# Benchmarking of quasi-Newton methods

Igor Sokolov
*Optimization Class Project. MIPT*

## Introduction

The most well-known minimization technique for unconstrained problems is Newtons Method. In each iteration, the step update is $x_{k+1} = x_k - \left(\nabla^2 f_k\right) \nabla 2 f_k$. wever, the inverse of the Hessian has to be calculated in every iteration so it takes $O\left(n^3\right)$. Moreover, in some applications, the second derivatives may be unavailable. One fix to the problem is to use a finite difference approximation to the Hessian.

We consider solving the nonlinear unconstrained minimization problem

$$\min f(x), x \in \mathbb{R}$$

Lets consider the following quadratic model of the objective function

$m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} B_k p$, where $B_k = B_k^T, B_k \succ 0$ is an $n \times n$

The minimizer $p_k$ of this convex quadratic model $p_k = -B_k^{-1} \nabla f_k$ is used as the search direction, and the new iterate is

$x_{k+1} = xk + \alpha p_k$, let $s_k = \alpha p_k$

The general structure of quasi-Newton method can be summarized as follows

- Given $x0$, $B_0$(or $H_0$), $k \to 0$;
- **For** $k = 0, 1, 2, \ldots$

  Evaluate gradient $g_k$.

  Calculate $s_k$ by line search or trust region methods.

  $x_{k+1} \leftarrow x_k + s_k$

  $y_k \leftarrow g_{k+1} - g_k$

  Update $B_{k+1}$ or $H_{k+1}$ according to the quasi-Newton formulas. **End(for)**

Basic requirement in each iteration, i.e., $B_k s_k = y_k$ (or $H_k y_k = s_k$)

## Quasi-Newton Formulas for Optimization

**BFGS**

$\min ||H - H_k||$,
s.t $H = H^T$, $Hy_k = s_k$

$H_{k+1} = (I - \rho s_k y_k^T) H_k (I - \rho y_k s_k^T) + \rho s_k s_k^T$
where $\rho = \frac{1}{y_k^T s_k}$

$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$

**DFP**

$\min ||B - B_k||$,
s.t $B = B^T$, $Bs_k = y_k$

$B_{k+1} = (I - \gamma y_k s_k^T) H_k (I - \gamma s_k y_k^T) + \gamma y_k y_k^T$
where $\gamma = \frac{1}{y_k^T s_k}$

$H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}$

**PSB**

$\min ||B - B_k||$,
s.t $(B - B_k) = (B - B_k)^T$,
$Bs_k = y_k$

$B_{k+1} = B_k - \frac{(y_k - B_k s_k) s_k^T + s_k (y_k - B_k s_k)^T}{s_k^T s_k} + \frac{s_k (y_k - B_k s_k) s_k s_k^T}{(s_k^T s_k)^2}$

$H_{k+1} = H_k - \frac{(s_k - H_k y_k) y_k^T + y_k (s_k - H_k y_k)^T}{y_k^T y_k} + \frac{s_k (s_k - H_k y_k) y_k y_k^T}{(y_k^T y_k)^2}$

**SR1**

$B_{k+1} = B_k + \sigma \nu \nu^T$,
s.t $B_{k+1} s_k = y_k$

$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}$,

$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}$

| Method | Advantages | Disadvantages |
|---|---|---|
| BFGS | • $H_0 \succ 0$ hence if $H_0 \succ 0$<br>• self correcting property if Wolfe chosen<br>• superlinear convergence | • $y_k^T s_k \approx 0$ formula produce bad results<br>• sensitive to round-off error<br>• sensitive bad line search<br>• can get stuck in saddle point |
| DFP | • can be highly inefficient at correcting large eigenvalues of matrices | • sensitive to round-off error<br>• sensitive bad line search<br>• can get stuck in saddle point |
| PSB | • superlinear convergence | • sensitive to round-off error<br>• can get stuck in saddle point |
| SR1 | • garantees to be $B_{k+1} \succ 0$ even if $s_k y_k > 0$ doesn't satisfied<br>• very good approximations to the Hessian matrices, often better than BFGS | • sensitive to round-off error<br>• can get stuck in saddle point |

## Line Search Vs. Trust Region

- Line search

  $f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla_k^T p_k$

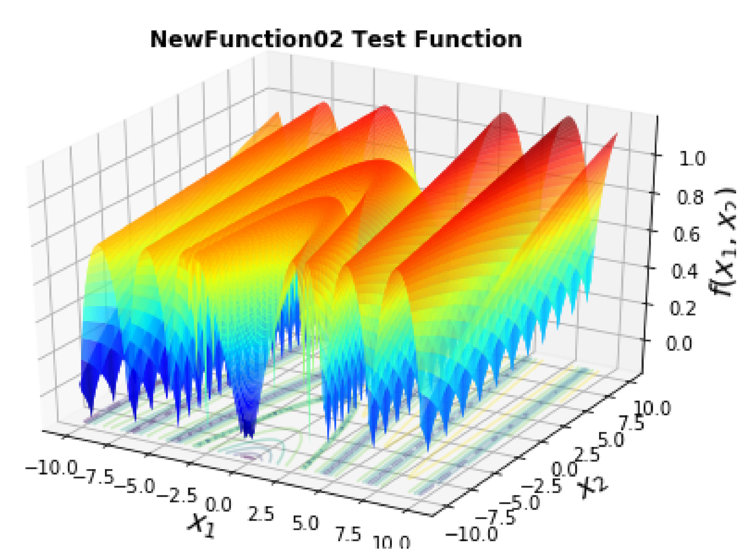
  $|f(x_k + \alpha_k p_k)^T p_k| \leq c_2 |\nabla f_k^T p_k|$
- Trust region

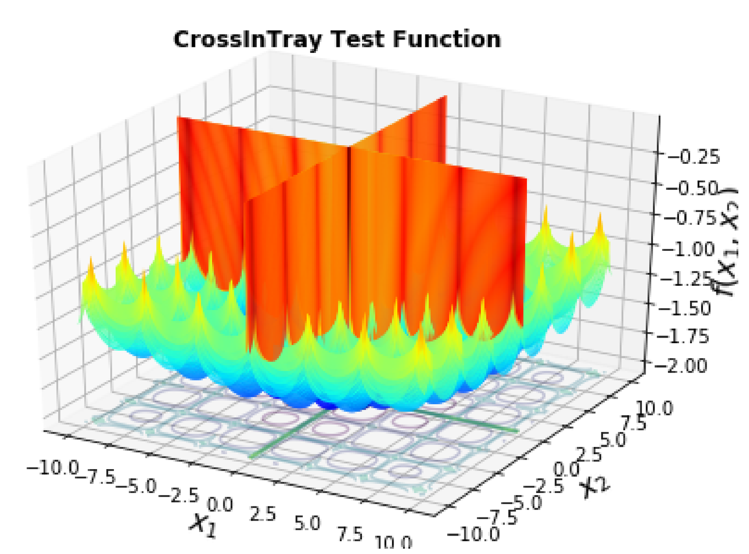
  Both direction and step size find from solving

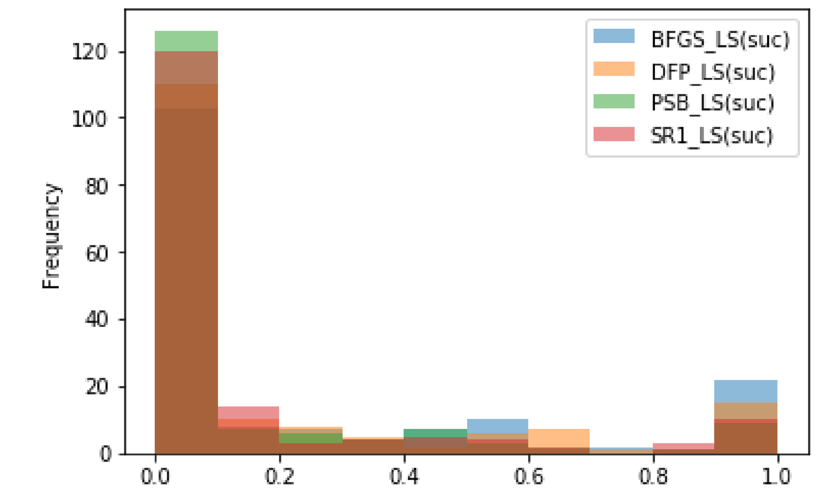
  $\min_{p \in \mathbb{R}^n} m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} B_k p \quad ||p|| \leq \Delta_k$

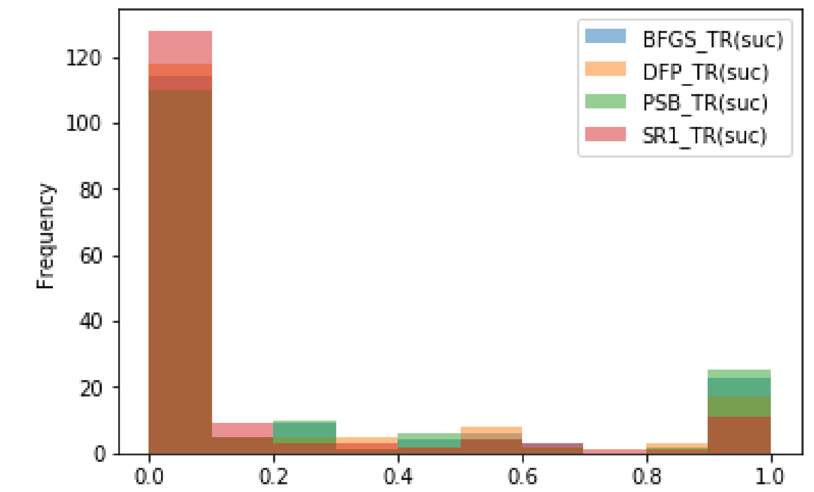
## Numerical Results

- All quasi-newton methods with two strategies (8 algorithms) were implemented in Python
- 165 various $N - d (N \geq 2)$ strong benchmark problems
- For each algorithm all problems were launched from random point of domain 50 times and results were averaged



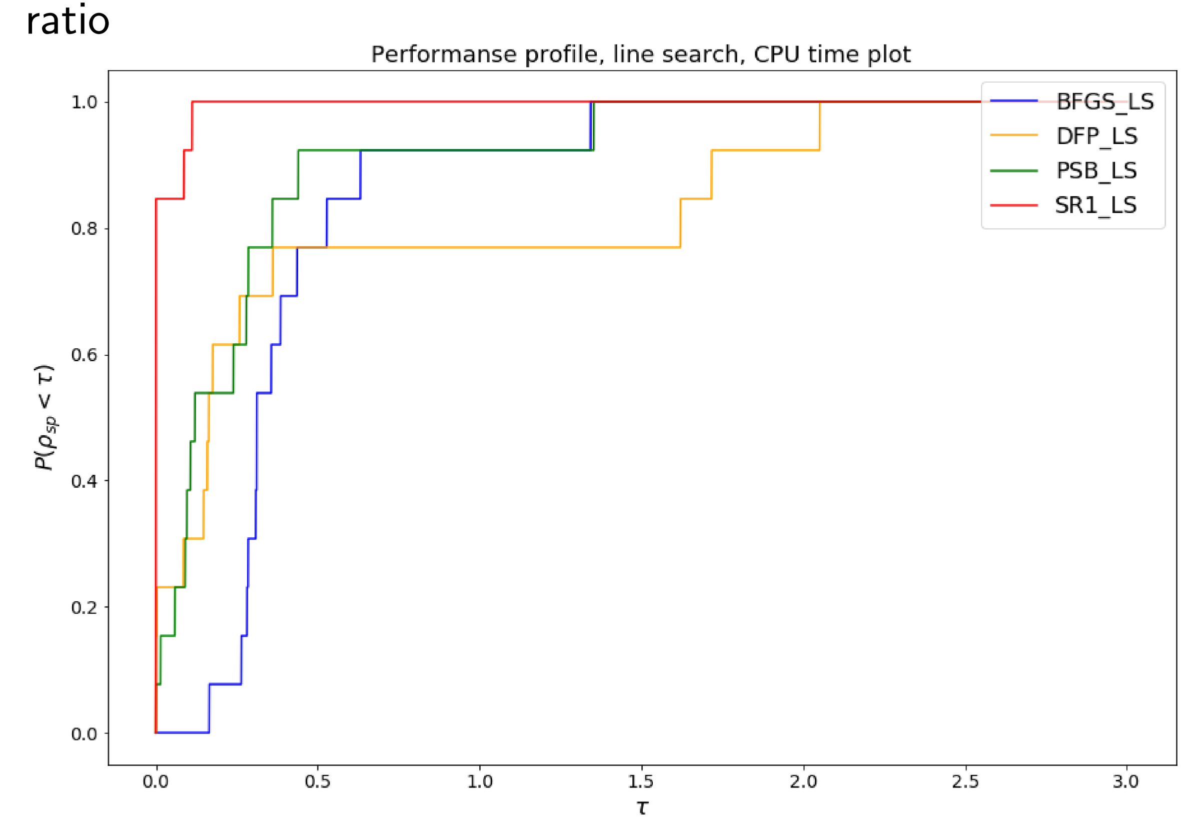



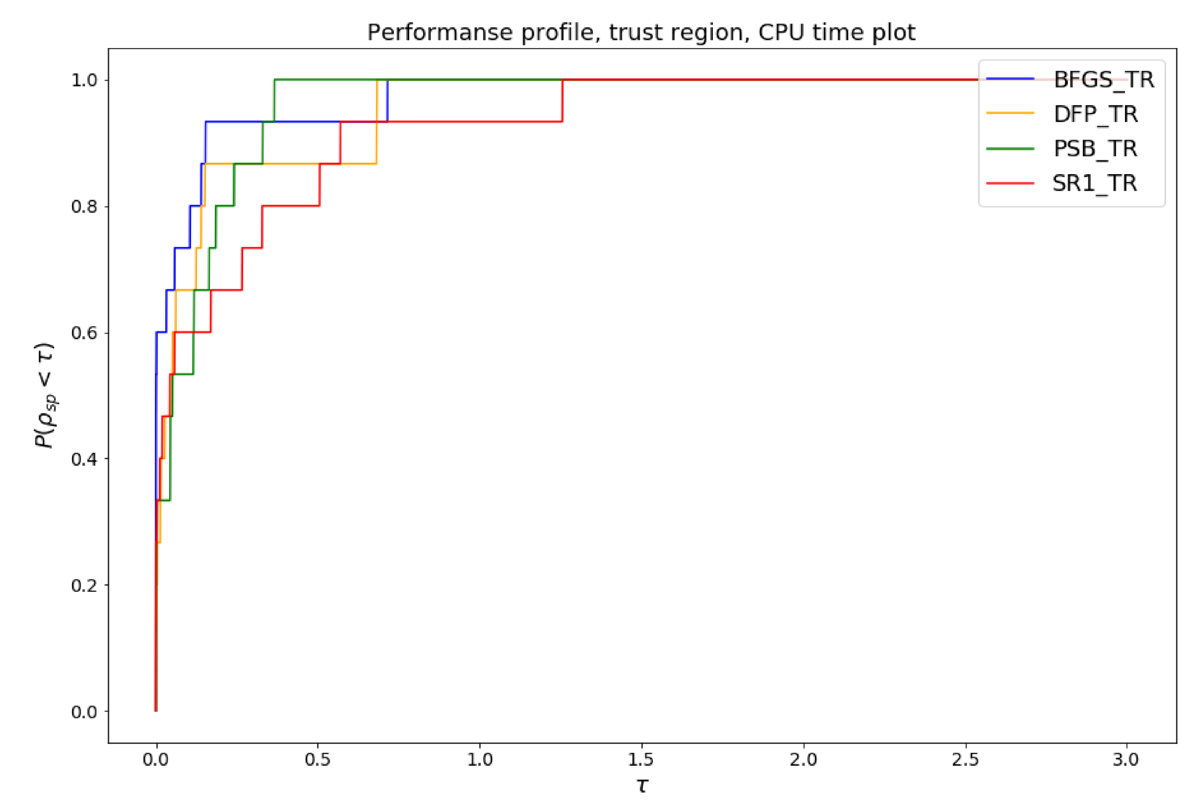




| Strategy | BFGS | DFP | PSB | SR1 | Total |
|---|---|---|---|---|---|
| Line search | 0.2 | 0.17 | 0.1 | 0.1 | 15(0.09) |
| Trust region | 0.18 | 0.16 | 0.19 | 0.11 | 15(0.09) |

Performance evaluation: $n_s$ - number of solvers, $n_p$ - number of problems, $t_{s,p}$ - time, $r_{s,p} = \frac{t_{s,p}}{\min\{t_{s,p} : s \in S\}}$ - performance profile function

$\rho_s(\tau) = \frac{1}{n_p} size\{p : 1 \leq p \leq n_p, \log(r_{s,p} \leq \tau)\}$ - defines the probability for solver $s$ that the performance ratio $r_{s,p}$ is within a factor $\tau$ of the best possible ratio




## Conclusions and Further Work

## Acknowledgements