NN_skomam4

January 25, 2023

Neuronové sítě

… jako složené funkce s parametry

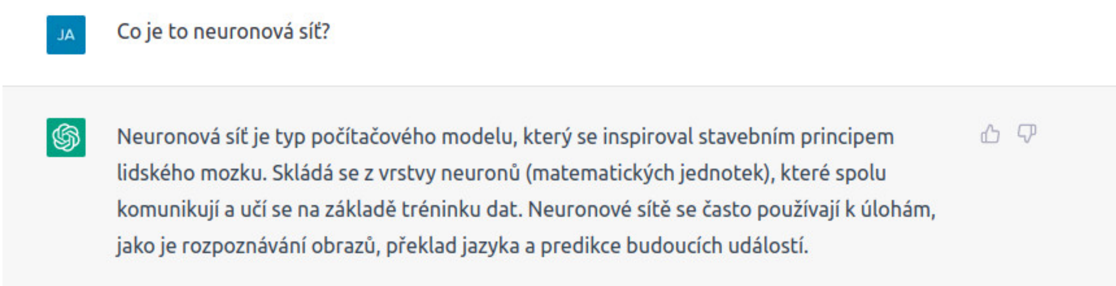
Jan Kracík

ŠKOMAM 2023

---

1 Úvod

(ChatGPT 19.1.2023)



Aha, tak nic …

---

[1]:
```python
import numpy as np
from matplotlib import pyplot as plt
from matplotlib import cm

import ipywidgets as widgets
from ipywidgets import interact

#from mpl_toolkits.mplot3d import Axes3D
```

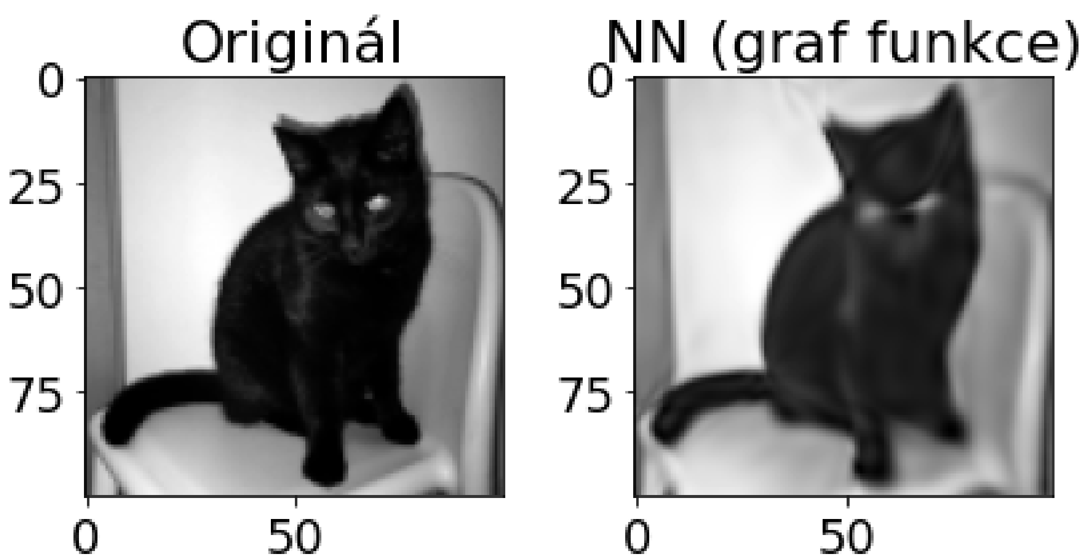
```
%matplotlib inline
```

[2]:
```
import matplotlib

font = {'size'   : 18}
matplotlib.rc('font', **font)

FS = (6, 6)
```

[3]:
```
jerryMLP = np.load('jerryMLP.npy')
jerryIMG = np.load('jerryIMG.npy')
plt.subplot(121)
plt.imshow(jerryIMG, cmap=cm.gray)
plt.title("Originál")
plt.subplot(122)
plt.imshow(jerryMLP, cmap=cm.gray)
plt.title("NN (graf funkce)")
plt.tight_layout();
```



[ ]:
```

```

---

2 Funkce jedné proměnné

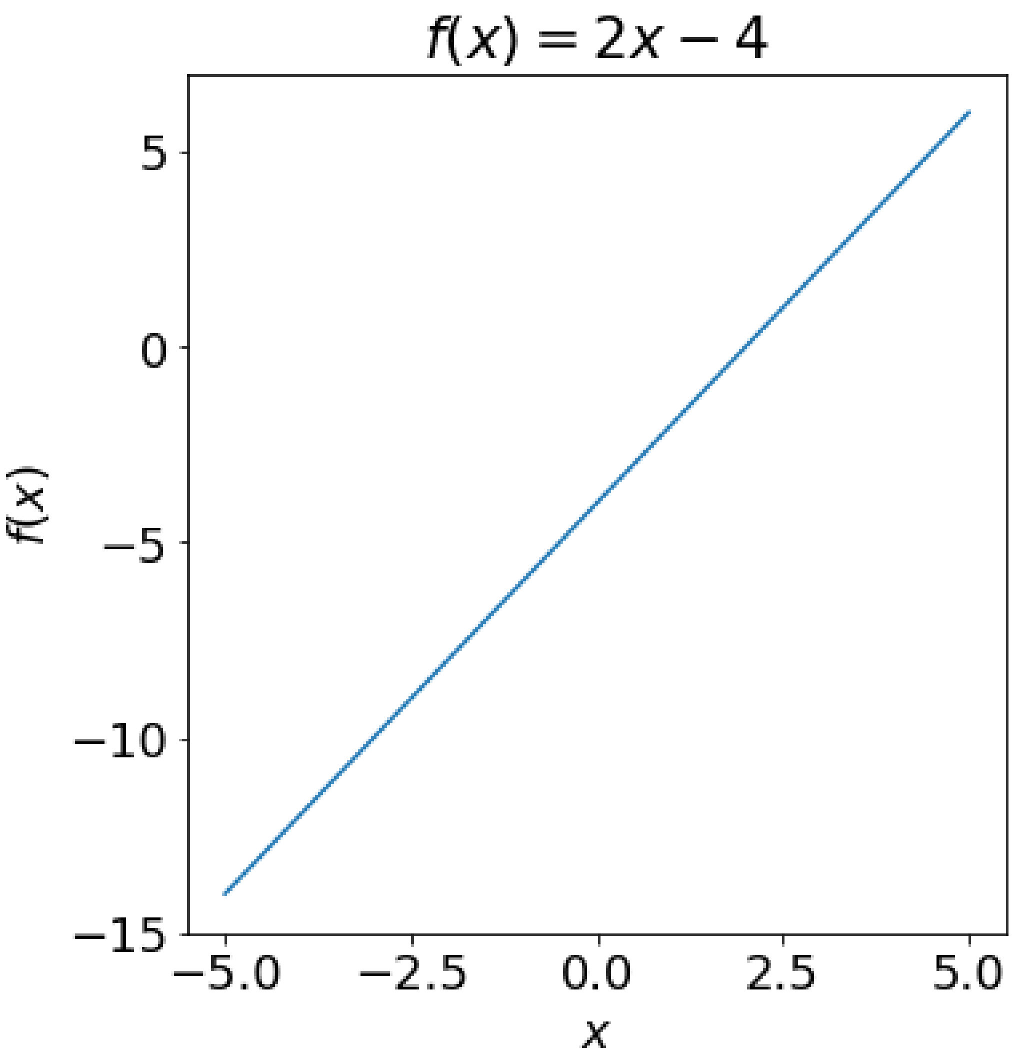
Z učebnice Matematika pro gymnázia:

Funkce ($f : A \to \mathbb{R}$) na množině $A \subset \mathbb{R}$ je předpis, který každému číslu z množiny A přiřazuje právě jedno reálné číslo. Množina A se nazývá definiční obor funkce.

Příklad (lineární funkce) $f : \mathbb{R} \to \mathbb{R}$

$$f(x) := 2x - 4$$

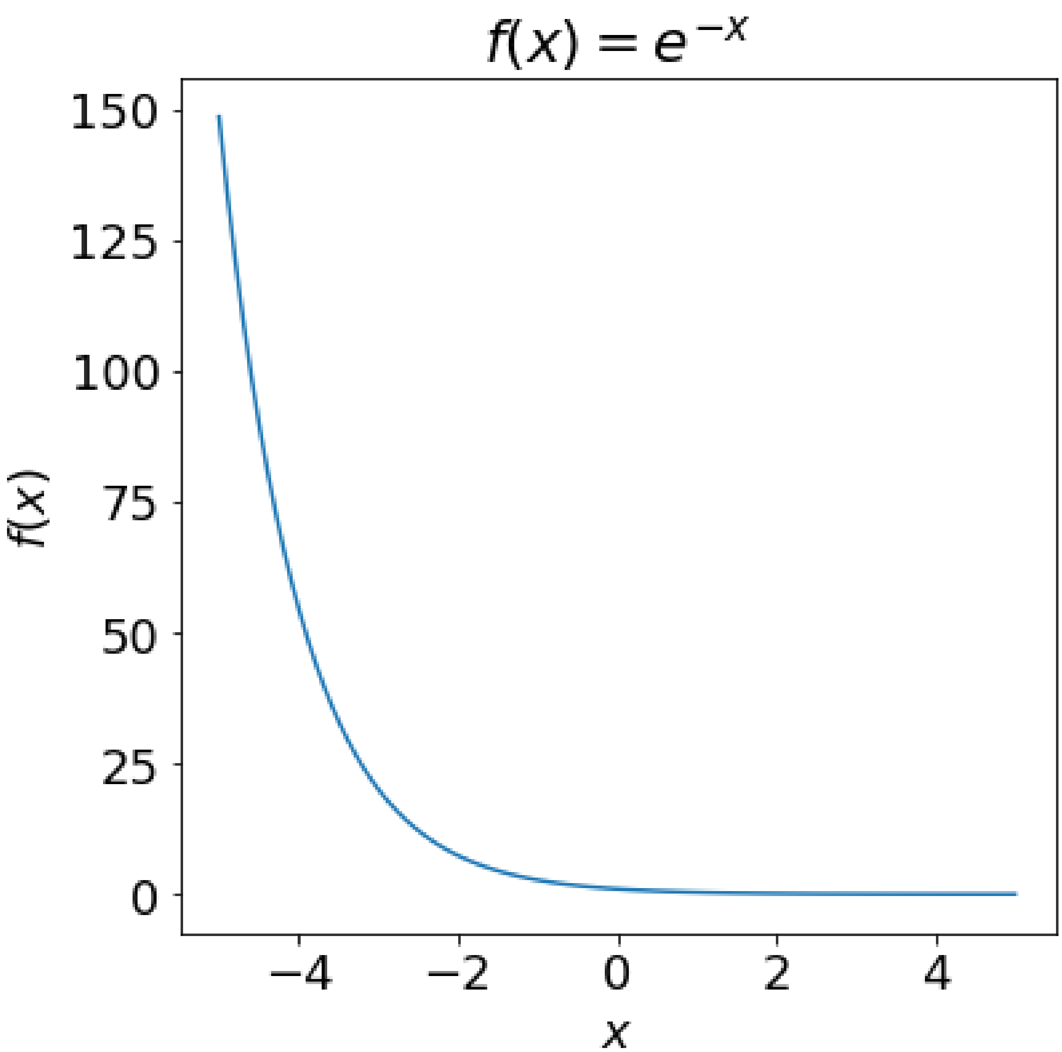
[4]:
```
xs = np.linspace(-5, 5, 100)
plt.figure(figsize=FS)
plt.plot(xs, 2*xs-4)
plt.xlabel("$x$")
plt.ylabel("$f(x)$")
plt.title("$f(x)=2x-4$")
plt.tight_layout();
```



Příklad (exponenciální funkce) $f : \mathbb{R} \to \mathbb{R}$

$$f(x) := e^x$$

[5]:
```
xs = np.linspace(-5, 5, 100)
plt.figure(figsize=FS)
plt.plot(xs, np.exp(-xs))
plt.xlabel("$x$")
plt.ylabel("$f(x)$")
plt.title("$f(x)=e^{-x}$");
plt.tight_layout();
```

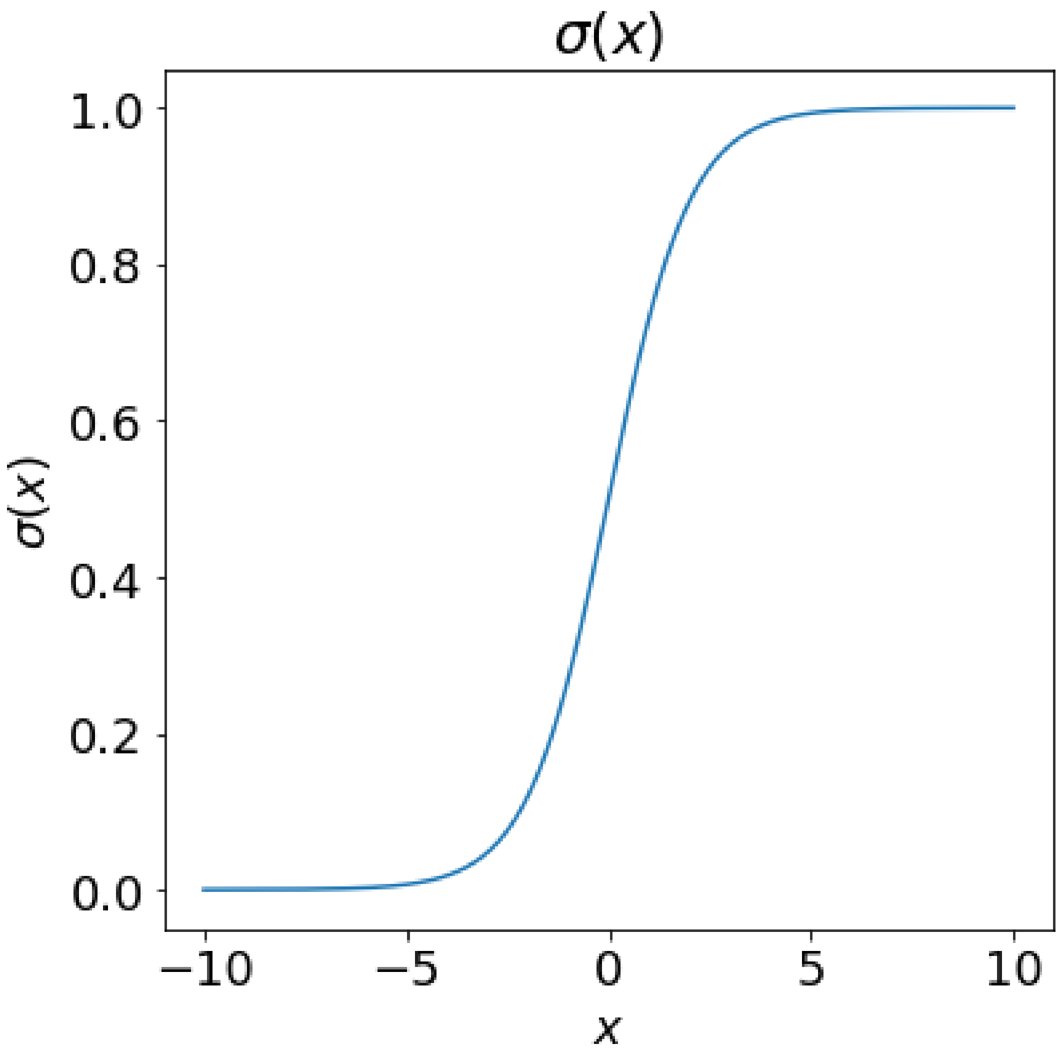


Příklad (logistická funkce) $\sigma : \mathbb{R} \to \mathbb{R}$

$$\sigma(x) := \frac{1}{1+e^{-x}}$$

[6]:
```
def sigma(x):
    return 1/(1+np.exp(-x))
```

[7]:
```
xs = np.linspace(-10, 10, 400)
plt.figure(figsize=FS)
plt.plot(xs, sigma(xs))
plt.xlabel("$x$")
plt.ylabel("$\sigma(x)$")
plt.title("$\sigma(x)$");
plt.tight_layout();
```



[8]:
```
sigma(-10), sigma(0), sigma(10)
```

[8]: (4.5397868702434395e-05, 0.5, 0.9999546021312976)

Příklad (program jako funkce)

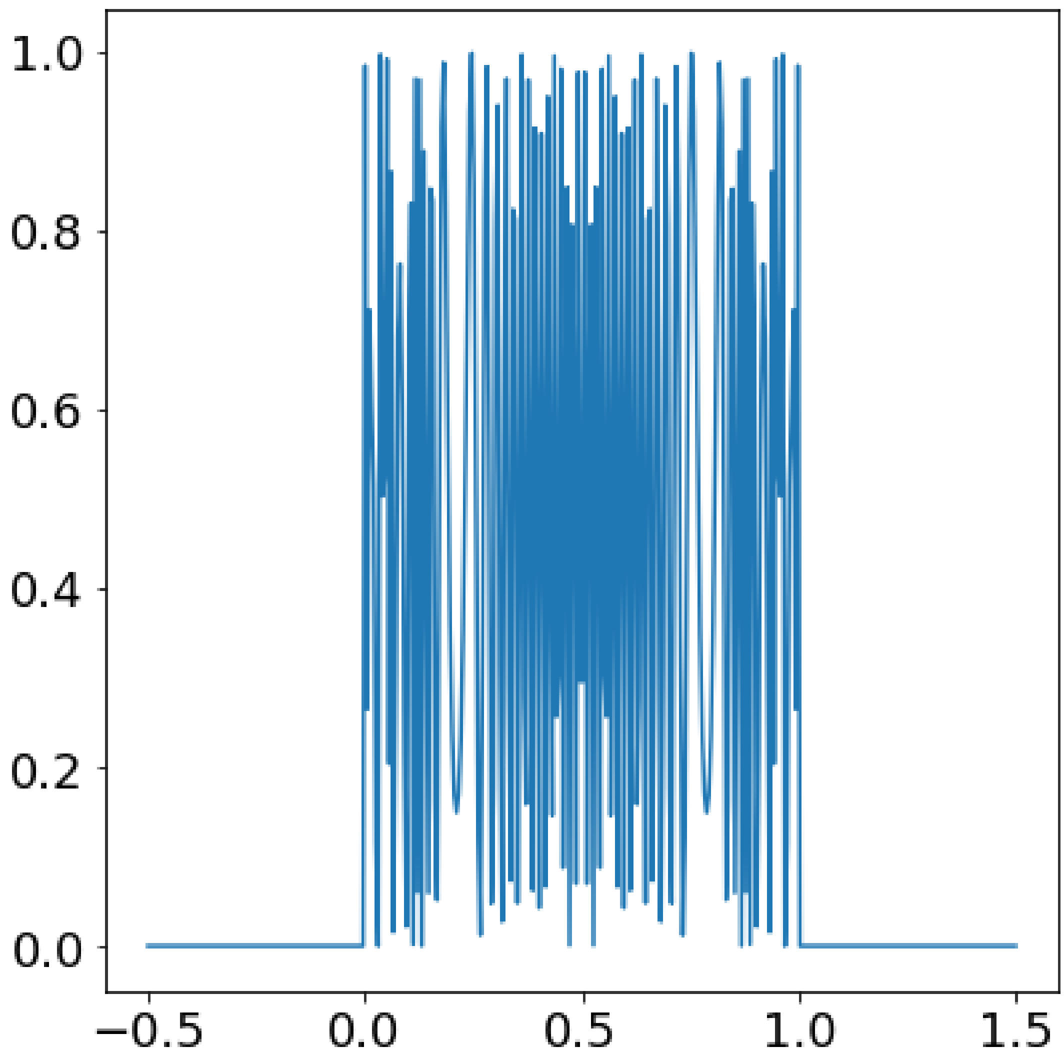
[9]:
```
def nejaka_funkce(x):
    if x<0:
        vysledek = 0

    if x>=0 and x<=1:
        v = x
        for n in range(10):
            v = 4*v*(1-v)
        vysledek = v

    if x>1:
        vysledek = 0

    return vysledek
```

[10]:
```
xs = np.linspace(-0.5, 1.5, 400)
plt.figure(figsize=FS)
ys = np.array([nejaka_funkce(x) for x in xs])
plt.plot(xs, ys)
plt.tight_layout();
```



[ ]:

3 Kartézský součin

Nechť M, N jsou libovolné množiny. Kartézský součin $M \times N$ je množina všech uspořádaných dvojic (m, n), kde $m \in M$ a $n \in N$. Tj.

$$M \times N := \{(m, n) | m \in M, n \in N\}.$$

Příklad

$$M = \{a, b, c\}$$

$$N = \{K, L\}$$

$$M \times N = \{(a,K),(a,L),(b,K),(b,L),(c,K),(c,L)\}$$

Příklad

$$\mathbb{R}^2 := \mathbb{R} \times \mathbb{R} = \{(x_1, x_2) | x_1 \in \mathbb{R}, x_2 \in \mathbb{R}\}$$

Obdobně pro libovolné $n \in \mathbb{N}$

$$\mathbb{R}^n := \mathbb{R} \times \mathbb{R} \times \cdots \times \mathbb{R} = \{(x_1, x_2, \ldots, x_n) | x_1 \in \mathbb{R}, x_2 \in \mathbb{R}, \ldots, x_n \in \mathbb{R})\}$$

[ ]:

```

```

4 Funkce více proměnných

Funkce ($f : M \to N$) na množině M s hodnotami v množině N je předpis, který každému prvku z množiny M přiřazuje právě jeden prvek z množiny N.

Příklad (lineární funkce)

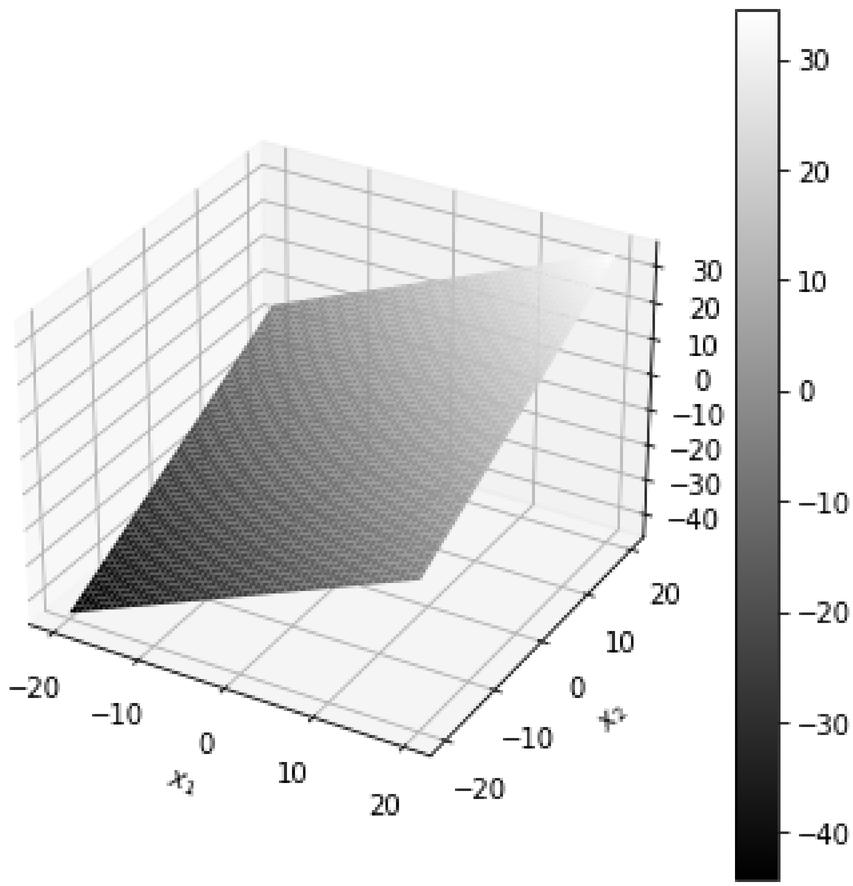
$$f : \mathbb{R}^2 \to \mathbb{R},$$

$$f(x_1, x_2) := x_1 + x_2$$

[11]:

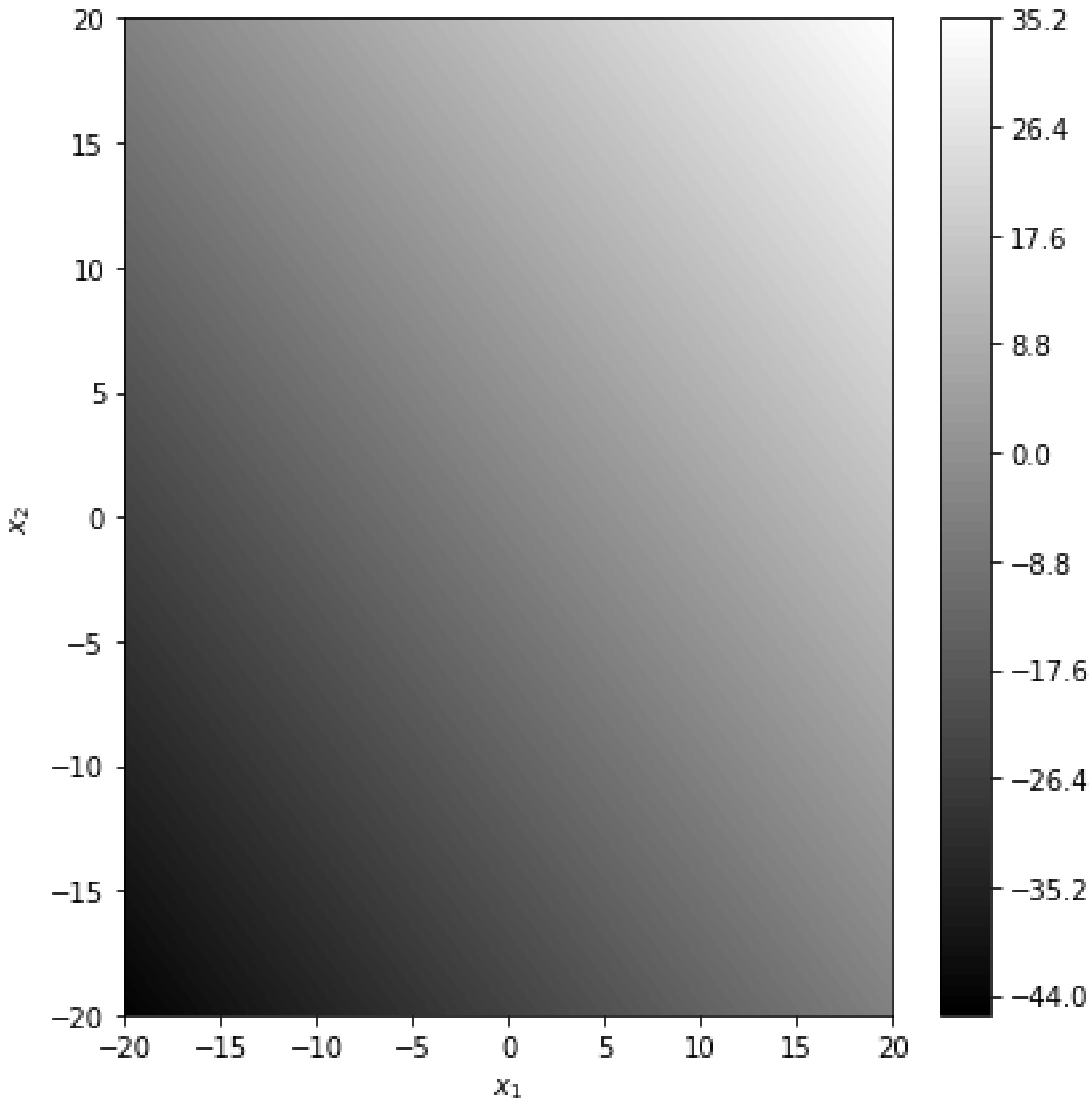
```
# graf jako projekce
%matplotlib inline
x1s = np.linspace(-20, 20, 100)
x2s = np.linspace(-20, 20, 100)

x1g, x2g = np.meshgrid(x1s, x2s)
yg = x1g+x2g-5
fig, ax = plt.subplots(figsize=FS, subplot_kw={"projection": "3d"})
surf = ax.plot_surface(x1g, x2g, yg, cmap=cm.gray)
fig.colorbar(surf)
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$f(x_1, x_2)$');
```



[12]:
```
# graf jako barevna mapa
%matplotlib inline
plt.figure(figsize=FS)
cp = plt.contourf(x1g, x2g, yg, cmap=cm.gray, levels=100)
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.tight_layout()
plt.colorbar(cp)
```

[12]: <matplotlib.colorbar.Colorbar at 0x7fe89bbe34f0>



[ ]:

Příklad (lineární funkce)

$$f : \mathbb{R}^3 \to \mathbb{R}^2,$$

$$f(x_1, x_2, x_3) := (x_1 + x_2, x_2 - x_3)$$

[ ]:

5 Funkce s parametry

Příklad (lineární funkce s parametry $a, b \in \mathbb{R}$)

$$f : \mathbb{R} \to \mathbb{R},$$

$$f(x) := ax + b$$

Funkci f lze chápat jako

$$F : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R},$$

$$F((a, b), x) := ax + b$$

Funkci s parametrem z množiny P definovanou na množině M s hodnotami v množině N lze chápat jako

$$F : P \times M \to N.$$

Pro zvolené $p \in P$ definujme

$$F_p : M \to N,$$

$$F_p(x) := F(p, x).$$

Příklad (lineární funkce s parametry $a, b \in \mathbb{R}$)

$$F : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R},$$

$$F((a, b), x) := ax + b$$

[13]:
```
%matplotlib inline
#%matplotlib qt
def F(a, b, x):
    return a*x+b

fig = plt.figure(figsize=FS)
ax = fig.add_subplot(111)
xs = np.linspace(-10, 10, 200)
c, = ax.plot(xs, F(1, 1, xs))
```

```
plt.xlabel('$x$')
plt.ylabel('$F_{(a, b)}(x)$')

@interact(a=(-2, 2, 0.01), b=(-5, 5, 0.01))
def plot_F(a=1, b=0):
    ys = F(a, b, xs)
    c.set_ydata(ys)
    plt.pause(0.1)
    plt.draw()

plot_F()
```

```
interactive(children=(FloatSlider(value=1.0, description='a', max=2.0, min=-2.0,␣
↪step=0.01), FloatSlider(value…
```

```
<Figure size 432x288 with 0 Axes>
```

```
<Figure size 432x288 with 0 Axes>
```

[ ]:

```

```

6 Skládání funkcí

Uvažujme funkce

$$f : M \to N,$$

$$g : N \to Q.$$

Složenou funkcí $g \circ f$ rozumíme funkci

$$g \circ f : M \to Q,$$

$$g \circ f(x) := g(f(x)).$$

Příklad (složená funkce s parametry)

$$f_{(a,b)} : \mathbb{R} \to \mathbb{R},$$

$$f_{(a,b)}(x) := ax + b,$$

$$(a, b) \in \mathbb{R}^2.$$

$$\sigma : \mathbb{R} \to \mathbb{R},$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$F_{(a,b)} := \sigma \circ f_{(a,b)} : \mathbb{R} \to \mathbb{R},$$

$$F_{(a,b)}(x) = \frac{1}{1 + e^{-(ax+b)}}$$

[14]:

```
%matplotlib inline
#%matplotlib qt

def F(a, b, x):
    return a*x+b

fig = plt.figure(figsize=FS)
ax = fig.add_subplot(111)
xs = np.linspace(-20, 20, 200)
#d, = ax.plot(xs, (F(1, 1, xs)))
#ax.twinx()
c, = ax.plot(xs, sigma(F(1, 1, xs)))

plt.xlabel('$x$')
plt.ylabel('$F_{(a, b)}(x)$')


@interact(a=(-10, 10, 0.01), b=(-100, 100, 0.01))
def plot_F(a=1, b=0):
    ds = F(a, b, xs)
    cs = sigma(ds)
    c.set_ydata(cs)
    #d.set_ydata(ds)
    plt.pause(0.1)
    plt.draw()

plot_F()
```

```
interactive(children=(FloatSlider(value=1.0, description='a', max=10.0, min=-10.
↪0, step=0.01), FloatSlider(val…

<Figure size 432x288 with 0 Axes>
```

<Figure size 432x288 with 0 Axes>

[ ]:

```

```

7 Skládání funkcí více proměnných

Uvažujme funkce

$$
\begin{aligned}
f_1 &: M \to Q, \\
f_2 &: M \to Q, \\
&\vdots \\
f_n &: M \to Q,
\end{aligned}
$$

$$g : Q \times Q \times \cdots \times Q \to N.$$

Složenou funkcí $g \circ (f_1, f_2, \ldots, f_n)$ rozumíme funkci

$$g \circ (f_1, f_2, \ldots, f_n) : M \to N,$$

$$g \circ (f_1, f_2, \ldots, f_n)(x) := g(f_1(x), f_2(x), \ldots, f_n(x)).$$

Příklad (složená funkce více proměnných s parametry)

$$
\begin{aligned}
f_{(a_1,b_1)} &: \mathbb{R} \to \mathbb{R}, \\
f_{(a_1,b_1)}(x) &:= \sigma(a_1 x + b_1)
\end{aligned}
$$

$$
\begin{aligned}
g_{(a_2,b_2)} &: \mathbb{R} \to \mathbb{R}, \\
g_{(a_2,b_2)}(x) &:= \sigma(a_2 x + b_2)
\end{aligned}
$$

$$
\begin{aligned}
h &: \mathbb{R}^2 \to \mathbb{R}, \\
h_{(p_1,p_2,q)}(x_1, x_2) &:= \sigma(p_1 x_1 + p_2 x_2 + q)
\end{aligned}
$$

$$
\begin{aligned}
F_{(a_1,b_1,a_2,b_2,p_1,p_2,q)} &:= h_{(p_1,p_2,q)} \circ (f_{(a_1,b_1)}, g_{(a_2,b_2)}) : \mathbb{R} \to \mathbb{R}, \\
F_{(a_1,b_1,a_2,b_2,p_1,p_2,q)}(x) &= \sigma(p_1 \sigma(a_1 x + b_1) + p_2 \sigma(a_2 x + b_2) + q)
\end{aligned}
$$

Funkce $F_{(a_1,b_1,a_2,b_2,p_1,p_2,q)}$ představuje jendoduchou neuronovou síť (perceptron s jednou skrytou vrstvou obsahující 2 neurony).

[15]:
```python
%matplotlib inline
#%matplotlib qt
def F(a1, b1, a2, b2, p1, p2, q,  x):
    s1 = sigma(a1*x+b1)
    s2 = sigma(a2*x+b2)
    return sigma(p1*s1+p2*s2+q)

fig = plt.figure(figsize=FS)
ax = fig.add_subplot(111)
xs = np.linspace(-20, 20, 200)
#d, = ax.plot(xs, (F(1, 1, xs)))
#ax.twinx()
c, = ax.plot(xs, F(1, 0, -1, 0, 1, 1, 0, xs))
plt.ylim(0, 1)
plt.xlabel('$x$')
plt.ylabel('$F_{(\cdots)}(x)$')


@interact(a1=(-10, 10, 0.01), b1=(-10, 10, 0.01),
          a2=(-10, 10, 0.01), b2=(-10, 10, 0.01),
          p1=(-10, 10, 0.01), p2=(-10, 10, 0.01), q=(-10, 10, 0.01))

def plot_F(a1=-8, b1=-6, a2=1.6, b2=-8.3, p1=-1.25, p2=-2.5, q=0.7):
#def plot_F(a1=1, b1=0, a2=-1, b2=0, p1=1, p2=1, q=0):
    cs = F(a1, b1, a2, b2, p1, p2, q, xs)
    c.set_ydata(cs)
    #d.set_ydata(ds)
    plt.pause(0.1)
    plt.draw()

plot_F()
```

```
interactive(children=(FloatSlider(value=-8.0, description='a1', max=10.0,␣
↪min=-10.0, step=0.01), FloatSlider(v…
```

```
<Figure size 432x288 with 0 Axes>
```

```
<Figure size 432x288 with 0 Axes>
```

8 Umělé neuronové sítě

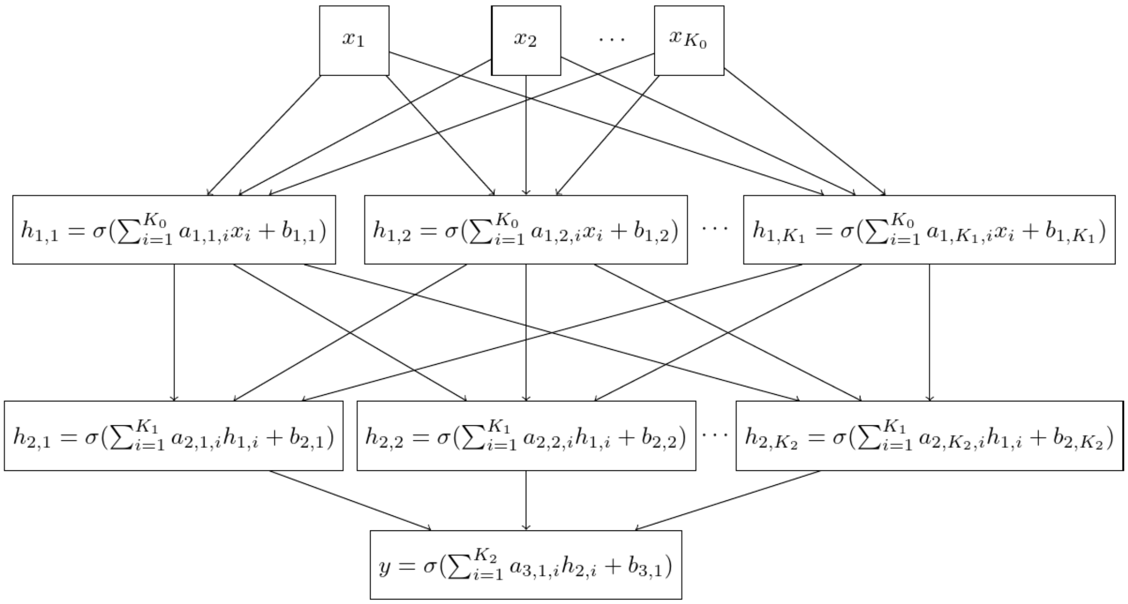
Umělé neuronové síte lze chápat jako parametrické funkce

$$F : P \times M \to N,$$

$$P \cdots \text{množina parametrů},$$

$$M \cdots \text{množina vstupních hodnot},$$

$$N \cdots \text{množina výstupních hodnot}.$$



[ ]:

8.1 Učení neuronové sítě

Učení neuronové sítě spočívá v nalezení vhodného parametru $\hat{p} \in P$ tak aby $F_{\hat{p}}$ byla “ve shodě” s trénovacími daty.

- Trénovací data: zadané dvojice

$$(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n) \in M \times N$$

- Ztrátová funkce (udává kvalitu aproximace dat):

$$L : P \times (M \times N)^n \to \mathbb{R}$$

  Např. kvadratická ztrátová funkce:

$$L(p, (X_1, Y_1), \ldots, (X_n, Y_n)) := \sum_{i=1}^{n} (Y_i - F_p(X_i))^2$$

- Hledáme parametr $\hat{p} \in P$ tak, aby pro každé $p \in P$ platilo

$$L(\hat{p}, (X_1, Y_1), \ldots, (X_n, Y_n)) \le L(p, (X_1, Y_1), \ldots, (X_n, Y_n)).$$

- Místo optimální hodnoty $\hat{p} \in P$ se zpravidla spokojíme s vhodnou aproximací

8.2 Použití naučené neuronové sítě

- x ... nová vstupní hodnota
- $y := F_{\hat{p}}(x)$ ... hodnota predikovaná naučenou neuronovou sítí

[ ]:
```

```