2.1

(a) Min is $g(O_{N*1})$ = 0, no matter the number of N

(b)

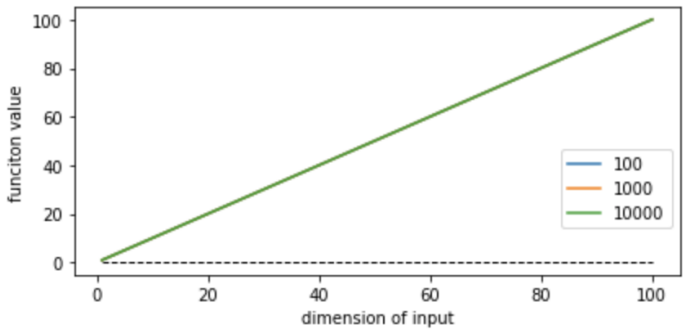
```
import matplotlib.pyplot as plt
import autograd.numpy as np
```

```
import sys
sys.path.append('../')
import matplotlib.pyplot as plt
import autograd.numpy as np
from mlrefined_libraries import calculus_library as calib
from mlrefined_libraries import math_optimization_library as optlib
static_plotter = optlib.static_plotter.Visualizer();
optimizers = optlib.optimizers
%matplotlib notebook
from matplotlib import rcParams
rcParams['figure.autolayout'] = True
%load_ext autoreload
%autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

```
%matplotlib inline
optlib.random_method_experiments.random_eval_experiment()
```

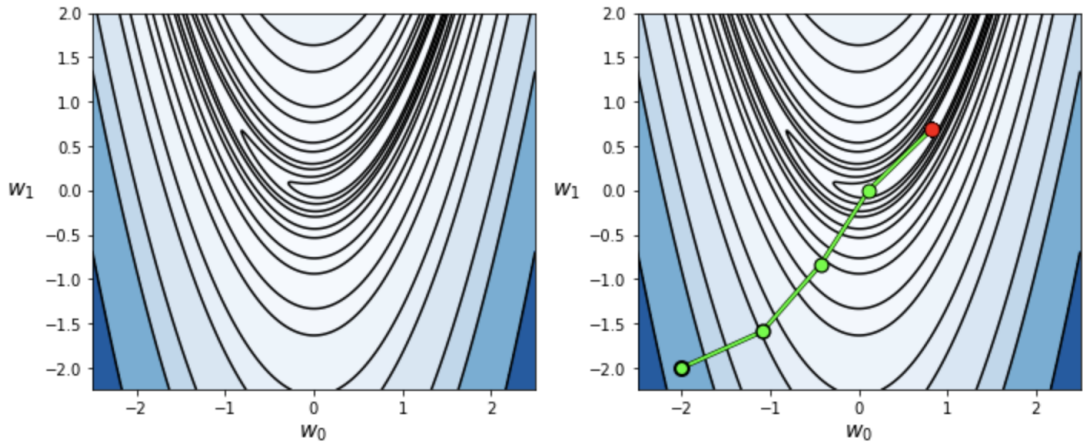
(c)



2.4

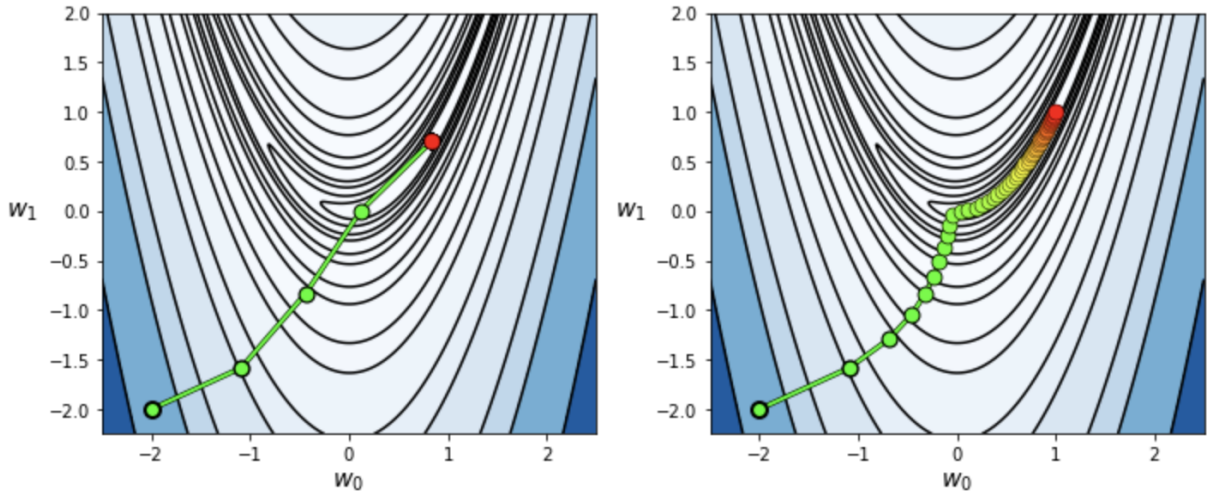
```
def random_search(g,a,maxx,w,num):
    whistory = []
    chistory = []
    al = 0
    for k in range(1,maxx+1):
        if a == 'diminishing':
            al = 1/float(k)
        else:
            al = a
        whistory.append(w)
        chistory.append(g(w))
        direction = np.random.randn(num_sam,np.size(w))
        norms = np.sqrt(np.sum(direction*direction,axis = 1))[:,np.newaxis]
        direction = direction/norms
        w_can = w + al*direction
        evals = np.array([g(w_val) for w_val in w_can])
        ind = np.argmin(evals)
        if g(w_can[ind]) < g(w):
            d = direction[ind,:]
            w = w + al*d
    whistory.append(w)
    chistory.append(g(w))
    return whistory,chistory
```

```
g = lambda w: 100*(w[1] - w[0]**2)**2 + (w[0] - 1)**2
a = 1
w = np.array([-2,-2])
num = 1000
maxx = 50
wh,ch = random_search(g,a,maxx,w,num)
static_plotter.two_input_contour_plot(g,weight_history_1,num_contours = 35,xmin = -2.5,xmax = 2.5,ymin = -2.25,ymax = 2)
```




```
g = lambda w: 100*(w[1] - w[0]**2)**2 + (w[0] - 1)**2
a = 'diminishing';
w = np.array([-2,-2]);
num = 1000;
maxx= 50;
wh2,ch2 = random_search(g,a,maxx,w,num)
```

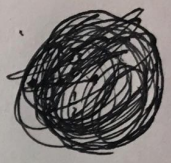
```
# show run in both three-dimensions and just the input space via the contour plot
static_plotter.compare_runs_contour_plots(g,[wh,wh2],contours = 35,xmin = -2.5,xmax = 2.5,ymin = -2.25,ymax = 2,show_original = False)
```



2.5

a) $N=2$ · $g(w^0+\alpha d^0) < g(w^0)$ for random d^0 is upper $\frac{\sqrt{3}}{4}$.

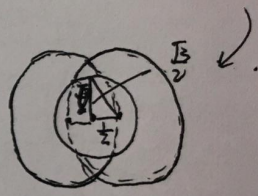
start with $w^0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$



Note:
Big circle here assumed to be unit circle.

descent probability $\leq \frac{1}{2} \frac{\text{len of small circle}}{\text{len of big circle}}$.

$$< \frac{1}{2} \frac{2\pi \frac{\sqrt{3}}{2}}{2\pi} = \frac{\sqrt{3}}{4}$$



b)

$$\text{des. prob} < \frac{1}{2} \frac{\text{Surface area of small circle}}{\text{Surface area of big circle}}.$$

$$= \frac{1}{2} \cdot \left(\frac{\sqrt{3}}{2}\right)^{N-1}$$

a). $g'(w) = \log(w) - \log(1-w) = 0$. ← set this to 0

$\Rightarrow \log\left(\frac{w}{1-w}\right) = 0$.

$\frac{w}{1-w} = e^0 = 1 \quad \Rightarrow w = 1-w \,/\, w = \frac{1}{2}$.

b). $g'(w) = \frac{ew}{1+e^w} = 0. \quad \Rightarrow w = -\infty$.

c) $g'(w) = \tanh(w) + w(1-\tanh^2(w)) = 0. \Rightarrow w = 0$.

d) $\Rightarrow Bw = -C \quad w = \begin{bmatrix} -0.4 \\ -0.2 \end{bmatrix}$

$$\nabla g(w) = \frac{2\left(Cw - \frac{w^T C w}{w^T w} w\right)}{w^T w}$$

$$= 0.$$

$$\Rightarrow \frac{2}{v^T v}\left(\lambda v - \lambda \frac{v^T v}{v^T v} v\right) = 0$$

$$\frac{2}{v^T v}(\lambda v - \lambda v) = 0.$$

$$g(v) = \frac{v^T C v}{v^T v} = \lambda \frac{v^T v}{v^T v}$$

$$= \lambda.$$

3.5

g’(w) = (4 + 2w + 10)

```
def gradientdescent(a,maxx,w):
    g = lambda w: 1/50*(w**4 + w**2 + 10*w)
    grad = lambda w: 1/50*(4*w**3 + 2*w + 10)
    ch = [g(w)]
    for k in range(1,maxx+1):
        geval = grad(w)
        w = w - a*geval
        ch.append(g(w))
    return ch
```

```
w = 2.0
maxx = 1000
a = 10**(0)
ch1 = gradientdescent(a,maxx,w)
a = 10**(-1)
ch2 = gradientdescent(a,maxx,w)
a = 10**(-2)
ch3 = gradientdescent(a,maxx,w)
static_plotter.plot_cost_histories([ch1,ch2,ch3],start = 0,points= False,labels =
                                   [r'$\alpha = 1$',r'$\alpha = 10^{-1}$',r'$\alpha = 10^{-2}$'])
```