How to use `polyline.m`?

- Define an inner function [fout,gout]=f(x) where fout=f(x) is a scalar and gout = grad f(x) is a COLUMN vector
- Ex: for this homework

```
function [z g]= f(x)
    z = (1-x(1))^2+100*(x(2)-x(1)^2)^2;
    g = [400*x(1)^3+2*x(1)-400*x(1)*x(2)-2
         200*x(2)-200*x(1)^2];
end
```

How to use `polyline.m`?

- When calling polyline,

```
[x idid] = polyline(x,fk,gk,pk,ft,@f,10);
if(idid == -1) % error handling
```

- x, gk, pk are vectors; fk, ft are scalar
  - ft = f(x+α_0*pk).
  - You can set α_0=1 or other value 0< α_0=<1. See some difference results.
  - If you use α_0=min(1,1/(1+norm(gout)), you need to explain how it is derived and why it works.

What does `polymod.m` do?

- Use the interpolation method to find min $\phi(\alpha)$=f(x+αp) in the interval [0, lamc].
  - More precisely, in [blow*lamc,bhigh*lamc]
- In the first step, we have $\phi(0)$, $\phi'(0)$, ϕ(lamc)=qc
  - Use quadratic model to find the minimum.
  - Note that lamc=1 in polyline.m
  - So the model function q(α) is

$$q(\alpha) = (\mathrm{qc} - \mathrm{q0} - \mathrm{qp0})\alpha^2 + \mathrm{qp0}\alpha + \mathrm{q0}$$

  - whose extremer is at $\alpha = -\mathrm{qp0}/2(\mathrm{qc} - \mathrm{q0} - \mathrm{qp0})$

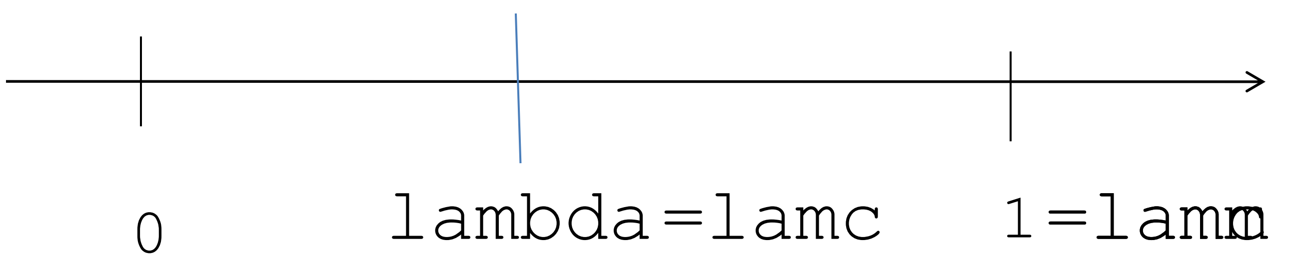
The quadratic model

- But the extremer `lplus` may not be in the interval []=[blow*lamc, bhigh*lamc]
  - The minimum value is on the bounary

```
lleft=lamc*blow; lright=lamc*bhigh;
% quadratic model
%  q(0) = q0, q'(0) = qp0, q(lamc) = qc
lplus =-qp0/(2*lamc*(qc-q0-qp0));
if lplus < lleft lplus = lleft; end
if lplus > lright lplus = lright; end
```

After the first iteration

- we have a new point lambda
- dditional info f(lamm),
  - lamm is got from previous iteration. (=$lamc_{old}$)
  - Use cubic model to find the minimum



The cubic model

```
% cubic model
% q(0) = q0, q'(0) = qp0, q(lamc) = qc,
q(lamm) = qm
a=[lamc^2, lamc^3; lamm^2, lamm^3];
b=[qc;qm]-[q0+qp0*lamc;q0+qp0*lamm];
c=a\b;
lplus=(-c(1)+sqrt(c(1)*c(1)-
3*c(2)*qp0))/(3*c(2));
if lplus < lleft lplus = lleft; end
if lplus > lright lplus = lright; end
```

- It’s the homework to figure out what those statements do.