10.34 – Fall 2006

Homework #3 - Solutions

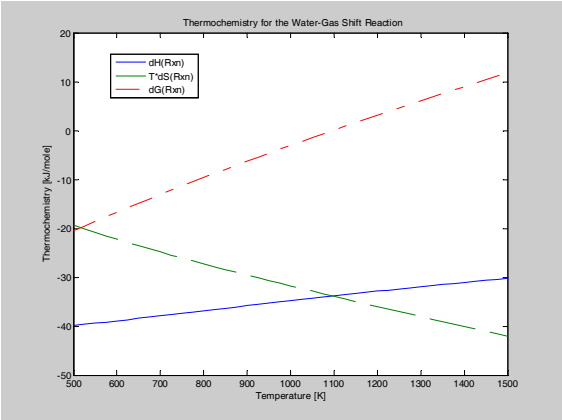
Problem 1: Peak Temperature

In this problem, there are two situations that we are concerned with: 100% conversion and equilibrium conversion. The 100% case is relatively easy to solve, since one only needs to solve the enthalpy with all of the CO being converted to products (it is easy because the outlet flow rates are known a priori. The equilibrium case is slightly more complicated because one must also solve the equilibrium condition:

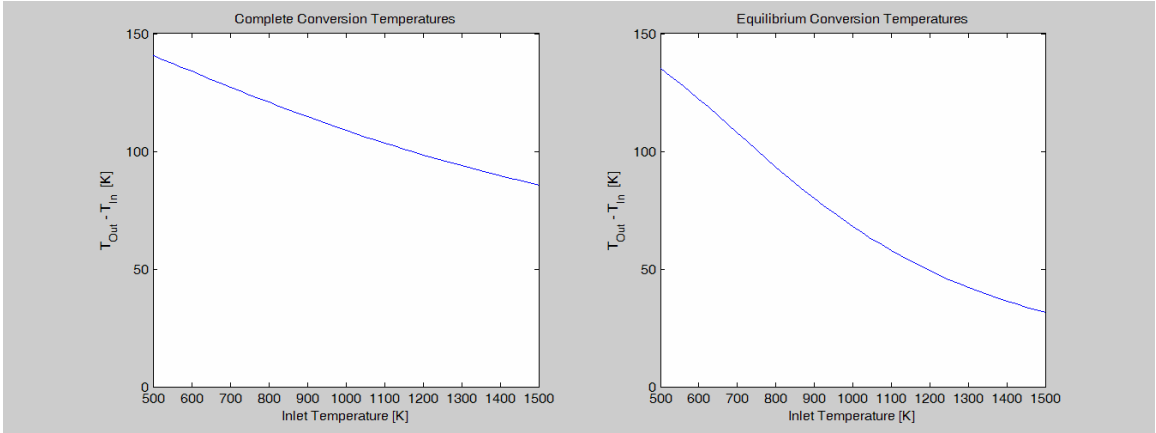
$$K_A = \exp\left(\frac{-\Delta G_{Rxn}}{RT}\right) = K_C \cdot \left(\frac{P_0}{RT}\right)^{-\Delta n} \quad \Rightarrow \quad \exp\left(\frac{-\Delta G_{Rxn}}{RT}\right) = \frac{[H_2][CO_2]}{[CO][H_2O]}$$

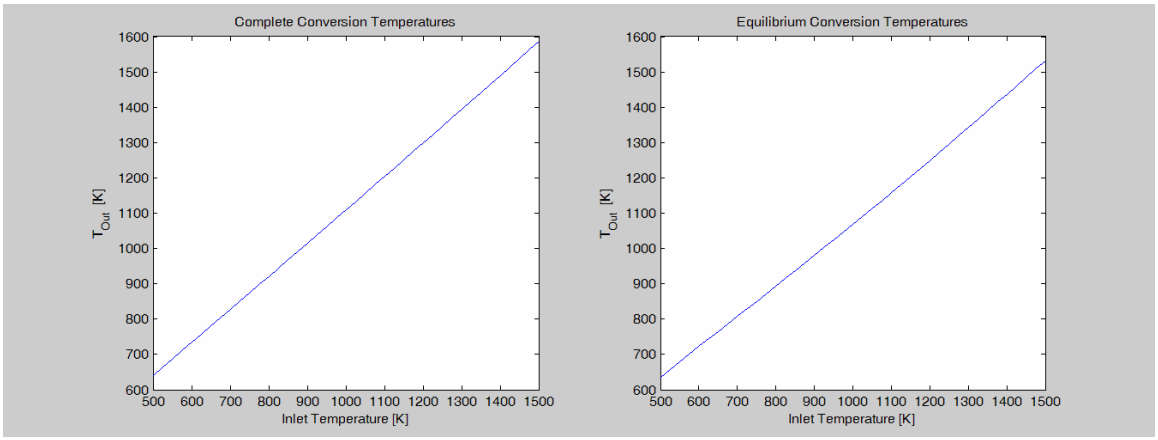
Since are dealing with an equimolar reaction, the dN term goes away. We can also treat the concentrations as molar flow rates in this case, since the volume terms in the concentrations would also cancel. Note that dG is temperature dependent. As was done previously, the molar flow rates (concentrations) of the species can be written in terms of a single extent of reaction variable, so we are only left with T and extent as variables.

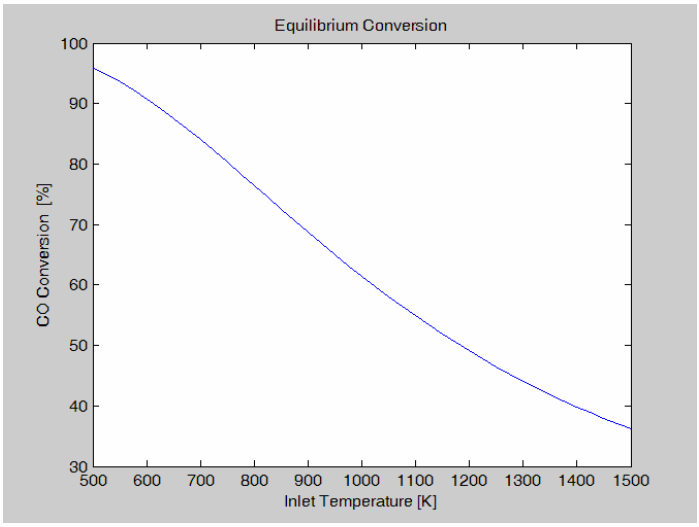
Part A:



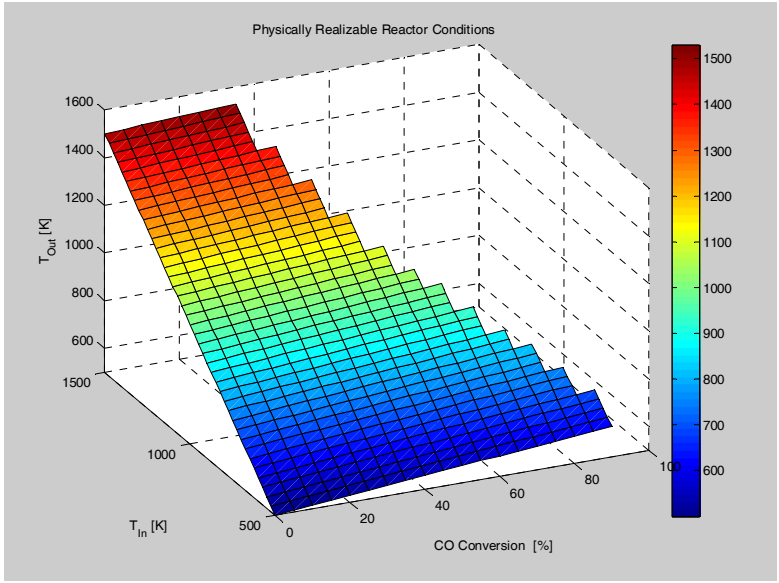
Parts B and C:







Part D:



Problem 2: Finite Differences

A. The problem is almost completely formulated in the problem statement. The value of heat supplied by the laser to the water is given by

$H = \frac{0.1 I_0}{w}$ (10% of laser is absorbed uniformly across the width w).

Now we can write the system of equations in the matrix form $AT = b$, where A is the coefficient matrix and T is the vector of temperatures.

The rule for making the matrix A and vector b is the following

A(1,1) = 1; b(1) = Ta;

A(i-1,i) = 1; A(i,i) = -2; A(i+1,i) = 1; b(i) = $-\frac{0.1 \times I_0 (\Delta y)^2}{kw}$

A(n+1,n+1) =1; b(n+1) = Ta;

Please look at the commented code Problem3a for the actual code.

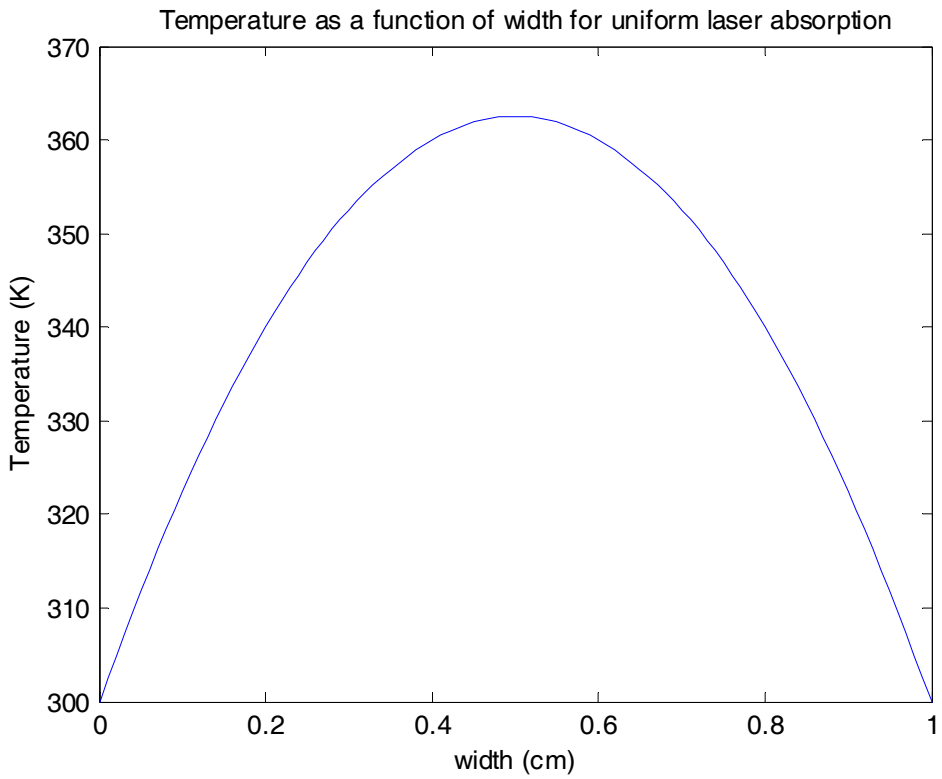
Once we make the A and b, we can solve for T using the back-slash command

$T = A \backslash b$.

The sample output is given below

```
>> u=problem2a;
The maximum temperature is T =362.5 K
The y-value at which temperature is maximum =0.005 cm
>>
```

The graph of temperature v/s width is



B. This problem is no longer linear. The temperature and intensity of light are related to each other and we have to know one before we can solve for the other. Let us make a vector *u* which holds the intensity of light and temperatures along the cross-section. If there are *n+1* points, then there are *n+1* uknown intensities and also *n+1* unknown temperatures. Thus the vector of *u* is of length 2*(*n+1)*.
In the solution u is defined as

$$u = \begin{bmatrix} I_1 \\ I_1 \\ \bullet \\ \bullet \\ I_{n+1} \\ T_1 \\ T_2 \\ \bullet \\ \bullet \\ T_{n+1} \end{bmatrix}$$

As mentioned in the problem the Temperature and Intensities have to be discretized.

Thus for our problem we have the following set of equations

Equations 1 to Equation n+1 are given below and describe the descretized equations for Intensity.

$u_1 - I_0 = 0$

$u_2 - u_1 - \varepsilon(u_{n+2}) \times u_2 \times \Delta y = 0$ (u_{n+2} is Temperature at grid point 2)

•

•

$u_{n+1} - u_n - \varepsilon(u_{2\times n+2}) \times u_{n+1} \times \Delta y = 0$

Equations *n+2* to Equation *2×(n+1)* are given below

$u_{n+2} - T_a = 0$

$u_{n+2} - 2u_{n+3} + u_{n+4} - \frac{\varepsilon(u_{n+3})u_2(\Delta y)^2}{k} = 0$

•

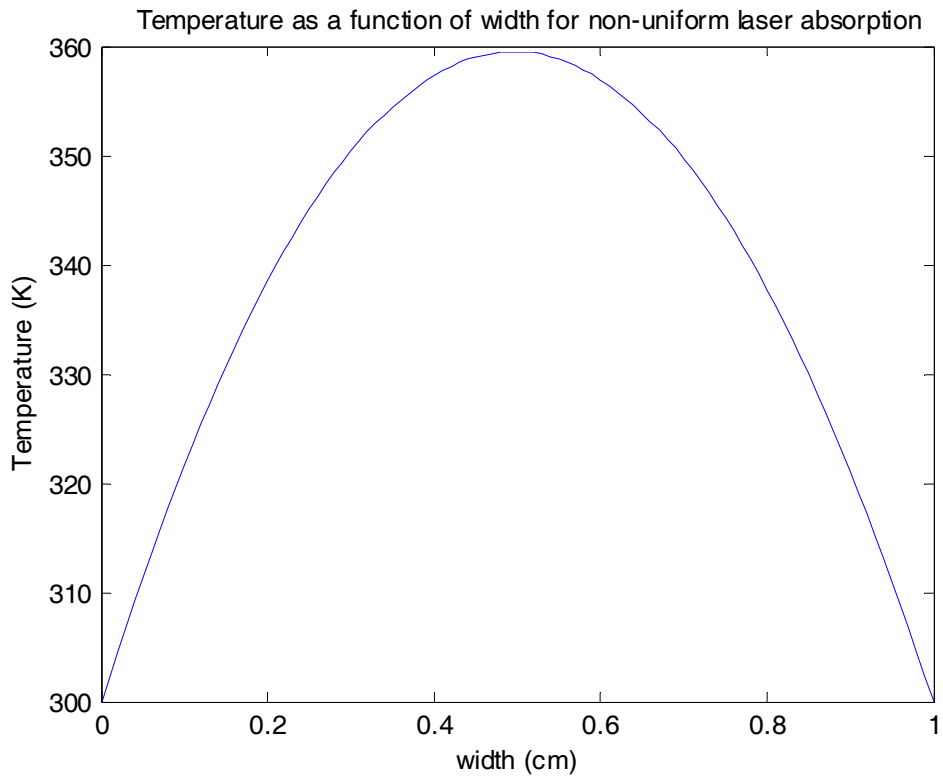
•

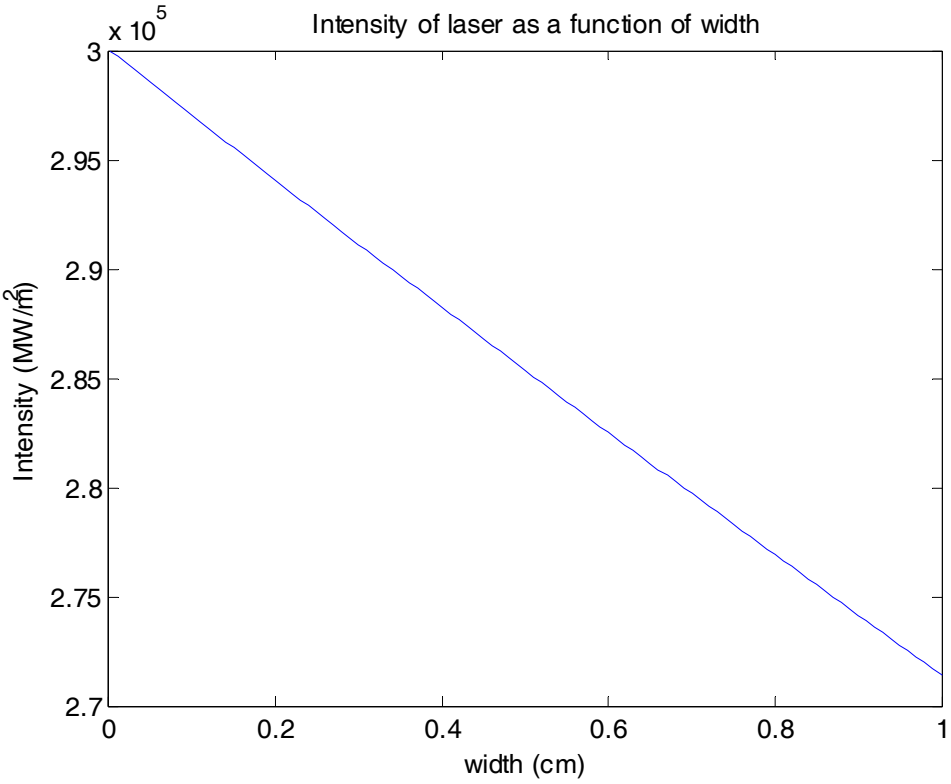
$u_{2\times n} - 2u_{2\times n+1} + u_{2\times n+2} - \frac{\varepsilon(u_{2\times n+1})u_n(\Delta y)^2}{k} = 0$

$u_{2\times n+2} - T_a = 0$

These equations are a set of $2 \times$ n+2 non-linear equations and can be solved using fsolve. The initial conditions for intensity is a vector of length n+1 with all the elements equal to I_o. The initial condition for T is a vector of lenth n+1 with all the elements equal to T_a. Look at the code problem2b to see the exact implementation of the solution.

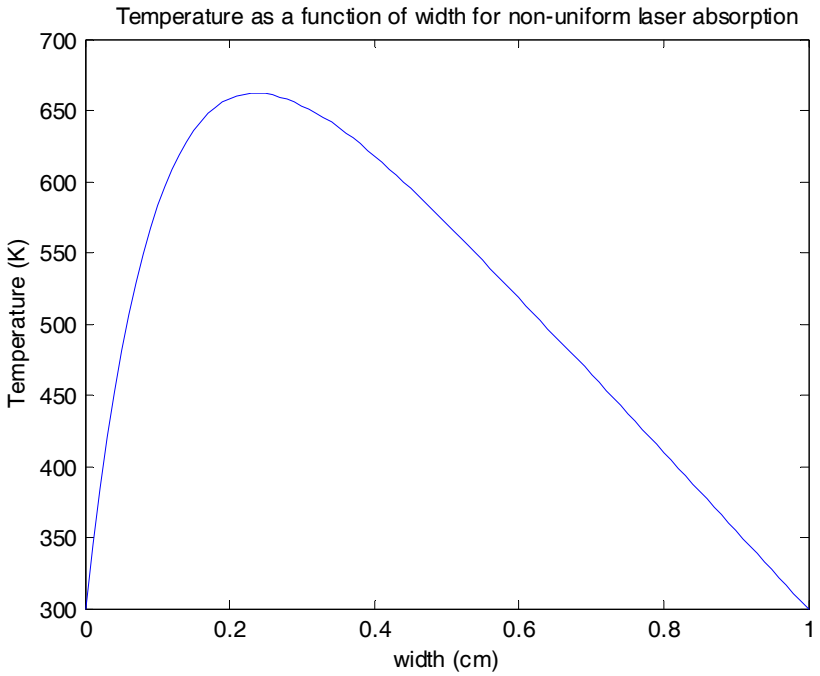
1) For the first problem the sample input and output is as follows.
   >> problem2b(100,-10,-1e-7);
   Optimization terminated: first-order optimality is less than options.TolFun.
   The maximum temperature is T =359.535 K
   The y-value at which temperature is maximum =0.5 cm

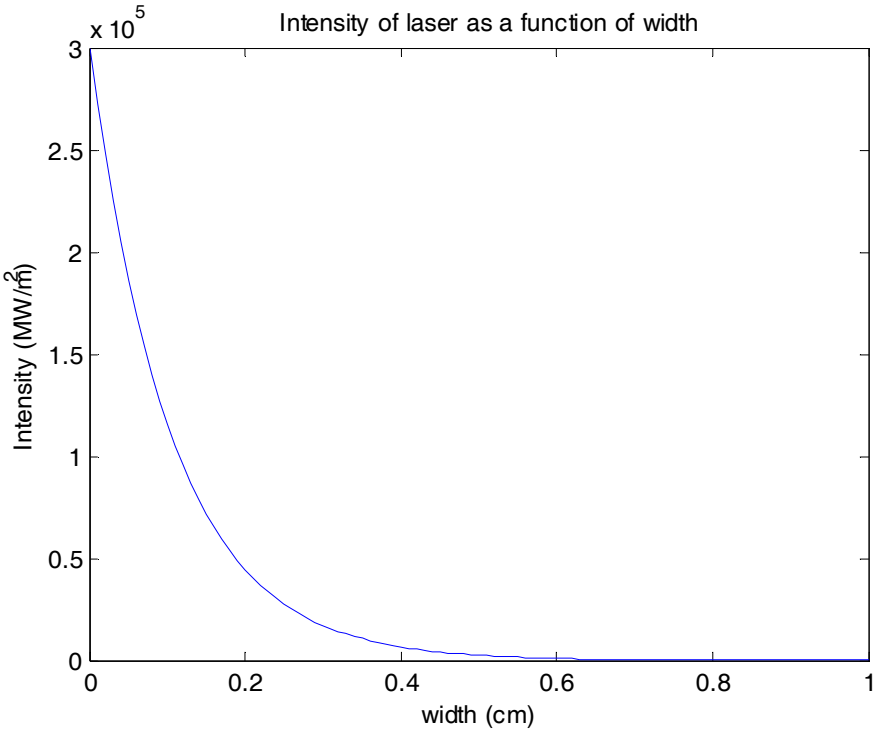




2) For the second problem the sample input and output is

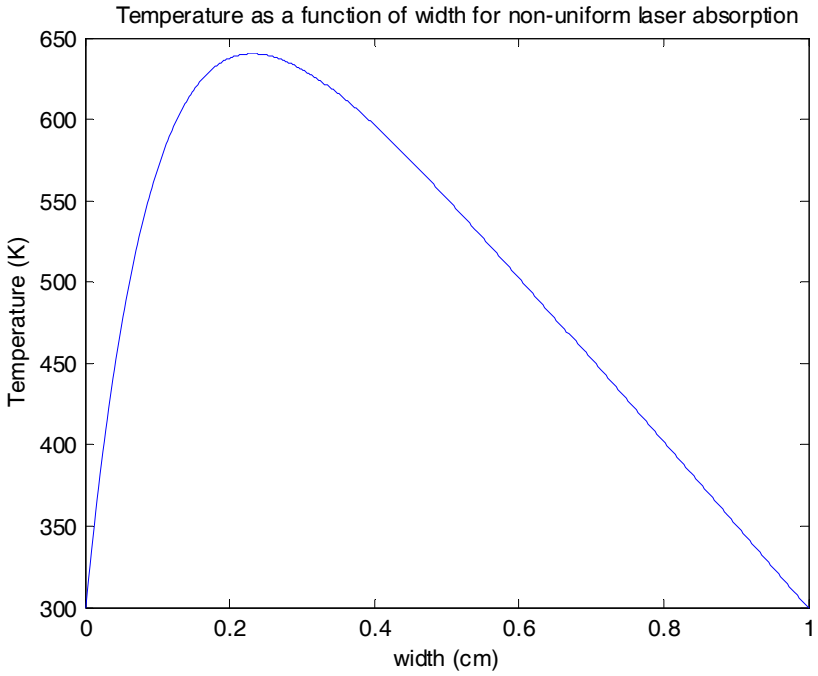
```
>> problem2b(100,-1000,-1e-7);
Optimization terminated: first-order optimality is less than options.TolFun.
The maximum temperature is T =662.1657 K
The y-value at which temperature is maximum =0.24 cm
>>
```

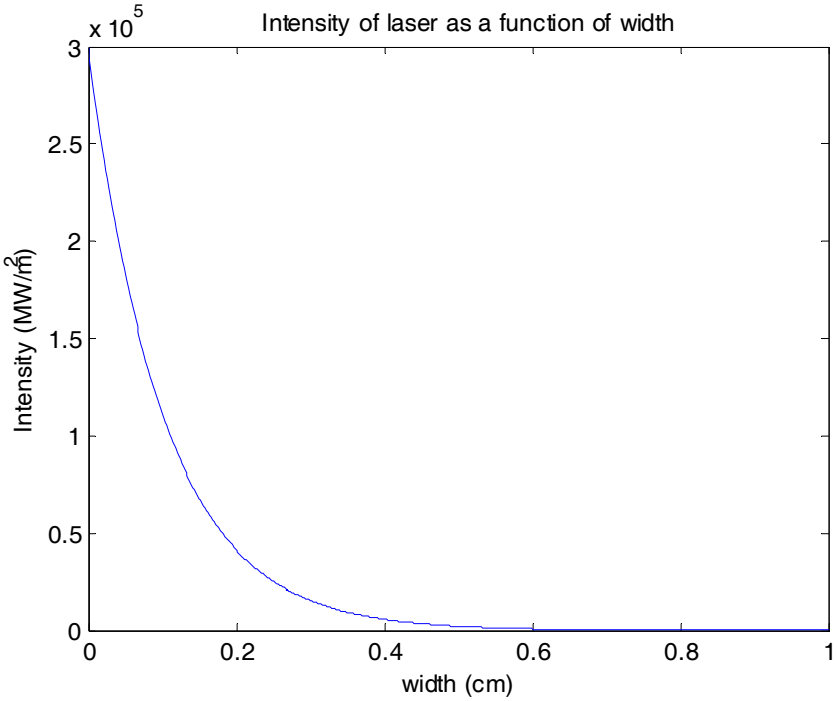




3) The sample input and output for part 3 is

```
>> problem2b(500,-1000,-1e-7);
Optimization terminated: first-order optimality is less than options.TolFun.
The maximum temperature is T =640.401 K
The y-value at which temperature is maximum =0.232 cm
>>
```





There is definitely a difference between the temperature profiles obtained in problem 2 and problem 3. This basically means that for this problem the discretization of n=100 points is not enough to get a converged solution. As the amount of discretization increases, the solution has a better chance of converging to the actual physical solution. In most cases it is a good idea to solve the problem with varying degrees of discretization and use the solution which remains unchanged with increasing discretization. If we were to solve the same problem with n=800, we will realize that the solution obtained at n=800 is very similar to the one obtained at n=500.

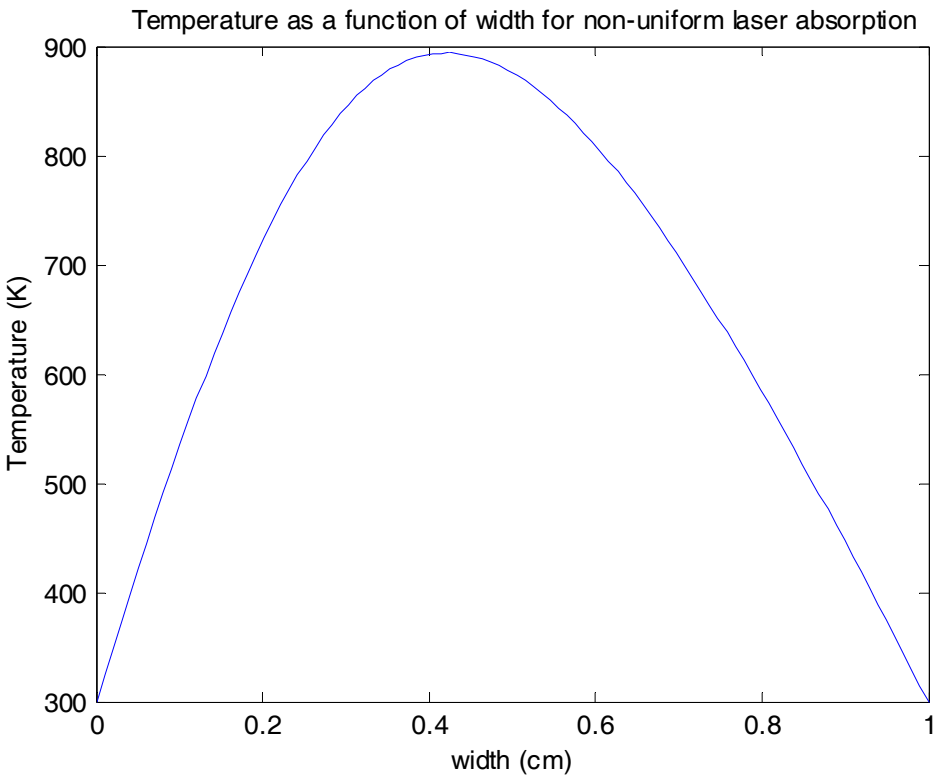
C. This problem can ideally be solved with the same solver we have used in the previous problem. But the difficult part is getting a good initial guess. My solver is not able to solve the problem with the basic initial guess of all T are equal to T_a and all I are I_0. The problem with this problem is the large value of the coefficient of T^2 (lets call it α). To circumvent this problem we make a matrix of α as shown below

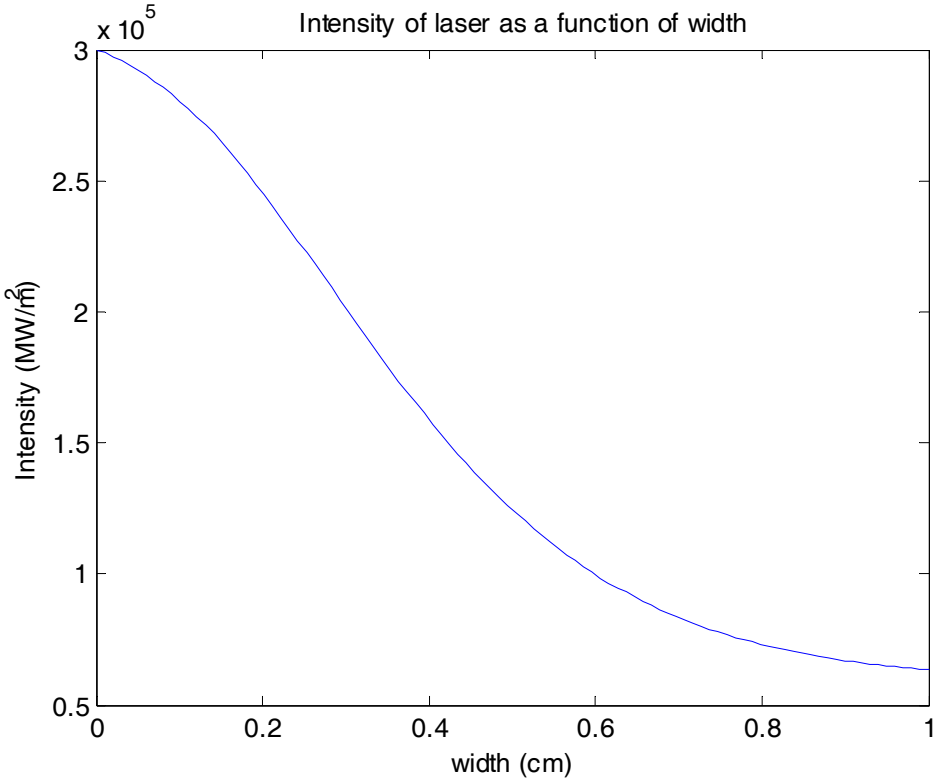
α =[-1e-7; -1e-6; -1e-5; -7e-5; -1e-4; -2e-4; -1e-4; -3e-4]

For the first value of α calculate a profile of temperature and Intensity. For the subsequent value of α , use the profiles of temperature and intensity from the previous α as the initial guess. For my program this strategy works and gives the correct answer.

The sample input and output from the code is

```
>> problem2c(100);
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
Optimization terminated: first-order optimality is less than options.TolFun.
The maximum temperature is T =894.0141 K
The y-value at which temperature is maximum =0.42424 cm
>>
```





Problem 3: Options for Broyden's Method

Part A:

In this part we are charged with proving the following equation:

$$\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)^{-1}=\underline{\underline{A}}^{-1}-\gamma\left(\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{A}}^{-1}\right)$$

We can start by right-multiplying by $\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)$:

$$\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)^{-1}\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)=\underline{\underline{A}}^{-1}\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)-\gamma\left(\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{A}}^{-1}\right)\left(\underline{\underline{A}}+\underline{u}\underline{v}^T\right)$$

This will simplify initially to the following by multiplying out the terms:

$$\underline{\underline{I}}=\underline{\underline{A}}^{-1}\underline{\underline{A}}+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T-\gamma\left(\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{A}}^{-1}\underline{\underline{A}}\right)-\gamma\left(\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\right)$$

This will simplify to this, writing out gamma explicitly:

$$\underline{\underline{I}}=\underline{\underline{I}}+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T-\frac{\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{I}}+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T}{1+\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}}$$

If you realize that $\underline{v}^T\underline{\underline{A}}^{-1}\_$ is a constant, you can rewrite the above as:

$$\underline{\underline{I}}=\underline{\underline{I}}+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T-\frac{\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\left(\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}\right)}{1+\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}}$$

Now factor the numerator to get:

$$\underline{\underline{I}}=\underline{\underline{I}}+\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T-\underline{\underline{A}}^{-1}\underline{u}\underline{v}^T\left(\frac{1+\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}}{1+\underline{v}^T\underline{\underline{A}}^{-1}\underline{u}}\right)$$

You can easily see that this will simplify to:

$$\underline{\underline{I}}=\underline{\underline{I}}$$

Therefore, the equation we were trying to prove must be correct.

Part B:

We want to be able to update the new inverse Broyden matrix, using the old one.

We can find the inverse update formula by starting with the normal update formula and utilizing the equation we proved in part A.

$$\underline{\underline{B}}\left(\underline{x}_{n+1}\right)=\underline{\underline{B}}\left(\underline{x}_n\right)+\chi\underline{F}\left(\underline{x}_{n+1}\right)\cdot\left(\underline{x}_{n+1}-\underline{x}_n\right)^T$$

If we take the inverse of both sides of the above, we are left with:

$$\underline{\underline{B}}^{-1}\left(\underline{x}_{n+1}\right)=\left(\underline{\underline{B}}\left(\underline{x}_n\right)+\chi\underline{F}\left(\underline{x}_{n+1}\right)\cdot\left(\underline{x}_{n+1}-\underline{x}_n\right)^T\right)^{-1}$$

Taking $\Delta x_n^T \equiv \left(\underline{x}_{n+1} - \underline{x}_n\right)^T$ and comparing the above with Eqn. 5 from the problem statement, we can see that the right hand side is that same form, with:

$$\underline{u} = \chi \underline{F}\left(\underline{x}_{n+1}\right) \quad and \quad \underline{v}^T = \underline{\Delta x}_n^T \quad and \quad \underline{\underline{A}} = \underline{\underline{B}}\left(\underline{x}_n\right)$$

So we can simply rewrite equation 5 with the appropriate terms substituted into the

$$\boxed{\underline{\underline{B}}^{-1}\left(\underline{x}_{n+1}\right) = \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) - \frac{\underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right) \cdot \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}}$$

Part C:

In order to solve this part, we have to start with the normal formula for determining the step size, where: $\Delta x_n^T \equiv \left(\underline{x}_{n+1} - \underline{x}_n\right)^T \quad and \quad \Delta x_{n+1}^T \equiv \left(\underline{x}_{n+2} - \underline{x}_{n+1}\right)^T$.

$$\underline{\Delta x}_{n+1} \equiv -\underline{\underline{B}}^{-1}\left(x_{n+1}\right) \cdot \underline{F}\left(x_{n+1}\right)$$

However, we know that calculating the inverse is expensive and we would like to avoid it, which is the whole idea of this procedure. Start by expanding the inverse:

$$\underline{\Delta x}_{n+1} \equiv -\left[\underline{\underline{B}}^{-1}\left(\underline{x}_n\right) - \frac{\underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right) \cdot \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}\right] \cdot \underline{F}\left(x_{n+1}\right)$$

Now we can left-multiply by $B(x_n)$, and distribute the $F(x_n)$:

$$\underline{\underline{B}}\left(\underline{x}_n\right)\underline{\Delta x}_{n+1} \equiv -\left[\underline{\underline{B}}\left(\underline{x}_n\right)\underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right) - \frac{\underline{\underline{B}}\left(\underline{x}_n\right)\underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right) \cdot \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}\right]$$

Taking care of the $B*B^{-1}$ terms, we are left with:

$$\underline{\underline{B}}\left(\underline{x}_n\right)\underline{\Delta x}_{n+1} \equiv -\left[\underline{F}\left(x_{n+1}\right) - \frac{\underline{F}\left(\underline{x}_{n+1}\right) \cdot \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}\right]$$

As was mentioned before, $\underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right)$ is a scalar, allowing for:

$$\underline{\underline{B}}\left(\underline{x}_n\right)\underline{\Delta x}_{n+1} \equiv -\frac{1}{\lambda}\underline{F}\left(x_{n+1}\right) \qquad where: \lambda = \frac{1}{1 - \dfrac{\underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}}$$

Rearranging yields the following expression, which looks almost like the step update:

$$\lambda \underline{\underline{B}}\left(\underline{x}_n\right)\underline{\Delta x}_{n+1} \equiv -\underline{F}\left(x_{n+1}\right)$$

However, if we take the LU decomposition of $B(x_n)$, we are left with:

$$\lambda \underline{\underline{L}}\left(x_n\right)\underline{\underline{U}}\left(x_n\right) \cdot \underline{\Delta x}_{n+1} \equiv -\underline{F}\left(x_{n+1}\right)$$

Well, we now have the needed expression for lambda:

$$\lambda = \frac{1}{1 - \dfrac{\underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(x_{n+1}\right)}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{\underline{B}}^{-1}\left(\underline{x}_n\right) \cdot \underline{F}\left(\underline{x}_{n+1}\right)}}$$

However, we still have one problem, which is that the lambda has an inv(B) that must be calculated in order to evaluate it. We need to find a way to replace this term. If we start with the following:

$$\underline{\underline{B}}\left(x_n\right) \cdot \underline{w} = \underline{\underline{L}}\left(x_n\right)\underline{\underline{U}}\left(x_n\right) \cdot \underline{w} = \underline{F}\left(x_{n+1}\right) \qquad or \qquad \underline{w} = \underline{\underline{B}}^{-1}\left(x_n\right) \cdot \underline{F}\left(x_{n+1}\right)$$

So if we can solve for $\underline{w}$, then we can replace the problem terms. We use the LU form to solve. Since L and U are triangular, we can solve them using backward substitution at a cost of only $O(N^2)$.

$$\underline{\underline{L}}\left(x_n\right)\underline{p} = \underline{F}\left(x_{n+1}\right)$$

$$\underline{\underline{U}}\left(x_n\right) \cdot \underline{w} = \underline{p}$$

Now we have the $\underline{w}$ vector to put in the lambda expression.

$$\lambda = \frac{1}{1 - \dfrac{\underline{\Delta x}_n^T \cdot \underline{w}}{\left\|\underline{\Delta x}_n^T\right\|^2 + \underline{\Delta x}_n^T \cdot \underline{w}}} \qquad \Rightarrow \qquad \boxed{\lambda = 1 + \frac{\underline{\Delta x}_n^T \cdot \underline{w}}{\left\|\underline{\Delta x}_n^T\right\|^2}}$$

The step updates can then be solved using backward substitution in two steps solving the following. Each step update can be found for the cost of several backward substitutions once you have the initial LU decomposition.

$$\lambda \underline{\underline{L}}\left(x_n\right)\underline{\underline{U}}\left(x_n\right) \cdot \underline{\Delta x}_{n+1} \equiv -\underline{F}\left(x_{n+1}\right)$$

Computational Cost:

Calculating the inverse every time will cost you $O(N^3)$ operations at every iteration, so this method is not preferred. It can also run into problem when the problem is ill-conditioned. Calculating the inverse once and then updating the inverse will cost you $O(N^3)$ once and $O(N^2)$ at every iteration to determine the update. Again, this method is much less costly, but can have problems when calculating the first inverse if the matrix is nearly singular. The LU method that avoid the inverse, but you have to do the LU decomposition once, which also takes $O(N^3)$ operations. Once you have the LU, it is pretty simple since you can just do a couple of backwards substitutions per update, which are $O(N^2)$. The LU method is preferred because it is more robust than the inverse methods, but still as fast as updating the inverse.