Linear search methods

Alexandre Gramfort

alexandre.gramfort@inria.fr

Master 2 Data Science, Univ. Paris Saclay
Optimisation for Data Science

About me

- Alexandre Gramfort
- Researcher
- Inria Saclay
- http://alexandre.gramfort.net
- alexandre.gramfort@inria.fr
- Research topics: machine learning, non-linear optimization, signal processing, sparse methods, applications in health care and particularly in neuroscience, open source software.

Table of Contents

Table of Contents

Why line search?

Descent algorithm reads:

$$x_{k+1} = x_k + t_k d_k, \;\; t_k \geq 0$$

where d_k is a descent direction ($\exists t_k > 0$ s.t. $f(x_{k+1}) < f(x_k)$).
In the case of gradient descent one uses:

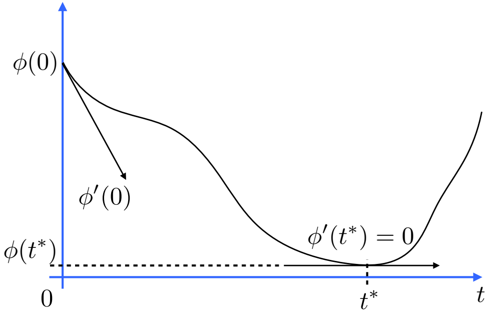
$$d_k = -\nabla F(x_k)$$

and if f has a Lipschitz continuous gradient with constant L then one can use $t_k = \frac{1}{L}$.
Problem: L is a global quantity (does not depend on x_k) and can be unknown.
Objective: Derive strategies to estimate “good enough” t_k (optimal step can be really costly in non-quadratic case).

Why line search?

Let $\phi(t) = f(x_k + td_k)$
Objective: find $t > 0$ such that $\phi(t) \leq \phi(0)$
For f is smooth, the optimal step size t^* is caracterized by:

$$\begin{cases} \phi'(t^*) = 0 & \text{(is a minimum)} \\ \phi(t) \geq \phi(t^*) \text{ for } 0 \leq t \leq t^* & \text{(decreases objective)} \end{cases}$$



Why line search?

Let

$$\phi(t) = f(x_k + td_k)$$

Objective: find $t > 0$ such that $\phi(t) \leq \phi(0)$

Exercise: Show that with $d_k = -\nabla F(x_k)$ and optimal step size $d_{k+1}^T d_k = 0$.

Security interval

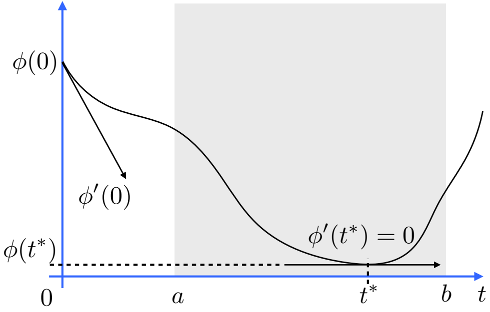
Definition (Security interval)

$[a, b]$ is a security interval if one can classify t values as:

- If $t < a$ then t is too small
- If $a \leq t \leq b$ then t is ok
- If $t > b$ then t is too big

Problem: How to translate these conditions from values of ϕ?
Problem: How to define a and b.

Security interval



Basic algorithm

Start from $[\alpha, \beta]$ with $[a, b] \subset [\alpha, \beta]$, e.g., $\alpha = 0$ and β large (always exists if f is coercive).

Basic algorithm

Start from $[\alpha, \beta]$ with $[a, b] \subset [\alpha, \beta]$, e.g., $\alpha = 0$ and β large (always exists if f is coercive).

Definition

F is coercive if

$$\lim_{\|x\| \to \infty} F(x) = +\infty$$

1. Choose t in $[\alpha, \beta]$
2. If t is too small then set $\alpha = t$ and go back to 1.
3. If t is too big then set $\beta = t$ and go back to 1.
4. If t is ok then stop

Problem: How to translate the "too small", "too big" and "ok" from values of ϕ?

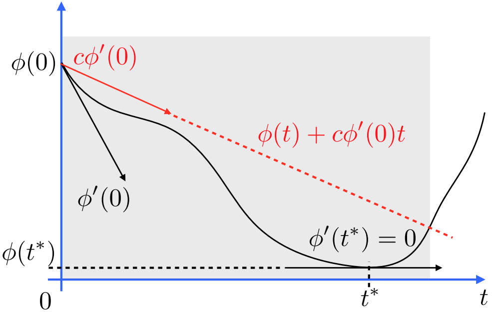
Table of Contents

Armijo's rule

Set $\alpha = 0$ and fix $0 < c < 1$.

Definition (Armijo's rule)

1. If $\phi(t) > \phi(0) + c\phi'(0)t$, then t is too big
2. If $\phi(t) \leq \phi(0) + c\phi'(0)t$, then ok



Armijo's rule

Set $\alpha = 0$ and fix $0 < c < 1$.

Definition (Armijo's rule)

1. If $\phi(t) > \phi(0) + c\phi'(0)t$, then t is too big
2. If $\phi(t) \leq \phi(0) + c\phi'(0)t$, then ok

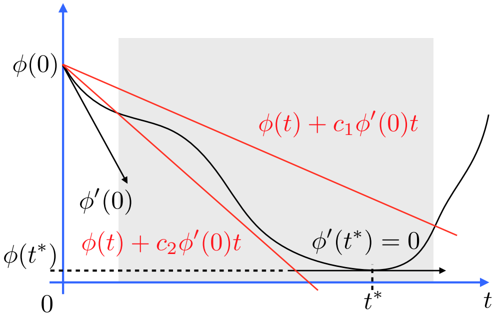
Problem: As $\alpha = 0$, t is never considered too small. So Armijo is not heavily used in practice.
Note: You have function scalar_search_armijo in scipy/optimize/linesearch.py but it does more (cubic interpolation, backtracking).

Goldstein's rule

Goldstein is Armijo with an extra inequality. Let $0 < c_1 < c_2 < 1$.

Definition (Goldstein's rule)

1. If $\phi(t) < \phi(0) + c_2\phi'(0)t$, then t is too small
2. If $\phi(t) > \phi(0) + c_1\phi'(0)t$, then t is too big
3. If $\phi(0) + c_1\phi'(0)t \geq \phi(t) \geq \phi(0) + c_2\phi'(0)t$, then ok



Goldstein's rule

c_2 should be chosen such that t^* in the quadratic case is in the security interval.
In the quadratic case:

$$\phi(t) = \frac{1}{2}at^2 + \phi'(0)t + \phi(0), a > 0$$

and t^* satisfies $\phi'(t^*) = 0$, so $t^* = -\frac{\phi'(0)}{a}$ and so

$$\phi(t^*) = \frac{\phi'(0)}{2}t^* + \phi(0)$$

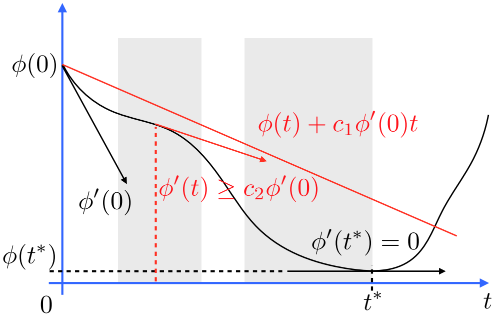
which means that one should have $c_2 \geq \frac{1}{2}$.
Common values used in practice are $c_1 = 0.1$ and $c_2 = 0.7$.

Wolfe's rule

Requires $\phi'(t) = d_k^\top \nabla f(x_k + td_k)$ (in theory more costly).

Definition: Wolfe's rule (with $0 < c_1 < c_2 < 1$)

1. If $\phi(t) > \phi(0) + c_1\phi'(0)t$, then t is too big (like Goldstein)
2. If $\phi(t) \leq \phi(0) + c_1\phi'(0)t$, and $\phi'(t) < c_2\phi'(0)$ then t is too small
3. If $\phi(t) \leq \phi(0) + c_1\phi'(0)t$, and $\phi'(t) \geq c_2\phi'(0)$, then ok



Wolfe's rule

Requires $\phi'(t) = d_k^\top \nabla f(x_k + td_k)$ (in theory more costly).

Definition: Wolfe's rule (with $0 < c_1 < c_2 < 1$)

1. If $\phi(t) > \phi(0) + c_1\phi'(0)t$, then t is too big (like Goldstein)
2. If $\phi(t) \leq \phi(0) + c_1\phi'(0)t$, and $\phi'(t) < c_2\phi'(0)$ then t is too small
3. If $\phi(t) \leq \phi(0) + c_1\phi'(0)t$, and $\phi'(t) \geq c_2\phi'(0)$, then ok

Note: The idea is to guarantee that t is not too small by requiring that the gradient is increased enough.
Note: This is implemented in scipy.optimize.line_search.

Table of Contents

Reducing security interval

First search for starting interval or first value of t ($\alpha = 0$).

1. If t is Ok then stop
2. If t is too big then set $\beta = t$ and ok.
3. If t is too small, then set t to ct with $c > 1$ and back to 1.

Reducing the interval

Multiple strategies

1. Dichotomy. Try $t = (\alpha + \beta)/2$ and then work with $[\alpha, t]$ or $[t, \beta]$
2. Polynomial approximation of ϕ, e.g., cubic approximation.

Cubic approximation

Cubic approximation is compatible with Wolfe's method which also needs ϕ'. Take 2 values t_0 and t_1 (for example α and β). Define the third order polynomial p such that:

- $p(t_0) = \phi(t_0)$
- $p(t_1) = \phi(t_1)$
- $p'(t_0) = \phi'(t_0)$
- $p'(t_1) = \phi'(t_1)$

Then propose for t the minimum of the polynomial. If it does not provide a valid t you can fallback to dichotomy.
$\rightarrow$ Demo on notebook

References

- Wright and Nocedal, Numerical Optimization, 1999, Springer, Chapter 3.