Prof. Dr. Karsten Urban
M. Sc. Lewin Ernst

Winter term 2024/2025
Sheet 05, 30 Points

Numerical Optimization
Solution to exercise sheet

review on 20.11.2024 during the exercise class

1. *(Convergence of nonlinear conjugate gradient method)*
In practice, the nonlinear cg-method is appealing for large nonlinear optimization problems, because each iteration requires only the evaluation of the objective function and its gradient. No matrix operations are required for the step computation, and just a few vectors have to be stored. One can prove a global convergence theorem (for the restarted version of the nonlinear cg-method, i.e. $\beta_{FR}^{(k)} \leftarrow 0$ for some k) based on Theorem 2.3.13 (Zoutendijk). For the Theorem to apply we need to choose decent directions $p^{(k)}$, i.e.

$$\nabla f(x^{(k)})^T p^{(k)} = -\|\nabla f(x^{(k)})\|^2 + \beta_{FR}^{(k)} \nabla f(x^{(k)})^T p^{(k-1)} < 0, \quad \forall k \in \mathbb{N}_0. \tag{1}$$

With the Lemma below and the strong Wolfe conditions with $0 < c_1 < c_2 < \frac{1}{2}$ one can prove that (1) is fulfilled and with the Theorem of Zoutendijk one conclude that the nonlinear cg-method is convergent.

Lemma 1. *Let the step sizes $\alpha^{(k)}$ of Algorithm 2.3.7 satisfy the strong Wolfe conditions with $0 < c_1 < c_2 < \frac{1}{2}$. Then the method generates directions $p^{(k)}$ that satisfy the following inequalities*

$$-\frac{1}{1-c_2} \leq \frac{\nabla f(x^{(k)})^T p^{(k)}}{\|\nabla f(x^{(k)})\|^2} \leq \frac{2c_2 - 1}{1 - c_2}, \quad \forall k \in \mathbb{N}_0 \tag{2}$$

a) Proof Lemma 1 by induction.

b) Show that for $0 < c_2 < \frac{1}{2}$, we have

$$-1 < \frac{2c_2 - 1}{1 - c_2} < 0.$$

Remark: This means we have descent directions $p^{(k)}$.

c) Why does convergence by Zoutendijk not directly follow? Argue with $\cos\theta^{(k)}$.

(6 + 2 + 4 = 12 Points)

Solution:

a) For $k = 0$, we use $p^{(0)} = -\nabla f(x^{(0)})$ and therefore

$$-\frac{1}{1-c_2} < -1 < \frac{2c_2 - 1}{1 - c_2}.$$

Now assume that (2) holds for some $k \geq 1$. With Line 8 and Line 9 of Algorithm 2.3.7 we have

$$\frac{\nabla f(x^{(k+1)})^T p^{(k+1)}}{\|\nabla f(x^{(k+1)})\|^2} = -1 + \beta_{FR}^{(k+1)} \frac{\nabla f(x^{(k+1)})^T p^{(k)}}{\|\nabla f(x^{(k+1)})\|^2} = -1 + \frac{\nabla f(x^{(k+1)})^T p^{(k)}}{\|\nabla f(x^{(k)})\|^2}.$$

Equation (2.3.8) of the strong Wolfe conditions states

$$|\nabla f(x^{(k+1)})p^{(k)}| \leq -c_2 \nabla f(x^{(k)})p^{(k)}$$

with which we deduce

$$-1 + c_2 \frac{\nabla f(x^{(k)})^T p^{(k)}}{\|\nabla f(x^{(k)})\|^2} \leq \frac{\nabla f(x^{(k+1)})^T p^{(k+1)}}{\|\nabla f(x^{(k+1)})\|^2} \leq -1 - c_2 \frac{\nabla f(x^{(k)})^T p^{(k)}}{\|\nabla f(x^{(k)})\|^2}.$$

Using the left hand side of the induction hypothesis to further lower and upper bound this gives us

$$-1 - \frac{c_2}{1-c_2} \leq \frac{\nabla f(x^{(k+1)})^T p^{(k+1)}}{\|\nabla f(x^{(k+1)})\|^2} \leq -1 + \frac{c_2}{1-c_2}.$$

A reformulation of the latter inequalities proves the claim.

b) We have

$$2c_2 - 1 \overset{0<c_2<1/2}{<} 0 < 1 - 2c_2 < 1 - c_2$$

which gives the desired inequalities.

c) From the Zoutendijk condition it follows that

$$(\cos\theta^{(k)})^2 \|\nabla f(^{(k)})\|^2 \to 0 \quad \text{for} \quad k \to \infty,$$

where

$$\cos\theta^{(k)} = \frac{-\nabla f(x^{(k)})^T p^{(k)}}{\|\nabla f(x^{(k)})\| \|p^{(k)}\|}.$$

The convergence would follow if we could show that there is a $\delta > 0$ such that $\cos\theta^{(k)} \geq \delta$ for all $k \in \mathbb{N}_0$. Unfortunately, multiplying (2) by $-\frac{\|\nabla f(x^{(k)})\|}{\|p^{(k)}\|}$ gives us

$$\frac{1-2c_2}{1-c_2} \frac{\|\nabla f(x^{(k)})\|}{\|p^{(k)}\|} \leq \cos\theta^{(k)} \leq \frac{1}{1-c_2} \frac{\|\nabla f(x^{(k)})\|}{\|p^{(k)}\|},$$

which means that $\cos\theta^{(k)} \approx 0$ if and only if $\|\nabla f(x^{(k)})\| \ll \|p^{(k)}\|$. It is a-priori not clear if the algorithm prevents this.

2. *(Convergence of trust-region methods)*
Trust-region methods are powerful global optimization schemes. The idea is in order to determine the next step $x^{(k+1)} = x^{(k)} + p^{(k)}$ we minimize the quadratic model

$$q^{(k)}(p) := f^{(k)} + (g^{(k)})^T p + \frac{1}{2} p^T H^{(k)} p,$$

where we define $f^{(k)} := f(x^{(k)})$, $g^{(k)} := \nabla f(x^{(k)})$ and $H^{(k)} := \nabla^2 f(x^{(k)})$, within a trust-region

$$B_{\Delta^{(k)}} := \{p \in \mathbb{R}^n : \|p\| \leq \Delta^{(k)}\}.$$

Therefore, we have to solve

$$\min_{p \in \mathbb{R}^n} q^{(k)}(p), \quad \text{s.t. } \|p\| \leq \Delta^{(k)}$$

in every iteration of the algorithm. Although, this problem can be solved efficiently the trust-region method is globally convergent when the approximate solution $p^{(k)}$ lies within the trust region and gives a *sufficient* reduction in the model. The *sufficient* reduction can be quantified

in terms of the Cauchy point/Cauchy step (see Assumption 2.4.2). We can calculate the Cauchy point $p_C^{(k)}$ as follows. Determine the solution $p_S^{(k)}$ of the linear problem

$$\min_{p\in\mathbb{R}^n} f^{(k)} + (g^{(k)})^T p, \quad \text{s.t. } \|p\| \leq \Delta^{(k)}$$

and then calculate the solution $\tau_C^{(k)} > 0$ of

$$\min_{\tau^{(k)}>0} q^{(k)}(\tau^{(k)} p_S^{(k)}), \quad \text{s.t. } \|\tau^{(k)} p_S^{(k)}\| \leq \Delta^{(k)}.$$

a) Show that

$$p_S^{(k)} = -\frac{\Delta^{(k)}}{\|g^{(k)}\|} g^{(k)}$$

and

$$p_C^{(k)} = -\tau^{(k)} \frac{\Delta^{(k)}}{\|g^{(k)}\|} g^{(k)},$$

where

$$\tau^{(k)} = \begin{cases} 1, & \text{if } (g^{(k)})^T H^{(k)} g^{(k)} \leq 0 \\ \min\{1, \|g^{(k)}\|^3 / \left(\Delta^{(k)} (g^{(k)})^T H^{(k)} g^{(k)}\right), & \text{otherwise} \end{cases}$$

Remark: This means that the computation of the Cauchy point is rather easy and we can check if Assumption 2.4.2 is fulfilled.

b) The proof of the global convergence theorem takes several auxiliary results. Therefore, we only prove the following: Let $f \in C^1(\mathbb{R}^n)$. If the trust-region algorithm produces a sequence $(\|g^{(k)}\|)_{k\in\mathbb{N}}$ with

$$\liminf_{k\to\infty} \|g^{(k)}\| = 0$$

and the iterates $x^{(k)}$ stay in a bounded set Ω, then there is *a* limit point $\bar{x}$ of $(x^{(k)})_{k\in\mathbb{N}}$ such that

$$g(\bar{x}) := \nabla f(\bar{x}) = 0.$$

Remark: Note that this is a weaker statement than Theorem 2.4.5 eq. (2.4.6).

(6 + 6 = 12 Points)

Solution:

a) We first solve the linear optimizatin problem

$$\min_{p\in\mathbb{R}^n} f^{(k)} + (g^{(k)})^T p, \quad \text{s.t. } \|p\| \leq \Delta^{(k)}.$$

We notice that the affine linear function $h(p) := f^{(k)} + (g^{(k)})^T p$ is convex on the convex set $\{p \in \mathbb{R}^n : \|p\| \leq \Delta^{(k)}\}$, so therefore every local minima is a global one and there is none within the disk $\|p\| < \Delta^{(k)}$. Therefore, there has to be one on the boundary $\|p\| = \Delta^{(k)}$ (existence is clear from the fact that the function is continuous on a closed and bounded set). By Theorem 2.3.5 and rescaling to the radius $\Delta^{(k)}$ we get

$$p_S^{(k)} = -\frac{\Delta^{(k)}}{\|g^{(k)}\|} g^{(k)}.$$

Now we turn to the univariate quadratic minimization problem

$$\min_{\tau^{(k)}>0} q^{(k)}(\tau^{(k)} p_S^{(k)}), \quad \text{s.t. } \|\tau^{(k)} p_S^{(k)}\| \leq \Delta^{(k)}.$$

It is clear that if the given $\tau^{(k)}$ solves this problem, the given Cauchy point $p_C^{(k)}$ solves the minimization problem in Assumption 2.4.2. The scalar funtion (after dropping the superscript (k)) is given by

$$\Phi(\tau) = f - \Delta\|g\|\tau + \frac{1}{2}\Delta^2 g^T H g / \|g\|^2 \tau^2.$$

The condition $\|\tau p_S^{(k)}\| \leq \Delta^{(k)}$ and the definition of $p_S^{(k)}$ imply $0 < \tau \leq 1$. If $g^T H g = 0$, the value $\tau = 1$ will suffice. If $g^T H g \neq 0$, then Φ is a parabola with critical point

$$\tau = \frac{\Delta\|g\|}{\Delta^2 g^T H g / \|g\|^2} = \frac{\|g\|^3}{\Delta g^T H g}$$

If $g^T H g < 0$ this is a maximum and due to $\Phi(0) > \Phi(1)$ we have in $\tau = 1$ the minimizer. When $g^T H g > 0$ the value above is positive and a minimizer, which we have to cut off in the case it exceeds 1. Therefore $\tau^{(k)}$ is given as stated.

b) From

$$\liminf_{k\to\infty} \|g^{(k)}\| = 0,$$

we deduce that there is a convergent subsequence $(a_{k_n})_{n\in\mathbb{N}}$ of $(\|g^{(k)}\|)_{k\in\mathbb{N}}$ such that

$$\lim_{n\to\infty} a_{k_n} = 0.$$

Since $(x_{k_n})_{n\in\mathbb{N}} \subset \Omega$ and Ω is bounded, there is again a convergent subsequence $(x_{k_{n_i}})_{i\in\mathbb{N}}$ with limit point $\bar{x}$. By continuity of g and the norm $\|\cdot\|$ we have $\|g(\bar{x})\| = 0$ and hence $g(\bar{x}) = 0$.

3. *(Nonlinear cg-method,* MATLAB*)*
Implement the nonlinear cg-method after Polak-Ribiére with a line search of your choice. Minimize the Rosenbrock function and compare the convergence behavior with one of the derivative-free algorithms and Newtons' method (with a line search algorithm of your choice). This should give you a feeling for the algorithms, which use no derivative information, only the gradient and also second order derivative information.

(6 Points)

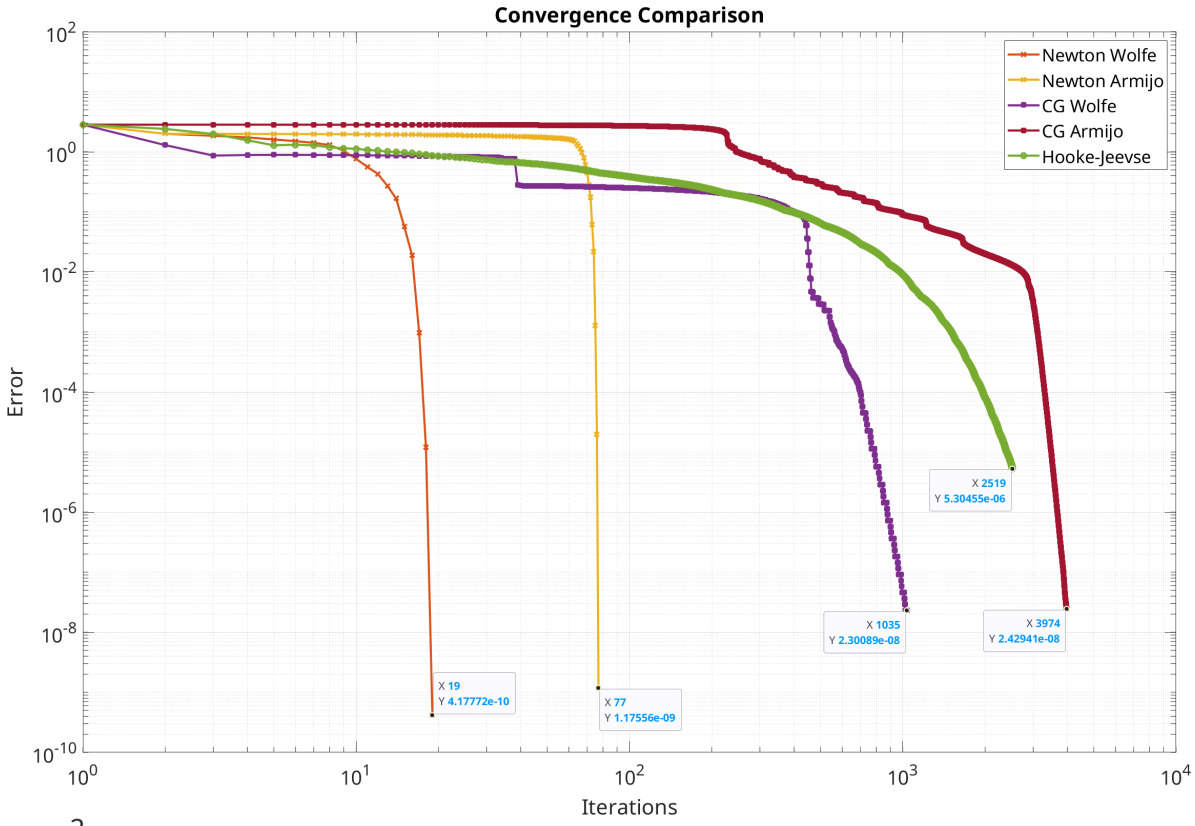
Solution: The MATLAB script could look like

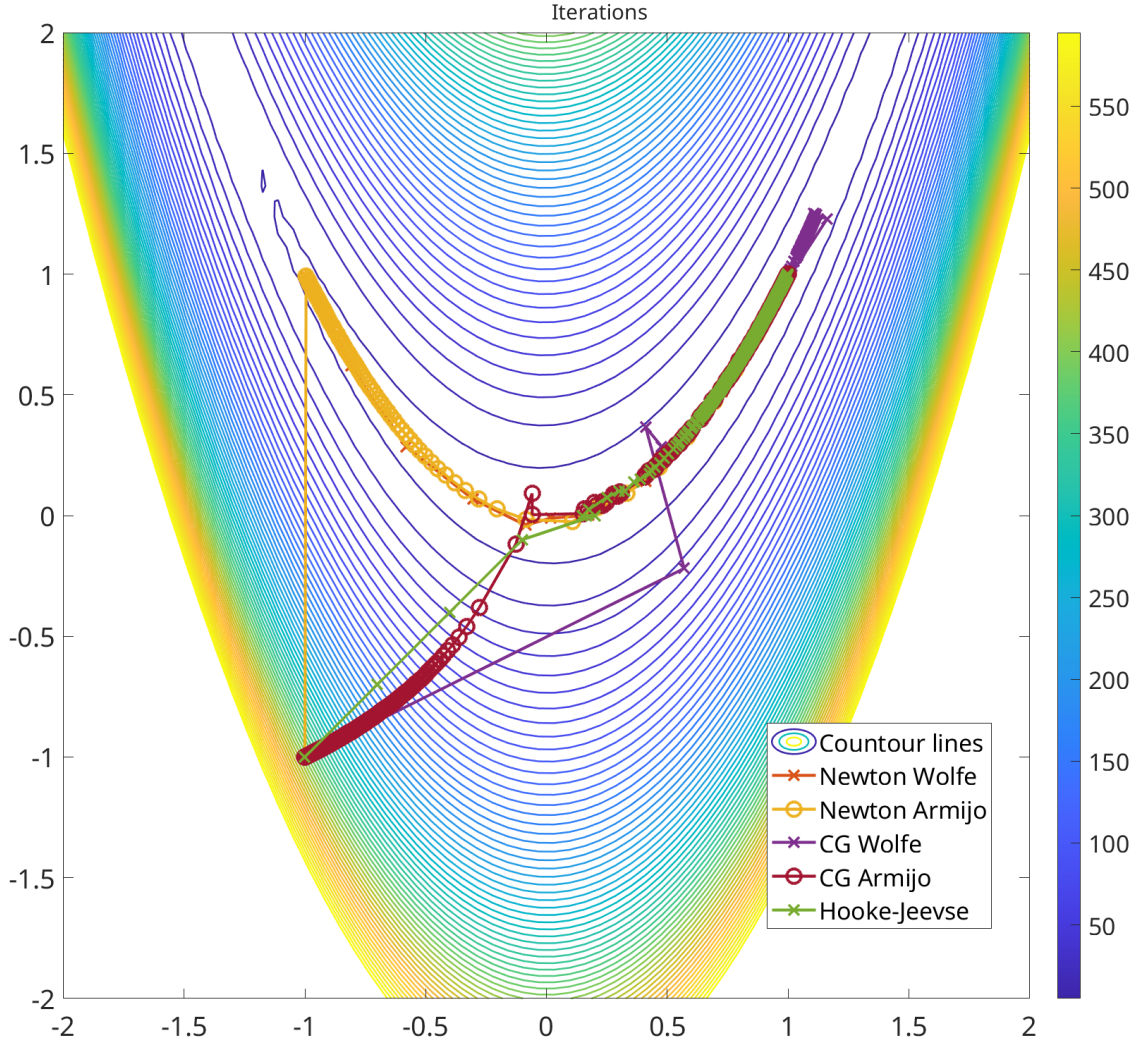
```
% Newton method using Armijo stepsize rule.
% It terminates when the norm of the gradient is below 10^(-6).
clear, close all
clc

% Rosenbrock function and derivative
Rosenbrock = @(x,y) (1-x).^2 + 100*(y-x.^2).^2;
f  = @(x) Rosenbrock(x(1),x(2));
gradf = @(x) [-400*(x(2)-x(1)^2)*x(1)-2*(1-x(1));
            200*(x(2)-x(1)^2)];
Hessf = @(x)  [1200*x(1)^2-400*x(2)+2, -400*x(1);...
                        -400*x(1), 200];
```

```
% Netwon settings
maxIter = 15000;
tol     = 1e-8;
% initial value
x0 = [-1, -1]';

% Wolfe stepsize rule parameters
alpha_newton     = 1;
alpha_max        = 2;
c1               = 1e-4;
c2               = 0.4;
alpha0           = 1;
delta_min        = 1e-2;
delta_max        = 5;
maxIter_stepsize = 1000;
% Hooke Jeeves
h0 = 0.1*ones(2,1);


[allx_newton_armijo, iter_newton_armijo] = newton_armijo(f, gradf, Hessf, ...
                                    x0, ...
                                    maxIter, tol, ...
                                    alpha0, c1, delta_min, delta_max, ...
                                    maxIter_stepsize);

[allx_newton_wolfe, iter_newton_wolfe] = newton_wolfe(f, gradf, Hessf, ...
                                        x0, ...
                                        maxIter, tol, ...
                                        alpha_newton, alpha_max, c1, c2);

[allx_cg_wolfe, iter_cg_wolfe] = nonlinear_cg_wolfe(f, gradf, ...
                                   x0, tol, maxIter, ...
                                   1, alpha_max, c1, c2);

[allx_cg_armijo, iter_cg_armijo] = nonlinear_cg_armijo(f, gradf, ...
                                   x0, ...
                                   maxIter, tol, ...
                                    0.1, c1, delta_min, delta_max, ...
                                    maxIter_stepsize);

[allx_hooke, iter_hooke] = hooke_jeeves(f, x0, h0, tol, maxIter);

x = linspace(-2,2,64);
y = linspace(-2,2,64);
[xx,yy] = meshgrid(x,y);
ff = Rosenbrock(xx,yy);
levels = 5:10:600;
figure(1)
contour(x,y,ff,levels,LineWidth=1.2),
colorbar
axis([-2 2 -2 2]),
axis square,
hold on
plot(allx_newton_wolfe(1,:), ...
     allx_newton_wolfe(2,:),'x-','Color', '#D95319', ...
     MarkerSize=10, LineWidth=2);
plot(allx_newton_armijo(1,:), ...
     allx_newton_armijo(2,:),'o-','Color', '#EDB120', ...
     MarkerSize=10, LineWidth=2);
plot(allx_cg_wolfe(1,:),allx_cg_wolfe(2,:),...
    'x-','Color', '#7E2F8E', MarkerSize=10, LineWidth=2);
plot(allx_cg_armijo(1,:),allx_cg_armijo(2,:),...
```

```
    'o-','Color', '#A2142F', MarkerSize=10, LineWidth=2);
plot(allx_hooke(1,:),allx_hooke(2,:),...
    'x-','Color', '#77AC30', MarkerSize=10, LineWidth=2);
set(gca, "FontSize", 18)
legend("Countour lines","Newton Wolfe", ...
    "Newton Armijo", "CG Wolfe", "CG Armijo", "Hooke-Jeevse")
hold off

figure(2)
loglog(1:iter_newton_wolfe+1, vecnorm(allx_newton_wolfe - [1;1],2,1),...
       'x-','Color', '#D95319', MarkerSize=5, LineWidth=2)
hold on
loglog(1:iter_newton_armijo+1, vecnorm(allx_newton_armijo - [1;1],2,1),...
       'x-','Color', '#EDB120', MarkerSize=5, LineWidth=2)
loglog(1:iter_cg_wolfe+1, vecnorm(allx_cg_wolfe - [1;1],2,1),...
       '*-','Color', '#7E2F8E', MarkerSize=5, LineWidth=2)
loglog(1:iter_cg_armijo+1, vecnorm(allx_cg_armijo - [1;1],2,1),...
       '*-','Color', '#A2142F', MarkerSize=5, LineWidth=2)
loglog(1:iter_hooke, vecnorm(allx_hooke - [1;1],2,1),...
       'o-','Color', '#77AC30', MarkerSize=5, LineWidth=2)
grid on
title("Convergence Comparison")
xlabel("Iterations")
ylabel("Error")
legend("Newton Wolfe", "Newton Armijo", "CG Wolfe", "CG Armijo", "Hooke-Jeevse")
set(gca, "FontSize", 18)
```

and the plots could look like


Convergence Comparison
Newton Wolfe
Newton Armijo
CG Wolfe
CG Armijo
Hooke-Jeevse
Error
Iterations
X 19
Y 4.17772e-10
X 77
Y 1.17556e-09
X 1035
Y 2.30089e-08
X 2519
Y 5.30455e-06
X 3974
Y 2.42941e-08


Countour lines
Newton Wolfe
Newton Armijo
CG Wolfe
CG Armijo
Hooke-Jeevse