Juan Meriles
ME 280 B
29 April 2002
Assignment 7

Problem 1

Want to minimize centroid of Planar curve $y = y(x)$ of length $2L$ that passes through $(-a, 0)$ and $(a, 0)$

exact sol

$$y(x) = r \cosh\left(\frac{x}{r}\right) - \sqrt{r^2 + L^2} \qquad r \sinh\left(\frac{a}{r}\right) = L$$

Approx sol with N finite elements of $L = \frac{2L}{N}$

i) Write explicit minimization function F and constraint function C in terms of coords of nodal points

$$F = \frac{\int_e y\, ds}{\int ds} = \frac{\int_c y\, ds}{2L}$$ if we only look at half by symmetry

$$= \frac{\int_{c_2} y\, ds}{L}$$

if we use Fea formulation

$$F^e = \frac{1}{L}\int_0^1 \left(1-\xi\right) y_0 + \left(\xi\right) y_1 \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}\, d\xi$$

$$= \frac{1}{L}\int_0^1 \left(y_0 + (y_1 - y_0)\xi\right) \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}\, d\xi$$

$$= \frac{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}}{L} \cdot \frac{(y_1 + y_0)}{2}$$

The constraint vector is that each element has length $\frac{L}{N}$

$$c^e = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} - \frac{L}{N}$$

ii) Use Lagrange multiplier formulation + NR to solve constrained minimization in part (i) for $a=1$ $L=4$ $N=8, 16, 32, 64$

Lagrange multiplier formulation

$$F_L = F + \Lambda c$$

We can write $F = \sum F^e$ $\Lambda c = \sum \lambda^e c^e$

to minimize this → derivative of F_L with respect to (x, y) coords $=0$ and derivative of F_L with respect to $\lambda = 0$

We then have

$$DF_L(u, \xi) + DF_L(\lambda, \theta) = 0$$
$$g(u) = 0$$

We can write system as

$$\begin{bmatrix} k(\hat{u}^{(k)}) + D^T d(\hat{u}^{(k)})\hat{\lambda}^{(k)} & d(\hat{u}^{(k)}) \\ d^T(\hat{u}^{(k)}) & 0 \end{bmatrix} \begin{bmatrix} \Delta\hat{u}^{(k)} \\ \Delta\hat{\lambda}^{(k)} \end{bmatrix} = -\begin{bmatrix} F(\hat{u}^{(k)}) + d(\hat{u}^{(k)})\lambda \\ g(\hat{u}^{(k)}) \end{bmatrix}$$

Where $k(\hat{u}^{(k)}) = D^2 F_L(u, \xi)$

$d(\hat{u}^{(k)}) = DF_L(\lambda, \theta)$

$F(\hat{u}^{(k)}) = DF_L(u, \xi) + DF_L(\lambda, \theta)$

$g(\hat{u}^{(k)}) = c(\hat{u}^{(k)})$

$$f^e = \frac{1}{2L\cdot\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}}\begin{bmatrix} -(x_1-x_0)(y_0+y_1) \\ (x_0-x_1)^2+2y_0(y_0-y_1) \\ (x_1-x_0)(y_0+y_1) \\ (x_0-x_1)^2+2y_1(y_1-y_0)\end{bmatrix}$$

found through wolfram alpha

$$K^e = C\begin{bmatrix} (x_0-x_1)^2(y_0+y_1) & (x_0-x_1)((x_0-x_1)^2+2y_1(y_1-y_0)) & (y_0-y_1)^2(y_0+y_1) & (x_0-x_1)((x_0-x_1)^2+2y_0(y_0-y_1)) \\ & x_0^2(3y_0-y_1)+2x_0x_1(y_1-3y_0)+x_1^2(3y_0-y_1)+2(y_0-y_1)^2 & (x_0-x_1)((x_0-x_1)^2+2y_1(y_1-y_0)) & -(x_0-x_1)^2(y_0+y_1) \\ & & (y_0-y_1)^2(y_0+y_1) & -(x_0-x_1)((x_0-x_1)^2+2y_0(y_0-y_1)) \\ \text{Sym} & & & -x_0^2(y_0-3y_1)+2x_0x_1(y_0-3y_1)-x_1^2(y_0-3y_1)-2(y_0-y_1)^3\end{bmatrix}$$

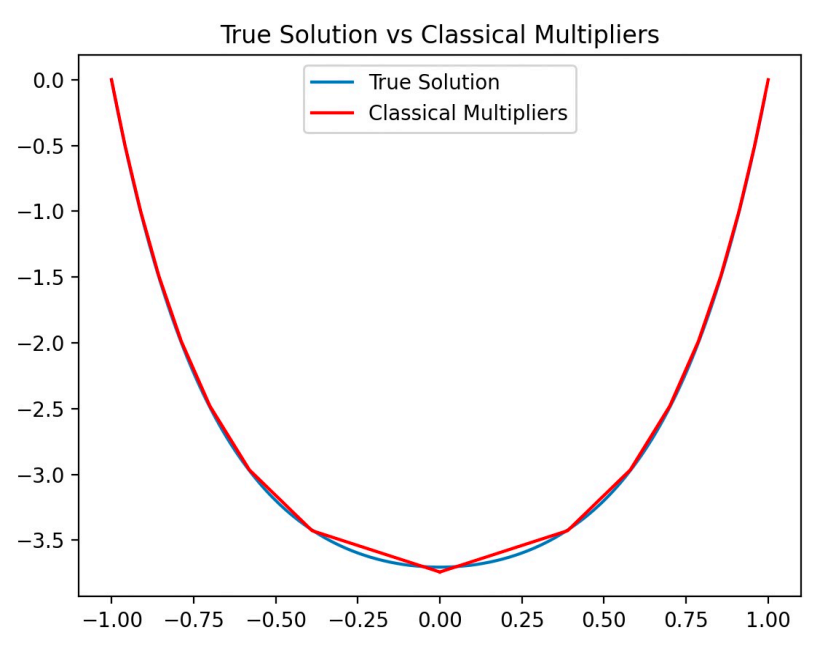
$$C = \frac{1}{2L\cdot\left((x_1-x_0)^2+(y_1-y_0)^2\right)^{3/2}}$$

$$dJ^e = \frac{1}{\sqrt{(x_1-x_0)^2+(y_1-y_0)^2}}\begin{bmatrix} (x_0-x_1) \\ (y_0-y_1) \\ (x_1-x_0) \\ (y_1-y_0)\end{bmatrix}$$

$$DJ^e = C_2\begin{bmatrix} (y_0-y_1)^2 & -(x_0-x_1)(y_0-y_1) & -(y_0-y_1)^2 & (x_0-x_1)(y_0-y_1) \\ & (x_0-x_1)^2 & (x_0-x_1)(y_0-y_1) & -(x_0-x_1)^2 \\ & & (y_1-y_0)^2 & (x_1-x_0)(y_0-y_1) \\ \text{Sym} & & & (x_1-x_0)^2\end{bmatrix}$$

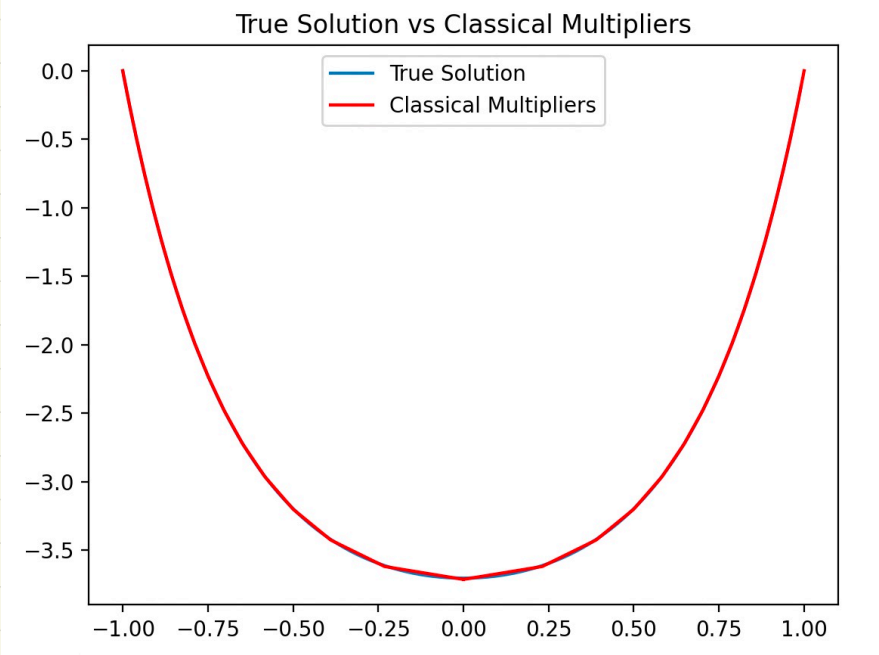
$$C_2 = \frac{1}{\left((x_1-x_0)^2+(y_1-y_0)^2\right)^{3/2}}$$

Ne = 8



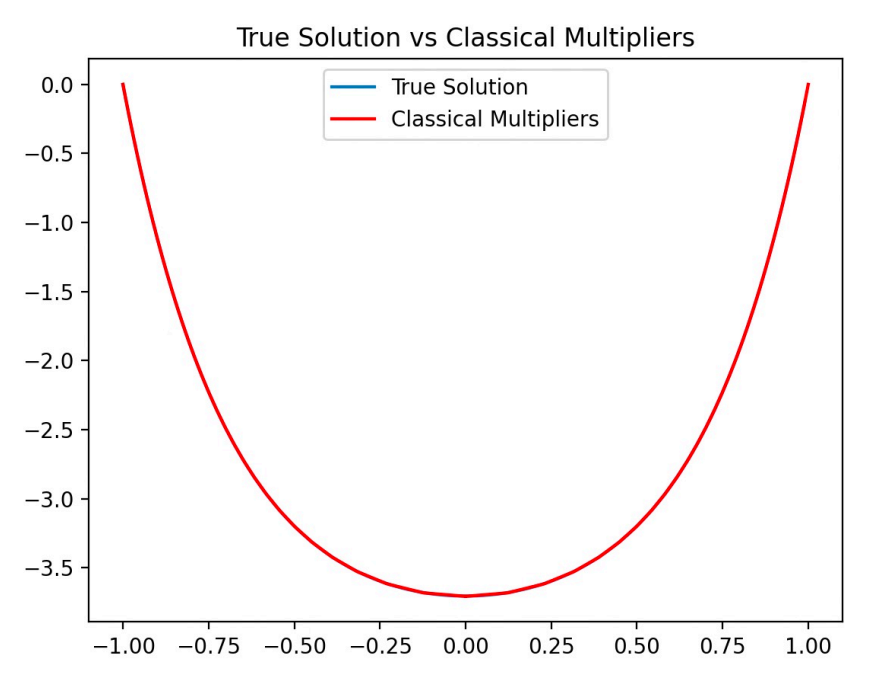
Error is 1.9504436526828772e-08 in 7 iterations

Ne = 16



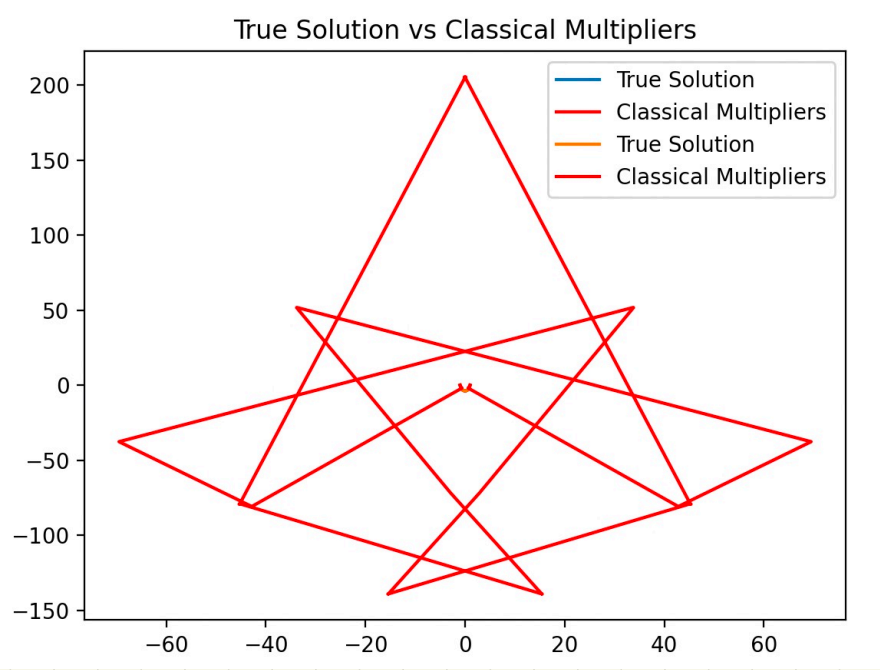
Error is 9.835181945083016e-11 in 9 iterations

Ne = 32



Error is 1.6090116495313247e-08 in 11 iterations

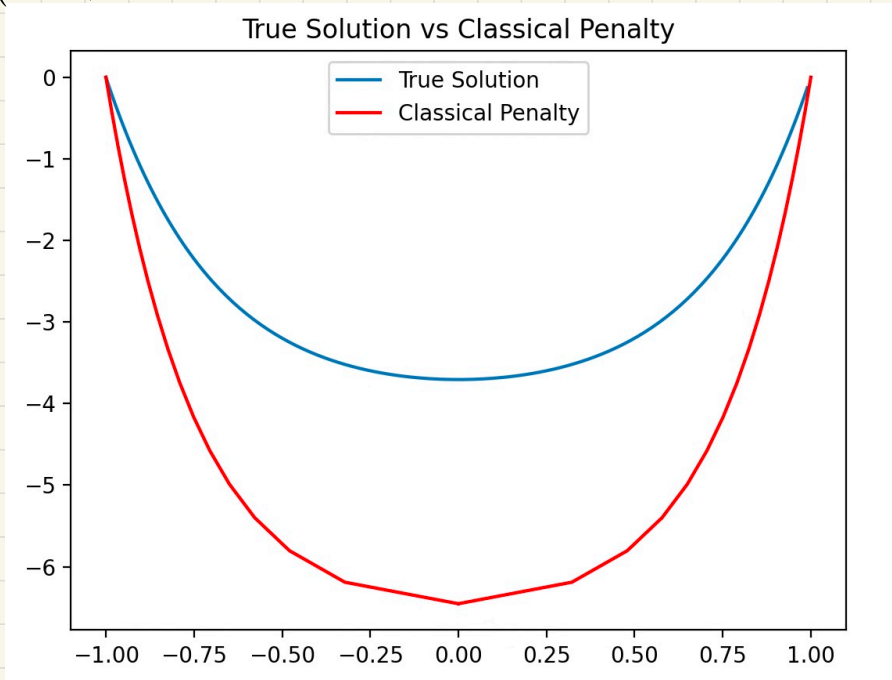
Ne = 64



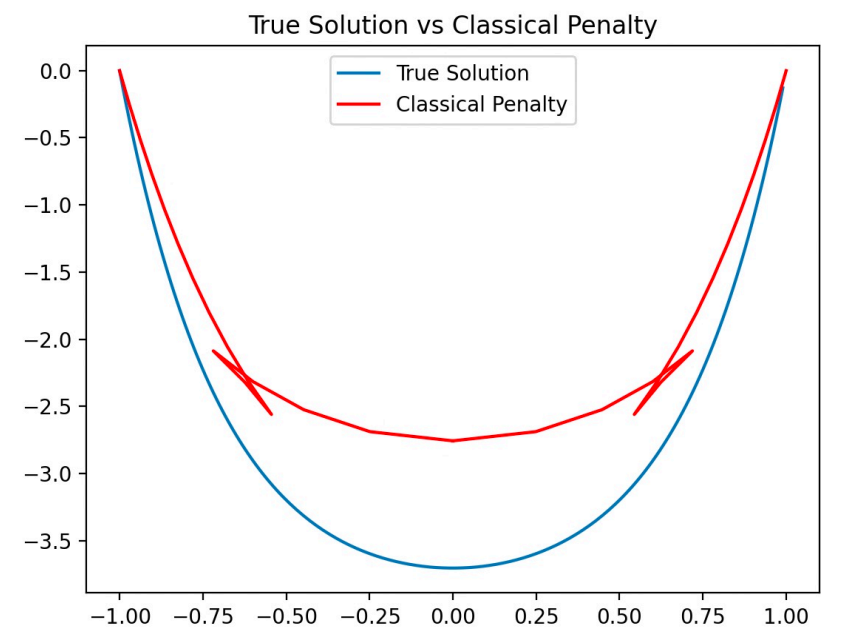
The solution Diverged

It is clear that as more elements are added, the solution improves, but once there are too many elements it begins to fail. This may be due to instability of the matrices built, or due to local minima that appear from having too many degrees of freedom.

ii) Classical Penalty

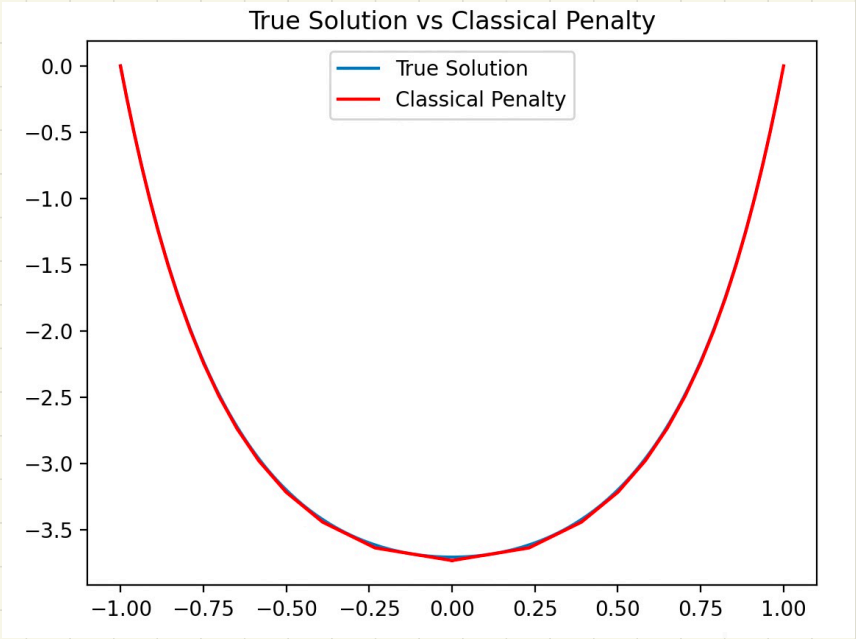


$\epsilon = 10$



$\epsilon = 100$

$\varepsilon = 1000$



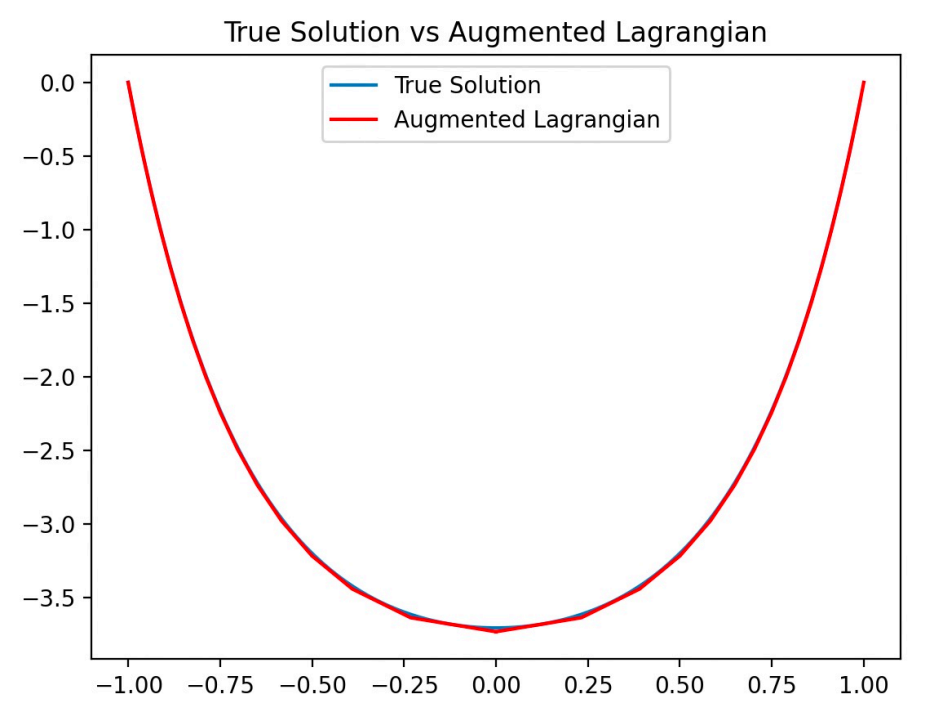
This penalty was implemented by using

$$A\Delta u = -b \quad \text{where} \quad A = K + \varepsilon\left(D^T d(\hat{u}^{(k)}) g(\hat{u}^{(k)}) + d\, d^T\right)$$

$$b = -f - \varepsilon\, d\, g(\hat{u}^{(k)})$$

As the penalty increases, so does the quality of the solution. This is because the penalty enforces the boundary condition of the element lengths more stringently as ε increases

iv)



The augmented lagrangian was implemented for 16 elements with the Uzawa algorithm

In this case

$$A = K + D^T d \Lambda \varepsilon g + \varepsilon d d^T$$

$$b = -(f + \varepsilon d \cdot g)$$

and $\varepsilon = 10^3$ and $\xi = 10^3$

$$\lambda^{(k)} = \xi g^{(k)}$$

The augmented lagrangian worked quite well. I tried it with various ε and noted that it works better

than the classical penalty method for lower ϵ. With this said, convergence seems to take significantly more iterations for this solution than either the extended system with lagrange multipliers or the classical penalty method.