalty applications

Exercise VIII-F.1. (Drift-plus-penalty for linear programs) Consider the linear program:

\[
\begin{align*}
\text{Minimize:} & & \sum_{i=1}^{N} c_i x_i \\
\text{subject to:} & & \sum_{i=1}^{N} a_{ki} x_i \leq b_k \quad \forall k \in \{1, \ldots, K\} \\
& & x_i \in [x_{i,\text{min}}, x_{i,\text{max}}] \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\]

where \( c_i, a_{ki}, b_k, x_{i,\text{min}}, x_{i,\text{max}} \) are given constants. Define \( \mathcal{X} = \{x \in \mathbb{R}^N | x_{i,\text{min}} \leq x_i \leq x_{i,\text{max}} \ \forall i \in \{1, \ldots, N\}\} \).

(a) Specify the virtual queues \( Q_k(t) \) for \( k \in \{1, \ldots, K\} \) used by the drift-plus-penalty algorithm.

(b) Specify the decisions \( x_i(t) \) made by the drift-plus-penalty algorithm (using a parameter \( V \geq 0 \)). You should find that \( x_i(t) \) is either \( x_{i,\text{min}} \) or \( x_{i,\text{max}} \) for all \( t \), depending on a simple threshold comparison. Theorem VII ensures the time averages converge to an \( O(\epsilon) \)-approximation, where \( \epsilon = 1/V \).

Exercise VIII-F.2. (Drift-plus-penalty for quality-aware video) Consider the same network of Exercise VIII-E.7b, where there are \( N \) video streams with rates \( \{x_1, \ldots, x_N\} \) for \( 0 \leq x_i \leq x_{\text{max}} \), non-negative and convex distortion functions \( d_i(x_i) \) and paths \( \mathcal{P}_i \) for each video \( i \in \{1, \ldots, N\} \), link capacities \( C_l \) for each link \( l \in \mathcal{L} \), and two special links 1 and 2 with concave capacity functions \( C_1(p_1) \), \( C_2(p_2) \) with power subject to \( p_1 + p_2 \leq 1 \) and \( p_1 \geq 0, p_2 \geq 0 \).

(a) Write the virtual queues.

(b) State the drift-plus-penalty algorithm, emphasizing the separable structure to solve for the \( x_i(t) \) variables and \( p_1(t), p_2(t) \) variables.

Exercise VIII-F.3. (Optimizing network flows)

\[
\begin{align*}
x & \xrightarrow{\text{Link 1}} \log(1+p_1) \\
\text{Link 2} & \xrightarrow{\text{Log(1+p_2)}} y \\
z & \xrightarrow{\text{Log(1+p_2)}}
\end{align*}
\]

Fig. 15. A network with three flows \( x, y, z \) and a power constraint \( p_1 + p_2 \leq \beta \), for Exercise VIII-F.3.

Consider the network with three flows with flow rates \( x, y, z \), as shown in Fig. 15. Flow \( x \) uses link 1. Flows \( y \) and \( z \) use link 2. The transmission rates of links 1 and 2 depend on power variables \( p_1 \) and \( p_2 \). There is a power constraint \( p_1 + p_2 \leq \beta \), for some constant \( \beta \). We want to use the drift-plus-penalty algorithm to solve the following:

\[
\begin{align*}
\text{Maximize:} & & \log(x) + \log(y) + \log(z) \\
\text{Subject to:} & & x \leq \log(1+p_1) , \ y + z \leq \log(1+p_2) \\
& & p_1 + p_2 \leq \beta \\
& & 0 \leq x \leq 1 , \ 0 \leq y \leq 1 , \ 0 \leq z \leq 1 \\
& & 0 \leq p_i \leq \beta \ \forall i \in \{1, 2\}
\end{align*}
\]
Assume $X$ is the set of all $(x, y, z, p_1, p_2)$ that satisfy (153), (154).

a) Write the virtual queues.

b) State the drift-plus-penalty algorithm. Be sure to take advantage of any separable structure. Specify the exact choices for your variables $x(t), y(t), z(t), p_1(t), p_2(t)$ for each slot $t \in \{0, 1, 2, \ldots\}$.

Exercise VIII-F.4. (Choosing a different $X$ set) Again solve Exercise VIII-F.3, but use $X$ as the set of all $(x, y, z, p_1, p_2)$ that satisfy (152), (154). What are the advantages and disadvantages of this approach?

Exercise VIII-F.5. (Deterministically bounded queues for a convex program) Consider the problem (84)–(87) and the corresponding drift-plus-penalty algorithm of Section VI-D.

a) Use the $y(t)$ selection rule in (91) to show that $x(t) + y(t) \geq 4$ whenever $Q_1(t)/(2V) \geq 4$. Find a threshold $\beta_1 V$ (for some constant $\beta_1 > 0$) such that $Q_1(t)$ in (88) cannot further increase once it exceeds $\beta_1 V$. Conclude that $Q_1(t) \leq \beta_1 V + 4$ for all $t$.

b) Use the $y(t)$ selection rule in (91) to show that $x(t) + 3y(t) \geq 6$ whenever $9Q_2(t)/(2V) \geq 6$. Find a threshold $\beta_2 V$ (for some constant $\beta_2 > 0$) such that $Q_2(t)$ in (89) cannot further increase once it exceeds $\beta_2 V$. Conclude that $Q_2(t) \leq \beta_2 V + 6$ for all $t$.

c) Substitute these deterministic queue bounds into Lemma VI.1 to show that for all $t \in \{1, 2, 3, \ldots\}$ we have:

$$-\pi(t) - y(t) \leq -4 + (\beta_1 V + 4)/t$$
$$-\pi(t) - 3y(t) \leq -6 + (\beta_2 V + 6)/t$$

and hence the constraints are arbitrarily close to being satisfied when $t$ is sufficiently large relative to $V$.

Exercise VIII-F.6. (Flow control with path options) Modify the flow control algorithm of Section VII-A to allow each flow $i$ to have a choice of two paths $P_i^+$ and $P_i^-$. Assume the objective function is $\sum_{i=1}^{N} \log(1 + x_i)$, where $x_i$ is the total rate achieved by flow $i$. You can assume that $a_i$ and $b_i$ are the rates over path $a$ and path $b$, respectively, for each flow $i \in \{1, \ldots, N\}$.

Specifically:

a) Write the corresponding convex program.

b) State the virtual queues.

c) State the drift-plus-penalty algorithm (with parameter $V > 0$), emphasizing distributed and separable implementation wherever possible.

There are two approaches to this problem that will give slightly different algorithms (you can use either one you want):

• Approach 1: Seek to maximize $\sum_{i=1}^{N} \log(1 + x_i)$ subject to constraints $x_i \leq a_i + b_i$ for all $i$, and subject to additional network constraints on $a_i$ and $b_i$.

• Approach 2: Seek to maximize $\sum_{i=1}^{N} \log(1 + a_i + b_i)$ subject to network constraints on $a_i$ and $b_i$.

Exercise VIII-F.7. (Drift-plus-penalty for one link) Derive the drift-plus-penalty algorithm to solve the problem of Exercise VIII-C.72.

Specifically, we want to solve:

Maximize: $\phi(x_1, \ldots, x_N)$

Subject to: $\sum_{i=1}^{N} x_i \leq C$

$0 \leq x_i \leq C \quad \forall i \in \{1, \ldots, N\}$

where the last constraint has been modified (without affecting the solution) to ensure optimization is over the compact set $\Omega = \{(x_1, \ldots, x_N) \in \mathbb{R}^N | 0 \leq x_i \leq C \quad \forall i \in \{1, \ldots, N\}\}$. Recall that:

$$\phi(x_1, \ldots, x_N) = \sum_{i=1}^{N} \frac{\theta_i}{b} \log(1 + bx_i)$$

where $\theta_i$ are given positive priority weights for users $i \in \{1, \ldots, N\}$, and $b$ is a given positive constant.

a) Write the virtual queue equation. Argue that since $x_i(t) \leq C$ for all $t$, the queue can increase by at most $(N - 1)C$ on any slot.

b) Write an exact formula for $x_i(t)$ for each $i \in \{1, \ldots, N\}$ in terms of $V, Q(t), \theta_i, b, C$.

c) Assume $Q(0) = 0$. Define $\theta_{\max} = \max_{i \in \{1, \ldots, N\}} \theta_i$. Prove that $Q(t) \leq V\theta_{\max} + (N - 1)C$ for all $t \in \{0, 1, 2, \ldots\}$.

d) Define $\pi_i(T) = \frac{1}{T} \sum_{\tau=0}^{T-1} x_i(\tau)$ as the resulting time average admission rate of user $i$ over the first $T$ slots, where $T$ is a positive integer. Prove that $\sum_{i=1}^{N} \pi_i(T) \leq C + (V\theta_{\max} + (N - 1)C)/T$ for all $T > 0$.

e) Define $\text{util}(T) = \phi(\pi_1(T), \ldots, \pi_N(T))$. It can be shown that $\text{util}(T) \geq \text{util}^{\text{opt}} - O(1/V)$. Discuss the tradeoff with $V$ in relation to the results of parts (c)-(d).

Exercise VIII-F.8. (Simulation of drift-plus-penalty) Write a computer program to simulate the above algorithm (from Exercise VIII-F.7) for $N = 3, C = 1, b = 5$ over $T = \lceil (V + 1)10^5 \rceil$ time slots. Define $Q_{\max}(T) = \max_{\tau \in \{0, \ldots, T-1\}} Q(\tau)$ as the
maximum observed queue size over the first $T$ slots. In this simulation, all numbers should be written with at least 4 or 5 significant digits.

a) Compute $(\overline{x}_1(T), \overline{x}_2(T), \overline{x}_3(T))$ and $\text{util}(T)$ for the case $V = 10$, $\theta_1 = 1, \theta_2 = \theta_3 = 5$. Compare to the exact answer from Exercise VIII-C.12.

b) Compute $(\overline{x}_1(T), \overline{x}_2(T), \overline{x}_3(T))$ and $\text{util}(T)$ for the case $V = 10$, $\theta_1 = 1, \theta_2 = \theta_3 = 2$. Compare to the exact answer from Exercise VIII-C.12.

c) Fix $\theta_1 = 1, \theta_2 = \theta_3 = 2$. Make two plots: One for $Q_{\text{max}}(T)$ versus $V$, another for $\text{util}(T)$ versus $V$, for data points taken with $V \in \{0, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 2, 5, 10, 20\}$.

Exercise VIII-F.9. (Drift-plus-penalty for joint routing and power allocation)

![Diagram of a network with 3 links and nodes labeled x, y, r1, r2, r3](image)

Fig. 16. The 3 link network for Exercise VIII-F.9 The transmission rate on each link $i \in \{1, 2, 3\}$ is $\log(1 + p_i)$, where $p_i$ is the power used on link $i$.

Consider the problem of joint routing and power allocation in the network of Fig. 16. Given constants $P_{\text{max}}, r_1, r_2, r_3$, we want to solve find routing variables $x, y$ and power variables $p_i$ for $i \in \{1, 2, 3\}$

to solve:

\[
\begin{align*}
\text{Minimize:} & \quad p_1 + p_2 + p_3 \\
\text{Subject to:} & \quad x + r_1 \leq \log(1 + p_1) \\
& \quad x + r_2 \leq \log(1 + p_2) \\
& \quad y + r_3 \leq \log(1 + p_3) \\
& \quad x + y \geq 8.7 \\
& \quad 0 \leq x \leq 10, \quad 0 \leq y \leq 10, \quad 0 \leq p_i \leq P_{\text{max}} \forall i \in \{1, 2, 3\}
\end{align*}
\]

a) Give the virtual queues.

b) Give the drift-plus-penalty algorithm. You must specify values chosen for each variable on each slot $t$.

Exercise VIII-F.10. (Link weight scaling) Consider the flow optimization problem of Section VII-A. Let $\gamma_1, \ldots, \gamma_L$ be positive numbers, and consider the problem:

\[
\begin{align*}
\text{Maximize:} & \quad \sum_{i=1}^{N} \phi_i(x_i) \\
\text{Subject to:} & \quad \sum_{i \in \mathcal{N}(l)} \gamma_i x_i \leq \gamma_i C_l \quad \forall l \in \{1, \ldots, L\} \\
& \quad x_i \in [0, x_{\text{max}}] \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\]

a) Argue that the above problem is equivalent to the problem (102)-(104).

b) State the drift-plus-penalty algorithm for this problem. This shows that any constant multiple of $Q_l(t)$ can be used.

Exercise VIII-F.11. (C-additive approximation of drift-plus-penalty) Suppose that it is difficult to choose $x(t) \in \mathcal{X}$ to minimize the right-hand-side of (76). Instead, let $C$ be a non-negative constant, and assume that every slot $t$ the decision $x(t) \in \mathcal{X}$ is made to ensure:

\[
V f(x(t)) + \sum_{k=1}^{K} Q_k(t)g_k(x(t)) \leq C + V f(z) + \sum_{k=1}^{K} Q_k(t)g_k(z) \quad \forall z \in \mathcal{X}
\]

If $C = 0$, then the $x(t)$ decision is exactly as specified by the drift-plus-penalty algorithm. If $C > 0$, the decision is called a $C$-additive approximation.

a) Modify equations (82)-(83) in the proof of Theorem VII.7 using this new assumption. Carry out the rest of the proof (with these modifications) to conclude that $f(\pi(t)) \leq f^* + (B + C)/V$.

b) Fix $\epsilon > 0$. What value of $V$ is needed to ensure that $f(\pi(t)) \leq f^* + \epsilon$?
Exercise VIII-F.12. (Time average optimization) Consider the 4 node network of Fig. 17. Traffic of rate $\lambda_1$ and $\lambda_2$ enters nodes 1 and 2, respectively, and must be routed to node 4. There are three links \{a, b, c\}. Every timeslot $t \in \{0, 1, 2, \ldots\}$ a transmission rate vector $\mu(t) = (\mu_a(t), \mu_b(t), \mu_c(t))$ is allocated within a finite set $\Gamma$ of possible transmission rate vectors, so that $\mu_a(t)$ is the service rate available on slot $t$ over link $a$, $\mu_b(t)$ is the available service rate at link $b$, and so on. We want to choose $\mu(t)$ over slots to satisfy the following time average inequality constraints:

\[
\begin{align*}
\lambda_1 & \leq \lim_{t \to \infty} \mu_a(t) \\
\lambda_2 & \leq \lim_{t \to \infty} \mu_b(t) \\
\lambda_1 + \lambda_2 & \leq \lim_{t \to \infty} \mu_c(t) \\
\mu(t) & \in \Gamma, \quad \forall t \in \{0, 1, 2, \ldots\}
\end{align*}
\]

Assume $\lambda_1$ and $\lambda_2$ are known constants. Use the drift-plus-penalty algorithm with 3 virtual queues and $V = 0$ to solve the problem.

Exercise VIII-F.13. (Single commodity backpressure) Consider the 4 node network of Fig. 17 with two flows that want to reach the destination node 4. Suppose the links a, b, c have fixed capacities $C_a, C_b, C_c$. Let $f_a, f_b, f_c$ be flow variables associated with links a, b, c. We want to allocate transport layer admission rates $(\lambda_1, \lambda_2)$ and flow variables $f_a, f_b, f_c$ to solve:

\[
\begin{align*}
\text{Maximize:} & \quad \theta_1 \log(1 + \lambda_1) + \theta_2 \log(1 + \lambda_2) \\
\text{Subject to:} & \quad \lambda_1 \leq f_a \\
& \quad \lambda_2 \leq f_b \\
& \quad f_a + f_b \leq f_c \\
& \quad f_a \in [0, C_a], f_b \in [0, C_b], f_c \in [0, C_c] \\
& \quad \lambda_1 \in [0, C_a], \lambda_2 \in [0, C_b] \quad (159) \\
& \quad \lambda_1 \in [0, C_a], \lambda_2 \in [0, C_b] \quad (160)
\end{align*}
\]

where $\theta_1$ and $\theta_2$ are given positive weights. Define $X$ as the set of all $(\lambda_1, \lambda_2, f_a, f_b, f_c)$ that satisfy (159)-(160). Use the drift-plus-penalty algorithm with $V > 0$ and three virtual queues $Q_1(t), Q_2(t), Q_3(t)$ to solve the problem. Is your resulting answer for $f_c(t)$ surprising? Discuss what happens with the $f_1(t)$ and $f_2(t)$ decisions in terms of the backpressure concept.

Exercise VIII-F.14. (Shrinking the time slot) Consider the flow optimization example of Section VII-A. However, suppose timeslots are are shrunk by a factor of $\delta$, where $0 < \delta < 1$, so that the per-slot capacity of each link becomes $C_i \delta$ and the admitted data is $x_i(t)\delta$. Let $\tilde{Q}_i(t)$ be the new queue values for this system. Still assume that $t \in \{0, 1, 2, \ldots\}$, with the understanding that the modified slot is now shorter than before. The underlying convex program is:

\[
\begin{align*}
\text{Maximize:} & \quad \sum_{i=1}^{N} \phi_i(x_i) \\
\text{Subject to:} & \quad \sum_{i \in N(t)} x_i \delta \leq C_i \delta \quad \forall i \in \{1, \ldots, L\} \\
& \quad x_i \in [0, x_{\text{max}}] \quad \forall i \in \{1, \ldots, N\}
\end{align*}
\]

With this modification, the equation (105) is changed to the following:

\[
\tilde{Q}_i(t + 1) = \max \left[ \tilde{Q}_i(t) + \sum_{i \in N(t)} x_i(t) \delta - C_i \delta, 0 \right]
\]

Let $\tilde{V}$ be the parameter of the drift-plus-penalty algorithm under this modification, so that $x(t)$ is chosen in $[0, x_{\text{max}}]$ to maximize $\phi_i(x_i(t) - x_i(t)) \sum_{i \in P(i)} \tilde{Q}_i(t)$. Suppose $Q_i(t)$ are the queue values under the original algorithm (without timeslot scaling) with parameter $V$. Assume that $\tilde{V} = \delta^2 V$, and $\tilde{Q}_i(0) = \delta Q_i(0)$.

a) Show that $\tilde{Q}_i(t) = \delta Q_i(t)$ for all $t \in \{0, 1, 2, \ldots\}$, and that both algorithms make exactly the same decisions for $x_i(t)$. Thus, the same data rates are achieved with queue values that are shrunk by a factor of $\delta$. 

Fig. 17. The 4 node network for Exercises VIII-F.12 and VIII-F.13
b) In the special case $\phi_i(x) = (\theta_i / b) \log(1 + bx)$, use (107) to prove that $\hat{Q}_i(t) \leq \delta[V_{\theta_{\text{max}}}/b + x_{\text{max}}]$ for all $t \in \{0, 1, 2, \ldots \}$, so that queues are arbitrarily small as $\delta \to 0$.

**Exercise VIII-F.15.** (Distributed optimization of a non-separable problem [8]) Consider a system with $N$ devices. Each device must choose its own variable $x_i \in [0, 1]$ to solve:

Minimize: $\sum_{i=1}^{N} f_i(x_i, \theta)$

Subject to: $g_i(x_i, \theta) \leq c_i \forall i \in \{1, \ldots, N\}$

$x_i \in [0, 1] \forall i \in \{1, \ldots, N\}$

$\theta \in [0, 1]$ where $\theta$ is a variable that must be collectively chosen in the interval $[0, 1]$, and $f_i(x_i, \theta)$ and $g_i(x_i, \theta)$ are convex functions from $[0, 1] \times [0, 1]$ to $\mathbb{R}$. Suppose the devices are nodes of a directed graph, and two distinct devices $i$ and $j$ can communicate if and only if $(i, j)$ is a directed link in the graph. Let $\mathcal{L}$ be the set of all directed links. Consider the modified problem that introduces estimation variables $\theta_i$ for each $i \in \{1, \ldots, N\}$:

Minimize: $\sum_{i=1}^{N} f_i(x_i, \theta_i)$

Subject to: $\theta_i = \theta_j \forall (i, j) \in \mathcal{L}$

$g_i(x_i, \theta_i) \leq c_i \forall i \in \{1, \ldots, N\}$

$x_i \in [0, 1], \theta_i \in [0, 1] \forall i \in \{1, \ldots, N\}$

a) Argue that the new problem is equivalent to the old whenever the directed graph can be changed into a connected undirected graph by removing the directionality on all links. Hint: Show that $\theta_i = \theta_j$ whenever nodes $i$ and $j$ can be connected via a path in the undirected graph, regardless of the directionality of the links in the underlying directed graph.

b) Let $\mathcal{X}$ be the set of all $\{x_1, \theta_1, \ldots, x_N, \theta_N\}$ that satisfy (163)–(164). Define virtual queues $H_{ij}(t)$ for each $(i, j) \in \mathcal{L}$, and show that the algorithm can be implemented in a distributed manner where each device $i \in \{1, \ldots, N\}$ chooses $(x_i(t), \theta_i(t))$ over the set (163)–(164) to solve:

Minimize: $V f_i(x_i(t), \theta_i(t)) + \theta_i(t) \left[ \sum_{j \in N_{\text{out}}(i)} H_{ij}(t) - \sum_{k \in N_{\text{in}}(i)} H_{ki}(t) \right]$

Subject to: $x_i(t) \in [0, 1], \theta_i(t) \in [0, 1], g_i(x_i(t), \theta_i(t)) \leq c_i$

where $N_{\text{out}}(i) = \{j \in \{1, \ldots, N\} | (j, i) \in \mathcal{L}\}$ is the set of all nodes $j$ for which node $i$ has an outgoing link, and $N_{\text{in}}(i) = \{k \in \{1, \ldots, N\} | (k, i) \in \mathcal{L}\}$ is the set of nodes $k$ from which node $i$ has an incoming link. Observe that each node $i$ does not require knowledge of the $f_j(x, \theta)$ and $g_j(x, \theta)$ functions for $j \neq i$.

**Exercise VIII-F.16.** (Another distributed optimization [8]) Suppose $N$ devices each have their own variables $x_i$, for $i \in \{1, \ldots, N\}$, and must choose these variables and collectively choose another variable $\theta$ to solve:

Minimize: $\sum_{i=1}^{N} f_i(x_i, \theta)$

Subject to: $\sum_{i=1}^{N} g_i(x_i, \theta) \leq C$

$x_i \in [0, 1] \forall i \in \{1, \ldots, N\}, \theta \in [0, 1]$ for some constant $C$ and some convex functions $f_i(x, \theta)$, $g_i(x, \theta)$. As in Exercise VIII-F.15 assume the devices are nodes in a directed graph with a link set $\mathcal{L}$, and suppose that the directed graph is connected when all directional links are changed to undirected links. Consider the modified problem:

Minimize: $\sum_{i=1}^{N} f_i(x_i^{(1)}, \theta_i)$

Subject to: $\sum_{i=1}^{N} g_i(x_i^{(1)}, \theta_1) \leq C$

$\theta_i = \theta_j \forall (i, j) \in \mathcal{L}$

$x_i^{(m)} = x_i^{(n)} \forall i \in \{1, \ldots, N\}, \forall (m, n) \in \mathcal{L}$

$x_i^{(j)} \in [0, 1], \theta_i \in [0, 1] \forall i, j \in \{1, \ldots, N\}$

Argue that the modified problem is equivalent to the original. Design a distributed drift-plus-penalty algorithm for this problem, with virtual queues $Q_{ij}(t)$ for all $(i, j) \in \mathcal{L}$ and $H_{ij}^{(m,n)}(t)$ for all $i \in \{1, \ldots, N\}$ and all $(m, n) \in \mathcal{L}$. Note that node 1 will have a different decision structure than all other nodes. Argue that the problem can also be solved by (i) changing the objective function to $\sum_{i=1}^{N} f_i(x_i^{(1)}, \theta_i)$ and for (ii) removing the constraint $\sum_{i=1}^{N} g_i(x_i^{(1)}, \theta_1) \leq C$ from the set $\mathcal{X}$ and enforcing it via a virtual queue $Z(t)$ (these approaches would result in a different instantiation of the drift-plus-penalty algorithm).

**Exercise VIII-F.17.** (Summing over a tree) Consider the problem (163)–(167). Assume that the functions $f_i(\cdot)$ and $g_i(\cdot)$ take values in the interval $[0, 1]$. Suppose the set of links $\mathcal{L}$ form a directed tree rooted at node 1, so that: (i) node 1 has no outgoing
links, (ii) nodes \( \{2, \ldots, N\} \) have exactly one outgoing link, (iii) there are no cycles. Such a graph has a single path to node 1 from all other nodes. Show that this problem is equivalent to the following modified problem:

Minimize:
\[
\sum_{i=1}^{N} f_i(x_i, \theta_i) \quad \text{Subject to:}
\]
\[
S_i \leq C \quad \forall i \in \{1, \ldots, N\}
\]
\[
g_i(x_i, \theta_i) + \sum_{j \in N_{c(i)}} S_j \leq S_i \quad \forall i \in \{1, \ldots, N\}
\]
\[
\theta_i = \theta_j \quad \forall (i, j) \in L
\]
\[
x_i \in [0, 1], \theta_i \in [0, 1], y_t \in [0, 1], S_i \in [0, N] \quad \forall i \in \{1, \ldots, N\}
\]

State the drift-plus-penalty algorithm for this problem and show that each node i does not require knowledge of the functions \( f_j(x, \theta) \) and \( g_j(x, \theta) \) for \( j \neq i \).

Exercise VIII-F.18. (Bounded queues for flow-based routing) Consider the flow-based multi-path routing and flow control algorithm of Section VII-D. Suppose all utility functions \( \phi_i(x) \) are differentiable over \( x \in [0, x_{\text{max}}] \), and note that \( \phi'_i(0) \geq \phi'_i(x_{\text{max}}) \) for all \( x \in [0, x_{\text{max}}] \).

a) Show from (137) that \( \gamma_i(t) = 0 \) whenever \( Z_i(t) > V \phi'_i(0) \), and \( \gamma_i(t) = x_{\text{max}} \) whenever \( Z_i(t) < V \phi'_i(x_{\text{max}}) \).

b) Conclude from (136) that \( \max[V \phi'_i(x_{\text{max}}) - x_{\text{max}}, 0] \leq Z_i(t) - V \phi'_i(0) + x_{\text{max}} \) for all \( t \in \{0, 1, 2, \ldots\} \), provided that \( Z_i(0) \) is in this interval.

c) Use the result of part (b) to conclude from (138) and (135) that for each link \( l \in \{1, \ldots, L\} \) we have \( Q_i(t) \leq N x_{\text{max}} + \max_{i \in \{1, \ldots, N\}} [V \phi'_i(0) + x_{\text{max}}] \) for all \( t \in \{0, 1, 2, \ldots\} \), provided that this holds for slot 0.

Appendix A—Differentiable functions and the equation \( \nabla f(x) + \mu \nabla g(x) = 0 \)

Again consider the problem of choosing \( x = (x_1, \ldots, x_N) \) in a set \( \mathcal{X} \subseteq \mathbb{R}^N \) to minimize a real valued function subject to one constraint. That is, we have the problem:

Minimize:
\[
f(x) \quad \text{Subject to:}
\]
\[
g(x) \leq c \quad x \in \mathcal{X}
\]

Assume \( f(x) \) and \( g(x) \) are real-valued functions that are continuous over \( x \in \mathcal{X} \) and differentiable at all interior points of \( \mathcal{X} \). That is, \( \nabla f(x) \) and \( \nabla g(x) \) exist whenever \( x \) is an interior point. The Lagrange multiplier approach of Theorem 1.2 defines a value \( \mu \geq 0 \) and then searches for a global minimum of \( f(x) + \mu g(x) \) over \( x \in \mathcal{X} \). However, this method does not necessarily find all points \( (c, \psi(c)) \) on the tradeoff curve. A search for critical points of \( f(x) + \mu g(x) \) often reveals points on the tradeoff curve that cannot be found by a global minimization. This is particularly useful when the functions \( f(x) \) and/or \( g(x) \) are non-convex.

G. The Karush-Kuhn-Tucker necessary conditions for problems with one constraint

Suppose \( \nabla f(x) \) and \( \nabla g(x) \) exist for points \( x \) in the interior of \( \mathcal{X} \). Let \( x^* = (x^*_1, \ldots, x^*_N) \) be an optimal solution of (168)-(170). Then at least one of the following three necessary conditions must hold:

- \( x^* \) is a boundary point of \( \mathcal{X} \).
- Else, \( \nabla g(x^*) = 0 \).
- Else, \( \nabla f(x^*) + \mu \nabla g(x^*) = 0 \) for some constant \( \mu \geq 0 \) that satisfies \( (g(x^*) - c)\mu = 0 \).

If \( \mathcal{X} = \mathbb{R}^N \) then there are no boundary points. The equation \( \nabla f(x^*) + \mu \nabla g(x^*) = 0 \) is called the stationary equation. This equation must hold if \( x^* \) is an interior point that is a global or local minimum of \( f(x) + \mu g(x) \). The equation \( (g(x^*) - c)\mu = 0 \) is called the complementary slackness equation and means that either \( g(x^*) = c \) or \( \mu = 0 \). A proof of these conditions is developed in the following subsections.

H. Preliminary facts

The following facts are useful for our purposes.

- Fact 1: Let \( x^* \) be an interior point of \( \mathcal{X} \) and let \( h(x) \) be a real-valued function defined over \( x \in \mathcal{X} \). If \( h(x) \) is differentiable at \( x^* \), then for any nonzero vector \( v = (v_1, \ldots, v_N) \in \mathbb{R}^N \), there exists a value \( \delta_{\max} > 0 \) such that \( x^* + \delta v \in \mathcal{X} \) for all \( \delta \in [0, \delta_{\max}] \), and:
\[
\lim_{\delta \to 0} \frac{h(x^* + \delta v) - h(x^*)}{\delta} = \nabla h(x^*) \cdot v
\]

where \( \nabla h(x^*) \cdot v \) is the dot product of the gradient vector \( \nabla h(x^*) \) with the vector \( v \):
\[
\nabla h(x^*) \cdot v = \sum_{i=1}^{N} \frac{\partial h(x^*)}{\partial x_i} v_i
\]
Note that the gradient $\nabla h(x^*)$ is treated as a row vector so that the transpose operation is not needed in the dot product.

- Fact 2: Suppose $a$ and $b$ are nonzero vectors in $\mathbb{R}^N$ such that $a$ is not a multiple of $b$. Then there is a vector $v \in \mathbb{R}^N$ such that $a^T \cdot v < 0$ and $b^T \cdot v < 0$. Specifically, this holds for $v = -\frac{1}{2} \left( \frac{a}{||a||} + \frac{b}{||b||} \right)$. Note that the vectors $a$ and $b$ are treated as column vectors, so that the transpose $a^T$ is a row vector and the dot product is $a^T \cdot v = \sum_{i=1}^N a_i v_i$.

The following lemma is an immediate consequence of Fact 1. It shows that if $x^*$ is an interior point and $v$ is a vector such that $\nabla h(x^*) \cdot v < 0$, then taking a small step in the direction of $v$ can reduce the value of $h(\cdot)$.

**Lemma VIII.1.** Let $x^*$ be an interior point of $\mathcal{X}$ and suppose $h(x)$ is differentiable at $x^*$. Let $v$ be a vector in $\mathbb{R}^N$ that satisfies $\nabla h(x^*) \cdot v < 0$. Then there is a real number $\delta_{\text{max}} > 0$ such that $x^* + \delta v \in \mathcal{X}$ for all $\delta \in (0, \delta_{\text{max}}]$, and:

$$h(x^* + \delta v) < h(x^*) \quad \text{for all } \delta \in (0, \delta_{\text{max}}]$$

**Proof.** Define $\theta = -\nabla h(x^*) \cdot v$. Then $\theta > 0$ and $\nabla h(x^*) \cdot v = -\theta$. By Fact 2:

$$\lim_{\delta \to 0} \frac{h(x^* + \delta v) - h(x^*)}{\delta} = -\theta$$

It follows that there is a $\delta_{\text{max}} > 0$ such that for all $\delta \in (0, \delta_{\text{max}}]$, we have $x^* + \delta v \in \mathcal{X}$ and

$$\frac{h(x^* + \delta v) - h(x^*)}{\delta} \leq -\theta/2 < 0$$

Multiplying the above inequality by $\delta$ proves the result.

\[ \square \]

I. Proof of the necessary conditions

The next lemma shows that if $x^*$ is an interior point and $v$ is a vector such that $\nabla f(x^*) \cdot v < 0$ and $\nabla g(x^*) \cdot v < 0$, then taking a small step in the direction of $v$ reduces the value of both $f(\cdot)$ and $g(\cdot)$, which is impossible if $x^*$ is optimal.

**Lemma VIII.2.** Suppose $x^*$ is an optimal solution to \((168)-(170)\). If $x^*$ is an interior point of $\mathcal{X}$, then there is no vector $v \in \mathbb{R}^N$ that satisfies $\nabla g(x^*) \cdot v < 0$ and $\nabla f(x^*) \cdot v < 0$.

**Proof.** Since $x^*$ is an optimal solution to \((168)-(170)\), it satisfies $g(x^*) \leq c$. Suppose there is a vector $v \in \mathbb{R}^N$ that satisfies $\nabla g(x^*) \cdot v < 0$ and $\nabla f(x^*) \cdot v < 0$ (we will reach a contradiction). By Lemma VIII.1, it follows that there is a $\delta_{\text{max}} > 0$ such that $x^* + \delta v \in \mathcal{X}$ and:

$$f(x^* + \delta v) < f(x^*) \quad \text{and} \quad g(x^* + \delta v) < g(x^*) \leq c$$

Thus, the point $x^* + \delta v$ satisfies the constraints of the problem \((168)-(170)\) and gives a strictly better value of the objective function than $f(x^*)$. This contradicts the fact that $x^*$ is optimal for the problem \((168)-(170)\).

\[ \square \]

**Lemma VIII.3.** Suppose $x^*$ is an optimal solution to \((168)-(170)\). If $x^*$ is an interior point of $\mathcal{X}$ and $\nabla g(x^*) \neq 0$, then there is a $\mu \geq 0$ such that $(g(x^*) - c)\mu = 0$ and $\nabla f(x^*) + \mu \nabla g(x^*) = 0$.

**Proof.**

- If $\nabla f(x^*) = 0$, we let $\mu = 0$. Then the equation $(g(x^*) - c)\mu = 0$ holds, and so does $\nabla f(x^*) + \mu \nabla g(x^*) = 0$.
- Else, if $g(x^*) = c$ and $\nabla f(x^*) \neq 0$, then the condition $(g(x^*) - c)\mu = 0$ is satisfied for all real numbers $\mu$. It suffices to show there is a $\mu \geq 0$ such that $\nabla f(x^*) + \mu \nabla g(x^*) = 0$. To this end, note that since both $\nabla f(x^*)$ and $\nabla g(x^*)$ are nonzero vectors, if one is not a multiple of the other then there is a $v \in \mathbb{R}^N$ such that $\nabla f(x^*) \cdot v < 0$ and $\nabla g(x^*) \cdot v < 0$ (from Fact 2), contradicting Lemma VIII.2. Thus, $\nabla f(x^*)$ is a multiple of $\nabla g(x^*)$. That is, $
abla f(x^*) = \alpha \nabla g(x^*)$ for some $\alpha \in \mathbb{R}$. If $\alpha > 0$ then we can define $v = -\nabla g(x^*)$ to get both $\nabla f(x^*) \cdot v < 0$ and $\nabla g(x^*) \cdot v < 0$, again contradicting Lemma VIII.2. Hence, $\alpha \leq 0$. Define $\mu = -\alpha$. Then $\mu \geq 0$ and $\nabla f(x^*) + \mu \nabla g(x^*) = 0$.

\[ \square \]

J. Equality constraints

Similar Karush-Kuhn-Tucker conditions hold for equality constraints. Here we state the result without proof (see, for example, [29] for a proof). Suppose $\mathcal{X} \subseteq \mathbb{R}^N$, and that $f(x)$ and $g(x)$ are real-valued functions that are differentiable on the interior of $\mathcal{X}$. Consider the problem:

\[
\begin{align*}
\text{Minimize:} & \quad f(x) \\
\text{Subject to:} & \quad g(x) = c \\
& \quad x \in \mathcal{X}
\end{align*}
\]
If \( x^* = (x_1^*, \ldots, x_N^*) \) is an optimal solution to this problem, then one of the following three conditions must hold:

- \( x^* \) is a boundary point of \( \mathcal{X} \).
- Else, \( \nabla g(x^*) = 0 \).
- Else, \( \nabla f(x^*) + \lambda \nabla g(x^*) = 0 \) for some \( \lambda \in \mathbb{R} \).

**APPENDIX B—HYPERPLANE SEPARATION AND LAGRANGE MULTIPLIERS**

This section shows that under certain convexity assumptions, all points on the tradeoff curve are also solutions to an unconstrained problem for a particular Lagrange multiplier \( \mu \geq 0 \). In such cases, there are no “hidden” Pareto optimal points that are not solutions to an unconstrained optimization.

**K. Separation of convex sets in the 2-d plane**

![Fig. 18. An example of the hyperplane separation theorem. The set \( \mathcal{A} \) is the orange shaded region. The set \( \tilde{\mathcal{A}} \) includes \( \mathcal{A} \) and also includes the extended blue region. The boundary point is \((x^*, y^*)\).](image-url)

Consider again the problem:

\[
\text{Minimize:} \quad y \\
\text{Subject to:} \quad x \leq c \\
(x, y) \in \mathcal{A}.
\]

where \( \mathcal{A} \) is a given nonempty subset of \( \mathbb{R}^N \) and \( c \) is a real number. Define \( x_{\text{min}} \) as the infimum value of \( x \) over all points \((x, y) \in \mathcal{A}\). Let \((x^*, y^*)\) be an optimal solution to the above problem and assume that \( x_{\text{min}} < x^* \). Define \( \tilde{\mathcal{A}} \) as the set of all points that are entrywise greater than or equal to some point in \( \mathcal{A} \). Assume the set \( \tilde{\mathcal{A}} \) is convex (it can be shown to be convex whenever \( \mathcal{A} \) itself is convex). It is clear that \((x^*, y^*)\) is a boundary point of the set \( \mathcal{A} \), since \((x^*, y^*) \notin \mathcal{A}\) for all \( \epsilon > 0 \).

The convex set separation theorem in the 2-d plane says that if \( \tilde{\mathcal{A}} \) is a convex subset of \( \mathbb{R}^2 \) and if \((x^*, y^*)\) is a point on the boundary of \( \mathcal{A} \), then there exists a line that passes through \((x^*, y^*)\) that contains \( \mathcal{A} \) on one side (see Fig. 18). This is a special case of the hyperplane separation theorem \( \text{[II]} \). This line cannot be vertical since then there would be points strictly on the left with \( x \) coordinates close to \( x_{\text{min}} \), and strictly on the right with \( x \) coordinates arbitrarily large. Thus, the line can be described by the set of all \((x, y) \in \mathbb{R}^2\) that satisfy:

\[(y - y^*) = -\mu(x - x^*) \]

for some value \( \mu \in \mathbb{R} \). The value \(-\mu\) is the slope of the line (see Fig. 18). Since the line contains the set \( \tilde{\mathcal{A}} \) on one side, we either have:

\[(y - y^*) \geq -\mu(x - x^*) \quad \forall (x, y) \in \tilde{\mathcal{A}} \quad (174)\]

or

\[(y - y^*) \leq -\mu(x - x^*) \quad \forall (x, y) \in \tilde{\mathcal{A}} \quad (175)\]

To show that \((175)\) is impossible, let \((a, b) \in \tilde{\mathcal{A}}\). Then \((a, b + h) \in \tilde{\mathcal{A}}\) for all \( h > 0 \), which makes \((175)\) fail for sufficiently large values of \( h \). Thus, \((174)\) holds. Similarly, it follows that \( \mu \geq 0 \) (else, if \( \mu < 0 \) we could choose \( x \) arbitrarily large and the inequality \((174)\) would be violated). Thus, \( \mu \geq 0 \) and:

\[y + \mu x \geq y^* + \mu x^*\]

for all \((x, y) \in \tilde{\mathcal{A}}\). In particular, the above holds for all \((x, y) \in \mathcal{A}\). It follows that the optimal solution \((x^*, y^*)\) to problem \((171)-(173)\) is also a global minimum of \( y + \mu x \) over the set \((x, y) \in \mathcal{A}\).
L. A counter-example when $x^* = x_{\min}$

Consider the same problem (171)-(173). For simplicity, assume $A$ is a convex and compact set. Again suppose $(x^*, y^*)$ is an optimal solution to (171)-(173), but that $x^* = x_{\min}$ (where $x_{\min}$ is the minimum value of $x$ over all points $(x, y) \in A$). In this case, there may or may not be a (finite) Lagrange multiplier $\mu \geq 0$ such that $(x^*, y^*)$ minimizes $y + \mu x$ over all $(x, y) \in A$. Such non-existence happens when the only tangent line to set $A$ that passes through the point $(x^*, y^*)$ has infinite slope. A simple example is:

$$A = \{(x, y) | 0 \leq x \leq 1, |y| \leq \sqrt{x}\}$$

The set $A$ is shown in Fig. 19. Clearly $x_{\min} = 0$. The point $(x^*, y^*) = (0, 0)$ solves the problem:

\[
\begin{align*}
\text{Minimize:} & \quad y \\
\text{Subject to:} & \quad x \leq 0 \\
& \quad (x, y) \in A
\end{align*}
\]

However, $x^* = x_{\min} = 0$, and the only tangent line to set $A$ that passes through $(0, 0)$ is the $y$-axis itself, which has infinite slope (see Fig. [19]). The previous section shows that such an example can only arise if $x^* = x_{\min}$. Further, it can be shown that such an example can only arise if the set $A$ is not the convex hull of a finite set of points (else, the set $A$ has a finite number of linear edges, and one can intuitively see that a tangent line of finite slope exists).

![Fig. 19. An example convex set $A$ for which $(x^*, y^*) = (0, 0)$ is a solution to a constrained optimization problem, but there is no corresponding Lagrange multiplier $\mu \geq 0$. Such an example can only arise if $x^* = x_{\min}$.

M. Hyperplanes

A hyperplane in $\mathbb{R}^N$ is defined by a nonzero vector $\gamma = (\gamma_1, \ldots, \gamma_N)$ and a scalar $b$. The hyperplane consists of all points $(x_1, \ldots, x_N) \in \mathbb{R}^N$ that satisfy:

$$\sum_{i=1}^{N} \gamma_i x_i = b$$

That is:

$$\sum_{i=1}^{N} \gamma_i x_i = b$$

A hyperplane slices $\mathbb{R}^N$ into two halves. The **upper half** is the set of all points $x \in \mathbb{R}^N$ that satisfy:

$$\sum_{i=1}^{N} \gamma_i x_i \geq b$$

The **lower half** is the set of all points $x \in \mathbb{R}^N$ that satisfy:

$$\sum_{i=1}^{N} \gamma_i x_i \leq b$$

The **hyperplane separation theorem** says that if $A$ is a convex subset of $\mathbb{R}^N$ and $x^*$ is a point on the boundary of $A$, then there exists a hyperplane that passes through $x^*$ and that contains the set $A$ on one side [11]. A hyperplane that passes through the point $x^* = (x_1^*, \ldots, x_N^*)$ must have the form:

$$\sum_{i=1}^{N} \gamma_i x_i = \sum_{i=1}^{N} \gamma_i x_i^*$$
Indeed, note that this hyperplane certainly passes through the point \( x^* \). Since there exists such a hyperplane that contains the convex set \( A \) on one side, we have one of the two possibilities. Either it contains \( A \) in its upper half:

\[
\sum_{i=1}^{N} \gamma_i a_i \geq \sum_{i=1}^{N} \gamma_i x_i^* \quad \forall (a_1, \ldots, a_N) \in A
\]  

(176)

or it contains \( A \) in its lower half:

\[
\sum_{i=1}^{N} \gamma_i a_i \leq \sum_{i=1}^{N} \gamma_i x_i^* \quad \forall (a_1, \ldots, a_N) \in A
\]

Multiplying the previous equation by \(-1\) gives an equation of the form (176). Hence, it suffices to assume that the hyperplane contains the set \( A \) on its upper half.

\[ N. \text{ Hyperplane separation for problems with multiple constraints} \]

Let \( N \) and \( K \) be positive integers. Let \( X \) be a subset of \( \mathbb{R}^N \) and let \( f(x), g_1(x), \ldots, g_K(x) \) be real-valued functions over \( x \in X \). Consider the following problem:

Minimize:

\[
f(x)
\]

Subject to:

\[
g_k(x) \leq c_k \quad \forall k \in \{1, \ldots, K\}
\]

\[
x \in X
\]

(177) (178) (179)

where \( c_1, \ldots, c_K \) are a given set of real numbers. Theorem [III.1] motivates a Lagrange multiplier approach to this problem by solving the unconstrained problem \( f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \) over \( x \in X \) for a given collection of real numbers \( \mu_1 \geq 0, \ldots, \mu_K \geq 0 \).

Now consider the special case when the set \( X \) is convex, and the functions \( f(x), g_1(x), \ldots, g_K(x) \) are convex over \( x \in X \). Let \( A \) be the set of all points \((g_1, \ldots, g_K, f) \in \mathbb{R}^{K+1}\) such that \( (g_1, \ldots, g_K, f) \geq (g_1(x), \ldots, g_K(x), f(x)) \) for some \( x \in X \). Similar to Exercise [VIII-D.9], it can be shown that this set \( A \) is convex. If \( x^* \) is a solution to the above constrained optimization problem (177), then the point \((c_1, \ldots, c_K, f(x^*)) \in A \), since:

\[
(c_1, \ldots, c_K, f(x^*)) \geq (g_1(x^*), g_2(x^*), \ldots, g_K(x^*), f(x^*))
\]

Furthermore, this point \((c_1, \ldots, c_K, f(x^*)) \) must be on the boundary of \( A \), since it is in \( A \), but for any \( \epsilon > 0 \) we have \((c_1, \ldots, c_K, f(x^*) - \epsilon) \notin A \) (else, \( f(x^*) \) would not be the optimal objective function value). The hyperplane separation theorem in \( \mathbb{R}^N \) implies that there exists a hyperplane in \( \mathbb{R}^{K+1} \) that passes through the point \((f(x^*), c_1, \ldots, c_K)\) and that contains \( A \) on one side. That is, there is a nonzero \((K+1)\)-dimensional vector \( \gamma = (\gamma_1, \ldots, \gamma_K, \gamma_{K+1}) \) that defines the hyperplane of all \((g_1, \ldots, g_K, f) \in \mathbb{R}^{K+1}\) that satisfy:

\[
\gamma^T \cdot (g_1, \ldots, g_K, f) = \gamma^T \cdot (c_1, \ldots, c_K, f(x^*))
\]

That is:

\[
\gamma_{K+1} f + \sum_{k=1}^{K} \gamma_k g_k = \gamma_{K+1} f(x^*) + \sum_{k=1}^{K} \gamma_k c_k
\]

Since this hyperplane contains the set \( A \) in its upper half, we have for all \((a_1, \ldots, a_K, a_{K+1}) \in A\):

\[
\gamma_{K+1} a_{K+1} + \sum_{k=1}^{K} \gamma_k a_k \geq \gamma_{K+1} f(x^*) + \sum_{k=1}^{K} \gamma_k c_k
\]

(180)

If a point \((a_1, \ldots, a_{K+1}) \) is in \( A \), then all points that are entrywise greater than or equal to \((a_1, \ldots, a_{K+1}) \) are also in \( A \). It follows that the values \( \gamma_i \) must be non-negative for all \( i \in \{1, \ldots, K+1\} \) (else, if there is an index \( j \in \{1, \ldots, K+1\} \) such that \( \gamma_j < 0 \), we could let \( a_j \to \infty \), which would make the left-hand-side of (180) arbitrarily small (negative), which would violate that inequality). We have \( \gamma_{K+1} \neq 0 \) if a non-vertical hyperplane exists. If a non-vertical hyperplane exists, then \( \gamma_{K+1} > 0 \) and we can divide (180) by \( \gamma_{K+1} \) and define \( \mu_k = \gamma_k / \gamma_{K+1} \) for all \( k \in \{1, \ldots, K\} \) to obtain the following for all \((a_1, \ldots, a_K, a_{K+1}) \in A\):

\[
a_{K+1} + \sum_{k=1}^{K} \mu_k a_k \geq f(x^*) + \sum_{k=1}^{K} \mu_k c_k
\]

Since \((g_1(x), \ldots, g_K(x), f(x)) \in A\) whenever \( x \in X \), this implies:

\[
f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \geq f(x^*) + \sum_{k=1}^{K} \mu_k c_k \quad \forall x \in X
\]

(181)
Since $\mu_k c_k \geq \mu_k g_k(x^*)$ for all $k \in \{1, \ldots, K\}$, the above implies:

$$f(x) + \sum_{k=1}^{K} \mu_k g_k(x) \geq f(x^*) + \sum_{k=1}^{K} \mu_k g_k(x^*) \quad \forall x \in \mathcal{X}$$

Thus, $x^*$ solves the global minimization of $f(x) + \sum_{k=1}^{K} \mu_k g_k(x)$ over $x \in \mathcal{X}$.

The non-vertical assumption is justified under certain regularity conditions. The simplest is the Slater condition, which says that there exists an $\bar{x} \in \mathcal{X}$ such that $g_k(\bar{x}) < c_k$ for all $k \in \{1, \ldots, K\}$. Indeed, suppose this condition is satisfied, but $\gamma_{K+1} = 0$. It is easy to obtain a contradiction to equation (180) (just consider $(a_1, \ldots, a_{K+1}) = (g_1(\bar{x}), \ldots, g_K(\bar{x}), f(\bar{x}))$ and recall that the $\gamma_k$ values are non-negative and not all zero). Additional regularity conditions (which do not require the Slater assumption) are given in [1].

APPENDIX C—IMPROVED CONVERGENCE TIME ANALYSIS OF DRIFT-PLUS-PENALTY

This appendix proves part (b) of Theorem VI.2. This result was developed in [4]. Recall that the convex program seeks to find a vector $x = (x_1, \ldots, x_N)$ to solve:

Minimize: $f(x)$

Subject to: $g_k(x) \leq c_k \quad \forall k \in \{1, \ldots, K\}$ $x \in \mathcal{X}$

where $\mathcal{X}$ is a compact and convex subset of $\mathbb{R}^N$, the functions $f(x), g_1(x), \ldots, g_K(x)$ are continuous and convex over $x \in \mathcal{X}$, and $c_k$ are given real numbers for all $k \in \{1, \ldots, K\}$. Suppose the problem (182)-(184) is feasible. Let $x^*$ be an optimal solution, with optimal objective function value $f^* = f(x^*)$. Suppose the drift-plus-penalty algorithm is implemented using a particular value of $V$ $\geq 0$, and with initial conditions $Q_k(0) = 0$ for all $k \in \{1, \ldots, K\}$.

A restatement of Theorem VI.2 is given below: Suppose the problem (182)-(184) is feasible, and that a Lagrange multiplier vector $\mu = (\mu_1, \ldots, \mu_K)$ exists (so that (181) holds). Then the drift-plus-penalty algorithm with $V \geq 0$ and initial conditions $Q_k(0) = 0$ for all $k \in \{1, \ldots, K\}$ ensures:

(a) $f(\pi(t)) \leq f^* + \frac{B}{t}$ for all slots $t \in \{1, 2, 3, \ldots\}$, where $B$ is the constant used in (76).

(b) $\frac{\|Q(t)\|}{t} \leq \frac{V\|\mu\| + \sqrt{V^2\|\mu\|^2 + 2Bt}}{t}$ for all $t \in \{1, 2, 3, \ldots\}$.

(c) Define $\overline{V} = 1/e$. Then for any integer $t \geq 1/e^2$ we have:

$$f(\pi(t)) \leq f^* + O(e)$$

$$g_k(\pi(t)) \leq c_k + O(e) \quad \forall k \in \{1, \ldots, K\}$$

$$\overline{x}(t) \in \mathcal{X}$$

Hence, the drift-plus-penalty algorithm produces an $O(e)$ approximation to the solution with a convergence time of $O(1/e^2)$.

Proof. Part (a) is already known from Theorem VI.1 and part (c) follows immediately from (a) and (b) together with the virtual queue lemma (Lemma VI.1). It remains to prove part (b). The proof of Theorem VI.1 establishes the following inequality for all slots $\tau \in \{0, 1, 2, \ldots\}$ (see (95)):

$$\Delta(\tau) + V f(x(\tau)) \leq B + V f(x^*)$$

where $\Delta(\tau) = \|Q(\tau + 1)\|^2/2 - \|Q(\tau)\|^2/2$. Let $t$ be a positive integer. Summing the above over $\tau \in \{0, 1, \ldots, t - 1\}$ gives:

$$\frac{\|Q(t)\|^2}{2} - \frac{\|Q(0)\|^2}{2} + V \sum_{\tau=0}^{t-1} f(x(\tau)) \leq Bt + tV f(x^*)$$

(185)

However, $\|Q(0)\| = 0$ since queues are initially empty. Further, applying Jensen’s inequality to the convex function $f(x)$ gives:

$$f(\overline{x}(t)) \leq \frac{1}{t} \sum_{\tau=0}^{t-1} f(x(\tau))$$

Substituting the above into (185) gives:

$$\frac{\|Q(t)\|^2}{2} + V tf(\overline{x}(t)) \leq Bt + tV f(x^*)$$

Rearranging terms gives:

$$\|Q(t)\|^2 \leq 2Bt + 2V t[f(x^*) - f(\overline{x}(t))]$$

(186)
Since $z(\tau) \in \mathcal{X}$ for all $\tau$ and $\mathcal{X}$ is a convex set, it follows that $\pi(t) \in \mathcal{X}$. Thus, from (181) we have:

$$f(\pi(t)) + \sum_{k=1}^{K} \mu_k g_k(\pi(t)) \geq f(x^*) + \sum_{k=1}^{K} \mu_k c_k$$

Substituting the above inequality into (186) yields:

$$||Q(t)||^2 \leq 2Bt + 2Vt \sum_{k=1}^{K} \mu_k [g_k(\pi(t)) - c_k]$$

However, Lemma VI.1 implies that $g_k(\pi(t)) \leq c_k + Q_k(t)/t$ for all $k \in \{1, \ldots, K\}$, and so:

$$||Q(t)||^2 \leq 2Bt + 2Vt \sum_{k=1}^{K} \mu_k Q_k(t)/t \\ \leq 2Bt + 2V||Q(t)|| \cdot ||\mu||$$

where the final inequality uses the fact that the dot product of two vectors is less than or equal to the product of their norms. Define $b = -2V||\mu||$ and $c = -2Bt$. Then:

$$||Q(t)||^2 + b||Q(t)|| + c \leq 0$$

The largest possible value of $||Q(t)||$ that satisfies the above inequality is found from taking the largest solution to the quadratic equation $x^2 + bx + c = 0$. Thus:

$$||Q(t)|| \leq \frac{-b + \sqrt{b^2 - 4c}}{2}$$

$$= \frac{2V||\mu|| + \sqrt{4V^2||\mu||^2 + 8Bt}}{2}$$

$$= V||\mu|| + \sqrt{V^2||\mu||^2 + 2Bt}$$

This completes the proof of part (b). \[\square\]

REFERENCES


