CSE 250A HW6

Jiping Lin A15058075

6.2 (a) $P(a,b|c,d) = \dfrac{P(a,b,c,d)}{P(c,d)}$

$$= \frac{P(a)P(b|a)P(c|a,b)P(d|a,b,c)}{P(c,d)}$$

$d \to a | b,c$

$$= \frac{P(a)P(b|a)P(c|a,b)P(d|b,c)}{\sum_{a'}\sum_{b'} P(a=a',b=b',c,d)} \quad \text{"d.sep (1)"}$$

$$= \frac{P(a)P(b|a)P(c|a,b)P(d|b,c)}{\sum_{a'}\sum_{b'} P(a=a')P(b=b'|a=a')P(c|a=a',b=b')P(d|b=b',c)}$$

(b) $P(a|c,d) = \sum_{b'} P(a,b=b'|c,d)$ (part(a))

$P(b|c,d) = \sum_{a'} P(a=a',b|c,d)$ (part(a))

(c) $\mathcal{L} = \sum_t \log \sum_{a'}\sum_{b'} P(A=a',B=b',C=c_t,D=d_t)$

$$= \sum_t \log \sum_{a'}\sum_{b'} P(A=a')P(B=b'|A=a')P(C=c_t|A=a',B=b')P(D=d_t|B=b',C=c_t)$$

↳ d.sep(1) like pt.(a).

(d) $P(A=a) \leftarrow \frac{1}{T}\sum_t P(A=a|C=c_t,D=d_t)$

$$P(B=b|A=a) \leftarrow \frac{\sum_t P(B=b,A=a|C=c_t,D=d_t)}{\sum_t P(A=a|C=c_t,D=d_t)}$$

$$P(C=c|A=a,B=b) \leftarrow \frac{\sum_t I(c,c_t)P(A=a,B=b|C=c_t,D=d_t)}{\sum_t P(A=a,B=b|C=c_t,D=d_t)}$$

$$P(D=d|B=b,C=c) \leftarrow \frac{\sum_t I(c,c_t)I(d,d_t)P(B=b|C=c_t,D=d_t)}{\sum_t I(c,c_t)P(B=b|C=c_t,D=d_t)}$$

6.3 (a). We need to show in second BN,

$P(Y=1|X) = 1-\prod_{i=1}^{n}(1-p_i)^{x_i}$.

Summing over all possible $(z_1,\dots,z_n) = z \in \{0,1\}^n$, we have

$$P(Y=1|X) = \sum_{z\in\{0,1\}^n} P(Y=1,z|X) = \sum_z P(z|X)P(Y=1|z,X) = \sum_z P(z|X)P(Y=1|z) \quad (*)$$

"Marginalization" "d.sep(1)"

For any z, $P(Y=1|z) = \begin{cases} 1, & \text{if } z_i=1 \text{ for some } i \\ 0, & \text{if } z_i=0\ \forall i \end{cases}$

then $P(Y=1|z)$ only is 0 when all z_i's are 0.

$X=0 \Rightarrow \prod = 1$

$X=1 \Rightarrow \prod = 1-p_i$

All situation / only case

So $(*) = \sum_z P(z|X) - P(\text{All } z_i=0|X) = 1-\prod_{i=1}^{n} P(z_i=0|X) = 1-\prod_{i=1}^{n}(1-p_i)^{x_i}$

(b) $P(Z_i=1, X_i=1 \mid X=x, Y=y)$

$$= I(x_i,1)\,\frac{P(Y=y \mid Z_i=1, X=x)\,P(Z_i=1 \mid X=x)}{P(Y=1 \mid X)}$$

[d.sep(1)]

$$= I(x_i,1)\,\frac{P(Y=y \mid Z_i=1)\,P(Z_i=1 \mid X=x)}{P(Y=1 \mid X)}$$

$$= \frac{I(x_i,1)\,I(y,1)\,P(Z_i=1 \mid X_i=x_i)}{P(Y=1 \mid X)}$$

→ Z_i only relates to X_i

$$\rightarrow = \begin{cases} 0, & \text{if } x_i=0 \\ p_i, & \text{if } x_i=1 \end{cases} \iff p_i I(X_i=1)$$

since $P(Y=1 \mid Z_i=1)=1$ so y must be 1

$$= \frac{I(X_i,1)\,I(y,1)\,(p_i I(X_i,1))}{P(Y=1 \mid X)}$$

$$= \frac{x_i y p_i}{1-\prod_j (1-p_j)^{x_j}}$$

→ part (a)

(c) $$p_i = P(Z_i=1 \mid X_i=1) \leftarrow \frac{\sum_t P(Z_i=1, X_i=1 \mid X=x^{(t)}, Y=y^{(t)})}{\sum_t P(X_i=1 \mid X=x^{(t)}, Y=y^{(t)})}$$

$$= \frac{\sum_t P(Z_i=1, X_i=1 \mid X=x^{(t)}, Y=y^{(t)})}{\sum_t I(x_i^{(t)},1)}$$

→ $= T_i$ by def.

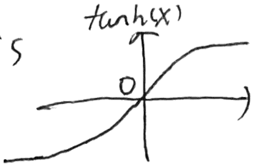
$$= \frac{1}{T_i}\sum_t P(Z_i=1, X_i=1 \mid X=x^{(t)}, Y=y^{(t)}).$$

(d)

iter	M	$\mathcal{L}$
0	175	-0.95809
1	56	-0.49592
2	43	-0.40822
4	42	-0.36461
8	44	-0.34750
16	40	-0.33462
32	37	-0.32258
64	37	-0.31482
128	36	-0.31116
256	36	-0.31016

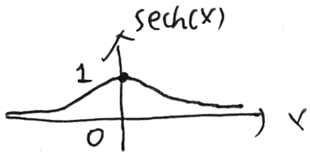
6.4 (a) $f'(x) = \frac{\sinh(x)}{\cosh(x)} = \tanh(x)$

The graph of $\tanh(x)$ is



so $f'(0)=0$, and f' has positive slope at $x=0$, so f has minimum at $x=0$

(b). $f''(x) = (\tanh(x))' = \operatorname{sech}^2(x)$. where the graph of $\operatorname{sech}(x)$ is



This implies $\operatorname{sech}^2(x) \leq 1$ since $0 \leq \operatorname{sech}(x) \leq 1$, $\forall x$.

so $f''(x) \leq 1$ $\forall x$.

(c) See code

(d) $Q(x,x) = f(x) + 0 + 0 = f(x)$.

$$f(x) = f(y) + \int_y^x du \left[f'(y) + \int_y^u dv\, f''(v)\right]$$

$$= f(y) + \int_y^x du\, f'(y) + \int_y^x du \int_y^u dv\, f''(v)$$

$$= f(y) + f'(y)(x-y) + \int_y^x du \int_y^u dv\, f''(v)$$

$$\leq f(y) + f'(y)(x-y) + \int_y^x du \int_y^u dv \qquad \ulcorner f''(v) \leq 1 \text{ by part (b)} \lrcorner$$

$$= f(y) + f'(y)(x-y) + \int_y^x du\, (u-y)$$

$$= f(y) + f'(y)(x-y) + \left.\frac{u^2-y^2}{2}\right|_y^{u=x}$$

$$= f(y) + f'(y)(x-y) + \tfrac{1}{2}(x^2-y^2) = Q(x,y).$$

(e) $Q(x,x_n) = f(x_n) + f'(x_n)(x-x_n) + \frac{1}{2}(x-x_n)^2$

$$\frac{\partial Q(x,x_n)}{\partial x} = 0 \Rightarrow f'(x_n) + (x - x_n) = 0 \qquad \frac{\partial^2 Q(x,x_n)}{\partial x^2} = 1 > 0 \Rightarrow \text{min.}$$

so $x = x_n - f'(x_n) \Rightarrow x_{n+1} = x_n - f'(x_n) = x_n - \tanh(x)$.

(f) See code

(g) $x_{n+1} = x_n - f'(x_n)/f''(x_n)$

$$= x_n - \frac{\tanh(x_n)}{\operatorname{sech}^2(x_n)} = x_n - \tanh(x_n)\cosh^2(x_n)$$
$$= x_n - \sinh(x_n)\cosh(x_n)$$

Update rule diverges to infinity (see code).
since $\sinh(x_n)\cosh(x_n)$ can be very large.
$|x_1| < |x_0| \iff |x_0 - \sinh(x_0)\cosh(x_0)| < |x_0|$.
this implies $|x_0| \leq 1.06$ 「Inequality Solver」

(h). The minimum in this case is NOT 0.

(i) It suffices to show that $g''(x) \leq 1$, since $Q(x,y)$ is aux. fn. for all f s.t. $f''(x) \leq 1$ $\forall x$.

$$g'(x) = \frac{1}{10}\sum_{k=1}^{10} \tanh\left(x + \frac{2}{\sqrt{k}}\right)$$

$$g''(x) = \frac{1}{10}\sum_{k=1}^{10} \operatorname{sech}^2\left(x + \frac{2}{\sqrt{k}}\right)$$

$\operatorname{sech}^2(x)$ has maxima $= 1$ at $x = 0$, so $g''(x) \leq g''(0)$, $\forall x$

$$\Rightarrow g''(0) = \frac{1}{10}\sum_{k=1}^{10} \operatorname{sech}^2\left(\frac{2}{\sqrt{k}}\right) \leq \frac{1}{10}\sum_{k=1}^{10} \operatorname{sech}^2(0) \leq \frac{1}{10} \times 10 = 1$$

$\Rightarrow g''(x) \leq 1$, $\forall x$.

(j) $$\frac{\partial R(x, x_n)}{\partial R} = 0 \Rightarrow g'(x_n) + (x - x_n) = 0$$
$$\Rightarrow x_{n+1} = x_n - g'(x_n)$$
$$= x_n - \frac{1}{10}\sum_{k=1}^{10} \tanh\left(x_n + \frac{2}{\sqrt{k}}\right).$$

(k) -0.9800

HW6 Code

November 11, 2021

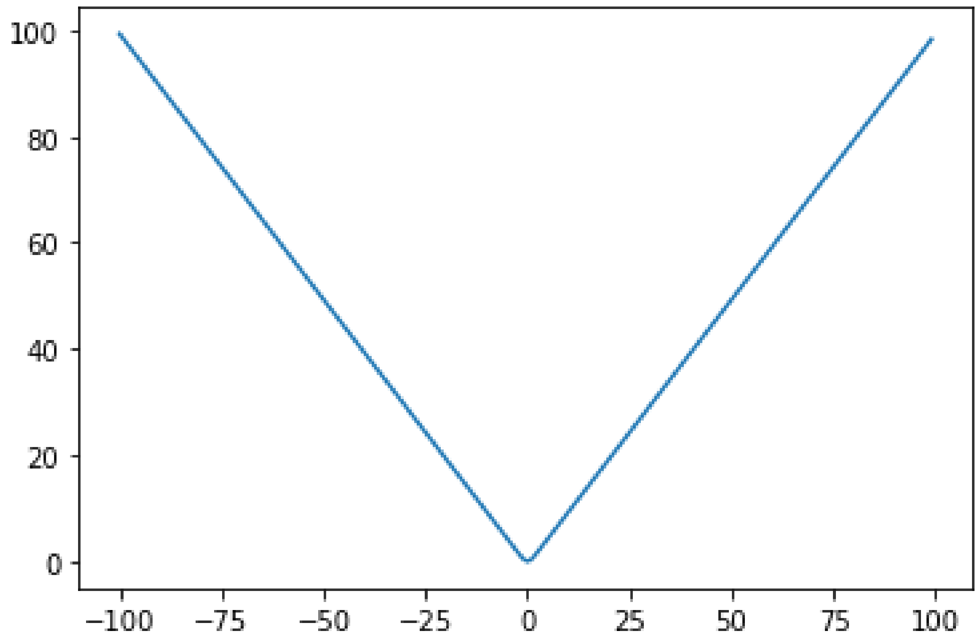
[2]:
```python
import copy
import math
import copy
import matplotlib.pyplot as plt
import numpy as np
# 6.3.d
X = []
with open("noisyOrX.txt", "r") as xf:
    for line in xf.readlines():
        X.append(line.strip('\n').split(' ')[:23])
for i in range(len(X)):
    for j in range(len(X[0])):
        X[i][j] = int(X[i][j])
Y = []
with open("noisyOrY.txt", "r") as yf:
    for line in yf.readlines():
        Y.append(int(line))


def prob(i, x: list, y, p):
    num = y * x[i] * p[i]
    res1 = 1
    for j in range(len(x)):
        res1 *= ((1 - p[j]) ** x[j])
    denom = 1 - res1
    return num / denom


T = []
for j in range(len(X[0])):
    res = 0
    for i in range(len(X)):
        if X[i][j] == 1:
            res += 1
    T.append(res)


def update(i, p: list, X: list, Y, T: list):
```

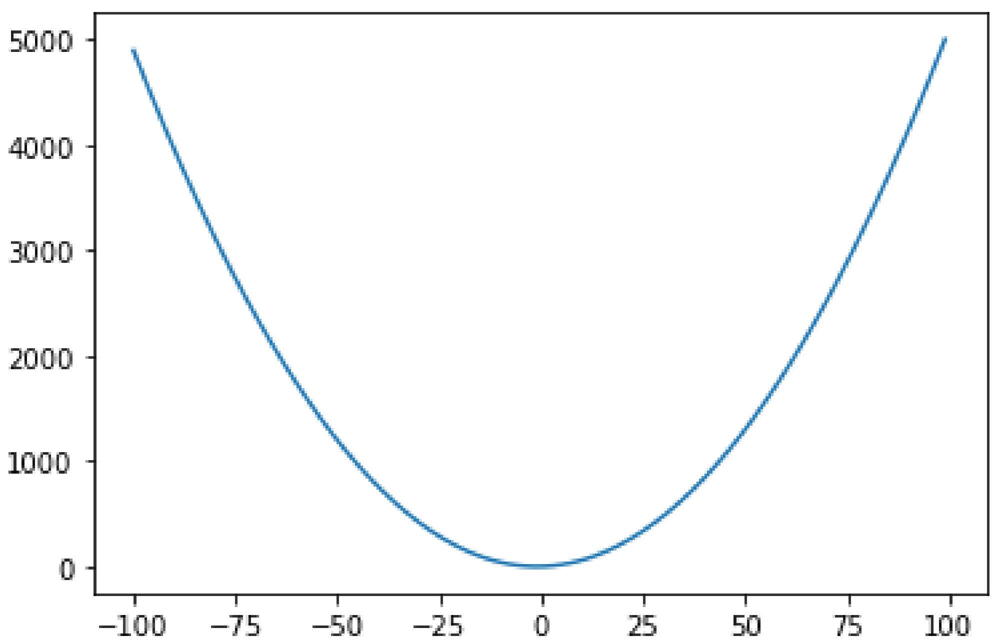
```python
    Ti = T[i]
    sum1 = 0
    for t in range(len(X)):
        sum1 += prob(i, X[t], Y[t], p)
    return sum1 / Ti


def likelihood(p, X, Y):
    sum1 = 0
    for t in range(len(X)):
        prod = 1
        for i in range(len(X[0])):
            prod = prod * ((1 - p[i]) ** X[t][i])
        if Y[t] == 1:
            sum1 += math.log(1 - prod)
        else:
            sum1 += math.log(prod)

    return sum1 / len(X)


def mistake(p, X, Y):
    M = 0

    for t in range(len(X)):
        prod = 1
        for i in range(len(X[0])):
            prod *= (1 - p[i]) ** X[t][i]
        if Y[t] == 0:
            if 1 - prod >= 0.5:
                M += 1
        if Y[t] == 1:
            if 1 - prod <= 0.5:
                M += 1
    return M


def em(iters, X, Y, T):
    p = [0.05] * 23
    L = likelihood(p, X, Y)
    M = mistake(p, X, Y)
    print(f"0: {M}, {L}")
    for k in range(1, iters + 1):
        temp_p = copy.deepcopy(p)
        for i in range(len(p)):
            p[i] = update(i, temp_p, X, Y, T)
        if math.log(k, 2) == int(math.log(k, 2)):
```

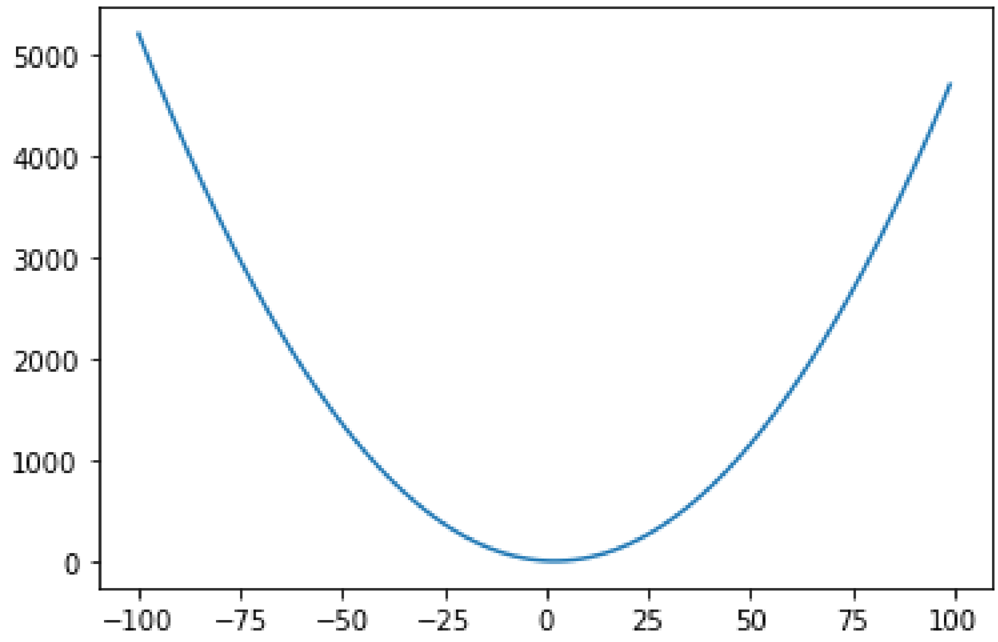
```python
            L = likelihood(p, X, Y)
            M = mistake(p, X, Y)
            print(f"{k}: {M}, {L}")


em(256, X, Y, T)
```

```
0: 175, -0.9580854082157914
1: 56, -0.49591639407753635
2: 43, -0.40822081705839114
4: 42, -0.3646149825001877
8: 44, -0.34750061620878253
16: 40, -0.33461704895854844
32: 37, -0.32258140316749784
64: 37, -0.3148266983628559
128: 36, -0.3111558472151897
256: 36, -0.310161353474076
```

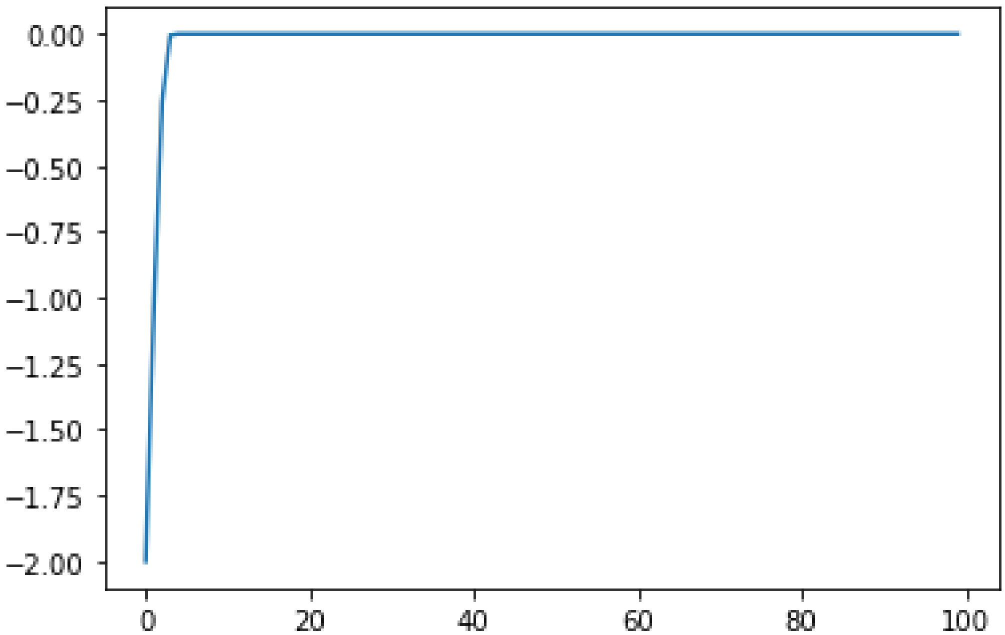
[3]:
```python
# 6.4.c
b1x = []
b1y = []
for i in range(-100, 100):
    b1x.append(i)
    b1y.append(np.log(np.cosh(i)))
plt.plot(b1x, b1y)
plt.show()
b2x = []
b2y = []
for i in range(-100, 100):
    b2x.append(i)
    res = np.log(np.cosh(-2))+np.tanh(-2) * (i + 2) + 0.5 * ((i + 2) ** 2)
    b2y.append(res)
plt.plot(b2x, b2y)
plt.show()
b3x = []
b3y = []
for i in range(-100, 100):
    b3x.append(i)
    res = np.log(np.cosh(3))+np.tanh(3) * (i - 3) + 0.5 * ((i - 3) ** 2)
    b3y.append(res)
plt.plot(b3x, b3y)
plt.show()
```

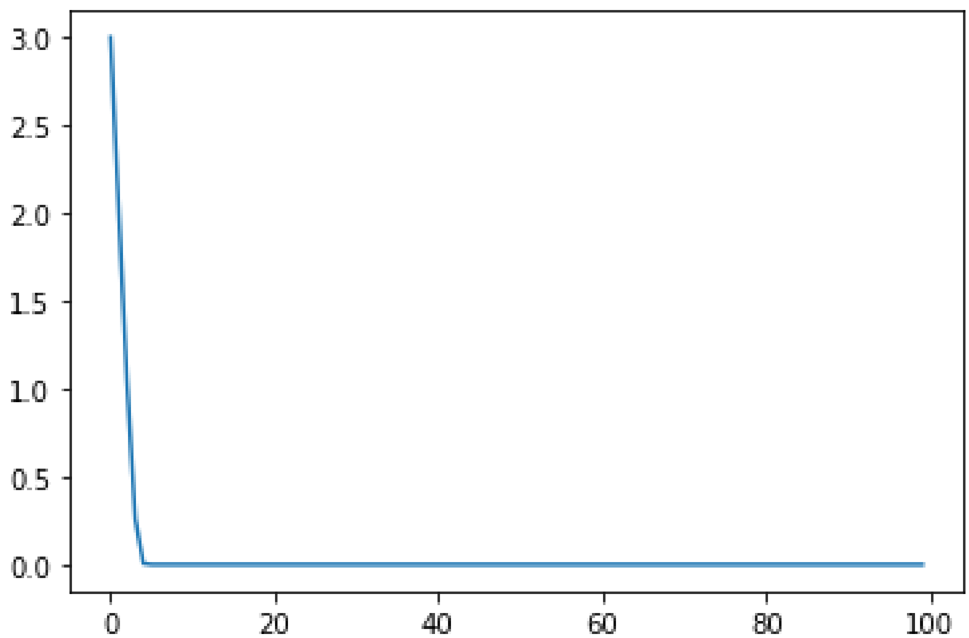

100
80
60
40
20
0
−100 −75 −50 −25 0 25 50 75 100


5000
4000
3000
2000
1000
0
−100 −75 −50 −25 0 25 50 75 100



[4]:
```
# 6.4.f
# x_n+1 = x_n - tanh(x)

x0 = -2
nArr = []
xArr = []
for i in range(100):
    nArr.append(i)
    xArr.append(x0)
    x0 = x0 - np.tanh(x0)
plt.plot(nArr, xArr)
plt.show()

x0 = 3
nArr = []
xArr = []
for i in range(100):
    nArr.append(i)
    xArr.append(x0)
    x0 = x0 - np.tanh(x0)
plt.plot(nArr, xArr)
plt.show()
```



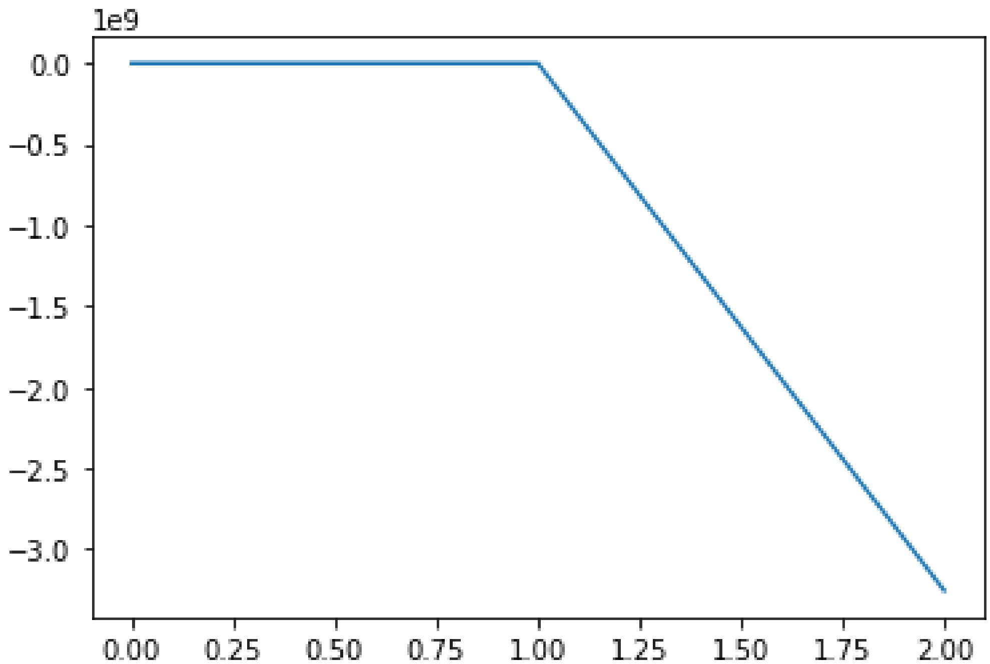


[5]:
```
# 6.4.g
x0 = -2
nArr = []
xArr = []
```

```
for i in range(100):
    nArr.append(i)
    xArr.append(x0)
    x0 = x0 - np.sinh(x0) * np.cosh(x0)
plt.plot(nArr, xArr)
print(xArr)
plt.show()

x0 = 3
nArr = []
xArr = []
for i in range(100):
    nArr.append(i)
    xArr.append(x0)
    x0 = x0 - np.sinh(x0) * np.cosh(x0)
plt.plot(nArr, xArr)
print(xArr)
plt.show()
```

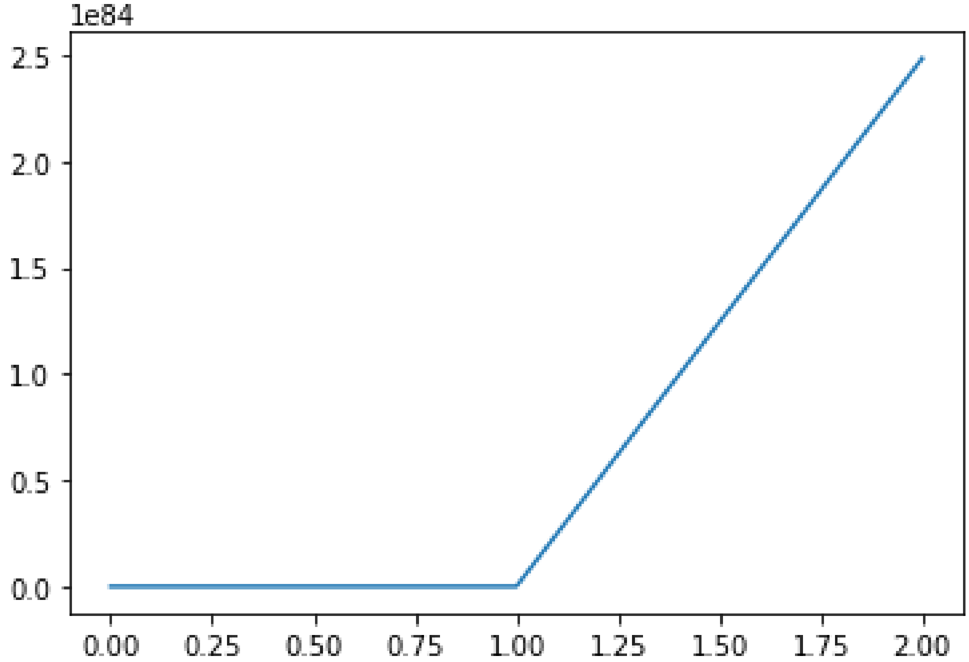
```
[-2, 11.644958598563875, -3255536207.1877036, inf, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan]
```

```
<ipython-input-5-64a119fe3e9a>:8: RuntimeWarning: overflow encountered in sinh
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
<ipython-input-5-64a119fe3e9a>:8: RuntimeWarning: overflow encountered in cosh
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
<ipython-input-5-64a119fe3e9a>:8: RuntimeWarning: invalid value encountered in
double_scalars
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
```

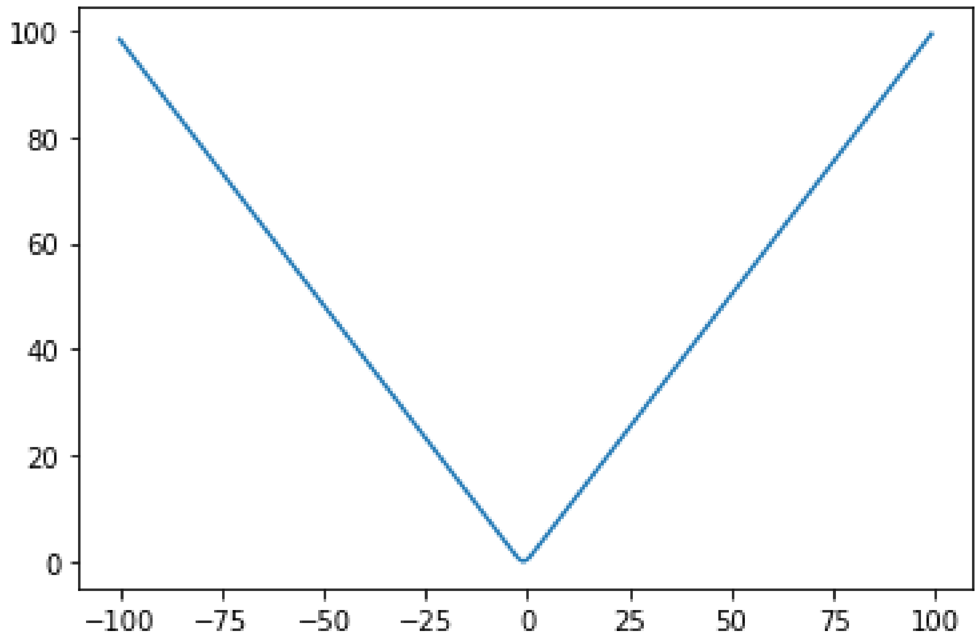



```
<ipython-input-5-64a119fe3e9a>:19: RuntimeWarning: overflow encountered in sinh
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
<ipython-input-5-64a119fe3e9a>:19: RuntimeWarning: overflow encountered in cosh
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
<ipython-input-5-64a119fe3e9a>:19: RuntimeWarning: invalid value encountered in
double_scalars
  x0 = x0 - np.sinh(x0) * np.cosh(x0)
```

```
[3, -97.85657868513961, 2.4836150932578143e+84, -inf, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan]
```



[6]:
```
# 6.4.h
xArr = []
gArr = []
for i in range(-100, 100):
    sumh = 0
    for k in range(1, 11):
        sumh += np.log(np.cosh(i + 2/np.sqrt(k)))
    xArr.append(i)
    gArr.append(sumh / 10)
plt.plot(xArr, gArr)
plt.show()
```



[7]:
```
# 6.4.k
xArr = []
gArr = []
x0 = 2
for i in range(100):
    xArr.append(i)
    gArr.append(x0)
    sumh = 0
    for k in range(1, 11):
        sumh += np.tanh(x0 + 2/np.sqrt(k))
    x0 = x0 - sumh * 0.1
print(gArr)
plt.plot(xArr, gArr)
plt.show()
```

```
[2, 1.0061451772465695, 0.049939702268616215, -0.6934594956924488,
-0.94527953722007, -0.9760448205051371, -0.9795316550334733,
-0.9799309378061212, -0.9799767255008595, -0.9799819770724016,
-0.9799825794074072, -0.9799826484930614, -0.9799826564169388,
-0.9799826573257793, -0.9799826574300201, -0.9799826574419761,
-0.9799826574433474, -0.9799826574435047, -0.9799826574435228,
-0.9799826574435249, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
```

```
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251,
-0.9799826574435251, -0.9799826574435251, -0.9799826574435251]
```