Package Pijavski
For global univariate optimization of non-convex Lipschitz functions. Version 1.0
User Manual

Gleb Beliakov
gleb@deakin.edu.au

Chapter 1

Pijavski method

We consider minimization of a univariate objective function f like the one presented on Figure 1.1. The function f needs not be smooth, but it has to be Lipschitz-continuous. If the objective function is not convex, there could be multiple local minima, as illustrated on Fig. 1.1. Locating the global minimum (on a bounded interval $[a, b]$) should be done by using a global optimization method.

If the objective function is known to be Lipschitz-continuous, and an estimate of its Lipschitz constant M is available, Pijavski-Shubert method [1] is an efficient way to find and confirm the global optimum. It works as illustrated on Fig. 1.1. We build a *saw-tooth underestimate* of the objective function f by using

$$H^K(x) = \min_{k=1,\dots,K} f(x_k) - L|x - x_k|,$$

where K denotes the iteration.

The function H^K by construction always underestimates f. At every iteration of the algorithm one value of the objective $f(x_k)$ is calculated at a suitable point x_k suggested by the algorithm, and the saw-tooth estimate is refined. The absolute minimizer of H^k is chosen as x_{k+1} for the next iteration. The sequence of global minima of the underestimates $H^K, K = 1, 2, \dots$ is known to converge to the global minimum of f.

Calculation of all local minimizers of H^K (the teeth of the saw-tooth underestimate) is done explicitly, and they are arranged into a priority queue, so that the global minimizer is always at the top. Note that there is exactly one local minimizer of H^K between each two neighboring points x_m, x_n. The computational complexity to calculate (and maintain the priority queue) the minimizers of H^K is logarithmic in K. While this method is not as fast as the Newton's method, it guarantees the globally optimal solution in the case of multiextremal objective functions, and does not suffer from lack of convergence.

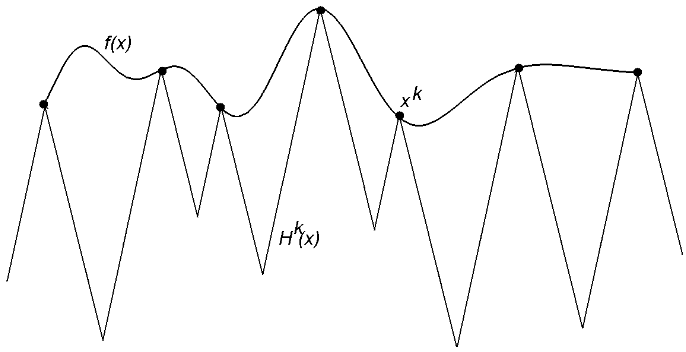

Figure 1.1: Optimization of a non-convex function with many local minima using Pijavski-Shubert method.

Chapter 2

Usage

Pijavski algorithm requires five parameters: the ends of the interval for optimization a, b, the objective function f, the limit on the number of iterations (function evaluations) and the Lipschitz constant L (or a higher value). Additionally we can supply the required tolerance *eps*.

The algorithm stops when the difference between the lowest value of f, f^*, and the lowest value of H^K is smaller than *eps*. This means that the global minimum of f cannot be smaller that $f^* - eps$.

Of course, the larger the Lipschitz constant L, the narrower are the teeth of the saw-tooth underestimate, and the more iterations are needed to reach the desired accuracy *eps*. Thus a meaningful upper limit on the number of iterations K is recommended.

To use this package, the user needs to specify the objective function f and pass it to the algorithm. Example:

```
func <- function(x,y){
y <- x * x
return(y)
}
```

Then call

```
output<-Pijavski(func, 5, -2.0, 1.0, 1000, 0.001, new.env(list(fn
= func)))
```

or with named parameters

```
output<-Pijavski(fn=func, Lips5, a=-2.0, b=1.0, iter=1000,
   prec=0.001, env=new.env(list(fn = func)))
```

The output will contain a list with these values

- **x** The global minimizer of fn.
- **value** The final value of the function being optimized.
- **precision** The precision of the result in terms of the difference of value and the lower estimate on fn.
- **iterations** Number of function evaluations performed.

Hence the global minimum is output[[2]] and the minimiser is output[[1]].

Check the precision output[[3]]. If this is negative, the Lipschitz constant supplied is too small (a value of H^K happened to be larger than some $f(x_k)$, hence the algorithm should be called with a larger value of L.

Bibliography

[1] S.A. Pijavski. An algorithm for finding the absolute extremum of a function. *USSR Comput. Math. and Math. Phys.*, 2:57–67, 1972.