Gradient Descent: OLS

Consider the convex problem

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2}\|Ax - b\|_2^2$$

where A is a matrix $m \times n$ and b is vector in $\mathbb{R}^m$. Observe that

$$\nabla f(x) = A^T(Ax - b)$$

and that $\nabla f(\cdot)$ is L-smooth with

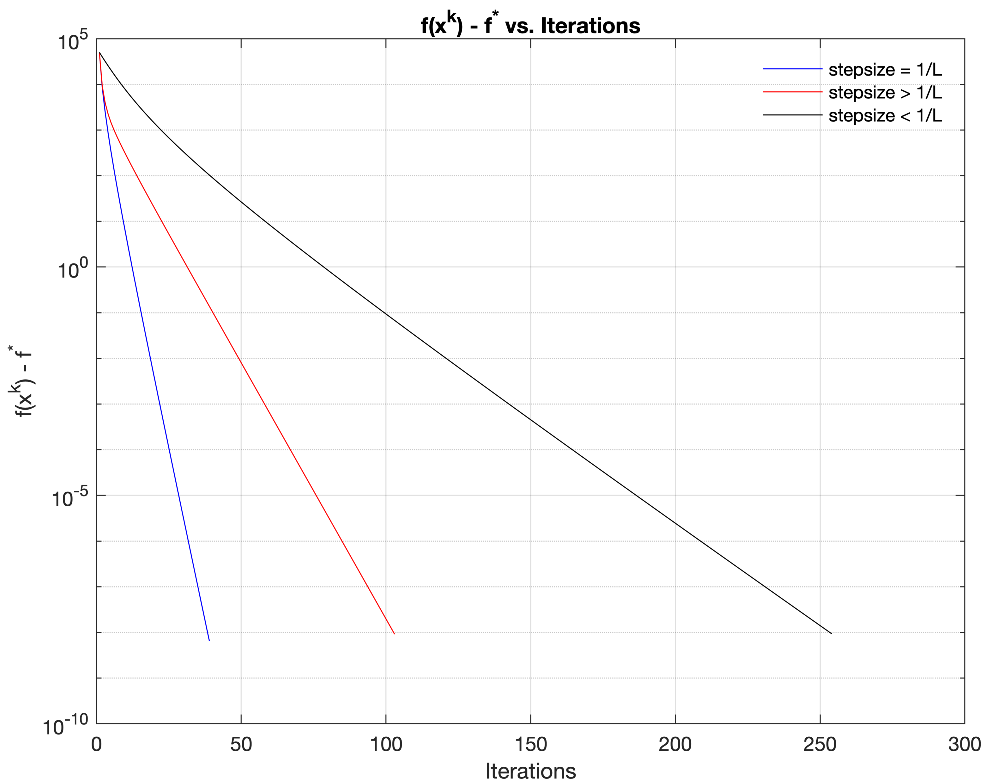
$$L = \|A^T A\|_2 = \sigma_{\max}(A^* A)$$

Therefore, a "good" stepsize selection is $t = 1/L$. However, observe that, depeding on the dimensions of the matrix A, calculating this value might no be an easy task and then a better aproach is to use the backtrack line search (we will not be doing this here, though).

Our goal in the algorithm below is to see the behavior of the GD under ***three different stepsize, all of them constant***.

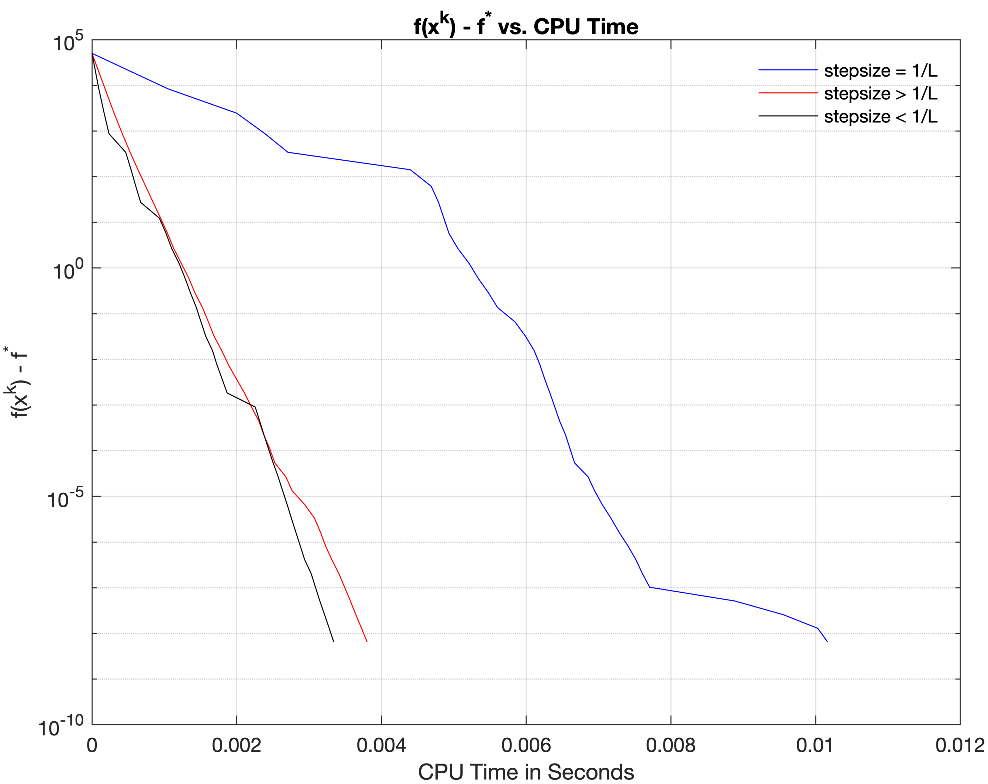
```
m=100;
n=1000;
A=randn(m,n);
xF=randn(n,1);
b=A*xF; %by doing that we guarantee that f^*=0

L =[1/norm(A'*A,2);1.1e-3;1e-4];
tol=1e-8;

maxIt= 1000;
diff = NaN(maxIt,size(L,1));
time = NaN(maxIt,size(L,1));
it   = NaN(1,size(L,1));

xI=zeros(n,1);
gapI=ols(A,xI,b);
for i=1:size(L,1)
    xOld=xI;
    gap=gapI;
    diff(1,i)=gapI;
    time(1,i)=0;
    it(1,i)=1;
    tic;
    while gap>tol
        it(1,i)=it(1,i)+1;
        gradf=transpose(A)*(A*xOld-b);
        xNew=xOld-L(i,1)*gradf;
        gap=ols(A,xNew,b);
        xOld=xNew;
```

```
        diff(it(1,i),i)=gap;
        time(it(1,i),i)=time(it(1,i)-1,i)+toc;
        tic;
    end
end

it1=1:1:it(1,1);it2=1:1:it(1,2);it3=1:1:it(1,3);

semilogy(it1,diff(it1,1),'-b',it2,diff(it2,2),'-r', ...
    it3,diff(it3,3),'-k');
grid on
xlabel('Iterations')
ylabel('f(x^k) - f^*')
legend(['stepsize = 1/L'],['stepsize > 1/L'], ...
    ['stepsize < 1/L'],"location","northeast")
legend("boxoff")
title('f(x^{k}) - f^* vs. Iterations')
```




```
semilogy(time(it1,1),diff(it1,1),'-b',time(it2,2),diff(it2,1),'-r', ...
    time(it3,3),diff(it3,1),'-k');
grid on
xlabel('CPU Time in Seconds')
ylabel('f(x^k) - f^*')
legend(['stepsize = 1/L'],['stepsize > 1/L'], ...
```

```
    ['stepsize < 1/L'],"location","northeast")
legend("boxoff")
title('f(x^{k}) - f^* vs. CPU Time')
```




```
function value = ols(A,x,b)

    value = .5*norm(A*x-b,2)^2;

end
```