Solutions for the exercises "Optimization in Systems and Control"

Remark:

- Changes made on September 20, 2020 are marked in blue.
- Changes made on October 13, 2020 are marked in green.

1 Exercises for Chapter 1

Exercise 1.1. *Let f be a convex function defined on a set I. If $x_1, x_2, ..., x_n \in I$, and $\lambda_1, \lambda_2, ..., \lambda_n \in [0,1]$ with $\sum_{i=1}^n \lambda_i = 1$, then prove that*

$$f\left(\sum_{i=1}^n \lambda_i x_i\right) \leq \sum_{i=1}^n \lambda_i f(x_i) \ . \tag{1.1}$$

Solution: As f is convex, we have $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in I$ and for all $\lambda \in [0,1]$.

For $n = 1$ we have $\lambda_1 = 1$ and then (1.1) reduces to the trivial statement $f(x_1) = f(x_1)$, which is true.
The case $n = 2$ corresponds to the definition of convexity.
We now proceed by induction, assuming the inequality (1.1) is true for some n and proving prove it holds for $n+1$. Since for $\lambda_{n+1} = 0$ the inequality (1.1) reduces to the case n and is thus true, we now assume that $\lambda_{n+1} > 0$. We have $\sum_{i=1}^{n+1} \lambda_i = 1$ and

$$\begin{aligned} f\left(\sum_{i=1}^{n+1} \lambda_i x_i\right) &= f\left(\lambda_{n+1} x_{n+1} + (1-\lambda_{n+1}) \frac{1}{1-\lambda_{n+1}} \sum_{i=1}^n \lambda_i x_i\right) \\ &= f\left(\lambda_{n+1} x_{n+1} + (1-\lambda_{n+1}) \sum_{i=1}^n \frac{\lambda_i}{1-\lambda_{n+1}} x_i\right) \ . \end{aligned} \tag{1.2}$$

Define $\theta_i = \dfrac{\lambda_i}{1-\lambda_{n+1}}$. Then we have $\theta_i \geq 0$ for $i = 1, \ldots, n$. Moreover, since $\sum_{i=1}^{n+1} \lambda_i = 1$, we have $\sum_{i=1}^n \lambda_i = 1 - \lambda_{n+1}$, and thus

$$\sum_{i=1}^n \theta_i = \sum_{i=1}^n \frac{\lambda_i}{1-\lambda_{n+1}} = \frac{1}{1-\lambda_{n+1}} \sum_{i=1}^n \lambda_i = \frac{1-\lambda_{n+1}}{1-\lambda_{n+1}} = 1.$$

Since $\theta_i \geq 0$ for all i, this also implies that $\theta_i \leq 1$ for all i. So $\theta_i \in [0,1]$ for all i. Now define

$$y = \sum_{i=1}^n \frac{\lambda_i}{1-\lambda_{n+1}} x_i = \sum_{i=1}^n \theta_i x_i \ .$$

Since I is the domain of definition of the convex function f, it is a convex set, and since y is a convex combination of $x_1, \ldots, x_n \in I$, we have $y \in I$. Hence, we can apply the definition of convex functions to (1.2), which yields

$$\begin{aligned} f\left(\sum_{i=1}^{n+1} \lambda_i x_i\right) &= f\big(\lambda_{n+1} x_{n+1} + (1-\lambda_{n+1}) y\big) \\ &\leq \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) f(y) \ . \end{aligned} \tag{1.3}$$

Since by induction (1.1) was assumed to hold for n and since the coefficients θ_i satisfy the conditions for the property, we have

$$f(y) = f\left(\sum_{i=1}^{n} \theta_i x_i\right) \leq \sum_{i=1}^{n} \theta_i f(x_i) \ .$$

If we combine this with (1.3), we find

$$\begin{aligned} f\left(\sum_{i=1}^{n+1} \lambda_i x_i\right) &\leq \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) \sum_{i=1}^{n} \theta_i f(x_i) \\ &\leq \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) \sum_{i=1}^{n} \frac{\lambda_i}{1-\lambda_{n+1}} f(x_i) \\ &\leq \sum_{i=1}^{n+1} \lambda_i f(x_i) \ , \end{aligned}$$

which proves the property.

Exercise 1.2. *Use the definition of convex functions to show that the function $f : \mathbb{R}^+ \to \mathbb{R}^+ : x \mapsto \sqrt{x}$ is not convex.*

Solution: Recall that the definition of convex functions states that f is convex if its domain $\mathrm{dom}(f)$ is convex and if we have $f((1-\lambda)x + \lambda y) \leq (1-\lambda) f(x) + \lambda f(y)$ for all $x, y \in \mathrm{dom}(f)$ and for all $\lambda \in [0, 1]$.

In our case $\mathrm{dom}(f) = \mathbb{R}^+$, which is a convex set.

Now consider $x = 0$, $y = 1$, and $\lambda = \frac{1}{4}$. We have $f(0) = 0$, $f(y) = 1$, $f((1-\lambda)x + \lambda y) = f\left(\frac{1}{4}\right) = \frac{1}{2}$. However,

$$(1-\lambda) f(x) + \lambda f(y) = \frac{3}{4} \cdot 0 + \frac{1}{4} \cdot 1 = \frac{1}{4} \ .$$

So $f((1-\lambda)x + \lambda y) = \frac{1}{2} \not\leq \frac{1}{4} = (1-\lambda) f(x) + \lambda f(y)$ for the given x, y, and λ. Hence, $f : \mathbb{R}^+ \to \mathbb{R}^+ : x \mapsto \sqrt{x}$ is not convex.

Exercise 1.3. *Determine for which values $p \geqslant 0$ the function $f : \mathbb{R}_0^+ \to \mathbb{R}^+$ defined by $f(x) = x^p$ is a convex function, with $\mathbb{R}_0^+ = \mathbb{R}^+ \setminus \{0\} = (0, +\infty)$.*
Hint: Use the fact that if the second derivative f'' of the function f with a scalar argument is defined and nonnegative, then f is convex.

Solution: We consider the cases $p = 0$ and $p = 1$ separately. Clearly, for $p = 0$ we have $f(x) = x^0 = 1$, and so for $p = 0$ the function is convex. For $p = 1$ we have $f(x) = x$, and so for $p = 1$ the function is convex. For $p \neq 0$ and $p \neq 1$ we have $f''(x) = p(p-1)x^{p-2}$. Since $x > 0$ we have that f'' is nonnegative only for $p > 1$. Moreover, for $0 < p < 1$ the derivative f'' is negative, which means that the function is then concave.
Hence, we conclude that f is convex for $p = 0$ or for $p \geqslant 1$.

Exercise 1.4. *Use previous result and the definition of convex functions to prove that the function f defined by*

$$f(x) = \sum_{i=1}^{n} |x_i|^p, \quad x \in \mathbb{R}^n, p \geq 1$$

is convex.

Solution: First we show that the function $f : v \mapsto |v|^p$ is a convex function on $\mathbb{R}$ if $p \geq 1$. The case $p = 1$ is treated first. For $p = 1$ we get $|v|$, which is a convex function on $\mathbb{R}$ as it is a norm function. For $p > 1$ the function $|v|^p$ can be differentiated twice everywhere and we have

$$\frac{\mathrm{d}\,|v|^p}{\mathrm{d}v} = \begin{cases} pv^{p-1} & \text{if } v > 0 \\ -p(-v)^{p-1} & \text{if } v < 0 \\ 0 & \text{if } v = 0 \end{cases}$$

and thus

$$\frac{\mathrm{d}^2\,|v|^p}{\mathrm{d}v^2} = \begin{cases} p(p-1)v^{p-2} & \text{if } v > 0 \\ p(p-1)(-v)^{p-2} & \text{if } v < 0 \\ 0 & \text{if } v = 0 \end{cases}$$

Since $p > 1$, the derivative is always nonnegative if $v \neq 0$, and since the function is continuous in $v = 0$, $|v|^p$ is a convex function on $\mathbb{R}$ if $p \geq 1$.

Now let $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$. We have

$$f(\lambda x + (1-\lambda)y) = \sum_{i=1}^{n} |\lambda x_i + (1-\lambda)y_i|^p \ .$$

Since $|v|^p$ is a convex function, it follows that $|\lambda x_i + (1-\lambda)y_i|^p \leq \lambda|x_i|^p + (1-\lambda)|y_i|^p$. Hence,

$$\begin{aligned} f(\lambda x + (1-\lambda)y) &\leq \sum_{i=1}^{n} \lambda|x_i|^p + (1-\lambda)|y_i|^p \\ &\leq \lambda \sum_{i=1}^{n} |x_i|^p + (1-\lambda) \sum_{i=1}^{n} |y_i|^p \\ &\leq \lambda f(x) + (1-\lambda) f(y) \ . \end{aligned}$$

So f is convex.

Exercise 1.5. *Find the Taylor polynomial P_2 of order 2 based at $(0, 0)$ for the function f defined by $f(x, y) = 3xy + 2xy^3$. Note: This Taylor polynomial is defined by:*

$$P_2(x, y) = f(0,0) + (\nabla f(0,0))^T \begin{pmatrix} x \\ y \end{pmatrix} + \frac{1}{2}(x, y) H_f(0,0) \begin{pmatrix} x \\ y \end{pmatrix} \ .$$

What is an upper bound for $\varepsilon > 0$ so that the error of between $P_2(x,y)$ and $f(x,y)$ is lower than 10^{-6} if $|x|, |y| \leq \varepsilon$? Note: The error is given by:

$$R_2(x,y) = \frac{1}{3!}\sum_{i,j,k=1}^{2} \frac{\partial^3 f(c_1,c_2)}{\partial x_i \partial x_j \partial x_k} h_i h_j h_k$$
$$= \frac{1}{3!}\left(\frac{\partial^3 f(c_1,c_2)}{\partial x^3}x^3 + 3\frac{\partial^3 f(c_1,c_2)}{\partial x^2 y}x^2 y + 3\frac{\partial^3 f(c_1,c_2)}{\partial x y^2}x y^2 + \frac{\partial^3 f(c_1,c_2)}{\partial y^3}y^3\right) ,$$

where (c_1,c_2) is any point in the line between $(0,0)$ and (x,y), and where h_i, h_j, and h_k refer to the x_i, x_j, and x_k component of the vector (x,y).

Solution: We have

$$f(0,0) = 0$$
$$\nabla f(x,y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix} = \begin{pmatrix} 3y + 2y^3 \\ 3x + 6xy^2 \end{pmatrix} \Rightarrow \nabla f(0,0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$H_f(x,y) = \begin{pmatrix} \frac{\partial^2 f(x,y)}{\partial x^2} & \frac{\partial^2 f(x,y)}{\partial x \partial y} \\ \frac{\partial^2 f(x,y)}{\partial y \partial x} & \frac{\partial^2 f(x,y)}{\partial y^2} \end{pmatrix} = \begin{pmatrix} 0 & 3+6y^2 \\ 3+6y^2 & 12xy \end{pmatrix} \Rightarrow H_f(0,0) = \begin{pmatrix} 0 & 3 \\ 3 & 0 \end{pmatrix} .$$

Then, $P_2(x,y) = 3xy$.

To evaluate the error, we need the following derivatives:

$$\frac{\partial^3 f(x,y)}{\partial x^3} = 0, \quad \frac{\partial^3 f(x,y)}{\partial x^2 \partial y} = 0, \quad , \frac{\partial^3 f(x,y)}{\partial x \partial y^2} = 12y, \quad , \frac{\partial^3 f(x,y)}{\partial y^3} = 12x .$$

If $|x| \leq \varepsilon$, $|y| \leq \varepsilon$, then we have $|c_1| \leq \varepsilon$ and $|c_2| \leq \varepsilon$. Hence,

$$|R_2(x,y)| \leq \frac{1}{3!}\sum_{i,j,k=1}^{2} |\frac{\partial^3 f(c_1,c_2)}{\partial x_i \partial x_j \partial x_k}||h_i||h_j||h_k| < \frac{\varepsilon^3}{3!}\sum_{i,j,k=1}^{2} |\frac{\partial^3 f(c_1,c_2)}{\partial x_i \partial x_j \partial x_k}|$$
$$\leq \frac{\varepsilon^3}{3!}(3|12c_1| + |12c_2|) < 8\varepsilon^4 .$$

Then, the error of using $P_2(x,y)$ will be clearly lower than 10^{-6} if $8\varepsilon^4 \leq 10^{-6}$ or $\varepsilon \leq 0.0188$.

Exercise 1.6. *Indicate whether or not the following functions $g(x)$ are subgradients of the corresponding functions $f(x)$:*

- $f(x) = |x|$, $x \in \mathbb{R}$: $g(x) = \begin{cases} -1 & if \quad x < 0 \\ 2 & if \quad x = 0 \\ 1 & if \quad x > 0 \end{cases}$
- $f(x) = \max\{f_1(x), f_2(x)\}$, $x \in \mathbb{R}^n$, $f_1(x)$ *and* $f_2(x)$ *convex and continuously differentiable:* $g(x) = \begin{cases} \nabla f_1(x) & if \quad f_1(x) > f_2(x) \\ \nabla f_2(x) & if \quad f_1(x) \leq f_2(x) \end{cases}$

Solution: It is easy to verify that both functions f defined above are convex.

Let f be a convex function. The function g is called a subgradient of f if $f(x) \geq f(y) + g(y)^T(x-y)$, $\forall x, y \in \text{dom}(f)$.

Consider the first function. The points $x = 1$ and $y = 0$ both belong to $\text{dom}(f)$. However, $f(y) + g(y)^T(x-y) = 0 + 2(1-0) = 2 > f(x) = 1$, and thus $g(x)$ is not a subgradient of $f(x)$.

Now consider the second function f. Recall that for convex functions that are continuously differentiable the subgradient is equal to the gradient. So we have $f_1(x) \geq f_1(y) + \nabla f_1(y)^T(x-y)$, $\forall x, y \in \text{dom}(f)$ and $f_2(x) \geq f_2(y) + \nabla f_2(y)^T(x-y)$, $\forall x, y \in \text{dom}(f)$.
Now we first assume $f_1(y) \geq f_2(y)$. Then we have $f(y) = f_1(y)$ and $g(y) = \nabla f_1(y)$ and then for any $x \in \text{dom}(f)$ we have $f(x) \geq f_1(x) \geq f_1(y) + \nabla f_1(y)^T(x-y) \geq f(y) + g^T(y)(x-y)$, i.e., the subgradient inequality holds in this case.
The case $f_1(y) < f_2(y)$ can be dealt with in a similar way.
So g is indeed a subgradient of f.

Exercise 1.7. *Let $A \in \mathbb{R}^{m \times n}$ and $x \in \mathbb{R}^n$. Show that $\nabla(Ax) = A^T$.*

Solution: Let $f(x) = Ax$, i.e., $f_j = \sum_{l=1}^{n} a_{jl}x_l$. Then $\frac{\partial f_j}{\partial x_i} = a_{ji}$. Hence, $\nabla f = A^T$.

Exercise 1.8. *Show that the following functions $g(x)$ are subgradients of the corresponding functions $f(x)$:*

- *$f(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x)$, $x \in \mathbb{R}^n$, $f_1(x)$ and $f_2(x)$ convex and differentiable: $g(x) = \alpha_1 \nabla f_1(x) + \alpha_2 \nabla f_2(x)$*
- *$f(x) = f_1(Ax + b)$, $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $f_1(x)$ convex and differentiable: $g(x) = A^T \nabla f_1(Ax + b)$*

Solution: For convex and continuously differentiable functions, the subgradient is equal to the gradient.

For the first function, using the sum rule of the derivative:

$$\nabla f(x) = \nabla(\alpha_1 f_1(x) + \alpha_2 f_2(x)) = \alpha_1 \nabla f_1(x) + \alpha_2 \nabla f_2(x) = g(x).$$

For the second function, using the chain rule, we can show (in a similar way as the preceding exercise) that $\nabla f_1(Ax + b) = A^T \nabla f_1(Ax + b)$. Indeed, define $f(x) = f_1(v)$ with $v = Ax + b$. Then we have

$$\frac{\partial f}{\partial x_i} = \sum_j \frac{\partial f_1}{\partial v_j} \frac{\partial v_j}{\partial x_i} = \sum_j \frac{\partial f_1}{\partial v_j} a_{ji} \ .$$

Hence, $\nabla f(x) = A^T \nabla f_1(Ax + b)$.

Exercise 1.9. *Find the saddle points and local minima and maxima of the following functions:*

- *$f_1(x) = 9 - 2x_1 + 4x_2 - x_1^2 - 4x_2^2$*
- *$f_2(x) = 2x_1^3 + x_1 x_2^2 + 5x_1^2 + x_2^2$*

Solution:

$$\nabla f_1(x) = \begin{pmatrix} \dfrac{\partial f_1(x)}{\partial x_1} \\ \dfrac{\partial f_1(x)}{\partial x_2} \end{pmatrix} = \begin{pmatrix} -2 - 2x_1 \\ 4 - 8x_2 \end{pmatrix}$$

Then $\nabla f_1(x) = 0$ if $x_1 = -1$ and $x_2 = \frac{1}{2}$. Then we evaluate the Hessian:

$$H_{f_1}(x) = \begin{pmatrix} \dfrac{\partial^2 f(x)}{\partial x_1^2} & \dfrac{\partial^2 f(x)}{\partial x_1 \partial x_2} \\ \dfrac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f(x)}{\partial x_2^2} \end{pmatrix} = \begin{pmatrix} -2 & 0 \\ 0 & -8 \end{pmatrix}$$

As $H_{f_1}(-1, \frac{1}{2})$ is negative definite, the point $(-1, \frac{1}{2})$ is a local maximum.

The same procedure is performed for the second function:

$$\nabla f_2(x) = \begin{pmatrix} \dfrac{\partial f_2(x)}{\partial x_1} \\ \dfrac{\partial f_2(x)}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 6x_1^2 + x_2^2 + 10x_1 \\ 2x_1x_2 + 2x_2 \end{pmatrix}$$

Then $\nabla f_2(x) = 0$ if: $x = (0, 0)$, $x = (-\frac{5}{3}, 0)$, $x = (-1, 2)$, or $x = (-1, -2)$. Indeed, we have $2x_1x_2 + 2x_2 = 0$ if $x_2 = 0$ or $x_1 = -1$.
For $x_2 = 0$, setting $6x_1^2 + x_2^2 + 10x_1 = 0$ yields $6x_1^2 + 10x_1 = 0$, or $x_1 = 0$ or $x_1 = -\frac{5}{3}$.
For $x_1 = -1$, setting $6x_1^2 + x_2^2 + 10x_1 = 0$ yields $x_2^2 = 4$ or $x_2 = \pm 2$.

Now we evaluate the Hessian:

$$H_{f_2}(x) = \begin{pmatrix} \dfrac{\partial^2 f_2(x)}{\partial x_1^2} & \dfrac{\partial^2 f_2(x)}{\partial x_1 \partial x_2} \\ \dfrac{\partial^2 f_2(x)}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f_2(x)}{\partial x_2^2} \end{pmatrix} = \begin{pmatrix} 12x_1 + 10 & 2x_2 \\ 2x_2 & 2x_1 + 2 \end{pmatrix}$$

As $H_{f_2}(0, 0)$ is positive definite, the point $(0, 0)$ is a local minimal. As $H_{f_2}(-\frac{5}{3}, 0)$ is negative definite, the point $(-\frac{5}{3}, 0)$ is a local maximum. The Hessians $H_{f_2}(-1, 2)$ and $H_{f_2}(-1, -2)$ are both indefinite, and so the points $(-1, 2)$ and $(-1, -2)$ are saddle points.

Figure 1 shows functions $f_1(x)$ and $f_2(x)$ and their contour plots. The positions of the analyzed points are indicated by the + marks in the contour plots.

Exercise 1.10. *The optimization problem* $\min f(x_1, x_2) = (x_1 - 3)^4 + (x_1 - 3x_2)^2$ *is solved using the following (gradient-based) algorithm*

$$x_{k+1} = x_k - \lambda_k \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|_2}$$

If the initial point is $x_0 = [0, 0]^T$ *and the step* $\lambda_k = (0.9)^{k+1}$*, use Matlab to indicate which of the following stopping criteria is fulfilled first:*

- $\|\nabla f(x_k)\|_2 \leq 3.5$
- $|f(x_k) - f(x_{k-1})| \leq 0.4$
- *Maximum number of iterations* $k_{max} = 10$

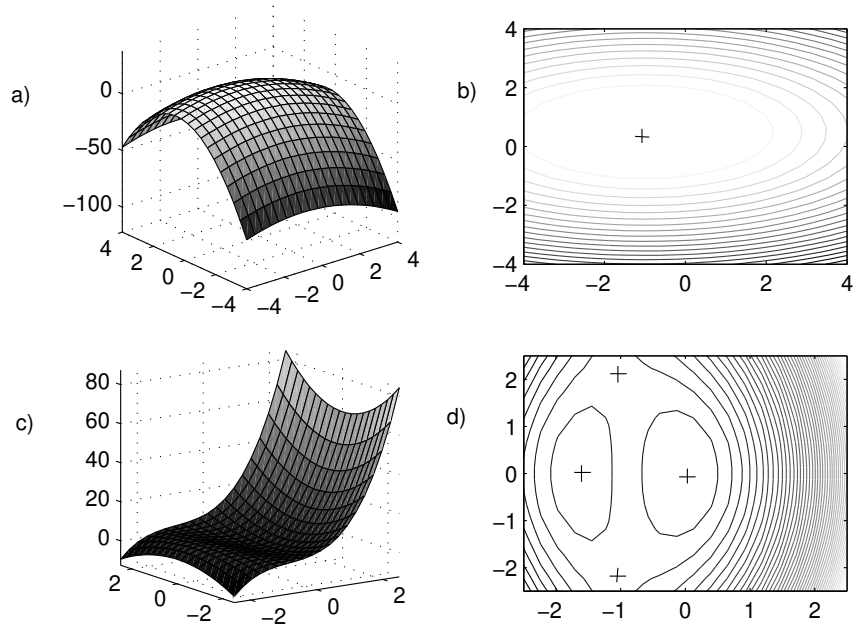

Figure 1: (a) Function $f_1(x)$, (b) Contour plot of function $f_1(x)$, (c) Function $f_2(x)$, (d) Contour plot of function $f_2(x)$

Table 1: Iterations

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	λ_k
0	(0.0000,0.0000)	81.0000	(-108.0000,0.0000)	0.9000
1	(0.9000,0.0000)	20.2581	(-35.2440,-5.4000)	0.8100
2	(1.7007,0.1227)	4.6258	(-6.1086,-7.9956)	0.7290
3	(2.1433,0.7020)	0.5401	(-2.4404,-0.2238)	0.6561
4	(2.7966,0.7619)	0.2627	(0.9881,-3.0654)	0.5904
5	(2.6154,1.3239)	1.8614	(-2.9402,8.1378)	0.5314
6	(2.7960,0.8241)	0.1065	(0.6134,-1.9422)	0.4782
7	(2.6520,1.2802)	1.4274	(-2.5458,7.1316)	0.4304
8	(2.7967,0.8748)	0.0314	(0.3110,-1.0338)	0.3874
9	(2.6851,1.2458)	1.1172	(-2.2295,6.3138)	0.3486
10	(2.8012,0.9170)	0.0041	(0.0690,-0.3012)	0.3138

Table 2: Stopping criteria

k	$\|\nabla f(x_k)\|_2$	$\|f(x_k) - f(x_{k-1})\|$
0	108.0000	–
1	35.6553	60.7419
2	10.0620	15.6323
3	2.4507	4.0857
4	3.2207	0.2774
5	8.6526	1.5987
6	2.0368	1.7549
7	7.5724	1.3209
8	1.0796	1.3960
9	6.6959	1.0858
10	0.3090	1.1131

Plot the various iteration points and their function values.

Solution: The results of the first 10 iterations are displayed in Table 1 (with 4 decimal digits). The values for the stopping criteria are in Table 2.

From Tables 1 and 2, we can see that the stopping criterion $\|\nabla f(x_k)\|_2 \leq 3.5$ is reached in the iteration $k = 3$, and the stopping criterion $|f(x_k) - f(x_{k-1})| \leq 0.4$ is reached in iteration step $k = 4$.

In the Figure 2, the function $f(x)$ and its contour plot is presented. The position of the optimal point $(3, 1)$ is indicated by the + mark in the contour plots, as well as the steps of the iterative method. In the figure it can be seen also the values of x_k and how close they get as the number of iterations increase.

Exercise 1.11. *Consider the problem of choosing (x, y) to maximize $f(x, y) = 3x + y$ subject to: $(x + 1)^2 + (y + 1)^2 \leq 5$ and $x \geq 0$, $y \geq 0$.*

- *Suppose that (x^*, y^*) solves this problem. Is there necessarily a value of μ such that (x^*, y^*) satisfies the Kuhn-Tucker conditions?*
- *Now suppose that (x^*, y^*) satisfies the Kuhn-Tucker conditions. Does (x^*, y^*) necessarily solve the problem?*
- *Given the information in your answers to (a) and (b), use the Kuhn-Tucker method to solve the problem.*

Solution: First we rewrite the problem in the standard form $\min_{(x,y)} f(x, y)$ s.t. $g(x, y) \leq 0$. This yields:

$$\begin{aligned} &\min_{(x,y)} (-3x - y) \\ &\text{s.t. } (x + 1)^2 + (y + 1)^2 - 5 \leq 0 \\ &\qquad -x \leq 0 \\ &\qquad -y \leq 0 \ . \end{aligned}$$

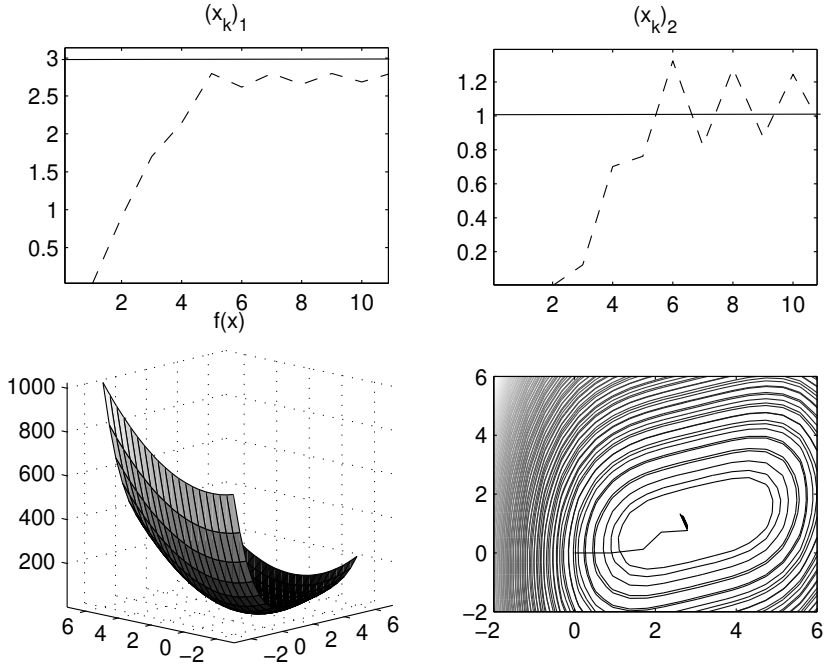

Figure 2: (a) x_1 as function of the iteration steps, (b) x_2 as function of the iteration steps, (c) Function $f(x)$, (d) Contour plot of function $f(x)$

The Kuhn-Tucker conditions for this problem are given by

$$\begin{aligned} \nabla f(x,y) + \nabla g(x,y)\mu &= 0 \\ \mu^T g(x,y) &= 0 \\ \mu &\geq 0 \\ g(x,y) &\leq 0 \ . \end{aligned}$$

- In general, the Kuhn-Tucker conditions provide necessary conditions for an optimum of the given optimum. So if (x^*, y^*) solves the given optimization problem, there should exist a μ^* such that (x^*, y^*, μ^*) satisfies the Kuhn-Tucker conditions.
- The objective function of the above minimization problem is convex and the constraints are also convex. So we have a convex optimization problem. Hence, the Kuhn-Tucker conditions are sufficient in this case.
- We have

$$\nabla f(x,y) = \begin{pmatrix} -3 \\ -1 \end{pmatrix} \quad \text{and} \quad \nabla g(x,y) = \begin{pmatrix} 2(x+1) & -1 & 0 \\ 2(y+1) & 0 & -1 \end{pmatrix} \ .$$

Hence, the Kuhn-Tucker conditions can be written as[1]

$$\begin{aligned}
-3 + 2\mu_1(x+1) - \mu_2 &= 0 \\
-1 + 2\mu_1(y+1) - \mu_3 &= 0 \\
\mu_1[(x+1)^2 + (y+1)^2 - 5] &= 0 \\
\mu_2(-x) &= 0 \\
\mu_3(-y) &= 0 \\
(x+1)^2 + (y+1)^2 - 5 &\leq 0 \\
-x &\leq 0 \\
-y &\leq 0 \\
\mu_1, \mu_2, \mu_3 &\geq 0
\end{aligned}$$

From the 4th and the 5th equation it follows that 4 different combinations are possible: (1) $x = 0$ and $y = 0$; (2) $\mu_2 = 0$ (or $x > 0$) and $y = 0$; (3) $x = 0$ and $\mu_3 = 0$ (or $y > 0$); and (4) $\mu_2 = 0$ (or $x > 0$) and $\mu_3 = 0$ (or $y > 0$).
The only combination that leads to a feasible solution is combination (2), which results in $(x, y, \mu_1, \mu_2, \mu_3) = (1, 0, 0.75, 0, 0.5)$. The solution of the optimization problem is thus $(x^*, y^*) = (1, 0)$.

2 Exercises for Chapter 2: Linear Programming

Exercise 2.1. *Use the graphical method to solve the following problem:*

$$\min f(x) = x_1 - 2x_2$$

subject to the constraints: $x_1 + x_2 \geq 2$, $-x_1 + x_2 \geq 1$, $x_2 \leq 3$, $x_1, x_2 \geq 0$.
Reformulate the same problem as a linear programming problem in standard form and solve it using the simplex method.

Solution: Figure 3 shows the contour plot and the feasible region of the optimization problem. The solution is in a vertex of the feasible set, which is obtained with the graphical method (we shift one of the contour lines in a parallel way in the direction of the arrow, where a lower minimum cost can be obtained, but such that there still is an intersection with the feasible set).

The optimal solution is given by the point $(0, 3)$, corresponding to $f(x^*) = 0 - 2 \cdot 3 = -6$. Now, we reformulate the same problem as a linear programming problem in standard form.

First the objective function is the same as we are facing a minimization problem. The constraints: $x_1 + x_2 \geq 2$, $-x_1 + x_2 \geq 1$, $x_2 \leq 3$, $x_1, x_2 \geq 0$, are equivalent to:

$$\begin{aligned}
-x_1 - x_2 + x_3 &= -2 \\
x_1 - x_2 + x_4 &= -1 \\
x_2 + x_5 &= 3 \\
x_1, x_2, x_3, x_4, x_5 &\geq 0
\end{aligned}$$

Note that the slack variables x_3, x_4, x_5 have been introduced to obtain equality constraints. If we define:

[1]Note that the condition $\mu^T g(x, y) = 0$ results in $\sum_i \mu_i g_i(x, y) = 0$ or equivalently $\sum_i \mu_i(-g_i(x, y))$. Since $-g(x, y) \geq 0$ and $\mu \geq 0$, then is in its turn equivalent to $\mu_i g_i(x, y) = 0$ for all i.

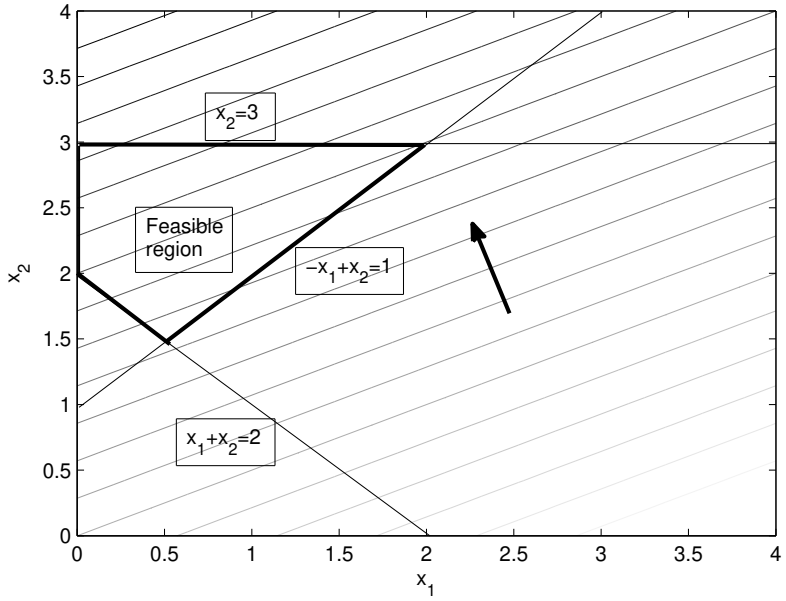

Figure 3: Feasible set and contour plot for Exercise 2.1

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}, \quad c = \begin{bmatrix} 1 \\ -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad b = \begin{bmatrix} -2 \\ -1 \\ 3 \end{bmatrix}, \quad A = \begin{bmatrix} -1 & -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

we have formulated the problem as a linear programming problem in standard form. Now we apply the simplex method. Suppose our first choice of B and N is:

$$B = \begin{bmatrix} -1 & -1 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad N = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix},$$

Then we find x_B, x_N, c_B and c_N as:

$$x_B = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = B^{-1}b = \begin{bmatrix} 2 \\ 3 \\ 3 \end{bmatrix}, x_N = \begin{bmatrix} x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad c_B = \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix}, c_N = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

This corresponds to $x_1 = 2$, $x_2 = 3$, $x_3 = 3$, $x_4 = 0$ and $x_5 = 0$ (feasible solution). The corresponding values of z_0 and p are:

$$z_0 = c_B^T B^{-1} b = -4, p^T = c_N^T - c_B^T B^{-1} N = [-1 \quad 1]$$

Since $p^T \not\geqslant 0$, the optimum is not found yet. Since -1 is the largest negative component of p, we select the first column of N ($i = 1$). We have $y = B^{-1}N_{\cdot,1} = [1\ 0\ 1]^T$. We have to choose between the first and the third component:

$$\frac{(x_B)_1}{y_1} = 2, \quad \frac{(x_B)_3}{y_3} = 3,$$

So we select the first column of B ($j = 1$). Now we interchange the first column of N with the first column of B, which leads to:

$$B = \begin{bmatrix} 0 & -1 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad N = \begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix},$$

Then we find x_B, x_N, c_B and c_N as:

$$x_B = \begin{bmatrix} x_4 \\ x_2 \\ x_3 \end{bmatrix} = B^{-1}b = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}, x_N = \begin{bmatrix} x_1 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad c_B = \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix}, c_N = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

This corresponds to $x_1 = 0$, $x_2 = 3$, $x_3 = 1$, $x_4 = 2$ and $x_5 = 0$. The corresponding values of z_0 and p are:

$$z_0 = c_B^T B^{-1} b = -6, p^T = c_N^T - c_B^T B^{-1} N = [1 \quad 2]$$

Since $p^T \geq 0$, the optimum was found. The optimal solution of the original problem is $(0, 3)$ and the corresponding cost is -6.

Exercise 2.2. *Two students A and B work at a shop for x and y hours per week, respectively. According to the rules, A can work at most 8 hours more than B. But student B can work at most 6 hours more than student A. Together they can work at most 40 hours per week. Find their maximum combined income per week if student A and student B earn 15 and 17 euro per hour, respectively.*

Solution: We formulate the problem as a linear programming problem. We have to maximize $15x+17y$, considering the constraints $x \leq 8+y$, $y \leq 6+x$ and $x+y \leq 40$. We will use Matlab to obtain the result, but this problem can be solved graphically, or by using the simplex algorithm.

The program `linprog` (Optimization Toolbox of Matlab) solves linear programming problems specified by:

min $f^T x$ such that $Ax \leq b$, $A_{eq}x = b_{eq}$, $l_b \leq x \leq u_b$, where f, x, b, b_{eq}, l_b and u_b are vectors, and A and A_{eq} are matrices. For our problem, we change the objective function to $-15x - 17y$ as the program will minimize instead of maximize. The linear programming routine is:

```
f = [-15; -17];
A =  [1 -1; -1 1; 1 1];
b = [8; 6; 40];
[x,fval,exitflag,output,lambda] = linprog(f,A,b,[],[],[]);
```

The result is:

$$x = \begin{bmatrix} 17 \\ 23 \end{bmatrix}, \quad fval = -646.0000$$

So student A and student B should work 17 and 23 hours respectively, and their maximum combined income per week is 646 euros. It is also interesting to check *lambda*:

$$lambda.ineqlin = \begin{bmatrix} 0 \\ 1 \\ 16 \end{bmatrix}, lambda.lower = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, lambda.upper = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Nonzero elements of the vectors in the fields of lambda indicate active constraints at the solution. In this case, the second and third inequality constraints (in lambda.ineqlin).

3 Exercises for Chapter 3: Quadratic Programming

Exercise 3.1. *Consider the process modeled by the following linear discrete-time system:* $y(n+1) = ay(n) + bu(n) + \frac{1}{1-q^{-1}}e(n)$, *where* $y(n)$ *is the output,* $u(n)$ *the input,* a *and* b *are the model parameters,* $e(n)$ *is white noise of mean value* 0 *and standard deviation* σ. *At time step* n *the output* $y(n)$ *is measured, the output* $y(n-1)$ *and control action* $u(n-1)$ *is also known, and we have to obtain a control action* $u(n)$. *It is easy to show that we can define the 1-step ahead prediction* $\hat{y}(n+1) = (1+a)y(n) - ay(n-1) + b\Delta u(n)$ *with* $\Delta = 1 - q^{-1}$ *with* $\hat{y}(n) = y(n)$.

1. *Obtain the control action* $\Delta u(n)$ *that minimizes* $J = (\hat{y}(n+1) - r)^2 + \lambda \left(\Delta u(n)\right)^2$, *where* λ *is a weighting factor and* r *the output reference.*

2. *Reformulate the following problem as a quadratic programming problem:*
   $\min J_n = \sum_{k=1}^{3}(\hat{y}(n+k) - r)^2 + \lambda \sum_{k=1}^{3} \left(\Delta u(n+k-1)\right)^2$,
   s.t. $\Delta u_{min} \leq \Delta u(n+k-1) \leq \Delta u_{max}$, $k = 1, 2, 3$.

3. *Reformulate the problem as a quadratic programming problem of Type 1 with as few variables as possible. Assume* $\Delta u_{min} = 0$.

Solution:

a) By replacing $\hat{y}(n+1) = (1+a)y(n) - ay(n-1) + b\Delta u(n)$ in the objective function $J = (\hat{y}(n+1) - r)^2 + \lambda \Delta u(n)^2$, we obtain J as function of $\Delta u(n)$, so then we just use the first order condition for the optimum:

$$\frac{\partial J}{\partial \Delta u(n)} = \frac{\partial (((1+a)y(n) - ay(n-1) + b\Delta u(n) - r)^2 + \lambda \Delta u(n)^2)}{\partial \Delta u(n)} =$$

$$2b(((1+a)y(n) - ay(n-1) + b\Delta u(n) - r) + 2\lambda \Delta u(n) = 0$$

$$\Rightarrow \Delta u(n) = \frac{-b(((1+a)y(n) - ay(n-1) - r)}{b^2 + \lambda}$$

b) From the 1-step ahead prediction equation it follows that $\hat{y}(n+2) = (1+a)\hat{y}(n+1) - ay(n) + b\Delta u(n+1)$ and $\hat{y}(n+3) = (1+a)\hat{y}(n+2) - a\hat{y}(n+1) + b\Delta u(n+2)$.

Let $x = [\hat{y}(n+1)\ \hat{y}(n+2)\ \hat{y}(n+3)\ \Delta u(n)\ \Delta u(n+1)\ \Delta u(n+2)]^T$. The objective function is then:

$$\begin{aligned} J_n &= \sum_{k=1}^{3} \hat{y}^2(n+k) - 2\hat{y}(n+k)r + r^2) + \lambda \sum_{k=1}^{3} \left(\Delta u(n+k-1)\right)^2 \\ &= \frac{1}{2} x^T \begin{bmatrix} 2I_3 & 0_3 \\ 0_3 & 2\lambda I_3 \end{bmatrix} x + \begin{bmatrix} -2r & -2r & -2r & 0 & 0 & 0 \end{bmatrix} x + 3r^2 \end{aligned}$$

then, $H = \begin{bmatrix} 2I_3 & 0_3 \\ 0_3 & 2\lambda I_3 \end{bmatrix}$ and $c^T = \begin{bmatrix} -2r & -2r & -2r & 0 & 0 & 0 \end{bmatrix}$.

The equality constraints are: $\hat{y}(n+1) = x_1 = (1+a)y(n) - ay(n-1) + bx_4$, $\hat{y}(n+2) = x_2 = (1+a)x_1 - ay(n) + bx_5$ and $\hat{y}(n+3) = x_3 = (1+a)x_2 - ax_1 + bx_6$, then if $A_{\text{eq}}x = b_{\text{eq}}$ we have:

$$A_{\text{eq}} = \begin{bmatrix} 1 & 0 & 0 & -b & 0 & 0 \\ -(1+a) & 1 & 0 & 0 & -b & 0 \\ a & -(1+a) & 1 & 0 & 0 & -b \end{bmatrix}, b_{\text{eq}} = \begin{bmatrix} (1+a)y(n) - ay(n-1) \\ -ay(n) \\ 0 \end{bmatrix}.$$

For the inequality constraints: $-\Delta u(n+k-1) \leq -\Delta u_{\min}$ and $\Delta u(n+k-1) \leq \Delta u_{\max}$, then if $Ax \leq b$ we have:

$$A = \begin{bmatrix} 0_3 & I_3 \\ 0_3 & -I_3 \end{bmatrix}, b = \begin{bmatrix} \Delta u_{\max} & \Delta u_{\max} & \Delta u_{\max} & -\Delta u_{\min} & -\Delta u_{\min} & -\Delta u_{\min} \end{bmatrix}.$$

c) To reformulate the problem as a quadratic programming problem - Type 1, let choose the optimization vector as $x = [\Delta u(n)\ \Delta u(n+1)\ \Delta u(n+2)]^T$. From the equality constraints:
$\hat{y}(n+1) = (1+a)y(n) - ay(n-1) + bx_1$,
$\hat{y}(n+2) = (1+a)bx_1 + bx_2 + (1+a+a^2)y(n) - a(1+a)y(n-1)$ and
$\hat{y}(n+3) = ((1+a)^2 - a)bx_1 + (1+a)bx_2 + bx_3 + (1+a)(1+a^2)y(n) - a((1+a)^2 - a)y(n-1)$.
Then if

$$Y = \begin{bmatrix} \hat{y}(n+1) \\ \hat{y}(n+2) \\ \hat{y}(n+3) \end{bmatrix}, Y_{\text{ini}} = \begin{bmatrix} y(n-1) \\ y(n) \end{bmatrix},$$

we have: $Y = Gx + FY_{\text{ini}}$ with

$$G = \begin{bmatrix} b & 0 & 0 \\ (1+a)b & b & 0 \\ ((1+a)^2 - a)b & (1+a)b & b \end{bmatrix}, F = \begin{bmatrix} -a & (1+a) \\ -a(1+a) & (1+a+a^2) \\ -a((1+a)^2 - a) & (1+a)(1+a^2) \end{bmatrix},$$

The objective function can be written as: $J_n = (Y-r)^T(Y-r) + \lambda x'x = (Gx + FY_{\text{ini}} - r)^T(Gx + FY_{\text{ini}} - r) + \lambda x^T x = x^T(G^TG + \lambda I_3)x + 2(FY_{\text{ini}} - r)^T Gx + (FY_{\text{ini}} - r)^T(FY_{\text{ini}} - r)$. Then, since a constant does not change the location of the optimum, the objective function is defined as:

$$J_n' = \frac{1}{2}x^T Hx + c^T x, \quad H = 2(G^TG + \lambda I_3), c^T = 2(FY_{\text{ini}} - r)^T G,$$

For the inequality constraints: $\Delta u(n+k-1) \geq 0$ and $\Delta u(n+k-1) \leq \Delta u_{\max}$, then if $Ax \leq b$ we have:

$$A = \begin{bmatrix} I_3 \\ -I_3 \end{bmatrix}, b = \begin{bmatrix} \Delta u_{\max} & \Delta u_{\max} & \Delta u_{\max} \end{bmatrix}.$$

and the non-negativity constraint $x \geq 0$.

Exercise 3.2. *Solve the following QP problem of type 2:* $\min \frac{1}{2}x^T Hx + c^T x$, *s.t.* $Ax = b$, $x \geq 0$, *where*

$$H = \begin{bmatrix} 1 & -4 & 2 & 1 \\ -4 & 16 & -8 & -4 \\ 2 & -8 & 4 & 2 \\ 1 & -4 & 2 & 1 \end{bmatrix}, \ c = \begin{bmatrix} -1 \\ 0 \\ 7 \\ 4 \end{bmatrix}, \ A = [1,1,1,1], \ b = 4,$$

Solution: Note that H is singular (the first row equals the last one, and the second is -2 times the third). Since $c \not\leq 0$, we have to apply Remark 3.4 of the lecture notes and consider

$$Hx + A^T\lambda - \mu + Du_2 = -c$$

instead of equation (3.7) of the lecture notes, where $D = D(-c) = \text{diag}(1, 1, -1, -1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$.

We construct the matrix A_0, the vectors b_0, c_0 and the vector x_0 according to equation (3.10), while taking into account the modified version of equation (3.7):

$$A_0 = \begin{bmatrix} A & 0 & 0 & I & 0 \\ H & A^T & -I & 0 & D \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & -4 & 2 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -4 & 16 & -8 & -4 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & -8 & 4 & 2 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & -4 & 2 & 1 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix},$$

$$b_0 = \begin{bmatrix} 4 \\ 1 \\ 0 \\ -7 \\ -4 \end{bmatrix}, c_0 = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1]^T,$$

Selecting

$$x = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \lambda = 0, \mu = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, u_1 = b = 4, u_2 = D^{-1}(-c) = -Dc = \begin{bmatrix} 1 \\ 0 \\ 7 \\ 4 \end{bmatrix}$$

yields a feasible initial solution. However, the optimum is not found yet. The optimal solution is found selecting the columns 1, 2, 5, 8, 9 of A_0:

$$x = \begin{bmatrix} 3.2400 \\ 0.7600 \\ 0 \\ 0 \end{bmatrix}, \lambda = 0.8, \mu = \begin{bmatrix} 0 \\ 0 \\ 8.1999 \\ 5.0000 \end{bmatrix}, u_1 = 0, u_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This result can be confirmed using directly the function quadprog of Matlab:

```
x=quadprog(H,c,[],[],A,b,zeros(4,1));
```

Exercise 3.3. *Prove that the gradient of $c^T x$ is c and the Jacobian of $(Ax - b) = A^T$.*

Solution: The gradient of a function f is defined by:

$$\bigtriangledown f(x) = [\frac{\partial f}{\partial x_1} \frac{\partial f}{\partial x_2} \cdots \frac{\partial f}{\partial x_n}]^T$$

The function $c^T x$ can be written as:

$$f(x) = c^T x = c_1 x_1 + c_2 x_2 + ... + c_n x_n$$

Thus $\frac{\partial f}{\partial x_i}$ is c_i. Consequently, the gradient of $c^T x$ is:

$$\bigtriangledown c^T x = [c_1 \ c_2 \ ... \ c_n]^T = c$$

The function $Ax - b$ with $A(m \times n)$ and $b(m \times 1)$ can be rewritten as:

$$Ax - b = \begin{pmatrix} \left(\sum_{j=1}^{n} A(1,j)x_j\right) - b_1 \\ \vdots \\ \left(\sum_{j=1}^{n} A(m,j)x_j\right) - b_m \end{pmatrix}$$

Referring to the definition of the Jacobian on page 8, the Jacobian of $Ax - b$ is obtained as follows:

$$\bigtriangledown(Ax - b) = \begin{pmatrix} A(1,1) & A(2,1) & \cdots & A(m,1) \\ A(1,2) & A(2,2) & \cdots & A(m,2) \\ \vdots & \vdots & \ddots & \vdots \\ A(1,n) & A(2,n) & \cdots & A(m,n) \end{pmatrix},$$

which is in fact A^T.

Exercise 3.4. *Solve the following optimization problem:*

$$\min f(x) = -8x_1 - 16x_2 + x_1^2 + 4x_2^2$$

subject to: $x_1 + x_2 \leq 5$, $x_1 \leq 3$, $x_1 \geq 0$, $x_2 \geq 0$

Solution: The quadratic problem's variables and matrices are given below. As can be seen, the H matrix is positive definite so the KKT conditions are necessary and sufficient for a global optimum. Note that we have a standard QP problem of type 1 with

$$c = \begin{bmatrix} -8 \\ -16 \end{bmatrix} \quad H = \begin{bmatrix} 2 & 0 \\ 0 & 8 \end{bmatrix} \quad A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \quad b = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

To solve the KKT equations for this problem, we use the approach of Section 3.1 of the lecture notes. First, we transform the problem into a type-2 problem, by introducing slack variables $y \geq 0$ such that $Ax + y = b$.

The KKT equations are then as follows:

$$\begin{aligned} x_1 + x_2 + y_1 &= 5 \\ x_1 + y_2 &= 3 \\ 2x_1 + \lambda_1 + \lambda_2 - \mu_1 &= 8 \\ 8x_2 + \lambda_1 - \mu_2 &= 16 \\ x, y, \mu &\geq 0 \\ x^T \mu &= 0 \end{aligned}$$

To create the appropriate linear program, we add artificial variables to each constraint and minimize their sum:

$$\min \; u_1 + u_2 + u_3 + u_4$$

$$\begin{aligned} \text{subject to} \; & x_1 + x_2 + y_1 + u_1 = 5 \\ & x_1 + y_2 + u_2 = 3 \\ & 2x_1 + \lambda_1 + \lambda_2 - \mu_1 + u_3 = 8 \\ & 8x_2 + \lambda_1 - \mu_2 + u_4 = 16 \\ & x, y, \mu, u \geq 0 \\ & x^T \mu = 0 \end{aligned}$$

Applying the modified simplex technique to this example, yields the sequence of iterations given in Table 3. The optimal solution to the original problem is $(x_1^*, x_2^*) = (3, 2)$.

Table 3: Simplex iterations for QP problem

Iteration	Basic variables	Solution	Objective value	Entering variable	Leaving variable
0	(u_1, u_2, u_3, u_4)	(5,3,8,16)	32	x_2	u_4
1	(u_1, u_2, u_3, x_2)	(3,3,8,2)	14	x_1	u_1
2	(x_1, u_2, u_3, x_2)	(3,0,2,2)	2	λ_1	u_2
3	$(x_1, \lambda_1, u_3, x_2)$	(3,0,2,2)	2	λ_2	u_3
4	$(x_1, \lambda_1, \lambda_2, x_2)$	(3,0,2,2)	0	-	-

4 Exercises for Chapter 4: Nonlinear optimization without constraints

Exercise 4.1. *Perform three iterations to find the minimum of* $f(x_1, x_2) = (x_1-3)^4+(x_1-3x_2)^2$ *using:*

- *Newton's method (use* $x_0 = [0, 0]^T$*).*
- *Levenberg-Marquardt's method (use* $x_0 = [0, 0]^T$*, and* λ*=1.1).*
- *Broyden-Fletcher-Goldfarb-Shanno's method (use* $x_0 = [0, 0]^T$*).*
- *Davidon-Fletcher-Powell's method (use* $x_0 = [0, 0]^T$*).*

Solution: The gradient and the Hessian of f are given by

$$\nabla f = \begin{bmatrix} 4(x_1 - 3)^3 + 2(x_1 - 3x_2) \\ -6(x_1 - 3x_2) \end{bmatrix} \quad \text{and} \quad H = \begin{bmatrix} 12(x_1 - 3)^2 + 2 & -6 \\ -6 & 18 \end{bmatrix}$$

The results for the Newton method are in Table 4. The Newton method is as follows:

$$x_{k+1} = x_k - H^{-1}(x_k)\nabla f(x_k)$$

The results for the Levenberg-Marquardt method are in Table 5. The Levenberg-Marquardt's method is as follows:

$$x_{k+1} = x_k - (\hat{H}(x_k))^{-1}\nabla f(x_k),$$
$$\hat{H}(x_k) = H(x_k) + \lambda I$$

The results for the Broyden-Fletcher-Goldfarb-Shanno method are in Table 6, and the method is:

$$x_{k+1} = x_k - (\hat{H}_k)^{-1}\nabla f(x_k),$$
$$\hat{H}_k = \hat{H}_{k-1} + \frac{q_k q_k^T}{q_k^T s_k} - \frac{\hat{H}_{k-1} s_k s_k^T \hat{H}_{k-1}^T}{s_k^T \hat{H}_{k-1} s_k},$$
$$s_k = x_k - x_{k-1}, \quad q_k = \nabla f(x_k) - \nabla f(x_{k-1}),$$
$$\hat{H}_0 = H(x_0)$$

The results for the Davidon-Fletcher-Powell method are in Table 7, and the method is:

$$x_{k+1} = x_k - \hat{D}_k \nabla f(x_k),$$
$$\hat{D}_k = \hat{D}_{k-1} + \frac{s_k s_k^T}{q_k^T s_k} - \frac{\hat{D}_{k-1} q_k q_k^T \hat{D}_{k-1}^T}{q_k^T \hat{D}_{k-1} q_k},$$
$$s_k = x_k - x_{k-1}, \quad q_k = \nabla f(x_k) - \nabla f(x_{k-1}),$$
$$\hat{D}_0 = H(x_0)^{-1}$$

Table 4: Results for the Newton method

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	$[H(x_k)]^{-1}$
0	(0.0000,0.0000)	81.0000	(-108.0000,0.0000)	$\begin{bmatrix} 0.0093 & 0.0031 \\ 0.0031 & 0.0566 \end{bmatrix}$
1	(1.0044,0.3348)	15.8597	(-31.7893,0.0186)	$\begin{bmatrix} 0.0209 & 0.0070 \\ 0.0070 & 0.0579 \end{bmatrix}$
2	(1.6688,0.5573)	3.1403	(-9.4422,0.0054)	$\begin{bmatrix} 0.04700 & 0.0157 \\ 0.0157 & 0.0608 \end{bmatrix}$
3	(2.1123,0.7044)	0.6210	(-2.7999,0.0024)	$\begin{bmatrix} 0.1058 & 0.0353 \\ 0.0353 & 0.0673 \end{bmatrix}$

Table 5: Results for the Levenberg-Marquardt method

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	$[\hat{H}(x_k)]^{-1}$
0	(0.0000,0.0000)	81.0000	(-108.0000,0.0000)	$\begin{bmatrix} 0.0092 & 0.0029 \\ 0.0029 & 0.0533 \end{bmatrix}$
1	(0.9936,0.3132)	16.2087	(-32.2002,-0.3240)	$\begin{bmatrix} 0.0202 & 0.0063 \\ 0.0063 & 0.0543 \end{bmatrix}$
2	(1.6461,0.5337)	3.3621	(-9.8370,-0.2700)	$\begin{bmatrix} 0.0431 & 0.0135 \\ 0.0135 & 0.0566 \end{bmatrix}$
3	(2.0737,0.6818)	0.7370	(-3.1226,-0.1698)	$\begin{bmatrix} 0.0869 & 0.0273 \\ 0.0273 & 0.0609 \end{bmatrix}$

Now, just to see the behavior of the algorithms, in the Figure 4 we can see the evolution of the variables x_1, x_2 as function of the iterations.

Exercise 4.2. *Use the golden section method to find the value of x that minimizes the function*

$$f(x) = -\min\left\{\frac{x}{2}, 2-(x-3)^2, 2-\frac{x}{2}\right\}.$$

Use the fact that the function is strictly unimodal on $[0,8]$. Perform five iterations. Compare the results with those obtained with the Fibonacci method and with a fixed-step method (take a step length $\Delta s = 2$).

Table 6: Results for the Broyden-Fletcher-Goldfarb-Shanno approach

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	$(\hat{H}_k)^{-1}$
0	(0.0000,0.0000)	81.0000	(-108.0000,0.0000)	$\begin{bmatrix} 0.0093 & 0.0031 \\ 0.0031 & 0.0566 \end{bmatrix}$
1	(1.0000,0.3333)	16.0000	(-32,0.0000)	$\begin{bmatrix} 0.0132 & 0.0044 \\ 0.0044 & 0.0570 \end{bmatrix}$
2	(1.4224,0.4741)	6.1942	(-15.7053,-0.0006)	$\begin{bmatrix} 0.0259 & 0.0086 \\ 0.0086 & 0.0584 \end{bmatrix}$
3	(1.8292,0.6092)	1.8790	(-6.4164,-0.0096)	$\begin{bmatrix} 0.0438 & 0.0146 \\ 0.0146 & 0.0604 \end{bmatrix}$

Table 7: Results for the Davidon-Fletcher-Powell approach

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	$(\hat{H}_k)^{-1}$
0	(0.0000,0.0000)	81.0000	(-108.0000,0.0000)	$\begin{bmatrix} 0.0092 & 0.0031 \\ 0.0031 & 0.0566 \end{bmatrix}$
1	(1.0044,0.3348)	16.0000	(-32.0000,0.0000)	$\begin{bmatrix} 0.0132 & 0.0044 \\ 0.0044 & 0.0570 \end{bmatrix}$
2	(1.6688,0.5573)	6.1942	(-9.4422,0.0054)	$\begin{bmatrix} 0.0259 & 0.0086 \\ 0.0086 & 0.0584 \end{bmatrix}$
3	(2.1123,0.7044)	1.8790	(-2.7999,0.0024)	$\begin{bmatrix} 0.0438 & 0.0146 \\ 0.0146 & 0.0604 \end{bmatrix}$

Solution: We first evaluate the fixed-step method, at the points $x = 0$, $x = 2$, $x = 4$, $x = 6$ and $x = 8$. The minimum value is in $x = 2$ as can be seen in the Figure 5.

Now we use the golden section method. This yields

l	a_l	b_l	c_l	d_l	$f(a_l)$	$f(b_l)$	$f(c_l)$	$f(d_l)$
0	0.0000	3.0557	4.9443	8.0000	7.0000	-0.4721	1.7802	23.0000
1	0.0000	1.8885	3.0557	4.9443	7.0000	-0.7647	-0.4721	1.7802
2	0.0000	1.1672	1.8885	3.0557	7.0000	1.3592	-0.7647	-0.4721
3	1.1672	1.8885	2.3344	3.0557	1.3592	-0.7647	-0.8328	-0.4721
4	1.8885	2.3344	2.6099	3.0557	-0.7647	-0.8328	-0.6950	-0.4721
5	1.8885	2.1641	2.3344	2.6099	-0.7647	-0.9180	-0.8328	-0.6950

The best solution found with the golden section method in 5 iterations is $x = 2.1641$.

Now we apply the Fibonacci method. If we choose $n = 7$, this yields

l	a_l	b_l	c_l	d_l	$f(a_l)$	$f(b_l)$	$f(c_l)$	$f(d_l)$
0	0.0000	3.0476	4.9524	8.0000	7.0000	-0.4762	1.8118	23.0000
1	0.0000	1.9048	3.0476	4.9524	7.0000	-0.8005	-0.4762	1.8118
2	0.0000	1.1429	1.9048	3.0476	7.0000	1.4490	-0.8005	-0.4762
3	1.1429	1.9048	2.2857	3.0476	1.4490	-0.8005	-0.8571	-0.4762
4	1.9048	2.2857	2.6667	3.0476	-0.8005	-0.8571	-0.6667	-0.4762
5	1.9048	2.2857	2.2857	2.6667	-0.8005	-0.8571	-0.8571	-0.6667

The best solution found with the Fibonacci method in 5 iterations is $x = 2.2857$.

However in order to iterate more with the Fibonacci algorithm (to get the optimal solution), we need to increase n. For this purpose we choose $n = 18$. The results for the first 5 iterations are similar to ones achieved by the Golden section method.

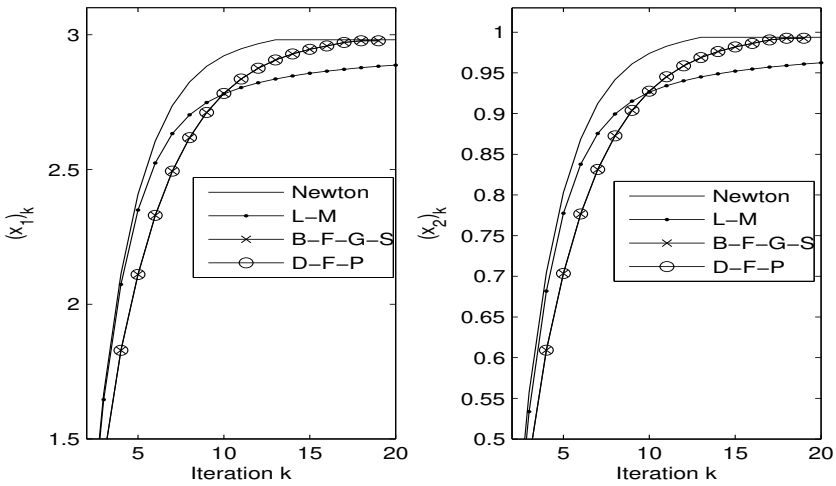

Figure 4: (a) x_1 as function of the iteration steps, (b) x_2 as function of the iteration steps

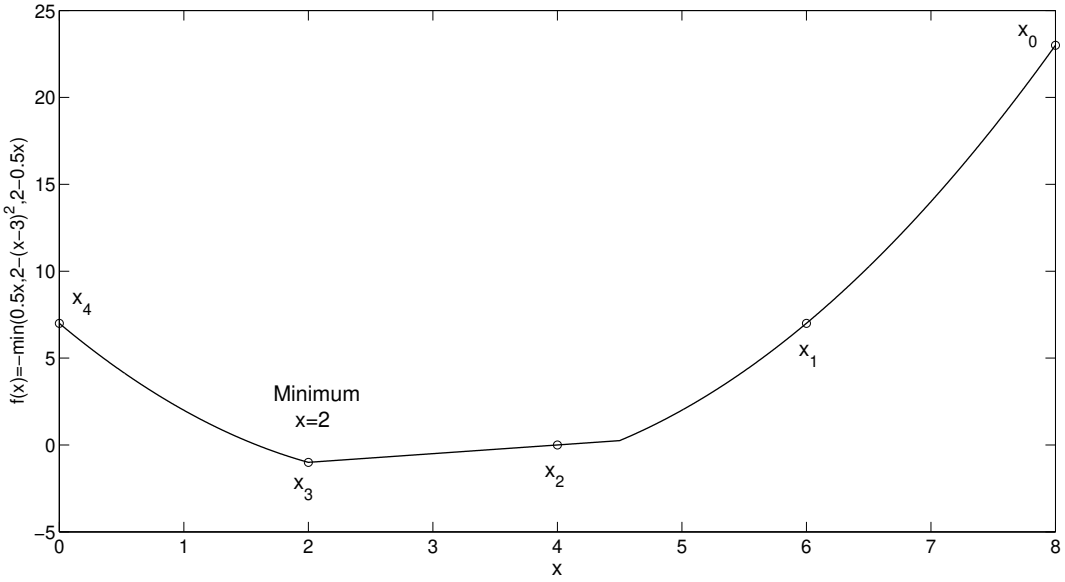

Figure 5: Function $f(x)$. The circles indicate the data points evaluated with the fixed-step method

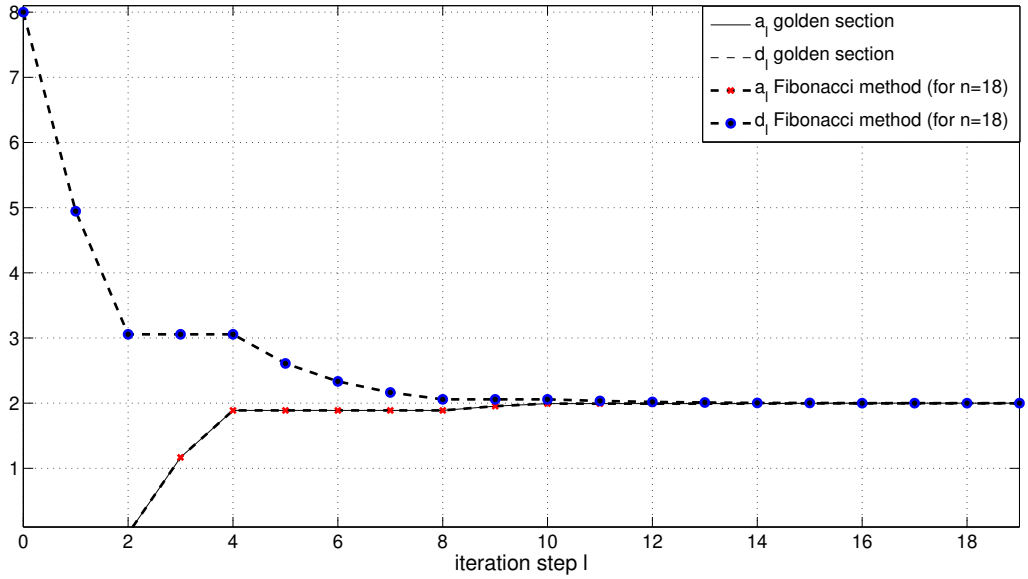

Figure 6: a_l and d_l for the golden section method and the Fibonacci method

To compare the algorithms, Figure 6 shows the values of a_l and d_l as a function of the iteration step l. As can be seen, both algorithms (golden section and Fibonacci) converge to the optimal solution.

Exercise 4.3. *Answer the following questions:*

- *Why is Newton's method for minimizing multivariate functions not a descent method and how should it be modified to become a descent method?*
  Note: An optimization method is called a descent method if $f(x_{k+1}) \leq f(x_k)$ for all k, where f is the objective function and x_k is the kth iteration point.
- *Is the steepest-descent algorithm is a descent method?*
- *Are the steps in the steepest-descent algorithm orthogonal?*

Solution:

- Newton's method is not necessarily a descent method since the Newton direction is not necessarily a descent direction and since — even if it is — the step to be taken may be to big.
  To get a descent method we can apply the Levenberg-Marquardt method with λ selected such that the direction is a descent direction and the step size taken small enough (i.e., we then apply a line search method using the Levenberg-Marquardt direction).
- The direction taken in the steepest-descent algorithm, i.e., the negative gradient $-\nabla f$ is a descent direction (see the lecture notes for the proof). If in the line search method we then always select a step size that is small enough, we get a descent method.
- Consider the kth iteration step, where we determine the optimal step size s_k in the direction $d_k = -\nabla f(x_k)$. The optimal step size s_k^* must verify $\dfrac{df(x_k + s_k d_k)}{ds_k} = 0$, i.e., we

Table 8: Iterations with the steepest-descent algorithm

k	x_k^T	$f(x_k)$	$\nabla f(x_k)^T$	$\|\nabla f(x_k)^T\|_2$
0	(0.0000,0.0000)	40.0000	(-4.0000,-24.0000)	24.3311
1	(0.1644,0.9864)	19.5878	(-3.6712,-16.1088)	16.5218
2	(0.3866,1.9614)	6.9178	(-3.2268,-8.3088)	8.9134
3	(0.7486,2.8936)	1.6113	(-2.5028,-0.8512)	2.6436
4	(1.6953,3.2156)	0.2788	(-0.6094,1.7248)	1.8293
5	(2.0284,2.2727)	2.1167	(0.0568,-5.8184)	5.8187
6	(2.0186,3.2726)	0.2976	(0.0372,2.1808)	2.1811
7	(2.0015,2.2727)	2.1159	(0.0030,-5.8184)	5.8184
8	(2.0010,3.2727)	0.2975	(0.0020,2.1816)	2.1816
9	(2.0001,2.2727)	2.1159	(0.0002,-5.8184)	5.8184

must find the point in which the line $x_k + s_k d_k$ is the tangent line to a contour line of f. Since the gradient in a given point is always orthogonal to the contour line, we find that $d_{k+1} = -\nabla f(x_k + s_k^* d_k)$ is orthogonal to d_k.

Exercise 4.4. *Using a steepest-descent method with update formula*

$$x_{k+1} = x_k - \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|_2} \ ,$$

find the minimum of the quadratic function

$$f(x_1, x_2) = (x_1 - 2)^2 + 4(x_2 - 3)^2,$$

starting from $x_0 = [0, 0]^T$. Using Matlab, plot the algorithmic moves (x_k as function of k) and verify the zigzag property of the algorithm.

Solution: The optimization problem is solved using the following steepest-descent method with

$$x_{k+1} = x_k - \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|_2} = \begin{pmatrix} (x_k)_1 \\ (x_k)_2 \end{pmatrix} - \frac{\begin{pmatrix} 2((x_k)_1 - 2) \\ 8((x_k)_2 - 3) \end{pmatrix}}{\sqrt{(2((x_k)_1 - 2))^2 + (8((x_k)_2 - 3))^2}}$$

The results of the first 10 iterations are listed in the Table 8.

In Figure 7, the function $f(x)$, its contour plot as well as the steps of the iterative method are presented. The position of the optimal point is in $(2, 3)$.

Exercise 4.5. *Show that the choice of λ in the golden section method indeed results in reuse of points from one iteration to the next.*

Solution: In the Golden section method, we can construct three sub-intervals of $[a_l, b_l]$ by choosing:

$$b_l = \lambda a_l + (1 - \lambda) d_l$$

$$c_l = (1 - \lambda) a_l + \lambda d_l$$

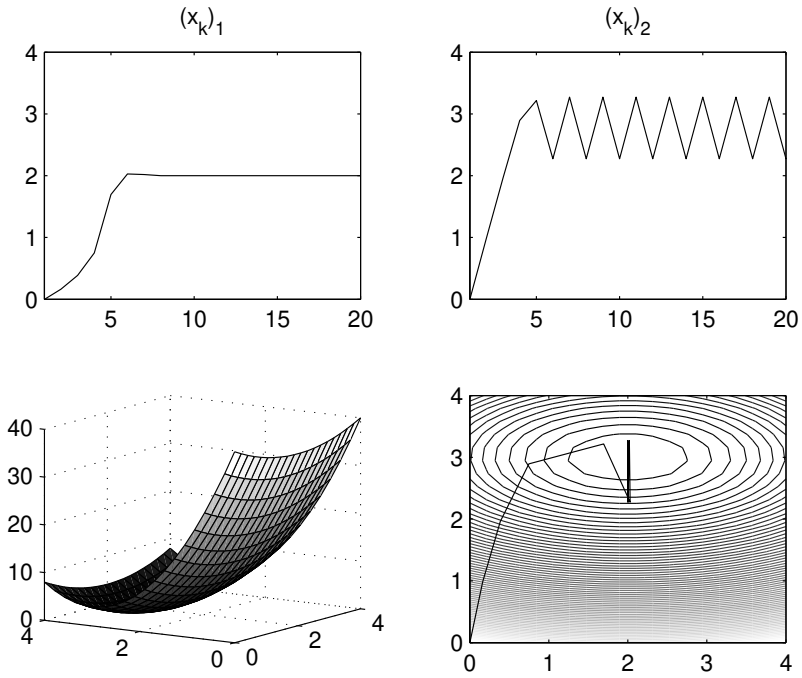

Figure 7: (a) x_1 as function of the iteration steps, (b) x_2 as function of the iteration steps, (c) Function $f(x)$, (d) Contour plot of function $f(x)$

with $\lambda = \frac{1}{2}(\sqrt{5}-1)$. If $\bar{f}_k(b_l) > \bar{f}_k(c_l)$ we know that the minimum must be in the interval $[b_l, d_l]$. Thus we can define $a_{l+1} = b_l,\ d_{l+1} = d_l$ and compute $b_{l+1} = \lambda a_{l+1} + (1-\lambda)d_{l+1}$ and $c_{l+1} = (1-\lambda)a_{l+1} + \lambda d_{l+1}$. For b_{l+1} we can write:

$$b_{l+1} = \lambda\big(\lambda a_l + (1-\lambda)d_l\big) + (1-\lambda)d_l = \lambda^2 a_l + (1-\lambda^2)d_l$$

$$= (\frac{1}{4}(6-2\sqrt{5})a_l + (1-\frac{3}{2}+\frac{1}{2}\sqrt{5})d_l = (1-\lambda)a_l + \lambda d_l = c_l$$

We can prove in a similar way that $c_{l+1} = b_l$ when $\bar{f}_k(b_l) < \bar{f}_k(c_l)$.

Exercise 4.6. *Show that the choice of λ in the Fibonacci method indeed results in reuse of points from one iteration to the next.*

Solution: Suppose we have:

$$b_l = \lambda_l a_l + (1-\lambda_l)d_l$$

$$c_l = (1-\lambda_l)a_l + \lambda_l d_l$$

with $\lambda_l = \frac{\mu_{n-l}}{\mu_{n+1-l}}$. If we substitute it in λ_l, we obtain:

$$b_l = \frac{\mu_{n-l}}{\mu_{n+1-l}}a_l + (1-\frac{\mu_{n-l}}{\mu_{n+1-l}})d_l$$

$$c_l = (1-\frac{\mu_{n-l}}{\mu_{n+1-l}})a_l + \frac{\mu_{n-l}}{\mu_{n+1-l}}d_l$$

Now for the $(l+1)$th iteration, $\lambda_{l+1} = \frac{\mu_{n-1-l}}{\mu_{n-l}}$. If $\bar{f}_k(b_l) > \bar{f}_k(c_l)$ we know that the minimum must be in the interval $[b_l, d_l]$. Thus we can define $a_{l+1} = b_l,\ d_{l+1} = d_l$ and compute $b_{l+1} =$

$\lambda_{l+1}a_{l+1} + (1-\lambda_{l+1})d_{l+1}$ and $c_{l+1} = (1-\lambda_{l+1})a_{l+1} + \lambda_{l+1}d_{l+1}$. If we substitute a_{l+1} with b_l we will get:

$$b_{l+1} = \lambda_{l+1}b_l + (1-\lambda_{l+1})d_l = \frac{\mu_{n-1-l}}{\mu_{n-l}}\left[\frac{\mu_{n-l}}{\mu_{n+1-l}}a_l + \big(1 - \frac{\mu_{n-l}}{\mu_{n+1-l}}\big)d_l\right] + (1 - \frac{\mu_{n-1-l}}{\mu_{n-l}})d_l$$

$$\Rightarrow b_{l+1} = \frac{\mu_{n-1-l}}{\mu_{n+1-l}}a_l + (1 - \frac{\mu_{n-1-l}}{\mu_{n+1-l}})d_l$$

On the other hand, we have the Fibonacci recursion equation $\mu_{n+1-l} = \mu_{n-l} + \mu_{n-1-l}$. Hence, if we substitute in the b_{l+1} equation we get:

$$b_{l+1} = \frac{\mu_{n+1-l} - \mu_{n-l}}{\mu_{n+1-l}}a_l + \frac{\mu_{n-l}}{\mu_{n+1-l}}d_l = (1 - \frac{\mu_{n-l}}{\mu_{n+1-l}})a_l + \frac{\mu_{n-l}}{\mu_{n+1-l}}d_l = c_l$$

Similarly, for the case $\bar{f}_k(b_l) < \bar{f}_k(c_l)$, we choose $a_{l+1} = a_l,\ d_{l+1} = c_l$ and by working out the equations we end up with $c_{l+1} = b_l$.

Exercise 4.7. *Prove the expressions on page 36 for gradient of $f(x)$ and the Hessian (4.1)*

Solution: The function $f(x) = e^T(x)e(x)$ is given by:

$$f(x) = e^T(x)e(x) = e_1(x)^2 + e_2(x)^2 + ... + e_N(x)^2$$

Hence the gradient is obtained as:

$$\bigtriangledown f(x) = \begin{pmatrix} 2\frac{\partial e_1}{\partial x_1}e_1 + ... + 2\frac{\partial e_N}{\partial x_1}e_N \\ 2\frac{\partial e_1}{\partial x_2}e_1 + ... + 2\frac{\partial e_N}{\partial x_2}e_N \\ \vdots \\ 2\frac{\partial e_1}{\partial x_n}e_1 + ... + 2\frac{\partial e_N}{\partial x_n}e_N \end{pmatrix}$$

It is can be easily observed that the above matrix is equal to the following multiplication:

$$2.\begin{pmatrix} \frac{\partial e_1}{\partial x_1} & \frac{\partial e_2}{\partial x_1} & \cdots & \frac{\partial e_N}{\partial x_1} \\ \frac{\partial e_1}{\partial x_2} & \frac{\partial e_2}{\partial x_2} & \cdots & \frac{\partial e_N}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_1}{\partial x_n} & \frac{\partial e_2}{\partial x_n} & \cdots & \frac{\partial e_N}{\partial x_n} \end{pmatrix}.[e_1\ e_2\ \cdots e_N]^T$$

Thus the gradient of f is:

$$\nabla f(x) = 2\nabla e(x)e(x)$$

The Hessian matrix can be determined as:

$$\begin{pmatrix} 2\frac{\partial^2 e_1}{\partial x_1^2}e_1 + 2\left(\frac{\partial e_1}{\partial x_1}\right)^2 ... + 2\frac{\partial^2 e_N}{\partial x_1^2}e_N + 2\left(\frac{\partial e_N}{\partial x_1}\right)^2 & \cdots & 2\frac{\partial^2 e_1}{\partial x_1 \partial x_n}e_1 + 2\frac{\partial e_1}{\partial x_n}\frac{\partial e_1}{\partial x_1} ... + 2\frac{\partial^2 e_N}{\partial x_1 \partial x_n}e_N + 2\frac{\partial e_N}{\partial x_n}\frac{\partial e_N}{\partial x_1} \\ \vdots & \ddots & \vdots \\ 2\frac{\partial^2 e_1}{\partial x_n \partial x_1}e_1 + 2\frac{\partial e_1}{\partial x_1}\frac{\partial e_1}{\partial x_n} ... + 2\frac{\partial^2 e_N}{\partial x_n \partial x_1}e_N + 2\frac{\partial e_N}{\partial x_1}\frac{\partial e_N}{\partial x_n} & \cdots & 2\frac{\partial^2 e_1}{\partial x_n^2}e_1 + 2\left(\frac{\partial e_1}{\partial x_n}\right)^2 ... + 2\frac{\partial^2 e_N}{\partial x_n^2}e_N + 2\left(\frac{\partial e_N}{\partial x_n}\right)^2 \end{pmatrix}$$

It is straightforward to see that the above matrix can be decomposed to the components:

$$H(x) = 2\nabla e(x)\nabla^T e(x) + \sum_{i=1}^{N} 2\nabla^2 e_i(x)e_i(x)$$

5 Exercises for Chapter 5: Constraints in nonlinear optimization

Exercise 5.1. *Consider the constrained minimization problem:*

$$\begin{aligned}&\min f(x_1, x_2, x_3) = x_1^2 + 8x_2^2 + 3x_1x_3\\ &\textit{subject to}\\ &x_1 - x_2 + x_3 = 1\\ &x_1 + x_2 = 2\end{aligned}$$

Solve this problem using the method of elimination of constraints.

Solution: From the equality constraints we get: $x_1 = -x_2 + 2$ and $x_3 = 2x_2 - 1$. Replacing this in the objective function yields:

$$f(x_1, x_2, x_3) = F(x_2) = (-x_2 + 2)^2 + 8x_2^2 + 3(-x_2 + 2)(2x_2 - 1) = 3x_2^2 + 11x_2 - 2 \ .$$

Analytically we can obtain the local optimum as follows:

$$\frac{\partial F}{\partial x_2} = 6x_2 + 11 = 0 \Rightarrow x_2^* = -\frac{11}{6} \approx -1.8333 \ .$$

With the second-order condition we can check that $x_2 = -\frac{11}{6}$ is a local minimum (since $F''(x_2) = 6 > 0$).

Then using this minimum we obtain the solution for the problem: $x = (\frac{23}{6}, -\frac{11}{6}, -\frac{14}{3})$. This result can be verified with the function fmincon of Matlab.
In an m-file called `Chapter5_solutioncodes.m`, we define the initial point `x0`, and the equality constraints `Aeq * x <= beq\`. The resulting code is as follows:

```
x0=[0;0;0];
Aeq=[1,-1,1;1,1,0];
beq=[1;2];
option=optimoptions('Algorithm','sqp');
x=fmincon(@(x)((x(1))^2+8*(x(2))^2+3*x(1)*(x(3))),x0,[],[],...
       Aeq,beq,[],[],[],option)
```

The final result is the vector $x = [3.8333 \ -1.8333 \ ; -4.6667]^T$.

Exercise 5.2. *Using Matlab, apply sequential quadratic programming to solve the problem:*

$$\begin{aligned}&\min f(x_1, x_2) = (x_1 - \tfrac{9}{4})^2 + (x_2 - 2)^2\\ &\textit{subject to}\\ &x_1^2 - x_2 \leq 0\\ &x_1 + x_2 \leq 6\\ &x_1, x_2 \geq 0\end{aligned}$$

Starting with the point $x_0 = [0, 0]^T$, show the evolution of the search direction d_k (Step 2 of the algorithm as listed in the lecture notes), step length in the line optimization s_k of (Step 3), and the optimization variables $(x_k)_1$, $(x_k)_2$ as function of the iteration step k.

Solution: We use the function fmincon of Matlab. In an m-file called `fun5_sequentialprog.m` we write the function:

Table 9: Iterations with the steepest-descent algorithm

k	$F-count$	$f(x)$	Max constraint	Linesearch steplength	Directional derivative	First − order optimalityProcedure
0	3	9.0625	0			
1	8	3.78906	-0.02734	0.25	-6.02	5.63
2	12	1.07034	0.2576	0.5	-3.75	1.21
3	15	1.6113	0.03713	1	-1.38	0.147
4	18	0.609201	0.0006924	1	0.255	0.0272
5	21	0.625	6.563e-008	1	0.231	0.000696

```
function f = fun5_sequentialprog(x)
f = (x(1)-9/4)^2 + (x(2)-2)^2;
```

Then, the non linear constraints are in a m-file called `con5_sequentialprog.m`:

```
function [c,ceq] = con5_sequentialprog(x)
        c(1) = x(1)^2 - x(2);
        c(2)= x(1)+x(2)-6;
        ceq = [];
```

Then, we define the initial point $x0$, the lower bounds LB, and the options to see the various measures of progress while the algorithm executes:

```
x0=[0;0];
LB=[0;0];
options = optimoptions('Algorithm','sqp',...
'PlotFcns',{@optimplotfunccount,@optimplotfval,...
@optimplotstepsize,@optimplotfirstorderopt})
```

With @*optimplotfunccount* we can see the function count, @*optimplotfval* plots the function value, @*optimplotstepsize* plots the step size, and @*optimplotfirstorderopt* plots the first-order optimality measure.

Finally, we use fmincon:

```
x = fmincon(@(x) fun5_sequentialprog(x),x0,[],[],[],[],LB,[],...
@(x)con5_sequentialprog(x),options)
```

The final results is the vector $x = [1.5000, 2.2501]^T$.

The results of the iterations are in the Table 9.

In the Figure 8, the variables x_1 and x_2 as function of the iteration steps. In the Figure 9, the function evaluations, function values, step size, first-order optimality, as function of the iterations.

Exercise 5.3. *Solve the problem:*

$$\max_{x,y,z}(x+y)$$

subject to $x^2 + 2y^2 + z^2 = 1$ *and* $x + y + z = 1$.

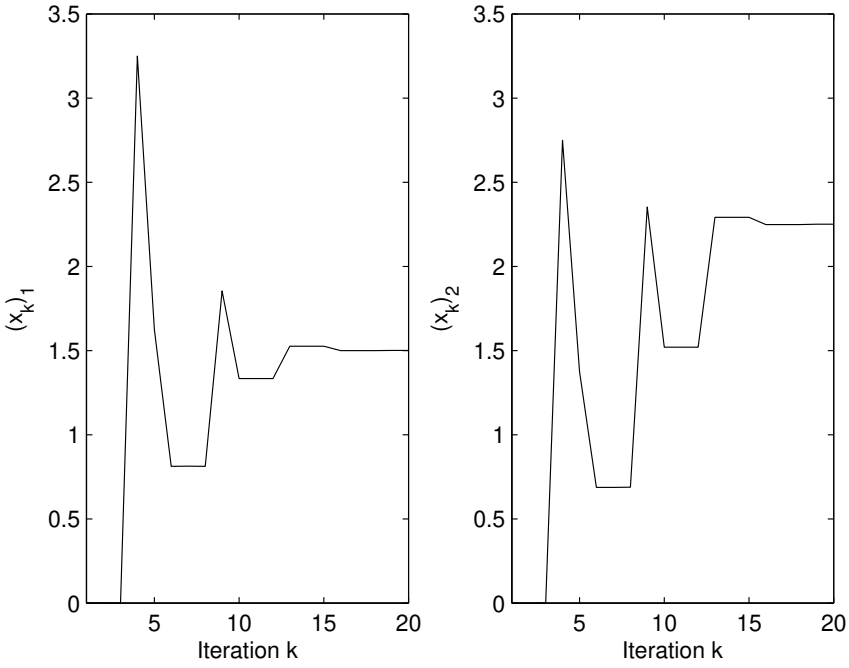

Figure 8: (a) x_1 as function of the iteration steps, (b) x_2 as function of the iteration steps

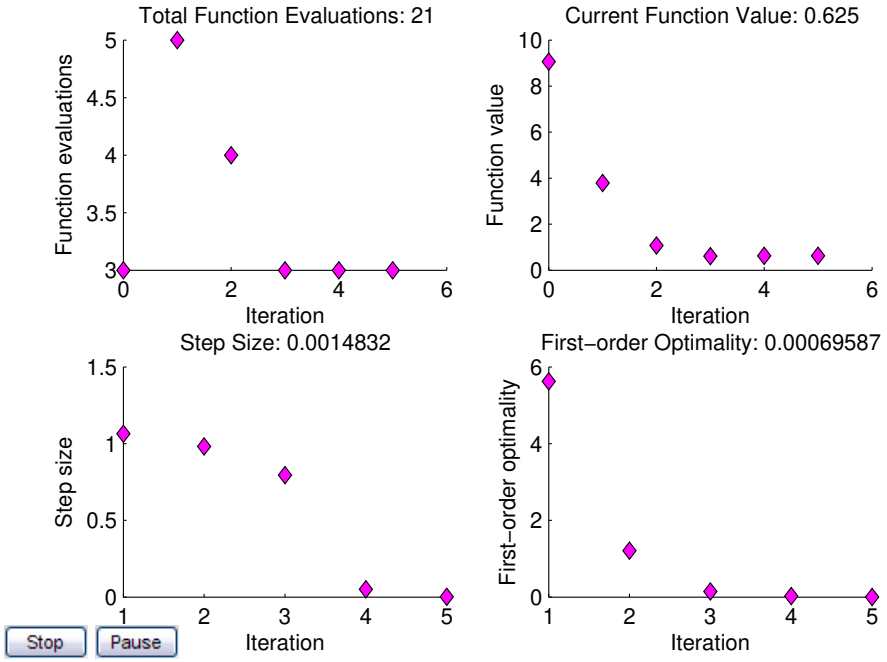

Figure 9: (a) function evaluations, (b) function values, (c) step size, (d) first-order optimality

Solution: For an equality constrained optimization problem necessary conditions for a minimum of the function f in (x, y, z), satisfying $h(x, y, z)$ are given by the Kuhn-Tucker conditions:

$$\begin{aligned} \nabla f(x, y, z) + \nabla h(x, y, z)\lambda &= 0 \\ h(x, y, z) &= 0 \end{aligned}$$

In our case, after rewriting the maximization problem as a minimization problem with objective function $-x - y$, the conditions are as follows:

$$\begin{aligned} -1 + 2\lambda_1 x + \lambda_2 &= 0 \\ -1 + 4\lambda_1 y + \lambda_2 &= 0 \\ 2\lambda_1 z + \lambda_2 &= 0 \\ x^2 + 2y^2 + z^2 &= 1 \\ x + y + z &= 1 \end{aligned}$$

To find the values of x, y, z, λ_1, λ_2 that solve these equations, we can first use the third condition to eliminate λ_2 (so $\lambda_2 = 2 - \lambda_1 z$), and then use the fifth condition to eliminate z (so $z = 1 - x - y$). Then we obtain:

$$\begin{aligned} -1 + \lambda_1[4x + 2y - 2] &= 0 \\ -1 + \lambda_1[2x + 6y - 2] &= 0 \\ 2x^2 - 3y^2 + 2y - 2xy + 2x &= 0 \end{aligned}$$

The first two equations yield $x = 2y$, so that the third equation is $3y(5y-2) = 0$, so that either $y = 0$ or $y = 2/5$. Thus there are two solutions of the first-order conditions and the constraints, namely $(0, 0, 1)$ with $\lambda_1 = -1/2$ and $\lambda_2 = 1$, and $(4/5, 2/5, -1/5)$ with $\lambda_1 = 5/18$ and $\lambda_2 = 1/9$. Now if we plug the two solutions in the objective function, the value of the function is higher at the second solution. Hence, the second solution is the solution of the problem.

6 Exercises for Chapter 6: Convex optimization

Exercise 6.1. *Perform two iterations of the ellipsoid algorithm to solve the program:*

$$\begin{aligned} &min f(x_1, x_2) = 4(x_1 - 10)^2 + (x_2 - 4)^2 \\ &\textit{subject to} \\ &x_1 - x_2 \leq 10 \\ &x_1 - x_2 \geq 3 \\ &x_1 \geq 0 \end{aligned}$$

Plot the feasible region and the algorithmic steps. Take $[0, 0]^T$ *as starting point.*

Solution: In m-files called `fun6_ellipsoid_f.m` and `fun6_ellipsoid_g.m` we write the functions:

```
function [f,ff]=fun6_ellipsoid_f(x)
f=4*(x(1)-10)^2+(x(2)-4)^2;
ff=[8*(x(1)-10),2*(x(2)-4)];
```

```
function [g,gg]=fun6_ellipsoid_g(x)
g=[x(1)-x(2)-10;3-x(1)+x(2);-x(1)];
gg=[1 -1; -1 1; -1 0];
```

We will show the results using the ellipsoid algorithm. Define the initial condition $x0$, the initial ellipsoid $A0$, and the parameter n:

```
x0=[0;0];
A0=[100 0; 0 100];
n=2;
```

Then we evaluate $f(x0)$, $\nabla f(x0)$, $g(x0)$, $\nabla g(x0)$:

```
[f,ff]=fun6_ellipsoid_f(x0);
[g,gg]=fun6_ellipsoid_g(x0);
```

We check if $g(x0) <= 0$. If it is, then the next iteration is:

```
if(g<=0)
  x1=x0-(1/(1+n))*A0*ff'/(sqrt(abs(ff*A0*ff')));
  A1=(n^2/(n^2-1))*(A0-(2/(n+1))*A0*ff'*ff*A0'/(ff*A0*ff'));
```

If the constraint $g(x0) <= 0$ is not satisfied, we select the vector of the subgradient $\nabla g(x0)$ related with the index of the bigger value of $g(x0)$, and the iteration is:

```
else
  [value,pos]=max(g);
  x1=x0-(1/(1+n))*A0*gg(pos,:)'/...
     (sqrt(abs(gg(pos,:)*A0*gg(pos,:)')));
  A1=(n^2/(n^2-1))*(A0-(2/(n+1))*A0*gg(pos,:)'*...
     gg(pos,:)*A0'/(gg(pos,:)*A0*gg(pos,:)'));
end
```

In the Figure 10, the variables x_1 and x_2 and the function evaluations are shown. In the figure it is also shown the feasible region (within the lines). We can see how the algorithm converges to the optimal solution $[10, 4]^T$.

Exercise 6.2. *Use the interior-point algorithm to solve the program:*

$$\begin{aligned} &min f(x_1, x_2) = -x_1 x_2 \\ &\textit{subject to} \\ &1 - x_1^2 - x_2^2 \geq 0 \end{aligned}$$

Plot the feasible region and the algorithmic steps. Use first $[0.1, 0.1]^T$ *and next* $[-0.1, -0.1]^T$ *as starting points.*

Solution:

We use the function fminunc of Matlab. In an m-file called `fun6_interior_barrier.m` we write the function:

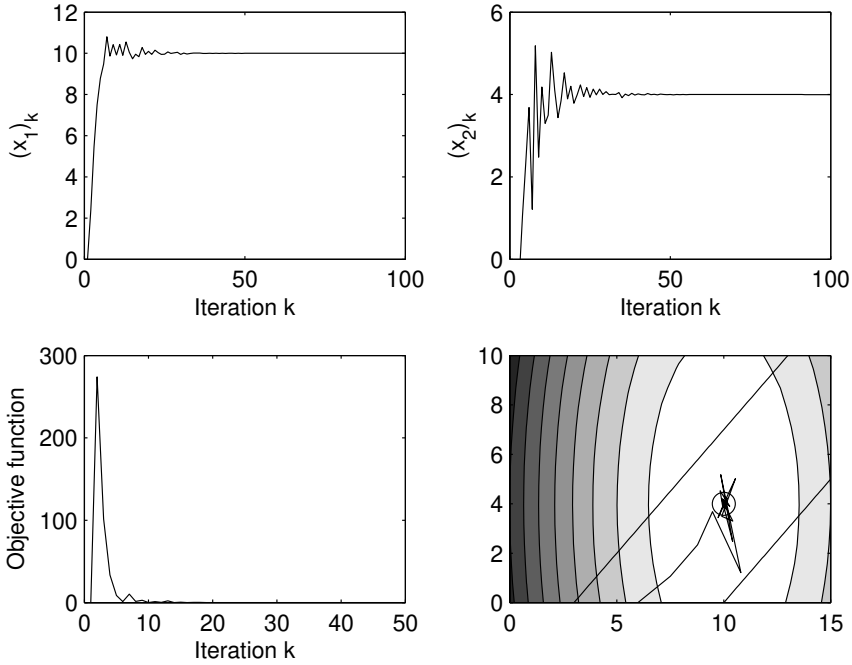

Figure 10: (a) x_1 as function of the iterations, (b) x_2 as function of the iterations, (c) objective function, (d) Feasible region

```
function f= fun6_interior_barrier(x)
if(x(1)^2+x(2)^2-1<0)
f=-log(-(x(1)^2+x(2)^2-1));
else
f=100000000000000000;
end
```

Finally, for a given t, and given initial condition $x0$, we use fminunc:

```
x=fminunc(@(x) t*(-x(1)*x(2))+fun6_interior_barrier(x),x0);
```

In the Figure 11, the variables x_1 and x_2 and the function evaluations as function of the parameter t are shown. The cases are $IC-1$ for the starting point $[0.1, 0.1]$, and $IC-2$ for the starting point $[-0.1, -0.1]$. In the figure it is also shown the feasible region (inside the unit circle). With arrow, the directions of central paths $x^*(t)$. We can see how the algorithm converges to the optimal solution as the parameter t increases.

Exercise 6.3. *Are the following functions convex or not? Why?*

1. $f : \mathbb{R} \to \mathbb{R} : x \mapsto (x^2 + 1)^2$
2. $f : \mathbb{R} \to \mathbb{R} : x \mapsto (x^2 - 3x)^2$
3. $f : \mathbb{R} \to \mathbb{R} : x \mapsto 2^x$
4. $f : \mathbb{R} \to \mathbb{R} : x \mapsto \left(\frac{1}{2}\right)^x$

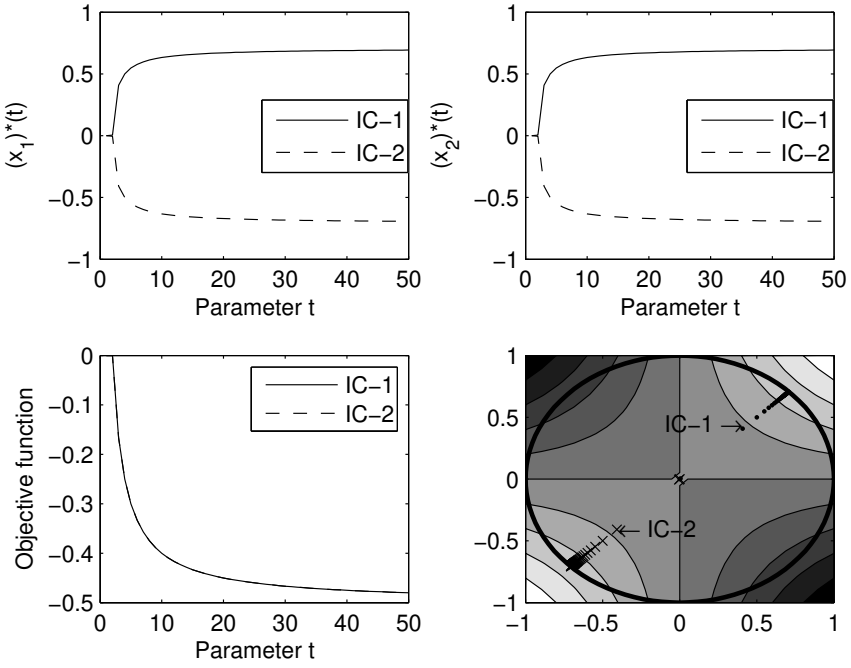

Figure 11: (a) x_1 as function of the parameter t, (b) x_2 as function of the parameter t, (c) objective function, (d) Feasible region and algorithmic steps

5. $f : \mathbb{R} \setminus \{0\} \to \mathbb{R} : x \mapsto \dfrac{1}{x}$
6. $f : [1, +\infty) \to \mathbb{R} : x \mapsto \dfrac{1}{x}$
7. $f : \mathbb{R}^2 \to \mathbb{R} : (x, y) \mapsto \cosh(x^2 + y^2)$

Solution: The definition of convex functions states that f is convex if its domain $\mathrm{dom}(f)$ is convex and if we have $f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$ for all $x, y \in \mathrm{dom}(f)$ and for all $\lambda \in [0, 1]$.

1. $\mathrm{dom}(f) = \mathbb{R}$ is a convex set. According to the Lecture Notes, αx^{2n} with $\alpha \in \mathbb{R}^+$ and $n \in \mathbb{N}$ is convex. Hence, the function $f = (x^2 + 1)^2 = x^4 + 2x^2 + 1$ is also convex.
2. Following from the previous case, the function $f = (x^2 - 3x)^2 = x^4 + 9x^2 - 6x^3$ is not convex (since the term $-6x^3$ is not convex).
3. The function can be rewritten as $f = 2^x = \exp^{(\ln(2).x)}$. The function $\exp g(x)$ is convex if g is convex. Hence, $f = 2^x$ is also convex since $\ln(2).x$ is convex (in fact $\exp^{ax}$ is convex for any $a \in \mathbb{R}$).
4. The function can be rewritten as $f = \left(\dfrac{1}{2}\right)^x = \exp^{(-\ln(2).x)}$ and it is convex since $\exp^{ax}$ is convex.
5. f is not convex, since the set $\mathbb{R} \setminus \{0\}$ is not convex.
6. The domain is a convex set and the function $f = \dfrac{1}{x}$ defined on this domain is convex.

7. $\cosh x$ is a convex function and non-decreasing. On the other hand, $x^2 + y^2$ is convex in $\mathbb{R}^2$. According to the Lecture notes, $f(x) = h(g(x))$ is convex if g is convex and h is convex and non-decreasing. Hence, $\cosh(x^2 + y^2)$ is convex.

Exercise 6.4. *On page 60 it is stated that if P is symmetric then the conditions $P > 0$ and $A^T P + PA < 0$ can be recast as an LMI. Prove this statement.*
Hint: Write P as a linear combination of symmetric basis matrices, each having only one (diagonal) entry or two (off-diagonal) entries equal to 1, the other entries being equal to 0.

Solution: We aim at showing that $A^T P + PA < 0$ is an LMI with p as the variable. To see this, we select a basis for symmetric $n \times n$ matrices. For $i \geq j$ define E^{ij} as the matrix with its (i, j) and (j, i) elements equal to one, and all of its other elements equal to zero. There are $m = \frac{n(n+1)}{2}$ linearly independent matrices E^{ij} and any symmetric matrix P can be written uniquely as:

$$P = \sum_{j=1}^{n} \sum_{i \geq j}^{n} P_{ij} E^{ij},$$

where P_{ij} is the (i, j) element of P. Thus, the matrices E^{ij} form a basis for symmetric $n \times n$ matrices. Substituting for P in terms of its basis matrices gives the alternative form for the Lyapunov inequality:

$$A^T P + PA = A^T \left(\sum_{j=1}^{n} \sum_{i \geq j}^{n} P_{ij} E^{ij} \right) + \left(\sum_{j=1}^{n} \sum_{i \geq j}^{n} P_{ij} E^{ij} \right) A = \sum_{j=1}^{n} \sum_{i \geq j}^{n} P_{ij} (A^T E^{ij} + E^{ij} A) < 0$$

which is in the form of an LMI with $F_0 = 0$ and $F_k = -A^T E^{ij} - E^{ij} A$, for $k = 1, ..., m$. The elements of the vector x in the $F(x)$ are the P_{ij}, $i \geq j$, stacked up on top of each other.

Exercise 6.5. *If the function f is convex, is f^2 then always convex?*
If the function f is convex and nonnegative, is f^2 then always convex?

Solution:

- No; for instance $|x| - 1$ is convex, but $(|x| - 1)^2$ is not convex as $f^2(1) = f^2(-1) = 0$, while $f^2(0) = 1$.
- Yes; and it can be proved using the definition of convex functions. (Hint: since we have assumed that f is nonnegative we can conclude that $f^2(x) \leq f^2(y)$ if $f(x) \leq f(y)$)

Exercise 6.6. *1. Prove that the sum of a linear function and a convex function is convex.*

2. Prove that the sum of a linear function and a nonconvex function is nonconvex.

3. Provide examples to show that the sum of a convex function and a nonconvex one, can be either convex or nonconvex.

4. Provide examples to show that the sum of two nonconvex functions can be either convex or nonconvex.

Solution:

1. we know from the properties of linear functions that $f(ax) = af(x)$ and $f(x+y) = f(x)+f(y)$. Hence, for the sum of a convex function f_1 and a linear function f_2 we have;

$$\begin{aligned} f_1(\lambda x + (1-\lambda)y) + f_2(\lambda x + (1-\lambda)y) &= f_1(\lambda x + (1-\lambda)y) + \lambda f_2(x) + (1-\lambda) f_2(y) \\ &\leq \lambda f_1(x) + (1-\lambda) f_1(y) + \lambda f_2(x) + (1-\lambda) f_2(y) \\ &= \lambda(f_1(x) + f_2(x)) + (1-\lambda)(f_1(y) + f_2(y)) \end{aligned}$$

   Thus the sum of the two functions is a convex function.

2. Similar to the approach in part (1), we can conclude that the sum of a nonconvex function and a linear function is nonconvex.

3. The functions $f_1(x) = ax^2$ with $a \geq 0$ and $f_2(x) = bx^2$ with $b < 0$ are convex and nonconvex respectively. However, the sum $f_1 + f_2$ is a convex function in case $a \geq |b|$ and nonconvex in case $a < |b|$.

4. Consider the functions $f_1(x) = x^3$ and $f_2(x) = -x^3$ with $\mathbb{R}$ as their domain. The sum of the two functions is zero and convex, while the subtraction results in a nonconvex functions.

7 Exercises for Chapter 7: Global optimization

Exercise 7.1. *Using the routine simulannealbnd of Matlab, minimize the following function,*

$$f(x) = -e^{-2\ln(2)(\frac{x-0.008}{0.854})} \sin^6(5\pi(x^{0.75} - 0.05)), \quad x \in [0,1].$$

Plot the current iteration point, the function value, and the temperature function.

Solution: First, we write an m-file called `fun7_simulanneal.m` for this function

```
function f=fun7_simulanneal(x)
f=-exp(-2*log(2).*(x-0.008)/0.854).*(sin(5*pi*(x.^(0.75) - 0.05))).^6;
```

Then, for example, we can ask in the options of simulannealbnd to display the objective function evaluated (@*saplotf*) and the value of the temperature @*saplottemperature* (see the help of simulannealbnd for more options)

```
options = saoptimset('PlotFcns',{@saplottemperature,@saplotf});
```

and finally we include the low and upper boundaries *lb*, *lu*, together with an initial guess $x0$,

```
lb=0;
lu=1;
x0=1;
x=simulannealbnd(@fun7_simulanneal,x0,lb,lu,options)
```

the solution $x = 0.0791$.

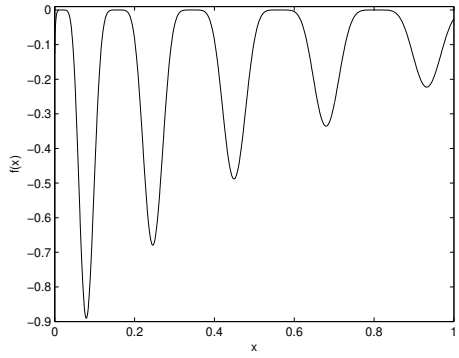

Figure 12: Function $f(x)$

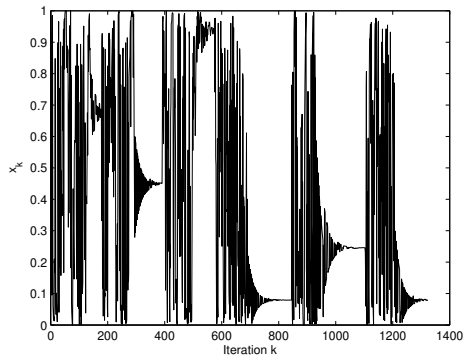

Figure 13: Iteration points x_k

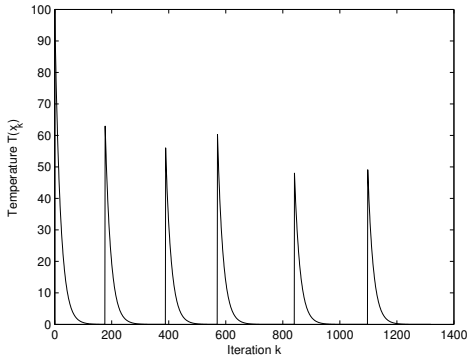

Figure 14: Temperature T

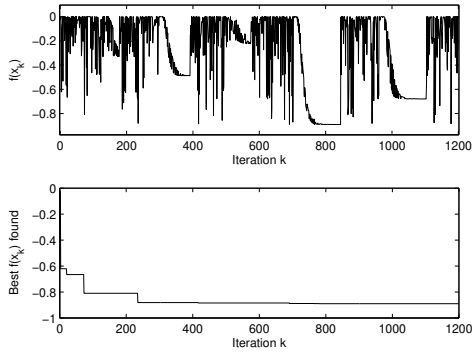

Figure 15: Objective function and the best objective function value reached

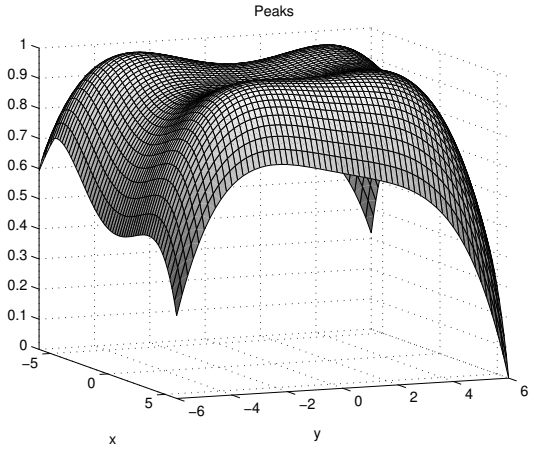

Figure 16: The Himmelblau function

Exercise 7.2. *The Himmelblau function has four peaks in the points* $(3;2)$*,* $(-3.799;-3.283)$*,* $(-2.805;3.131)$*, and* $(3.584;-1.848)$*, and it is defined by*

$$f(x_1,x_2)=\frac{2186-(x_1^2+x_2-11)^2-(x_1+x_2^2-7)^2}{2186},\quad x_1,x_2\in[-6,6].$$

Using the routine ga of Matlab, generate an optimizer capable to detect the four optimal solutions.

Solution: In the Figure 16 we can see the function.

To use the routine ga, first, we write an m-file called `fun7_Himmelblau.m` for this function (remember that ga will minimize).

```
function f=fun7_Himmelblau(x)
f=-(2186-(x(1)^2+x(2)-11)^2-(x(1)+x(2)^2-7)^2)/2186;
```

Then we run ga, including the number of optimization variables NV, the lower bound LB, and the upper bound UB:

```
NV=2;
LB=[-6 -6];
UB=[6 6];
X = ga(@fun7_Himmelblau,NV,[],[],[],[],LB,UB,[])
```

If we run several times the algorithm (500 times), we will note that the solution $(3.000, 2.000)$ is the most typical (485/500 times).
In the tests, the solutions $(-2.8051, 3.1313)$ and $(3.5844, -1.8480)$, appeared 10, and 5 times. Then, it is necessary to introduce a modification, in order to find all the peaks. In the literature "Nitching algorithms" are proposed, to keep the diversity in the solutions. We can also change some parameters of the ga like the mutation or the crossover probabilities (to increase diversity). Another option is to split the regions randomly, so the *ga* will find the peaks in different specific regions. For example, the following code will find the four peaks (in the Figure 17 the results are displayed):

```
[X,ef] = ga(@example72,2,[],[],[],[],[0 0],[6 6],[]);
[X,ef] = ga(@example72,2,[],[],[],[],[-6 -6],[0 0],[]);
[X,ef] = ga(@example72,2,[],[],[],[],[-6 0],[0 6],[]);
[X,ef] = ga(@example72,2,[],[],[],[],[0 -6],[6 0],[]);
```

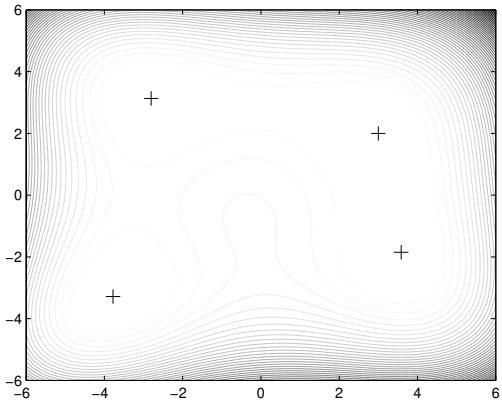

Figure 17: The Himmelblau function

Exercise 7.3.

Discuss the main differences between multi-start local optimization methods, simulated annealing, and genetic algorithms.

Solution: In multi-start local optimization, we select several starting points in the feasible set (using, e.g., a uniform distribution) and for each starting point we run a local minimization method. From the set of returned solutions we afterwards select the one that yields the lowest value for the objective function. It is not always very efficient in general. However, if we already have a good idea of the region where the global optimum will be situated, we can select our initial points in this region and then the process will be much more efficient.

In contrast to local optimization techniques, simulated annealing can escape from local minima. It uses probabilistic transition rules. Only the function values of the objective function are required; the gradient and the Hessian are not used. And one important feature of this algorithm is that it can be used for problems with discrete parameters.

The genetic algorithms search from a population of points: instead of considering one point at the time as is done in random search, multi-start local optimization or simulation annealing, genetic algorithms consider sets of possible solutions in each iteration step. They can escape from local minima since a whole population of possible solutions is considered. And they use probabilistic transition rules.

8 Exercises for Chapter 11: Integer optimization

Exercise 8.1. *Consider the process modeled by the following linear discrete-time system:* $y(n+1) = ay(n) + bu(n) + e(n)$*, where* $y(n)$ *is the output,* $u(n) \in \{0, 1\}$ *the input (binary input),* $a = 0.9$ *and* $b = 0.1$ *are the model parameters, and* $e(n)$ *is white noise of mean value* 0 *and standard deviation* σ*. At instant time* n *the output* $y(n) = 0.5$ *is measured and we have to obtain a control action* $u(n) \in \{0, 1\}$*. Let us define the prediction* $\hat{y}(n+1) = ay(n) + bu(n)$*, and* $\hat{y}(n+k) = a\hat{y}(n+k-1) + bu(n+k-1)$ *for* $k \in \{2, 3, 4, 5\}$*.*

- *Obtain the control action* $u(n) \in \{0, 1\}$ *that minimizes* $J = (\hat{y}(n+1) - r)^2 + \lambda u(n)^2$*, where* $\lambda = 0.01$ *is a weighting factor and* $r = 1$ *the output reference.*

- *Using branch-and-bound, obtain the control sequence* $U = [u(n), u(n+1), u(n+2)]$*, that minimize* $\min J_n^{n+2} = \sum_{k=1}^{3}(\hat{y}(n+k) - r)^2 + \lambda \sum_{k=1}^{3} u(n+k-1)^2$.

Solution:

- We have 2 options:

  - If $u(n) = 0$, then $J = (ay(n) - r)^2$.
  - If $u(n) = 1$, then $J = (ay(n) + b - r)^2 + \lambda$.

  Among those two options, we select the one that provides the minimum J. If $a = 0.9$, $b = 0.1$, $r = 1$, $\lambda = 0.01$, and $y(n) = 0.5$ then $J = 0.3025$ for $u(n) = 0$, and $J = 0.2125$ for $u(n) = 1$, so the optimum is $u(n) = 1$.

- We can eliminate the equality constraints and to minimize the objective function: $J_n^{n+2} = (ay(n) + bu(n) - r)^2 + (a^2y(n) + abu(n) + bu(n+1) - r)^2 + (a^3y(n) + a^2bu(n) + abu(n+1) + bu(n+2) - r)^2 + \lambda(u(n)^2 + u(n+1)^2 + u(n+2)^2)$

  First choose U as real variable. This results in an unconstrained nonlinear optimization problem that can be solved by the *fminunc* function in MATLAB and the obtained optimum is therefore $U^* = [3.5568, 1.7930, 0.9301]^T$, with $J = 0.2229$. Now we split the problem into two subproblems: In the first we introduce the constraint $u(n) = 1$, and in the second $u(n) = 0$. In both problems, $u(n+1)$ and $u(n+2)$ are still chosen as real variables. This leads to two subproblems SP1 and SP2:

  - SP1 $u(n) = 1$, results in $u(n+1) = 3.1374$, $u(n+2) = 1.3607$, $J = 0.3845$.
  - SP2 $u(n) = 0$, results in $u(n+1) = 3.6632$, $u(n+2) = 1.5292$, $J = 0.5357$.

  In the same way we introduce the constraints for the second variable $u(n+1) = 1$ and $u(n+1) = 0$ for both subproblems SP1 and SP2. This leads to:

  - SP11 $u(n) = 1$, $u(n+1) = 1$, results in $u(n+2) = 2.3225$, $J = 0.4944$.
  - SP12 $u(n) = 1$, $u(n+1) = 0$, results in $u(n+2) = 2.7725$, $J = 0.6213$.
  - SP21 $u(n) = 0$, $u(n+1) = 1$, results in $u(n+2) = 2.7275$, $J = 0.7063$.
  - SP22 $u(n) = 0$, $u(n+1) = 0$, results in $u(n+2) = 3.1775$, $J = 0.8585$.

  Now we introduce the constraints for the third variable $u(n+2) = 1$ and $u(n+2) = 0$. In the subproblem SP11:

  - SP111 $u(n) = 1$, $u(n+1) = 1$, $u(n+2) = 1$, results in $J = 0.5294$.
  - SP112 $u(n) = 1$, $u(n+1) = 1$, $u(n+2) = 0$, results in $J = 0.6023$.

  In this point we can conclude that the optimal solution is given by $u(n) = 1$, $u(n+1) = 1$, $u(n+2) = 1$, because the value of the objective function in SP111 is lower than the others SP12, SP21, SP22 (when including the constraints in $u(n+2)$, the objective function will increase in those problems). Next the results just to verify:

  - SP121 $u(n) = 1$, $u(n+1) = 0$, $u(n+2) = 1$, results in $J = 0.6841$.
  - SP122 $u(n) = 1$, $u(n+1) = 0$, $u(n+2) = 0$, results in $J = 0.7750$.

- SP211 $u(n) = 0$, $u(n+1) = 1$, $u(n+2) = 1$, results in $J = 0.7660$.
- SP212 $u(n) = 0$, $u(n+1) = 1$, $u(n+2) = 0$, results in $J = 0.8551$.
- SP221 $u(n) = 0$, $u(n+1) = 0$, $u(n+2) = 1$, results in $J = 0.9533$.
- SP222 $u(n) = 0$, $u(n+1) = 0$, $u(n+2) = 0$, results in $J = 1.0604$.