Exercise sheet 1 for Numerical Optimization

Dr. Michael Lehn Tobias Born

Deadline: October 22, 2025

Exercise 1 (**Nonlinear least squares problems**, 5+5 points)

In the lecture Numerical Linear Algebra we considered least squares problems of the form

$$\text{find} \quad \hat{x} \in \mathbb{R}^n \quad \text{such that} \quad \|A\hat{x} - b\|_2 = \min_{x \in \mathbb{R}^n} \|Ax - b\|_2 \tag{1}$$

with $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and $m \geq n$, so an overdetermined system of linear equations. If $\operatorname{rank}(A) = n$, (1) admits a unique solution. This solution can, e.g., be computed using the QR-decomposition of A.
Now we generalize least squares problems on an open set $\Omega \subset \mathbb{R}^n$ by replacing $Ax - b$ by a nonlinear function $F : \Omega \to \mathbb{R}^m$ with

$$F(x) = \begin{pmatrix} F_1(x) \\ \vdots \\ F_m(x) \end{pmatrix}$$

and get

$$\text{find} \quad \hat{x} \in \Omega \quad \text{such that} \quad \|F(\hat{x})\|_2 = \min_{x \in \Omega} \|F(x)\|_2 \; . \tag{2}$$

a) Consider $g : \Omega \subset \mathbb{R}^n \to \mathbb{R}$ with $g(x) := \frac{1}{2} F(x)^T F(x)$ and the optimization problem

$$\text{find} \quad \hat{x} \in \Omega \quad \text{such that} \quad g(\hat{x}) = \min_{x \in \Omega} g(x) \; . \tag{3}$$

Show that the two optimization problems (2) and (3) are equivalent.

If a minimum of g on Ω exists, it has to be a local interior minimum as Ω is open. Assume now that $F \in C^2(\Omega, \mathbb{R}^m)$, then $g \in C^2(\Omega)$ and $\hat{x}$ is a stationary point if $g'(\hat{x}) = 0$. Sufficient optimality conditions for a local minimum would consider $g''(\hat{x})$.

b) Determine

$$g'(x) = \begin{pmatrix} \frac{\partial g(x)}{\partial x_1} & \dots & \frac{\partial g(x)}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{1 \times n}$$

and

$$g''(x) = \begin{pmatrix} \frac{\partial^2 g(x)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 g(x)}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 g(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 g(x)}{\partial x_n \partial x_n} \end{pmatrix} \in \mathbb{R}^{n \times n}$$

using the derivative of F

$$F'(x) = \begin{pmatrix} \frac{\partial F_1(x)}{\partial x_1} & \cdots & \frac{\partial F_1(x)}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial F_m(x)}{\partial x_1} & \cdots & \frac{\partial F_m(x)}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{m \times n}$$

and the second derivative of the i-th component F_i of F

$$F_i''(x) = \begin{pmatrix} \frac{\partial F_i(x)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial F_i(x)}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial F_i(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial F_i(x)}{\partial x_n \partial x_n} \end{pmatrix} \in \mathbb{R}^{n \times n}$$

for $i = 1, \ldots, m$.

Exercise 2 (**Gauß-Newton method**, 7+5+3+2 points)

The idea of the Gauß-Newton method is to find a stationary point $\hat{x}$ with $g'(\hat{x}) = 0$. This is a necessary condition for finding a local minimum, but not a sufficient one. One has to solve the system of nonlinear equations

$$g'(x) = F(x)^T F'(x) = 0 \; . \tag{4}$$

As $F \in C^2(\Omega, \mathbb{R}^m)$ it is $g' \in C^1(\Omega, \mathbb{R}^{1\times n})$ and consequently (4) can be solved using Newton's method (recall Numerical Analysis). The Newton correction $s_k \in \mathbb{R}^n$ in the (k+1)-st iteration step is the result of the n-dimensional system of linear equations

$$g''(x_k)s_k = -g'(x_k)^T$$

and the Newton update

$$x_{k+1} = x_k + s_k \; .$$

The correction therefore takes the form

$$s_k = -\left(g''(x_k)\right)^{-1} g'(x_k)^T = -\left(F'(x_k)^T F'(x_k) + \sum_{l=1}^{m} F_l(x_k) F_l''(x_k)\right)^{-1} F'(x_k)^T F(x_k) \; .$$

Evaluating the second derivatives $F_1'', \dots, F_m''$ at every iterate x_k is potentially very costly, so they are neglected and the Newton correction simplifies to

$$s_k = -\left(F'(x_k)^T F'(x_k)\right)^{-1} F'(x_k)^T F(x_k) \; .$$

Algorithm 1: Gauß-Newton method

```
1 Input: initial value $x_0 \in \mathbb{R}^n$, $F \in C^2(\Omega, \mathbb{R}^m)$, $F' \in C^1(\Omega, \mathbb{R}^{m\times n})$
2 Output: approximation $x_k$ of a solution to problem (2)
3 for $k \leftarrow 0, 1, \dots$ do
4     compute $F_{(k)} \leftarrow F(x_k)$ and $F'_{(k)} \leftarrow F'(x_k)$
5     solve system of linear equations $s_k \leftarrow -\left(\left(F'_{(k)}\right)^T F'_{(k)}\right)^{-1} \left(F'_{(k)}\right)^T F_{(k)}$
6     $x_{k+1} \leftarrow x_k + s_k$
```

Apart from simplifying the Newton correction, the Gauß-Newton method basically is Newton's method applied to a derivative such that it searches for a stationary point.

a) Create a MATLAB programme `gauss_newton.m` that gets inputs `x0`, a function handle `F`, a function handle `dF`, a tolerance `tol` as well as a maximum number of iteration steps `maxit` and implements the Gauß-Newton method. The programme is supposed to terminate when either $\|s_k\|_2 <$`tol` or when `maxit` iteration steps have been carried out.
Note: You can use the MATLAB operator \ in order to solve the system of linear equations. But keep in mind that a big part of the lecture Numerical Linear Algebra dealt with this kind of problems.

b) We want to test the MATLAB programme on an example. Consider two-dimensional data points $(x_i, y_i) \in \mathbb{R}^2$, $i = 1, \dots, m$, which are roughly arranged in a circle defined by the centre $(x_c, y_c) \in \mathbb{R}^2$ and radius $r > 0$. If all data points would exactly lie on this circle, it would hold that

$$(x_i - x_c)^2 + (y_i - y_c)^2 - r^2 = 0 \quad \forall i = 1, \dots, m \; .$$

But as they only approximate the circle, in general it is

$$(x_i - x_c)^2 + (y_i - y_c)^2 - r^2 \neq 0 \quad \forall i = 1, \dots, m \ .$$

Now we want to find a centre (x_c, y_c) and a radius r that minimize the cumulated squared deviations

$$\sum_{i=1}^{m} \left((x_i - x_c)^2 + (y_i - y_c)^2 - r^2 \right)^2 \ . \tag{5}$$

Find a function F such that minimizing (5) corresponds to a problem in the form of (2). Then also determine its derivative F'.

c) Now we will test `gauss_newton.m`. Therefore execute the programme `regression_circle.m`. It creates random points arranged in a rough circle, solves the respective nonlinear least squares problem using `gauss_newton.m` and finally plots as well the data points as the determined circle.
However, before that, you have to fill in the gaps in `regression_circle.m`. Then execute the programme and evaluate the result. Just pay attention to Figure 1, Figure 2 will be important in the next exercise.

d) In `regression_circle.m` set the initial value to $x_0 = (1, 1, 1)^T$ and execute the programme several times. What do you observe?
Have a look at Figure 2 and think about what it says in terms of the existence of a unique solution of the optimization problem.
Add one line of code to `regression_circle.m` such that there is no longer an error thrown, regardless of which local minimum is found.

Some statements about the Gauß-Newton method:

- In general, the underlying problems of the form (2) do not have a unique solution.
- When the method converges to a point, it is only a stationary point, so a point with zero derivative. This does not have to be a minimum, it could also be a saddle point or a maximum. That is because (4) only includes the necessary optimality criterion, not a sufficient one. To ensure a local minimum, more information about the individual problem is needed.
- For the Newton correction to be computable, the matrix $(F'(x_k))^T F'(x_k)$ has to be invertible and therefore $F'(x_k) \in \mathbb{R}^{m \times n}$ with $m \geq n$ has to have full rank n for every iterate x_k, $k = 0, 1, \dots$. This must be assumed.
- Recall from Numerical Analysis that under certain assumptions, Newton's method converges within an environment around the zero point. But as for the Gauß-Newton method the second derivative terms in the Newton correction are omitted, it is no longer an application of Newton's method, but merely an approximation of it. Therefore, convergence of the Gauß-Newton method cannot be justified by the convergence of Newton's method. Under certain assumptions, convergence of the Gauß-Newton method can also be proven, but with a different approach.
- Remember that under these conditions, Newton's method converges quadratically within this environment. Due to the simplification of the Newton correction, we can no longer give any convergence speed. It turns out that in practice it often is linear. However, it may be better or worse than that.

- Since we have so little information about the convergence behaviour, without further information we cannot say anything about reasonable initial value candidates.

The idea behind simplifying the Newton correction is as follows. Convergence statements and convergence speed are lost, but each iteration step can be performed more efficiently because no second derivatives of F need to be computed. So it is a trade-off. On the next exercise sheet we will see other methods that decide this trade-off differently.

Exercise 3 (**The Gauß-Newton method and Taylor's expansion**, 7+5 points)

Analogously to Newton's method the Gauß-Newton method can also be derived using Taylor's expansion (obviously, as the Gauß-Newton method basically is Newton's method except for the simplification of the Newton correction).

a) Derive the Gauß-Newton method by demanding $g'(x_{k+1})^T = 0$, Taylor expanding $g'(x_{k+1})^T$ and omitting higher order terms (exactly as for the derivation of Newton's method). To eventually get the iteration rule of the Gauß-Newton method, simplify by neglecting terms with second derivatives of F (c.f. exercise 2).

b) Recall the lecture Numerical Linear Algebra, then it will become apparent that

$$F'(x_k)^T F'(x_k) s_k = -F'(x_k)^T F(x_k) \tag{6}$$

is the normal equation to the linear least squares problem

$$\text{find} \quad \hat{s}_k \in \mathbb{R}^n \quad \text{such that} \quad \left\| F'(x_k)\hat{s}_k + F(x_k) \right\|_2 = \min_{s_k \in \mathbb{R}^n} \left\| F'(x_k) s_k + F(x_k) \right\|_2 \ . \tag{7}$$

So the problems (6) and (7) are equivalent. In other words, completely analogous to Newton's method, what does the Gauß-Newton method do with nonlinear least squares problems?

Exercise 4 (**Testing Gauß-Newton on the Rosenbrock function**, 2+4+3+2 points)

We finally want to test the Gauß-Newton method on a standard problem for optimization, the Rosenbrock function

$$r(x,y) = \left\| \begin{pmatrix} 1-x \\ 10\,(y-x^2) \\ \sqrt{x^2+y^2} \end{pmatrix} \right\|_2 .$$

a) Execute the MATLAB programme `visualize_rosenbrock.m` and think about where the difficulty might lie in optimizing this function.

b) We want to apply the Gauß-Newton method. What is the corresponding function F and its derivative F'?

c) The MATLAB programme `test_rosenbrock.m` tries to find the minimum of the Rosenbrock function using your programme `gauss_newton.m`. Then it plots the sequence of iterates.
For that, you have to adjust `gauss_newton.m` such that it does not only output the final iterate but a matrix of all iterates in the form of

$$\big(x_0, x_1, x_2, \ldots\big) \in \mathbb{R}^{2\times(\#\text{ iterations}+1)} .$$

Fill in the gaps in `test_rosenbrock.m`, execute the programme and reflect on the outcome.

d) The MATLAB programme `error_rosenbrock.m` computes for every iteration step the approximation error with respect to the euclidean 2-norm,

$$e_k := \left\| \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} - \begin{pmatrix} x_k \\ y_k \end{pmatrix} \right\|_2 \quad \text{with} \quad k = 0, \ldots, \#\text{iterations} ,$$

with (x_k, y_k) being the k-th iterate of the Gauß-Newton method and $(\hat{x}, \hat{y})$ the exact solution.
Execute the programme and reflect on the outcome. Especially focus on the convergence speed.