Chapter 23 Algorithms for Constrained Optimization

An Introduction to Optimization

Spring, 2015

Wei-Ta Chu

Projections

- The optimization algorithms described in Part II have the general form $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$, where $\boldsymbol{d}^{(k)}$ is typically a function of $\nabla f(\boldsymbol{x}^{(k)})$. The value of $\boldsymbol{x}^{(k)}$ is not constrained to lie inside any particular set.
- Consider the optimization problem

$$\begin{aligned} minimize \;\; & f(\boldsymbol{x}) \\ subject\ to \;\; & \boldsymbol{x} \in \Omega \end{aligned}$$

- Modify the algorithm we already know to take constraints into account.

Projections

- If $\boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$ is in Ω, then we set form $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$ as usual.
- If, on the other hand, $\boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$ is not in Ω, then we "project" it back into Ω before setting $\boldsymbol{x}^{(k+1)}$.
- Consider the case where the constraint set $\Omega \in R^n$ is given by

$$\Omega = \{\boldsymbol{x}: l_i \le x_i \le u_i, \qquad i = 1, \dots, n\}$$

- In this case, Ω is a "box" in R^n; for this reason, this form of Ω is called a *box constraint*.

Projections

- Given a point $\boldsymbol{x} \in R^n$, define $\boldsymbol{y} = \prod[\boldsymbol{x}] \in R^n$ by

$$y_i = \min\{u_i, \max\{l_i, x_i\}\} = \begin{cases} u_i & if\ x_i > u_i \\ x_i & if\ l_i \le x_i \le u_i \\ l_i & if\ x_i < l_i \end{cases}$$

- The point $\prod[\boldsymbol{x}]$ is called the projection of $\boldsymbol{x}$ onto Ω. Note that $\prod[\boldsymbol{x}]$ is actually the "closest" point in Ω to $\boldsymbol{x}$. Using the projection operator $\prod$, we can modify the previous unconstrained algorithm as follows:

$$\boldsymbol{x}^{(k+1)} = \prod[\boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}]$$

- Note that the iterates $\boldsymbol{x}^{(k)}$ now all lie inside Ω. We call the algorithm above a *projection algorithm*.

Projections

- In the more general case, we can define the projection onto Ω:

$$\prod[\boldsymbol{x}] = \arg\min_{\boldsymbol{z}\in\Omega} \|\boldsymbol{z} - \boldsymbol{x}\|$$

- In this case, $\prod[\boldsymbol{x}]$ is again the "closest" point in Ω to $\boldsymbol{x}$. This projection operator is well-defined only for certain types of constraint sets: for example, closed convex sets.
- For some sets Ω, the "arg min" above is not well-defined. If the projection $\prod$ is well-defined, we can simply apply the projection algorithm

$$\boldsymbol{x}^{(k+1)} = \prod[\boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}]$$

Projections

- Consider the projection method applied specifically to the gradient algorithm. Recall that the vector $-\nabla f(\boldsymbol{x})$ points in the direction of maximum rate of decrease of f at $\boldsymbol{x}$. This was the basis for gradient methods for unconstrained optimization, which have the form $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)})$, where α_k is the step size. The choice of the step size depends on the particular gradient algorithm. For example, recall that in the steepest descent algorithm, $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)}))$.

Projections

- The projected version of the gradient algorithm has the form

$$\boldsymbol{x}^{(k+1)} = \prod[\boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)})]$$

- We refer to the above as the *projected gradient algorithm*.

Example

- Consider the problem

$$\begin{aligned} &\textit{minimize } \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \\ &\textit{subject to } \|\boldsymbol{x}\|^2 = 1 \end{aligned}$$

where $\boldsymbol{Q} = \boldsymbol{Q}^T > 0$. Suppose that we apply a *fixed-step-size projected gradient algorithm* to this problem.

- a. Derive a formula for the update equation for the algorithm. You may assume that the argument in the projection operator to obtain $\boldsymbol{x}^{(k)}$ is never zero.

Example

- The projection operator in this case simply maps any vector to the closest point on the unit circle. Therefore, the projection operator is given by $\prod[\boldsymbol{x}] = \boldsymbol{x}/\|\boldsymbol{x}\|$, provided that $\boldsymbol{x} \neq \boldsymbol{0}$. The update equation is

$$\boldsymbol{x}^{(k+1)} = \beta_k\left(\boldsymbol{x}^{(k)} - \alpha\boldsymbol{Q}\boldsymbol{x}^{(k)}\right) = \beta_k(\boldsymbol{I} - \alpha\boldsymbol{Q})\boldsymbol{x}^{(k)}$$

where $\beta_k = 1/\left\|(\boldsymbol{I} - \alpha\boldsymbol{Q})\boldsymbol{x}^{(k)}\right\|$ (i.e., it is whatever constant scaling is needed to make $\boldsymbol{x}^{(k+1)}$ have unit norm).

Example

- b. It is possible for the algorithm not to converge to an optimal solution even if the step size $\alpha > 0$ is taken to be arbitrarily small?
- If we start with $\boldsymbol{x}^{(0)}$ being an eigenvector of $\boldsymbol{Q}$, then $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(0)}$ for all k. Therefore, if the corresponding eigenvalue is not the smallest, then clearly the algorithm is stuck at a point that is not optimal.

Example

- c. Show that for $0 < \alpha < 1/\lambda_{max}$ (where λ_{max} is the largest eigenvalue of $\boldsymbol{Q}$), the fixed-step-size projected gradient algorithm (with step size α) converges to an optimal solution, provided that $\boldsymbol{x}^{(0)}$ is not orthogonal to the eigenvectors of $\boldsymbol{Q}$ corresponding to the smallest eigenvalue. (Assume that the eigenvalues are distinct.)

Example

- We have

$$\begin{aligned}\boldsymbol{x}^{(k+1)} &= \beta_k(\boldsymbol{I}-\alpha\boldsymbol{Q})\boldsymbol{x}^{(k)} \\ &= \beta_k(\boldsymbol{I}-\alpha\boldsymbol{Q})\left(y_1^{(k)}\boldsymbol{v}_1+\cdots+y_n^{(k)}\boldsymbol{v}_n\right) \\ &= \beta_k(y_1^{(k)}(\boldsymbol{I}-\alpha\boldsymbol{Q})\boldsymbol{v}_1+\cdots+y_n^{(k)}(\boldsymbol{I}-\alpha\boldsymbol{Q})\boldsymbol{v}_n)\end{aligned}$$

- But $(\boldsymbol{I}-\alpha\boldsymbol{Q})\boldsymbol{v}_i=(1-\alpha\lambda_i)\boldsymbol{v}_i$, where λ_i is the eigenvalue corresponding to $\boldsymbol{v}_i$. Hence,

$$\boldsymbol{x}^{k+1}=\beta_k(y_1^{(k)}(1-\alpha\lambda_1)\boldsymbol{v}_1+\cdots+y_n^{(k)}(1-\alpha\lambda_n)\boldsymbol{v}_n)$$

which means that $y_i^{(k+1)}=\beta_k y_i^{(k)}(1-\alpha\lambda_i)$.

Example

- In other words, $y_i^{(k)} = \beta^{(k)} y_i^{(0)} (1 - \alpha\lambda_i)^k$, where $\beta^{(k)} = \prod_{i=0}^{k-1} \beta_k$. We rewrite $\boldsymbol{x}^{(k)}$ as

$$\boldsymbol{x}^{(k)} = \sum_{i=1}^{n} y_i^{(k)} \boldsymbol{v}_i = y_1^{(k)} \left(\boldsymbol{v}_1 + \sum_{i=2}^{n} \frac{y_i^{(k)}}{y_1^{(k)}} \boldsymbol{v}_i \right)$$

- Assuming that $y_1^{(0)} \neq 0$, we obtain

$$\frac{y_i^{(k)}}{y_1^{(k)}} = \frac{y_i^{(0)} (1 - \alpha\lambda_i)^k}{y_1^{(0)} (1 - \alpha\lambda_1)^k} = \frac{y_i^{(0)}}{y_1^{(0)}} \left(\frac{1 - \alpha\lambda_i}{1 - \alpha\lambda_1} \right)^k$$

Example

- Using the fact that $\frac{1-\alpha\lambda_i}{1-\alpha\lambda_1} < 1$ (because the $\lambda_i > \lambda_1$ for $i > 1$ and $\alpha < 1/\lambda_{max}$), we deduce that

$$\frac{y_i^{(k)}}{y_1^{(k)}} \to 0,$$

which implies that $\boldsymbol{x}^{(k)} \to \boldsymbol{v}_1$, as required.

Projected Gradient Methods with Linear Constraints

- Consider optimization problems of the form

$$\begin{aligned} &minimize \;\; f(\boldsymbol{x}) \\ &subject\ to \;\; \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

where $f: R^n \to R, \boldsymbol{A} \in R^{m\times n}$, $m < n$, rank $\boldsymbol{A} = m$, $\boldsymbol{b} \in R^m$

- We assume throughout that $f \in C^1$.
- The specific structure of the constraint set allows us to compute the projection operator $\prod$ using the *orthogonal projector*. Specifically, $\prod[\boldsymbol{x}]$ can be defined using the orthogonal projector matrix $\boldsymbol{P}$ given by

$$\boldsymbol{P} = \boldsymbol{I}_n - \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}\boldsymbol{A}$$

Projected Gradient Methods with Linear Constraints

- Two important properties of the orthogonal projector $\boldsymbol{P}$ that we use in this section are
  - $\boldsymbol{P} = \boldsymbol{P}^T$
  - $\boldsymbol{P}^2 = \boldsymbol{P}$
- Lemma 23.1. Let $\boldsymbol{v} \in R^n$. Then, $\boldsymbol{Pv} = \boldsymbol{0}$ if and only if $\boldsymbol{v} \in \mathcal{R}(\boldsymbol{A}^T)$. In other words, $\mathcal{N}(\boldsymbol{P}) = \mathcal{R}(\boldsymbol{A}^T)$. Moreover, $\boldsymbol{Av} = \boldsymbol{0}$ if and only if $\boldsymbol{v} \in \mathcal{R}(\boldsymbol{P})$; that is, $\mathcal{N}(\boldsymbol{A}) = \mathcal{R}(\boldsymbol{P})$.

Projected Gradient Methods with Linear Constraints

- Recall that in unconstrained optimization, the first-order necessary condition for a point $\boldsymbol{x}^*$ to be a local minimizer is $\nabla f(\boldsymbol{x}^*) = 0$.
- In optimization problems with equality constraints, the Lagrange condition plays the role of the first-order necessary condition.
- When the constraint set takes the form $\{\boldsymbol{x}: \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$, the Lagrange condition can be written as $\boldsymbol{P}\nabla f(\boldsymbol{x}^*) = 0$.

Projected Gradient Methods with Linear Constraints

- Proposition 23.1. Let $\boldsymbol{x}^* \in R^n$ be a feasible point. Then $\boldsymbol{P}\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ if and only if $\boldsymbol{x}^*$ satisfies the Lagrange condition.
- Proof. By Lemma 23.1, $\boldsymbol{P}\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ if and only if we have $\nabla f(\boldsymbol{x}^*) \in \mathcal{R}(\boldsymbol{A}^T)$. This is equivalent to the condition that there exists $\boldsymbol{\lambda}^* \in R^m$ such that $\nabla f(\boldsymbol{x}^*) + \boldsymbol{A}^T\boldsymbol{\lambda}^* = \boldsymbol{0}$, which together with the feasibility equation $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, constitutes the Lagrange condition.

Projected Gradient Methods with Linear Constraints

- Recall that the projected gradient algorithm has the form

$$x^{(k+1)} = \prod[\boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)})]$$

- For the case where the constraints are linear, it turns out that we can express the projection $\prod$ in terms of the matrix $\boldsymbol{P}$ as follows:

$$\prod\left[\boldsymbol{x}^{(k)} - \alpha_k \nabla f\left(\boldsymbol{x}^{(k)}\right)\right] = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{P} \nabla f(\boldsymbol{x}^{(k)})$$

assuming that $\boldsymbol{x}^k \in \Omega$.

Projected Gradient Methods with Linear Constraints

- In our constrained optimization problem, the vector $-\nabla f(\boldsymbol{x})$ is not necessarily a feasible direction. In other words, if $\boldsymbol{x}^{(k)}$ is a feasible point and we apply the algorithm $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)})$, then $\boldsymbol{x}^{(k+1)}$ need not be feasible. This problem can be overcome by replacing $-\nabla f(\boldsymbol{x}^{(k)})$ by a vector that points in a feasible direction.
- Note that the set of feasible directions is simply the nullspace $\mathcal{N}(\boldsymbol{A})$ of the matrix $\boldsymbol{A}$. Therefore, we should first project the vector $-\nabla f(\boldsymbol{x})$ onto $\mathcal{N}(\boldsymbol{A})$. This projection is equivalent to multiplication by the matrix $\boldsymbol{P}$.

Projected Gradient Methods with Linear Constraints

- In summary, in the projection gradient algorithm, we update $\boldsymbol{x}^{(k)}$ according to the equation

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{P} \nabla f(\boldsymbol{x}^{(k)})$$

Projected Gradient Methods with Linear Constraints

- Proposition 23.2. In a projected gradient algorithm, if $\boldsymbol{x}^{(0)}$ is feasible, then each $\boldsymbol{x}^{(k)}$ is feasible; that is, for each $k \geq 0$, $\boldsymbol{A}\boldsymbol{x}^{(k)} = \boldsymbol{b}$.
- Proof. We proceed by induction. The result holds for $k = 0$ by assumption. Suppose now that $\boldsymbol{A}\boldsymbol{x}^{(k)} = \boldsymbol{b}$. We now show that $\boldsymbol{A}\boldsymbol{x}^{(k+1)} = \boldsymbol{b}$. Observe that $\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(k)}\right) \in \mathcal{N}(\boldsymbol{A})$. Therefore,

$$\begin{aligned}\boldsymbol{A}\boldsymbol{x}^{(k+1)} &= \boldsymbol{A}\left(\boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(k)}\right)\right) \\ &= \boldsymbol{A}\boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{A}\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(k)}\right) \\ &= \boldsymbol{b}\end{aligned}$$

Projected Gradient Methods with Linear Constraints

- The projected gradient algorithm updates $\boldsymbol{x}^{(k)}$ in the direction of $-\boldsymbol{P}\nabla f(\boldsymbol{x}^{(k)})$. This vector points in the direction of maximum rate of decrease of f at $\boldsymbol{x}^{(k)}$ along the surface defined by $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, as described in the following argument.
- Let $\boldsymbol{x}$ be any feasible point and $\boldsymbol{d}$ a feasible direction such that $||\boldsymbol{d}|| = 1$. The rate of increase of f at $\boldsymbol{x}$ in the direction $\boldsymbol{d}$ is $\langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle$. Next, we note that because $\boldsymbol{d}$ is a feasible direction, it lies in $\mathcal{N}(\boldsymbol{A})$ and hence by Lemma 23.1, we have $\boldsymbol{d} \in \mathcal{R}(\boldsymbol{P}) = \mathcal{R}(\boldsymbol{P}^T)$. So, there exists $\boldsymbol{v}$ such that $\boldsymbol{d} = \boldsymbol{P}\boldsymbol{v}$.

Projected Gradient Methods with Linear Constraints

- Hence,

$$\langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle = \langle \nabla f(\boldsymbol{x}), \boldsymbol{P}^T \boldsymbol{v} \rangle = \langle \boldsymbol{P} \nabla f(\boldsymbol{x}), \boldsymbol{v} \rangle$$

- By the Cauchy-Schwarz inequality,

$$\langle \boldsymbol{P} \nabla f(\boldsymbol{x}), \boldsymbol{v} \rangle \leq \|\boldsymbol{P} \nabla f(\boldsymbol{x})\| \|\boldsymbol{v}\|$$

with equality if and only if he direction of $\boldsymbol{v}$ is parallel with the direction of $\boldsymbol{P} \nabla f(\boldsymbol{x})$. Therefore, the vector $-\boldsymbol{P} \nabla f(\boldsymbol{x})$ points in the direction of maximum rate of decrease of f at $\boldsymbol{x}$ among all feasible directions.

Projected Gradient Methods with Linear Constraints

- Suppose that we have a starting point $\boldsymbol{x}^{(0)}$, which we assume is feasible; that is, $\boldsymbol{A}\boldsymbol{x}^{(0)} = \boldsymbol{b}$. Consider the point $\boldsymbol{x} = \boldsymbol{x}^{(0)} - \alpha \boldsymbol{P} \nabla f(\boldsymbol{x}^{(0)})$, where $\alpha \in R$. As usual, the scalar α is called the step size.
- By the discussion above, $\boldsymbol{x}$ is also a feasible point. Using a Taylor series expansion of f about $\boldsymbol{x}^{(0)}$ and the fact that $\boldsymbol{P} = \boldsymbol{P}^2 = \boldsymbol{P}^T\boldsymbol{P}$, we get

$$\begin{aligned}
&f\left(\boldsymbol{x}^{(0)} - \alpha \boldsymbol{P} \nabla f\left(\boldsymbol{x}^{(0)}\right)\right) \\
&\qquad = f\left(\boldsymbol{x}^{(0)}\right) - \alpha \nabla f\left(\boldsymbol{x}^{(0)}\right)^T \boldsymbol{P} \nabla f\left(\boldsymbol{x}^{(0)}\right) + o(\alpha) \\
&\qquad = f\left(\boldsymbol{x}^{(0)}\right) - \alpha \left\|\boldsymbol{P} \nabla f\left(\boldsymbol{x}^{(0)}\right)\right\|^2 + o(\alpha)
\end{aligned}$$

Projected Gradient Methods with Linear Constraints

- Thus, if $\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(0)}\right) \neq 0$, that is, $\boldsymbol{x}^{(0)}$ does not satisfy the Lagrange condition, then we can choose an α sufficiently small such that $f(\boldsymbol{x}) < f(\boldsymbol{x}^{(0)})$, which means that $\boldsymbol{x} = \boldsymbol{x}^{(0)} - \alpha\boldsymbol{P}\nabla f(\boldsymbol{x}^{(0)})$ is an improvement over $\boldsymbol{x}^{(0)}$. This is the basis for the projected gradient algorithm $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k\boldsymbol{P}\nabla f(\boldsymbol{x}^{(k)})$, where the initial point $\boldsymbol{x}^{(0)}$ satisfies $\boldsymbol{A}\boldsymbol{x}^{(0)} = \boldsymbol{b}$ and α_k is some step size.

Projected Gradient Methods with Linear Constraints

- A well-known variant: *projected steepest descent algorithm*, where α_k is given by
$$\alpha_k = \arg\min_{\alpha \geq 0} f\left(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{P} \nabla f\left(\boldsymbol{x}^{(k)}\right)\right)$$
- Theorem 23.1. If $\{\boldsymbol{x}^{(k)}\}$ is the sequence of points generated by the projected steepest descent algorithm and if $\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(0)}\right) \neq 0$, then $f\left(\boldsymbol{x}^{(k+1)}\right) < f\left(\boldsymbol{x}^{(k)}\right)$.

Projected Gradient Methods with Linear Constraints

- If for some k, we have $\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(k)}\right) = \boldsymbol{0}$, then by Proposition 23.1 the point $\boldsymbol{x}^{(k)}$ satisfies the Lagrange condition. This condition can be used as a stopping criterion for the algorithm. Note that if this case, $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)}$.
- For the case where f is a convex function, the condition $\boldsymbol{P}\nabla f\left(\boldsymbol{x}^{(k)}\right) = \boldsymbol{0}$ is, in fact, equivalent to $\boldsymbol{x}^{(k)}$ being a global minimizer of f over the constraint set $\{\boldsymbol{x}: \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$.

Projected Gradient Methods with Linear Constraints

- Proposition 23.3. The point $\boldsymbol{x}^* \in R^n$ is a global minimizer of a convex function f over $\{\boldsymbol{x}: \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$ if and only if $\boldsymbol{P}\nabla f(\boldsymbol{x}^*) = \mathbf{0}$.
- Proof. We first write $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$. Then, the constraints can be written as $\boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}$, and the problem is of the form considered in earlier chapters.
- Note that $D\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{A}$. Hence, $\boldsymbol{x}^* \in R^n$ is a global minimizer of f if and only if the Lagrange condition holds (see Theorem 22.8). By Proposition 23.1, this is true if and only if $\boldsymbol{P}\nabla f(\boldsymbol{x}^*) = \mathbf{0}$, and this completes the proof.

Lagrangian Algorithm for Equality Constraints

- The basic idea is to use gradient algorithms to update simultaneously the decision variable and Lagrange multiplier vector.
- Consider the following problem

$$\begin{gathered} \mathit{minimize} \;\; f(\boldsymbol{x}) \\ \mathit{subject\ to} \;\; \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \end{gathered}$$

- where $\boldsymbol{h}: R^n \to R^m$. Recall that for this problem the Lagrangian function is given by

$$l(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{h}(\boldsymbol{x})$$

- Assume that $f, \boldsymbol{h} \in C^2$; as usual, denote the Hessian of the Lagrangian by $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$

Lagrangian Algorithm for Equality Constraints

- The Lagrangian algorithm for this problem is given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \left(\nabla f\left(\boldsymbol{x}^{(k)}\right) + D\boldsymbol{h}\left(\boldsymbol{x}^{(k)}\right)^T \boldsymbol{\lambda}^{(k)} \right)$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \beta_k \boldsymbol{h}(\boldsymbol{x}^{(k)})$$

- Notice that the update equation for $\boldsymbol{x}^{(k)}$ is a gradient algorithm for minimizing the Lagrangian with respect to its $\boldsymbol{x}$ argument, and the update equation for $\boldsymbol{\lambda}^{(k)}$ is a gradient algorithm for maximizing the Lagrangian with respect to its $\boldsymbol{\lambda}$ argument.
- Because only the gradient is used, the method is also called the *first-order Lagrangian algorithm*.

Lagrangian Algorithm for Equality Constraints

- Lemma 23.2. For the Lagrangian algorithm for updating $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\lambda}^{(k)}$, the pair $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ is a fixed point if and only if it satisfies the Lagrange condition.
- Below, we use $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ to denote a pair satisfying the Lagrange condition. Assume that $\boldsymbol{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) > 0$. Also assume that $\boldsymbol{x}^*$ is a *regular* point. With these assumptions, we are now ready to state and prove that the algorithm is locally convergent. For simplicity, we will take α_k and β_k to be fixed constants (not depending on k), denoted α and β, respectively.

Lagrangian Algorithm for Equality Constraints

- Theorem 23.2. For the Lagrangian algorithm for updating $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\lambda}^{(k)}$, provided that α and β are sufficiently small, there is a neighborhood of $(\boldsymbol{x}^{*}, \boldsymbol{\lambda}^{*})$ such that if the pair $(\boldsymbol{x}^{(0)}, \boldsymbol{\lambda}^{(0)})$ is in this neighborhood, then the algorithm converges to $(\boldsymbol{x}^{*}, \boldsymbol{\lambda}^{*})$ with at least a linear order of convergence.

Lagrangian Algorithm for Inequality Constraints

- Consider the following optimization problem with inequality constraints:

$$\begin{gathered} minimize \;\; f(\boldsymbol{x}) \\ subject\ to \;\; \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \end{gathered}$$

where $\boldsymbol{g}: R^n \to R^p$.

- Recall that for this problem the Lagraingian function is given by

$$l(\boldsymbol{x}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\mu}^T \boldsymbol{g}(\boldsymbol{x})$$

- As before, assume that $f, \boldsymbol{g} \in C^2$; as usual, denote the Hessian of the Lagrangian by $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\mu})$

Lagrangian Algorithm for Inequality Constraints

- The Lagrangian algorithm for this problem is given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \left(\nabla f\left(\boldsymbol{x}^{(k)}\right) + D\boldsymbol{g}\left(\boldsymbol{x}^{(k)}\right)^T \boldsymbol{\mu}^{(k)} \right)$$

$$\boldsymbol{\mu}^{(k+1)} = \left[\boldsymbol{\mu}^{(k)} + \beta_k \boldsymbol{g}\left(\boldsymbol{x}^{(k)}\right)\right]_+$$

where $[\cdot]_+ = \max\{\cdot, 0\}$ (applied componentwise).

- Notice that, as before, the update equation for $\boldsymbol{x}^{(k)}$ is a gradient algorithm for minimizing the Lagrangian with respect to its $\boldsymbol{x}$ argument. The update equation for $\boldsymbol{\mu}^{(k)}$ is a *projected gradient algorithm* for maximizing the Lagrangian with respect to its $\boldsymbol{\mu}$ argument. The reason for the projection is that the KKT multiplier vector is required to be nonnegative to satisfy the KKT condition.

Lagrangian Algorithm for Inequality Constraints

- Lemma 23.3. For the Lagrangian algorithm for updating $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\mu}^{(k)}$, the pair $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ is a fixed point if and only if it satisfies the KKT condition.
- As before, we use the notation $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ to denote a pair satisfying the KKT condition. Assume that $L(\boldsymbol{x}^*, \boldsymbol{\mu}^*) > 0$. Also assume that $\boldsymbol{x}^*$ is a regular point. With these assumptions, we are now ready to state and prove that the algorithm is locally convergent.

Lagrangian Algorithm for Inequality Constraints

- As before, we will take α_k and β_k to be fixed constants (not depending on k), denoted α and β, respectively. Out analysis examines the behavior of the algorithm in two phases. In the first phase, the “nonactive” multipliers decrease to zero in finite time and remain at zero thereafter. In the second phase, the $\boldsymbol{x}^{(k)}$ iterates and the “active” multipliers converge jointly to their respective solutions, with at least a linear order of convergence.

Lagrangian Algorithm for Inequality Constraints

- Theorem 23.3. For the Lagrangian algorithm for updating $\boldsymbol{x}^{(k)}$ and $\boldsymbol{\mu}^{(k)}$, provided that α and β are sufficiently small, there is a neighborhood of $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ such that if the pair $(\boldsymbol{x}^{(0)}, \boldsymbol{\mu}^{(0)})$ is in this neighborhood, then (1) the nonactive multipliers reduce to zero in finite time and remain at zero thereafter and (2) the algorithm converges to $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ with at least a linear order of convergence.

Penalty Methods