Q1.1:

~~Def~~ Def $g(w) = f(w)$

$$h(w) = L_C(w) = \begin{cases} 0 & \text{if } w \in C \\ \infty & \text{o.w} \end{cases}$$

then we assume that strong duality hold and by 15.9 we have:

$$\inf_{w \in C} f(w) = \inf_{w} g(Iw) + h(w)$$

$$= -\inf_{y} g^*(y) + h^*(-I^T y)$$ ~~from~~ Fenchel-Rockafeller duality

$$= -\inf_{y} \left(f^*(y) + L_C^*(-y) \right)$$

Q1.2

by @398, we assume f is L-smooth, and all function values and their subgradients ~~are~~ can be easily computed.

then by 11.11. f^* is $\frac{1}{L}$-strongly convex

by 11.10 L_C^* is always closed and convex

We Define

$$H(y) = L_C^*(-y)$$

Since L_C^* and affine function are convex

So $H(y)$ is also convex

Now for $f^* + H$, $\forall x, y \in \mathbb{R}^d$, $\lambda \in [0,1]$, by 5.20.

$$(f^* + H)(\lambda x + (1-\lambda)y) + \frac{1}{L} \frac{\lambda(1-\lambda)}{2} \|x-y\|_2^2$$

$$= f^*(\lambda x + (1-\lambda)y)^{①} + H(\lambda x + (1-\lambda)y) + \frac{1}{L} \frac{\lambda(1-\lambda)}{2} \|x-y\|_2^2 \quad ② \qquad ③$$

Since f^* is $\frac{1}{L}$-strongly convex

Combine ①, ②, the sum is smaller or equal to

$$\lambda f^*(x) + (1-\lambda) f^*(y)$$

So ③ $\leq \lambda f^*(x) + (1-\lambda) f^*(y) + \lambda H(x) + (1-\lambda) H(y)$

So the strongly-convexity is hold.

We claim that both f^* and L_c are L_c - ~~[illegible]~~ smooth

So $\forall x, y$ we have $\|(f^*+H)(x) - (f^*+H)(y)\|$ $\qquad L_c = \sup_{w \in C} \|w\|$

$\le \|f^*(x) - f^*(y)\| + \|H(x) - H(y)\| \le 2L_c\|x-y\|$

(~~this claim is proved in Appendix~~).

By @398, we can use such assumption.

Note that: by 6.7

$\partial(f^*+H) \supseteq \partial(f^*) + \partial(H)$

We may apply subgradient algorithm here.

Note by 6.18 we pick $\eta_t = \frac{L}{r(t+1)}$

the domain is $\mathbb{R}^d$, obj function is f^*+H

Algorithm:

for $t = 0, 1, \cdots$ do:

choose $d_t \in \partial f^*(w_t) + \partial L_c^*(-w_t)$

$\eta_t = \frac{L}{t+1}$

$w_{t+1} = w_t - \eta_t d_t$

by 6.18. with $\eta_t = \frac{L}{(t+1)}$,

we have the upper bound of

$$\frac{(2L_c)^2 \sum_{t=0}^{T-1} \frac{1}{t+1}}{2T/L}$$ converges at $O\left(\frac{\log T}{T}\right)$

Q1.3:

$$-d_\mu^2(s) = -\min_{w\in C} \langle w, s\rangle + \frac{1}{2\mu}\|w\|_2^2$$

$$= \sup_{w\in C} \left(-\langle w, s\rangle - \frac{1}{2\mu}\|w\|_2^2\right)$$

Note that for a fix $w \in C$.

$-\langle w, s\rangle - \frac{1}{2\mu}\|w\|_2^2$ is affine (by 1.24)

so by 1.27 $-d_\mu^2(s)$ is convex (1.27.4).

We use 11.8.

$$d_\mu^2(s) = \min_{w\in C} \langle w, s\rangle + \frac{1}{2\mu}\|w\|_2^2$$

$$= \frac{1}{2\mu}\min_{w\in C}\left(2\mu\langle w, s\rangle + \|w\|_2^2\right)$$

$$= -\frac{1}{2\mu}\|\mu s\|^2 + \frac{1}{2\mu}\min\left(\|\mu s\|_2^2 + 2\mu\langle w, s\rangle + \|w\|_2^2\right)$$

$$= -\frac{1}{2\mu}\|\mu s\|_2^2 + \inf_w\left(\frac{1}{2\mu}\|w + \mu s\|_2^2 + I_C(w)\right) \text{ (by 4.6)}$$

$$= -\frac{1}{2\mu}\|\mu s\|_2^2 + M_{I_C}^{\mu}(-\mu s)$$

Since I_C is closed and convex, by 11.12

$M_{I_C}^{\mu}(-\mu s)$ is convex and $\frac{1}{\mu}$- smooth

~~And~~ $\cancel{\nabla M_{I_C}^{\mu}(-\mu s) = \frac{1}{\mu}((-\mu s) - P_C(-\mu s))} \cdot \mu$

$\cancel{= -\mu s - P_C(-\mu s)}$

And $\nabla M_f^{\eta}(z) = \frac{1}{\eta}(z - P_f^{\eta}(z))$

So: $\nabla M_{I_C}^{\mu}(z) = \frac{1}{\mu}(z - P_{I_C}^{\mu}(z)) = \frac{1}{\mu}(z - P_C(z))$

$$\therefore \nabla(-d_{\mu}^{2})(s) = \mu s - (-\mu)\cdot\left[\frac{1}{\mu}(z - P_C(z))\right]_{z=-\mu s}$$

$$= \mu s + z - P_C(z) = -P_C(-\mu s)$$

Note that μs is affine and thus convex.

for s, t

$$\|-P_C(-\mu s) + P_C(-\mu t)\| = \|\underbrace{\arg\min_{w\in C}\|-\mu t - w\| - \arg\min_{w\in C}\|-\mu s - w\|}_{①}\|$$

~~Note that $0 \in C$~~ so

~~if $s>0$, $t>0$~~

① ~~this part $\leq \|\mu$~~ $t\| + \mu\|s\|$

Now by 6.16 we have

$\forall s,t \quad \|-P_C(-\mu s) + P_C(-\mu t)\| \leq \mu\|s-t\|$

So $-d_{\mu}^{2}(z)$ is μ-smooth by 2.14

1.4

Combine 5.19 and 5.7 we have (set $s_0=0$)

for $t=0,1,\cdots$

$$w_t = -\nabla(-d_\mu^2)(s_t)$$

$$g_t = \underset{s}{\operatorname{argmin}} f^*(s) + \langle s, -w_t \rangle$$

$$\gamma_t = \frac{1}{t+1}$$

$$s_{t+1} = (1-\gamma_t) s_t + \gamma_t g_t$$

by 1.3 We have: $w_t = P_C(-\mu_t s_t)$

$\because s_0=0 \quad \therefore w_0=0$

By Fenchel-Young Inequality,

$$f^*(s) \geq \langle w_t, s \rangle - f(w_t)$$

So: $f^*(s) + \langle s, -w_t \rangle \geq -f(w_t)$

and the equality iff $s \in \partial f(w_t)$

So Now We have

$w_0=0, s_0=0$

for $t=0,1,\cdots$

$$g_t \in \partial f(w_t)$$

$$\gamma_t = \frac{1}{t+1}$$

$$s_{t+1} = (1-\gamma_t) s_t + \gamma_t g_t$$

$$w_{t+1} = P_C(-\mu_{t+1} s_{t+1})$$

Claim: $S_{t+1} = \frac{1}{t+1} \sum_{i=0}^{t} g_t$

Base Case $t=0$, $r_t = 1$ so

$$S_1 = (1-1) S_0 + g_0 = g_0$$

Suppose $S_{t+1} = \frac{1}{t+1} \sum_{i=0}^{t} g_t$ $t \geq 0$,

$$r_{t+1} = \frac{1}{t+2}$$

$$S_{t+2} = (1 - r_{t+1}) S_{t+1} + r_{t+1} g_{t+1}$$

$$= (1 - \frac{1}{t+2}) \cdot \frac{1}{t+1} \sum_{i=0}^{t} g_i + \frac{1}{t+2} g_{t+1}$$

$$= \frac{1}{t+2} \sum_{i=0}^{t+1} g_i \qquad \square$$

So we can change it to

for $t = 0, 1, \cdots$

$g_t \in \partial f(w_t)$

$S_{t+1} = S_t + g_t$

$w_{t+1} = P_C(-\mu_{t+1} \cdot \frac{1}{t+1} S_{t+1})$

We want $\frac{\mu_{t+1}}{t+1} = \eta_{t+1} = \frac{\eta_0}{\sqrt{t+1}}$

i.e $\mu_{t+1} = \eta_0 \sqrt{t+1}$

this is

for $t = 0, 1, \cdots$

$g_t \in \partial f(w_t)$

$S_{t+1} = S_t + g_t$

$\eta_{t+1} = \frac{\eta_0}{\sqrt{t+1}}$

$w_{t+1} = P_C(-\eta_{t+1} S_{t+1})$

$\rightarrow$ which is algorithm 1.

1.5:

$f^*(s) + L_C^*(-s) \geq \langle s, 0\rangle - f(0) + \langle s, 0\rangle = -f(0)$

$f^*(s) - d_\mu^2(-s) \geq \langle s, 0\rangle - f(0) + \langle s, 0\rangle - \|0\|_2^2 = -f(0)$

Also $L_C^*(s) = \sup_{w\in C} \langle w, -s\rangle$

$-d_\mu^2(s) = \sup_{w\in C} \langle w, -s\rangle - \frac{1}{2\mu}\|w\|_2^2$

$\therefore\ L_C^*(s) \geq -d_\mu^2(s)$

And we can apply EVT on $L_C^*(s)$, $-d_\mu^2(s)$

i.e $\exists\, w_1(s)$, $w_2(s)$ s.t

$-d_\mu^2(s) = \langle w_1(s), -s\rangle - \frac{1}{2\mu}\|w_1(s)\|_2^2$

$L_C^* = \langle w_2(s), -s\rangle$

And $-d_\mu^2(s) \geq \langle w_2(s), -s\rangle - \frac{1}{2\mu}\|w_2(s)\|_2^2$

$\geq \langle w_2(s), -s\rangle - \frac{1}{2\mu} L_C^2$

$L_C = \sup_{w\in C}\|w\|$

So we may pick $\mu = \frac{3L_C^2}{2\varepsilon}$

then $-d_\mu^2(s) \geq L_C^*(-s) - \frac{\varepsilon}{3}$

So we have

$0 \leq f^*(s) + L_C^*(-s) - [f^*(s) - d_\mu^2(s)] \leq \frac{\varepsilon}{3}$

So by 11.5 we have

$0 \leq \inf(f^*(s) + L_C^*(-s)) - \inf(f^*(s) - d_\mu^2(s)) \leq \varepsilon/3$

We know that $f^*(s) - d_\mu^2(s)$ is μ-smooth.

So: it may converge to a $\varepsilon/3$- approximate minimizer

$$\Rightarrow (f^*(s) - d_\mu^2(s)) - \inf (f^*(s) - d_\mu^2(s)) \le \varepsilon/3$$

So: $f^*(s) + L_c^*(-s) - \inf(f^*(s) + L_c^*(-s))$

$$\le \left|(f^*(s) - d_\mu^2(s)) - (f^*(s) + L_c^*(-s))\right| + \left|f^*(s) - d_\mu^2(s) - \inf (f^*(s) - d_\mu^2(s))\right|$$

$$+ \left|\inf (f^*(s) - d_\mu^2(s)) - \inf(f^* + L_c^*(-s))\right| \le \varepsilon/3 + \varepsilon/3 + \varepsilon/3 = \varepsilon$$

So we pick

$$\mu = \frac{3L_c^2}{2\varepsilon} \qquad L_c = \sup_{w \in C} \|w\|$$

the step is $O\left(\frac{\mu}{\varepsilon/3}\right) =$

$$= O\left(\frac{9L_c^2}{2\varepsilon^2}\right)$$

Q2.1

SVM: $\min_{w} \frac{1}{2}\|w\|_2^2 + c \cdot \sum_{i}(1-y_i\hat{y}_i)_+$ $c \geq 0$

for $c(z)_+ = \begin{cases} cz & \text{if } z \geq 0 \\ 0 & \text{o.w} \end{cases}$

for $\alpha_i \in [0, c]$

$c(z)_+ = \max_{\alpha_i \in [0,c]} \alpha \cdot z$ since if $z \geq 0$, α_i is c
if $z < 0$ $\alpha_i \geq 0$,
we have $\alpha = 0$

So: $c\sum(1-y_i\hat{y}_i)_+$
$= \sum_{i} \max (1-y_i\hat{y}_i)\alpha_i$

$= \max_{c \geq \alpha \geq 0} \sum_{i} (1-y_i\hat{y}_i)\alpha_i$

$= \max_{c \geq \alpha \geq 0} \sum_{i} (1-\langle y_i x_i, w\rangle)\alpha_i$

So the lagrangian formulation of SVM
is $\min_{w}\left(\frac{1}{2}\|w\|_2^2 + \max_{c \geq \alpha \geq 0} \sum_{i}(1-\langle y_i x_i, w\rangle)\alpha_i\right)$

$= \min_{w} \max_{c \geq \alpha \geq 0} \left(\frac{1}{2}\|w\|_2^2 + \sum_{i}(1-\langle y_i x_i, w\rangle)\alpha_i\right)$

Q22

Claim: $f(w, \alpha)$ is L-smooth and strong-duality holds

$$f(w,\alpha) = \frac{1}{2}\|w\|_2^2 + \sum_i (1 - \langle y_i x_i, w \rangle)\, \alpha_i$$

$$\frac{\partial f}{\partial w} = w - \sum_i y_i x_i^T \cdot \alpha_i$$

$$\frac{\partial f}{\partial \alpha_i} = 1 - y_i x_i^T w$$

If Def $X = \begin{bmatrix} y_1 x_1^T \\ \vdots \\ y_n x_n^T \end{bmatrix}$ we will have

$$\frac{\partial f}{\partial w} = w - X^T \alpha \qquad \frac{\partial f}{\partial \alpha} = \vec{1} - Xw$$

So $\forall (w, \alpha), (z, \beta)$

$$\left\| \begin{bmatrix} w - X^T\alpha \\ 1 - Xw \end{bmatrix} - \begin{bmatrix} z - X^T\beta \\ 1 - Xz \end{bmatrix} \right\| = \left\| \begin{bmatrix} \cancel{x - z}\ w - z - X^T(\alpha - \beta) \\ -X(w - z) \end{bmatrix} \right\|$$

$$= \left\| \begin{bmatrix} I & -X^T \\ -X & 0 \end{bmatrix} \cdot \begin{bmatrix} w - z \\ \alpha - \beta \end{bmatrix} \right\|$$

$$\leq \left\| \begin{bmatrix} I & -X^T \\ -X & 0 \end{bmatrix} \right\|_{sp} \cdot \left\| \begin{bmatrix} w - z \\ \alpha - \beta \end{bmatrix} \right\|$$

So $L = \left\| \begin{bmatrix} I & -X^T \\ -X & 0 \end{bmatrix} \right\|_{sp}$

$f(w,\cdot)$ is linear and thus quasi-concave

$f(\cdot,\alpha)$ is continuous and strongly convex since:

~~tw~~ all parts are continuous

$\sum_i (1-\langle y_i x_i, w\rangle)$ is linear function w.r.t w

$\nabla^2 f(w) = \nabla^2 \frac{1}{2}\|w\|_2^2 = I$ ~~(by 5~~

So f is 1-strongly-convex

since: $\langle z, Iz\rangle \geq 1\cdot\|z\|_2^2$ by ~~5.2-1~~ 5.21 (III)

~~1~~. Also, since f is ~~a~~ strong-convex,

so inf-compactness holds

So by Minimax Theorem (15.15).

f is strong duality holds $\square$

Q2.6

T is monotone and L-Lipschitz, so we can choose

$\eta_t \in [0, \frac{1}{L}]$. (by 17.5)

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
X = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
X = np.append(X,np.ones((4,1)),axis=1)
y = np.array([1,1,1,-1])
init_w = np.zeros(3)
init_alpha = np.zeros(4)
```

In [3]:
```
def GDA(func_w, func_a, eta_c, w, alpha, maxiter,c):
    w_list = []
    w_list.append(w)
    alpha_list = []
    alpha_list.append(alpha)
    beta = alpha
    for t in range(maxiter):
        fg_w = func_w(w,alpha)
        fg_alpha = func_a(w,alpha)

        if eta_c =='1/t':
            eta = 1/(t+1)
        elif eta_c =='1/t^2':
            eta = 1/(t+1)**2
        elif eta_c == 'sq':
            eta = 1/(t+1)**(1/2)

        w = w-eta*fg_w
        w_list.append(w)

        N = alpha + eta*fg_alpha
        alpha = Proj(N,c)
        alpha_list.append(alpha)

    return w_list, alpha_list
```

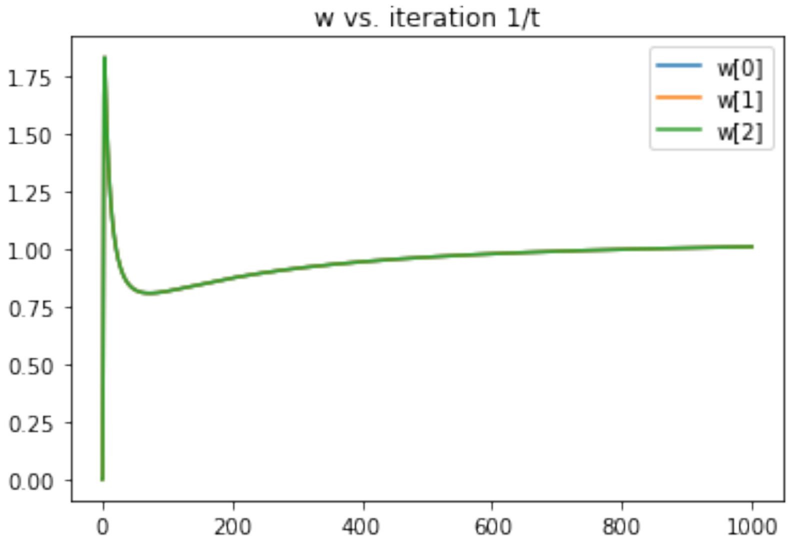
In [4]:
```
def Proj(N,c):
    result = []
    for i in N:
        result.append(min(10, max(0,i)))
    return np.array(result)
```

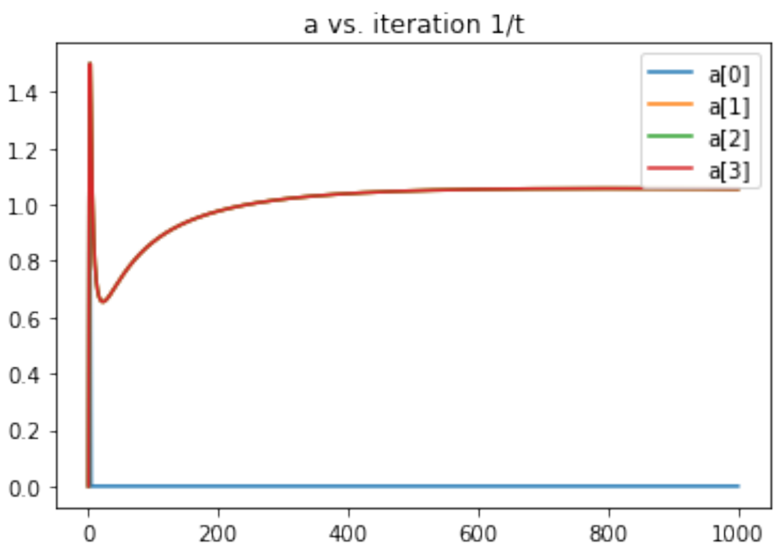
In [5]:
```
def func_w(w, alpha):
    result = np.zeros(len(w))
    for i in range(len(alpha)):
        result += y[i]*X[i]*alpha[i]
    return w - result
```

In [6]:
```
def func_alpha(w, alpha):
    result = []
    for i in range(len(alpha)):
        result.append(1- np.dot(y[i]*X[i],w))
    return np.array(result)
```

In [7]:
```
w1, a1 = GDA(func_w, func_alpha, '1/t', init_w, init_alpha, 1000,10)
```

In [9]:
```
draw(w1,a1,1)
```

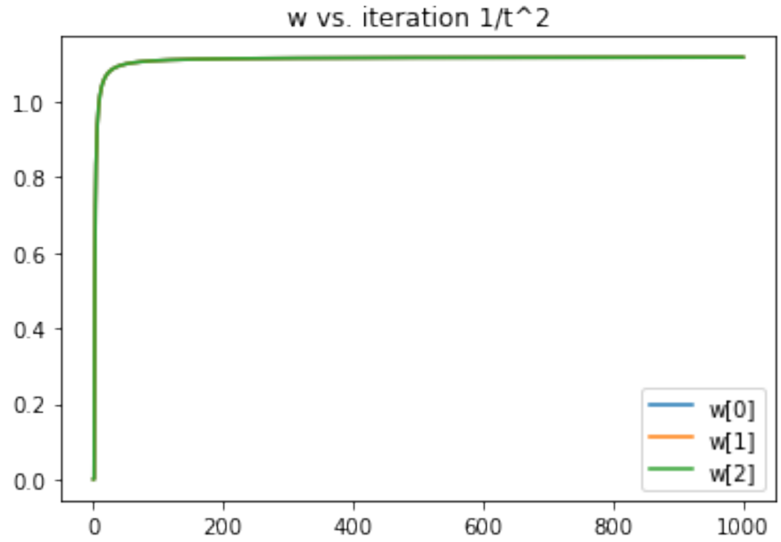


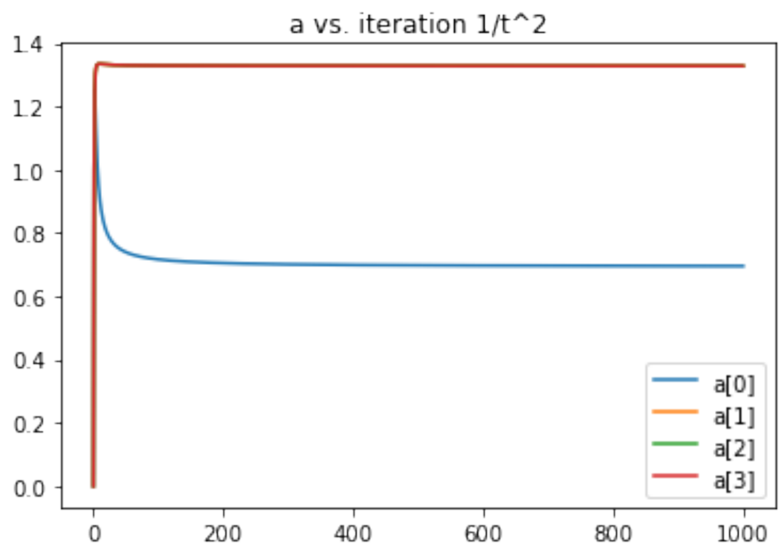


In [ ]:

In [10]:
```
w1, a1 = GDA(func_w, func_alpha, '1/t^2', init_w, init_alpha, 1000,10)
```

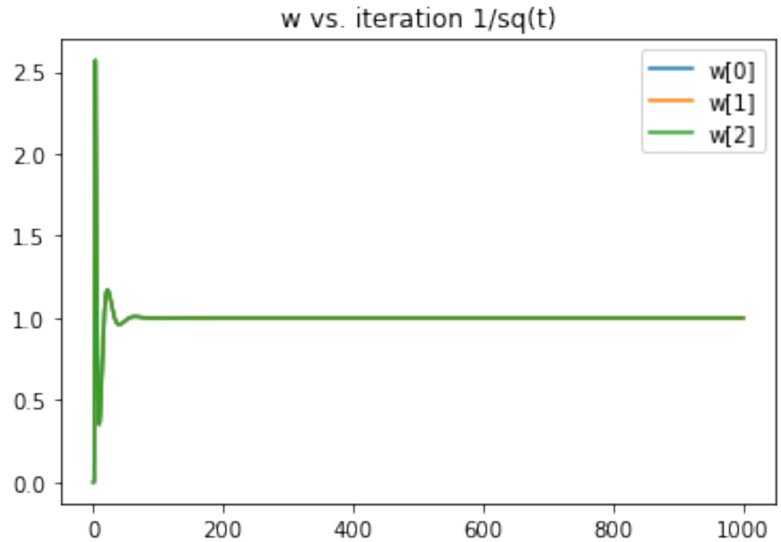
In [11]:
```
draw(w1,a1,2)
```

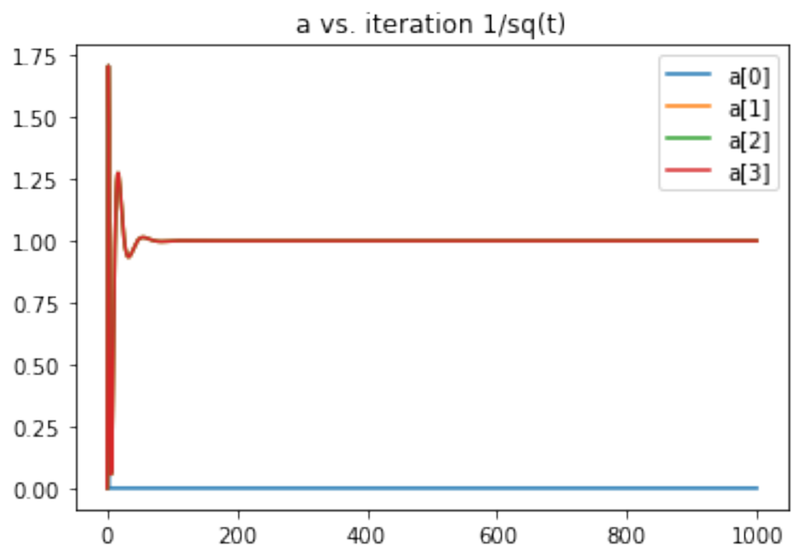




In [12]:
```
w1, a1 = GDA(func_w, func_alpha, 'sq', init_w, init_alpha, 1000,10)
```

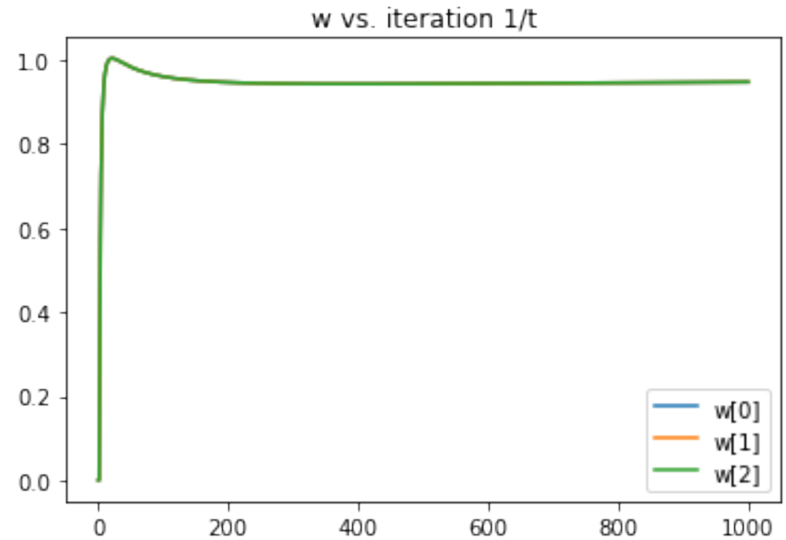
In [13]:
```
draw(w1,a1,3)
```

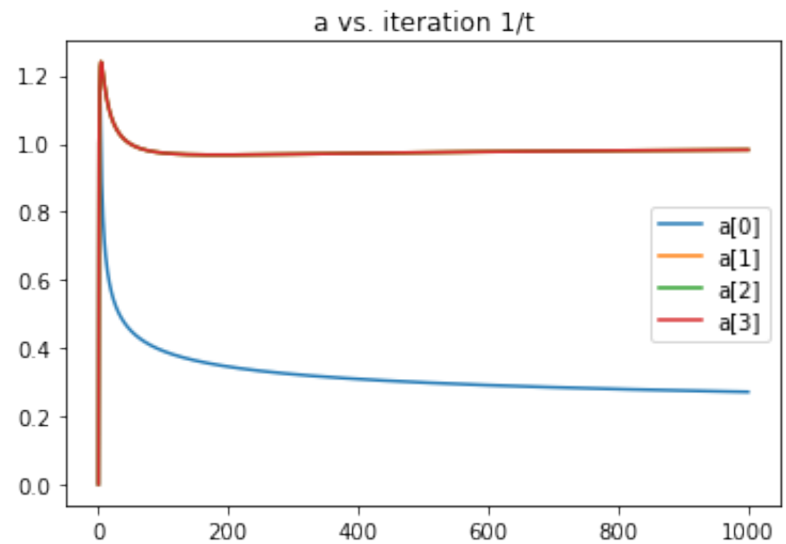




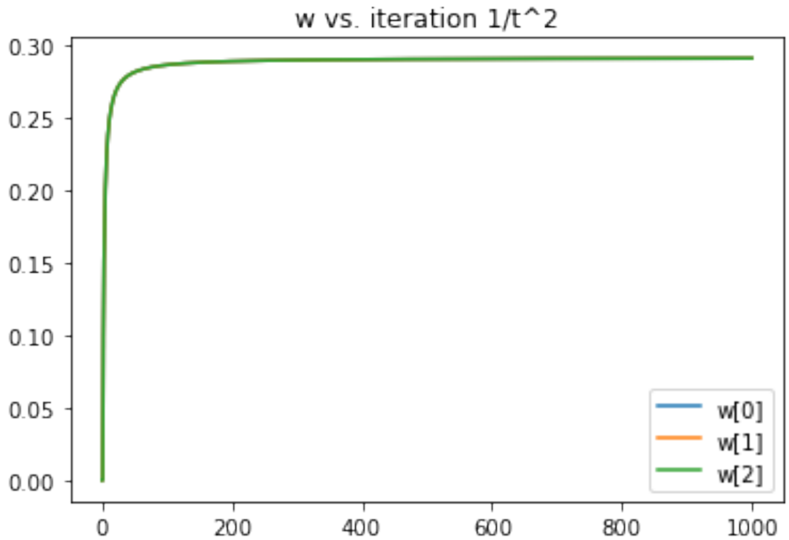
In [14]:
```
def Avg_GDA(func_w, func_a, eta_c, w, alpha, maxiter,c):
    w_list = []
    w_list.append(w)
    alpha_list = []
    alpha_list.append(alpha)
    beta = alpha
    sum_eta = 0
    sum_w = 0
    sum_a = 0
    for t in range(maxiter):
        fg_w = func_w(w,alpha)
        fg_alpha = func_a(w,alpha)

        if eta_c =='1/t':
            eta = 1/(t+1)
        elif eta_c =='1/t^2':
            eta = 1/(t+1)**2
        elif eta_c == 'sq':
            eta = 1/(t+1)**(1/2)

        sum_eta += eta

        w = w-eta*fg_w


        N = alpha + eta*fg_alpha
        alpha = Proj(N,c)
        sum_w += eta*w
        sum_a += eta*alpha

        w_avg = sum_w/sum_eta
        a_avg = sum_a/sum_eta
        w_list.append(w_avg)
        alpha_list.append(a_avg)



    return w_list, alpha_list
```

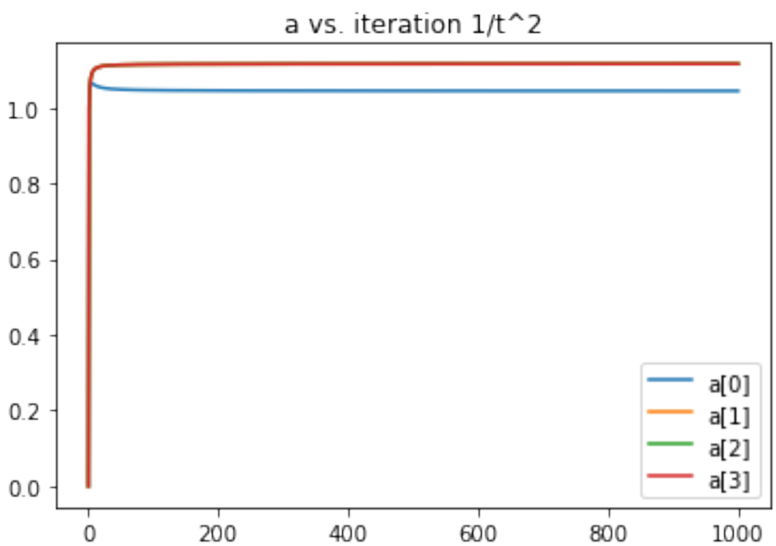
In [16]:
```
w1, a1 = Avg_GDA(func_w, func_alpha, '1/t', init_w, init_alpha, 1000,1
draw(w1,a1,1)
```



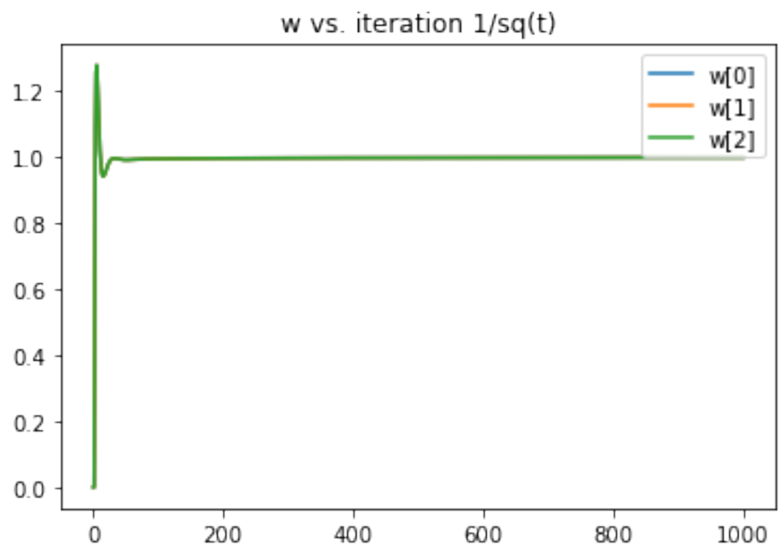


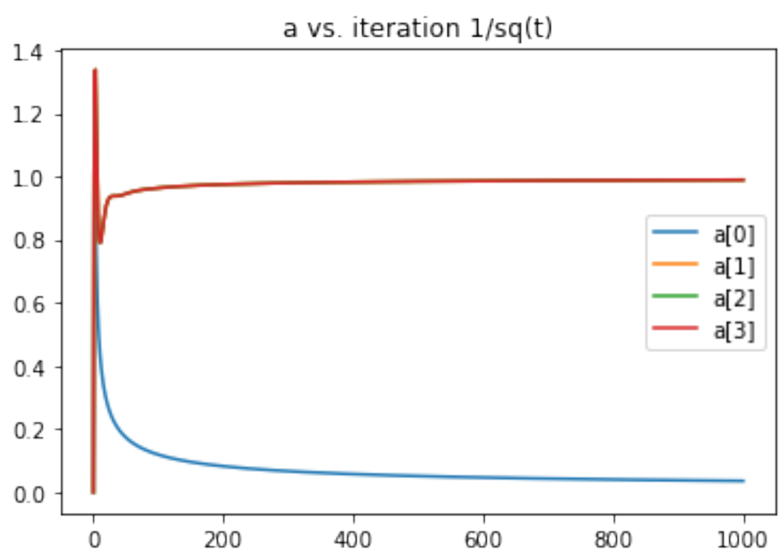
In [17]:
```
w1, a1 = Avg_GDA(func_w, func_alpha, '1/t^2', init_w, init_alpha, 1000
draw(w1,a1,2)
```





In [18]:
```
w1, a1 = Avg_GDA(func_w, func_alpha, 'sq', init_w, init_alpha, 1000,10
draw(w1,a1,3)
```



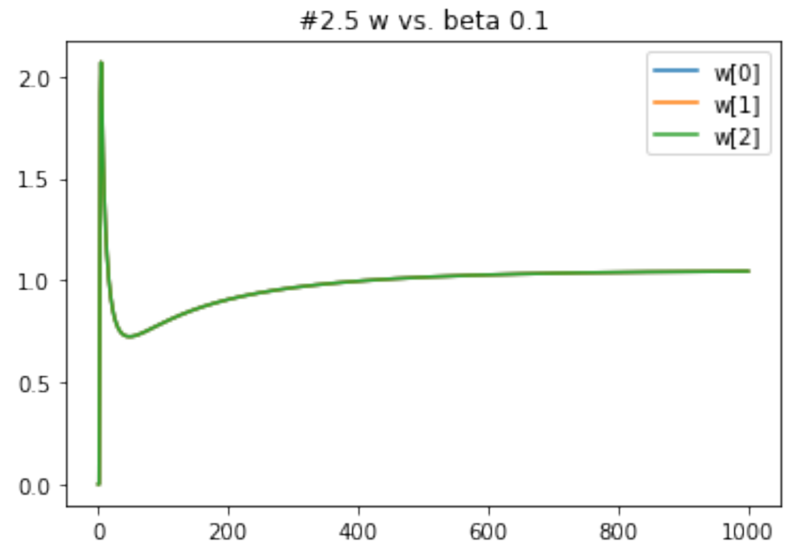


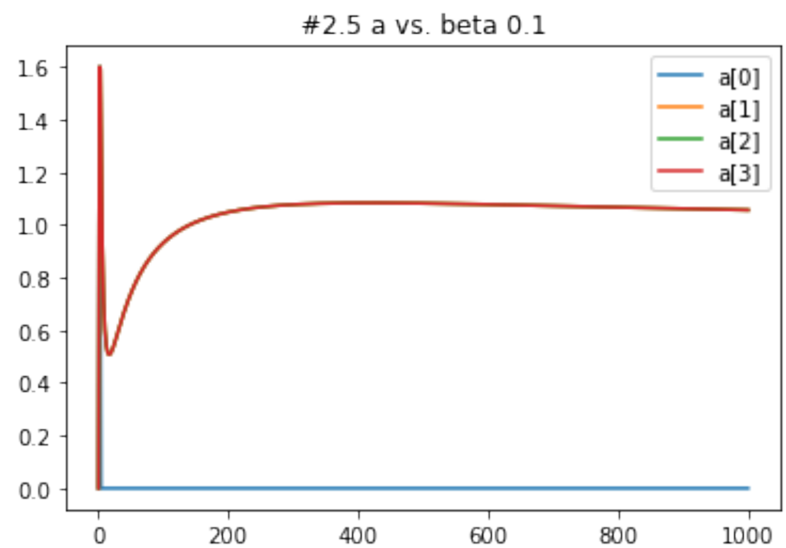
In [19]:
```
a1[-1]
```

Out[19]:
```
array([0.03571312, 0.98894366, 0.98894366, 0.98894366])
```

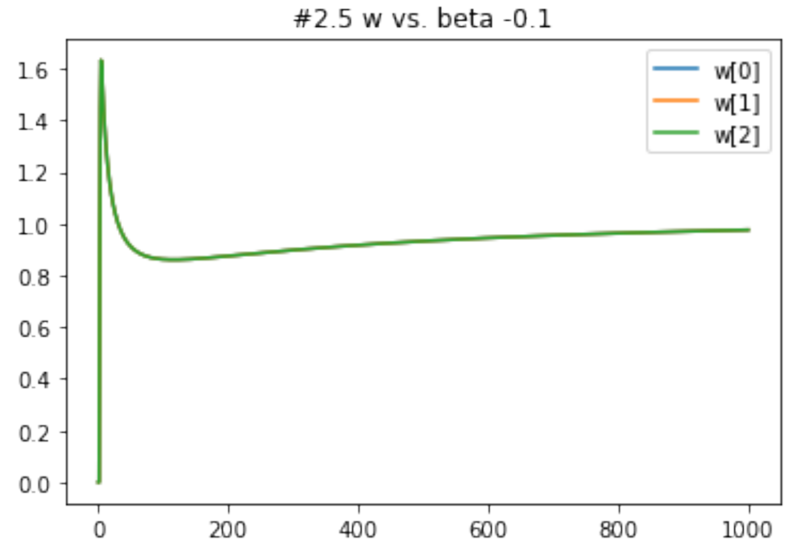
In [40]:
```
def Nest_GDA(func_w, func_a, eta_c, w, a, maxiter,c,beta):
    w_list = []
    w_list.append(w)
    alpha_list = []
    alpha_list.append(a)
    prev_w = w
    prev_a = a
    for t in range(maxiter):


        if eta_c =='1/t':
            eta = 1/(t+1)

        '''
        if t == 0:
            continue
        '''

        w_tuta = w + beta*(w-prev_w)
        a_tuta = a + beta*(a-prev_a)
        prev_w = w
        prev_a = a

        fg_w = func_w(w_tuta,a_tuta)
        fg_alpha = func_a(w_tuta,a_tuta)
        w = w_tuta-eta*fg_w


        N = a_tuta + eta*fg_alpha
        a = Proj(N,c)



        w_list.append(w)
        alpha_list.append(a)



    return w_list, alpha_list
```

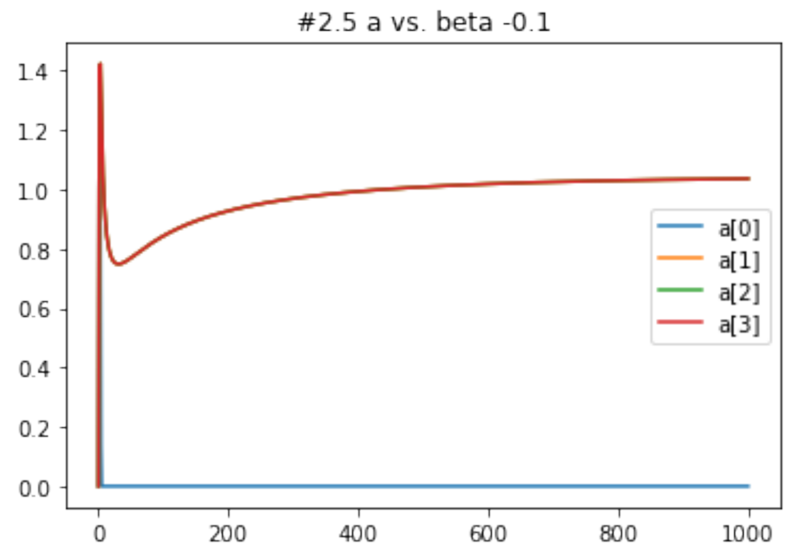
In [44]:
```
w1, a1 = Nest_GDA(func_w, func_alpha, '1/t', init_w, init_alpha, 1000,
draw3(w1,a1,1)
```





In [45]:
```
w1, a1 = Nest_GDA(func_w, func_alpha, '1/t', init_w, init_alpha, 1000,
draw3(w1,a1,2)
```

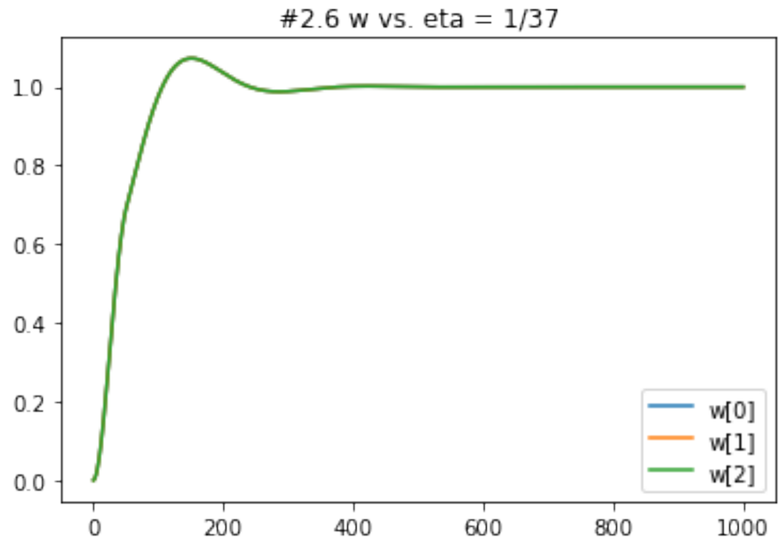


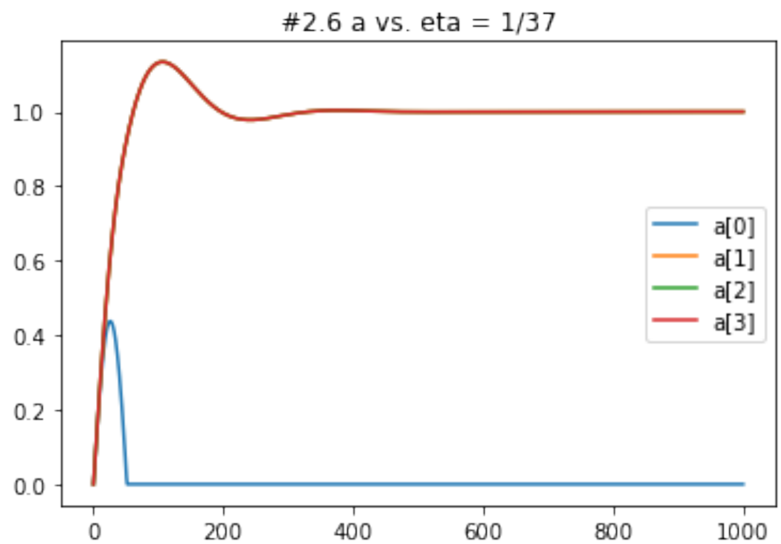


In [36]:
```
def Extra_GDA(func_w, func_a, eta_c, w, a, maxiter,c,beta):
    w_list = []
    w_list.append(w)
    alpha_list = []
    alpha_list.append(a)

    for t in range(maxiter):


        if eta_c =='1/t':
            eta = 1/37
        else:
            eta = 1
            p = 0
```

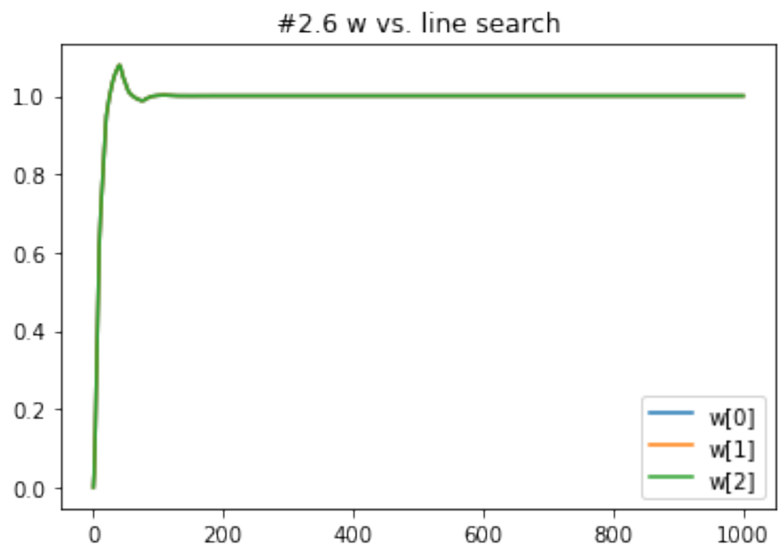
```
    .
    fg_w = func_w(w,a)
    fg_alpha = func_a(w,a)

    while(2*eta > p):


        w_tuta = w - eta*fg_w
        N = a + eta*fg_alpha
        a_tuta = Proj(N,c)

        eta  = eta/2

        fg_w2 = func_w(w_tuta,a_tuta)
        fg_alpha2 = func_a(w_tuta,a_tuta)


        upper = np.linalg.norm(w - w_tuta)**2 + np.linalg.norm
        lower = np.linalg.norm(fg_w - fg_w2)**2 + np.linalg.nc

        upper2 = upper**(1/2)
        lower2 = lower**(1/2)



        p = upper2/lower2/2



fg_w = func_w(w,a)
fg_alpha = func_a(w,a)

w_tuta = w -eta*fg_w


N = a + eta*fg_alpha
a_tuta = Proj(N,c)

fg_w2 = func_w(w_tuta,a_tuta)
fg_alpha2 = func_a(w_tuta,a_tuta)

w = w - eta*fg_w2

N2 = a + eta*fg_alpha2
a = Proj(N,c)


w_list.append(w)
alpha_list.append(a)
```

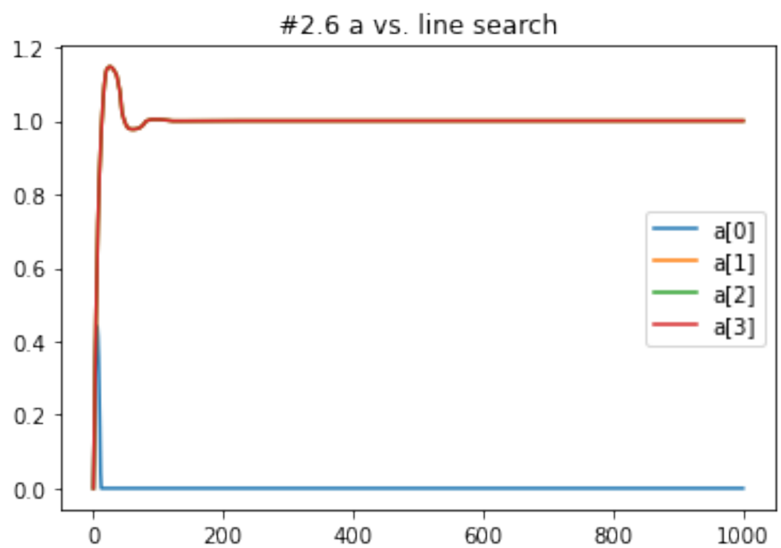
In [38]:
```
w1, a1 = Extra_GDA(func_w, func_alpha, '1/t', init_w, init_alpha, 1000
draw4(w1,a1,1)
```





In [39]:
```
w1, a1 = Extra_GDA(func_w, func_alpha, 'line search', init_w, init_alp
draw4(w1,a1,2)
```





In [ ]:

In [ ]: