Problem 1: Root Solving

Problem Statement Consider the "Rosenbrock" function for x_1 and x_2:

$$f(\mathbf{x}) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$$

Part 1.1

Problem Statement Demonstrate the topology of the function for the ranges of $x_1 = [-3, 3]$ and $x_2 = [-1, 5]$
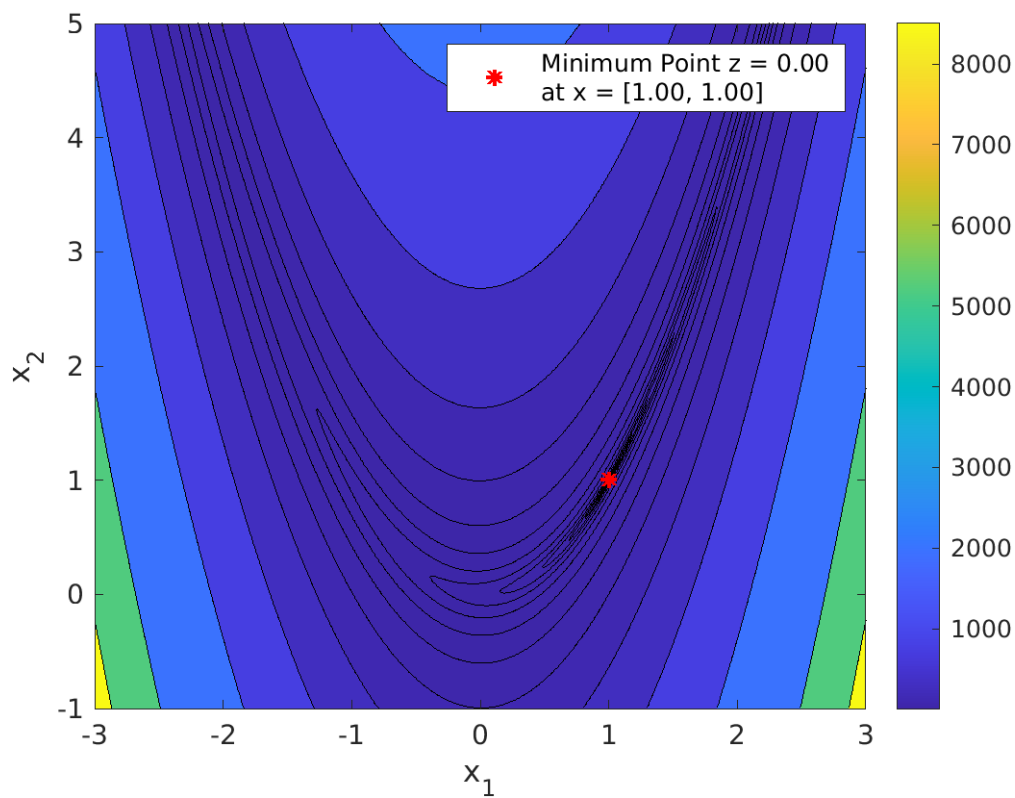

Figure 1: Contour map of Rosenbrock function over $x_1 = [-3, 3]$ and $x_2 = [-1, 5]$

Part 1.2

> **Problem Statement** Visually, where is the global minimum?

Looking specifically at the density of the countour lines, indicates a region around $[11]$ being the minimum.

Part 1.3

> **Problem Statement** Using calculus, demonstrate the answer is a local minimum with the sufficient and necessary conditions.

Second Order Necessary Condition: $\nabla f(\mathbf{x}_*) = \vec{0}$

Second Order Sufficient Condition: $\nabla^2 f(\mathbf{x}_*)$ is positive definite

Calculating the first gradient of the function yields a 2×1 vector:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2\,x_1 - 400\,x_1\,(x_2 - x_1{}^2) - 2 \\ 200\,x_2 - 200\,x_1{}^2 \end{bmatrix}$$

$$\begin{bmatrix} 2\,x_1 - 400\,x_1\,(x_2 - x_1{}^2) - 2 \\ 200\,x_2 - 200\,x_1{}^2 \end{bmatrix} \Bigg|_{\mathbf{x}=[1,1]^T} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \checkmark$$

With the first order gradient having zeros in every index, the second order necessary condition is satisified. Taking the second gradient, using the first, yields a 2×2 matrix of second order parital derivatives:

$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 1200\,x_1{}^2 - 400\,x_2 + 2 & -400\,x_1 \\ -400\,x_1 & 200 \end{bmatrix}$$

$$\texttt{chol}(\,\nabla^2 f([1,\,1]^T)\,) = \begin{bmatrix} \sqrt{802} & -\frac{200\,\sqrt{802}}{401} \\ 0 & \frac{10\,\sqrt{2}\,\sqrt{401}}{401} \end{bmatrix} \checkmark$$

Seeing as the Cholesky factorization was completed successfully, the second order gradient evaluated at $[1,1]^T$ is positive definite, satisfying the second order sufficient condition.

Problem 2: Single Line Search

> **Problem Statement** Build a general line search subroutine that minimizes an arbitrary n-dimensional function along an arbitrary search direction **s** starting at an arbitrary location **x** with an initial stride t.

Part 2.1

> **Problem Statement** Using the Quadratic Polynomial and Golden Ratio Methods, minimize the Rosenbrock function along $\mathbf{s} = [-1, 1]$ from a starting point $\mathbf{x} = [4, -1]$ using an initial stride length of $t_0 = 0.1$ and an $\epsilon = 10^{-4}$. Plot the position of each function call and use a different marker for the initial guess and final solution. Report the number of function calls required.
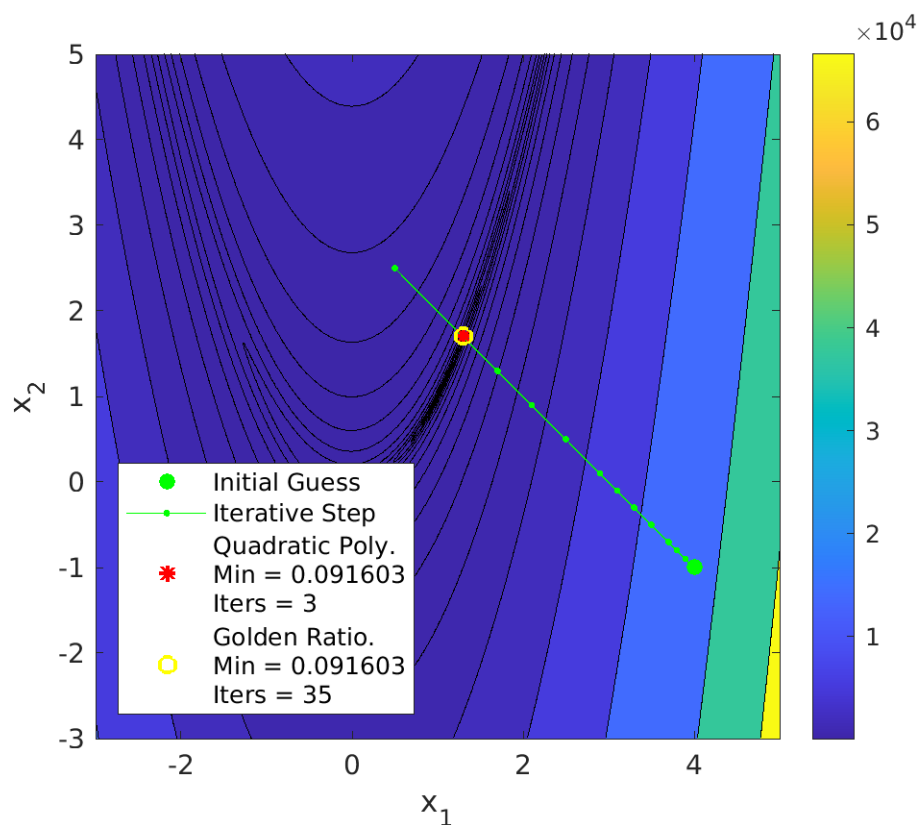

Figure 2: Stepping to solution space in green, then solving for minimum using both Quadratic Polynomial and Golden Ratio solvers.
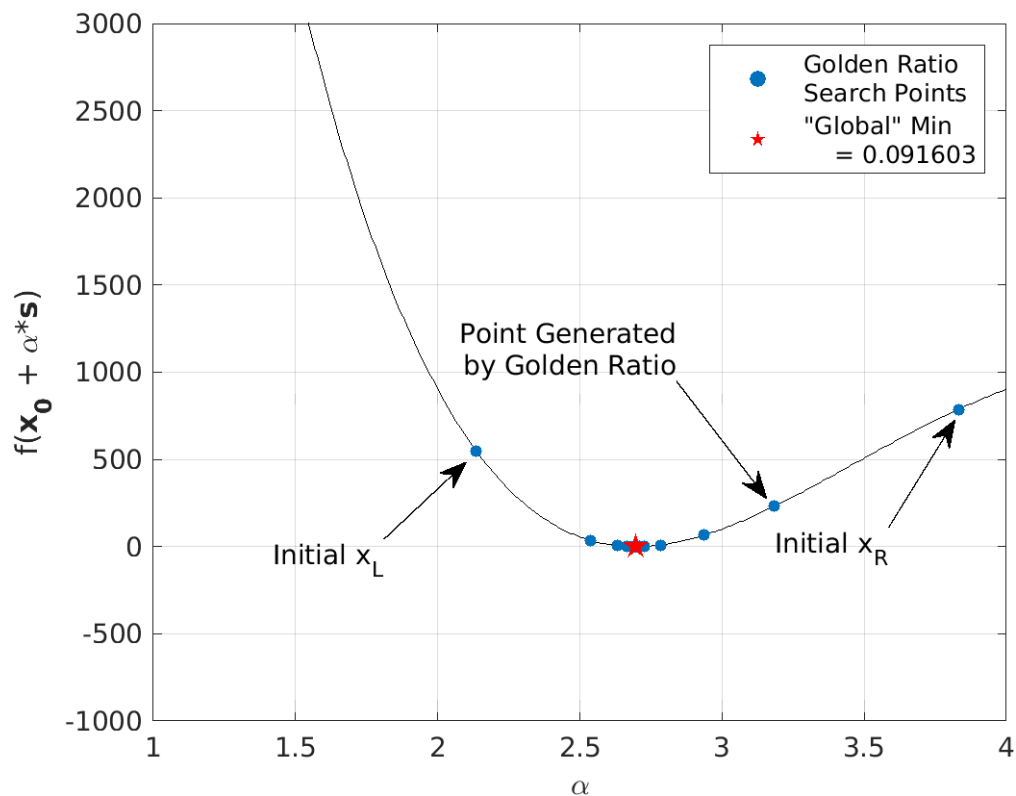
Figure 3: Slice of Rosenbrock function along $\mathbf{s} = [-1, 1]$ showing targetted minimum and search points created by Golden Ratio solver.

Part 2.2

> **Problem Statement** Use different strategies for stride adjustment, and discuss the results with regard to the number of function calls required. Present a summary table or plot of your investigation.

When adjusting the stride of the line search, the following method was used. Starting with the initial, given stride length, a certain number of steps were taken, and if the minimum was not bracketed, the stride length would be increased by some multiplication factor. The below table shows the results of varying the number of strides before increasing the step size, per row variance, and how much the stride was increased by, per column variance. Within each cell is the number of iterations to bracket the minimum, find the minimum using the quadratic polynomial method, and find the minimum using the golden ratio method, in that order.

Number Steps	1.1 Multiplier	1.25 Multiplier	1.5 Multiplier	2.0 Multiplier	2.5 Multiplier	3.0 Multiplier	5.0 Multiplier
2	19, 4, 33	14, 5, 34	11, 5, 35	9, 5, 36	8, 52, 37	7, 5, 37	7, 5, 44
4	23, 4, 32	19, 3, 33	16, 5, 33	13, 3, 35	11, 5, 35	10, 5, 35	9, 5, 37
8	26, 4, 32	23, 4, 32	21, 4, 33	18, 3, 34	17, 3, 34	16, 4, 33	13, 3, 34
16	28, 4, 31	27, 4, 31	25, 3, 32	23, 3, 32	22, 4, 33	21, 3, 33	19, 5, 34

One key result from the table is the inverse relationship between the number of iterations to bracket the minimum and find the minimum, with both methods, when varying either the multiplier or number of steps. The faster the minimum is bounded the longer it will take to find the minimum, due to the size of said bracket on the line. One outlier from the results is for the 2 step and 2.5 multiplier result of the quadratic polynomial method. Which hit an iteration limit, and bounced around very near the solution for infinity.

Problem 3: Adding the Outer Loop

Problem Statement Using the polynomial line search from Part 2, build wrapper routines for general n-dimensional unconstrained optimizers using:

1) Seepest Descent
2) Fletcher-Reeves
3) Polak-Ribiere
4) BFGS

Also build a full second order Levenberg-Marquardt solver with dynamic adjustment of the damping parameter. **Extra Credit:** Build a full trust region solver with dynamic trust region management.

Part 3.1

Problem Statement For each of the 5 methods, minimize the Rosenbrock function starting at $\mathbf{x} = [4, -1]$. Plot the $\mathbf{x}$ at each major iteration. Tune your codes and try to reduce the number of function calls to a minimum. State success or failure (using a maximum iteration limit of 1000), the final $\mathbf{x}$ and f to 8 digits, and the total number of function calls and derivative function calls (first and second order), also the CPU time required, and machine you are using. Discuss the results and your general strategy. Present a summary table with rows of each method and columns for the characteristics of each method mentioned above.

Algorithm	F_{min}	Func Calls	Grad Calls	Hess Calls	Run Time (seconds)
Steepest Descent	0.17807939	40863	1001	0	0.279271
Fletcher Reeves	1×10^{-8}	3766	93	0	0.079834
Polak Ribiere	0	3571	88	0	0.071204
BFGS	0	635	16	0	0.066949
Levenberg Marquardt	0	21	15	15	0.031304

Minimizing Rosenbrock Function
[#f #g #H] = [40863 1001 0]
f = 0.17807939
Run Time = 0.279271 Seconds

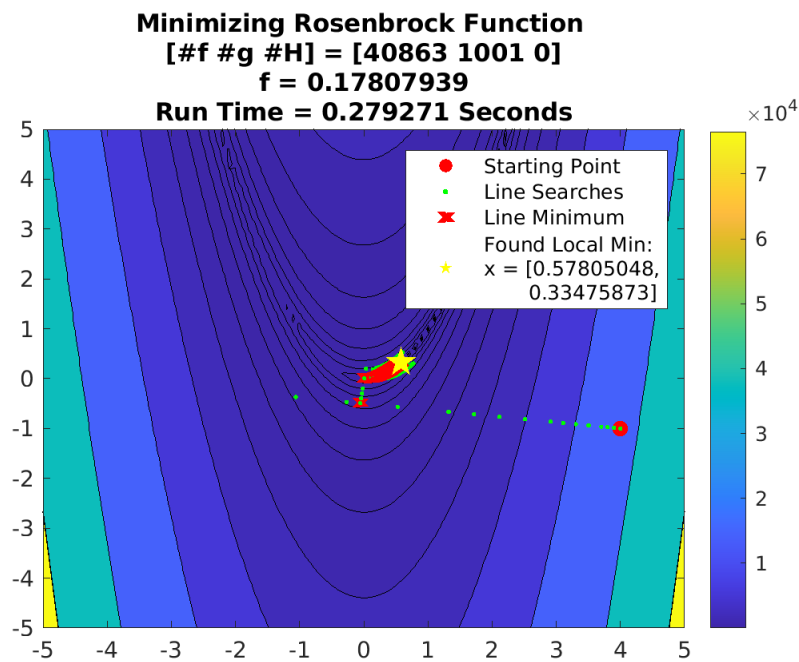**Figure 4:** Finding minimum of Rosenbrock function using Steepest Descent. Solution fails after iteration budget is reached.

Minimizing Rosenbrock Function
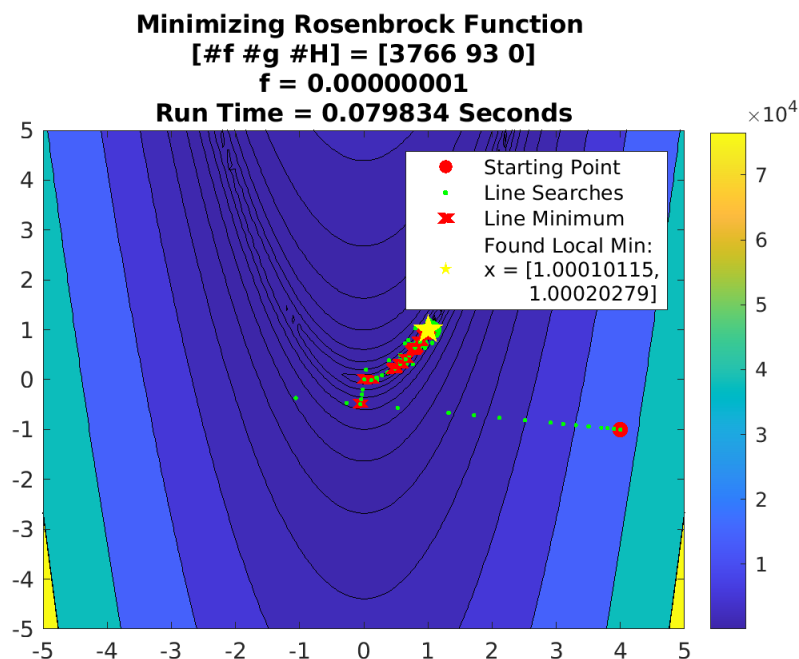**[#f #g #H] = [3766 93 0]**
f = 0.00000001
Run Time = 0.079834 Seconds

Figure 5: Finding minimum of Rosenbrock function using Fletcher-Reeves algorithm. Minimum successfully found within bounds of problem.
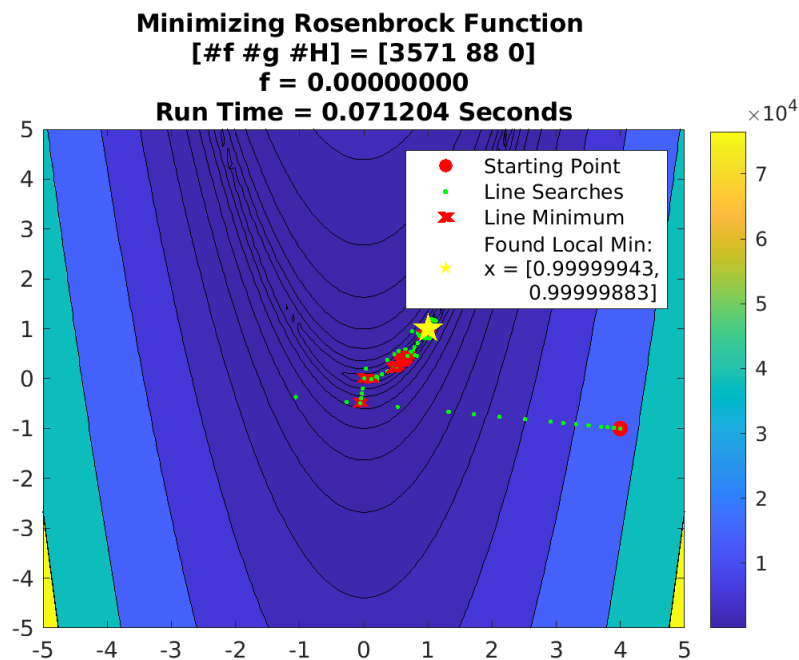
Figure 6: Finding minimum of Rosenbrock function using Polak-Ribiere algorithm. Minimum successfully found within bounds of problem.
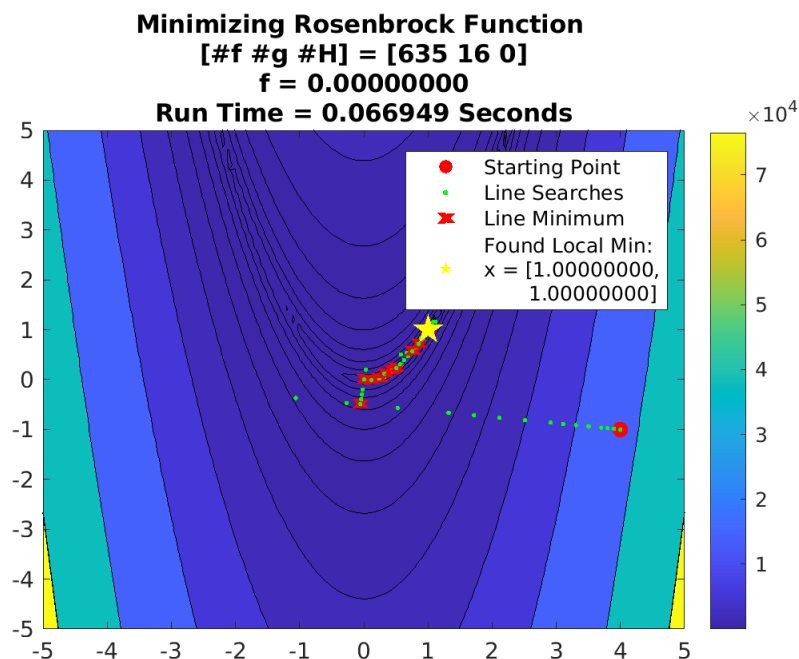

Figure 7: Finding minimum of Rosenbrock function using BFGS algorithm. Minimum successfully found within bounds of problem.
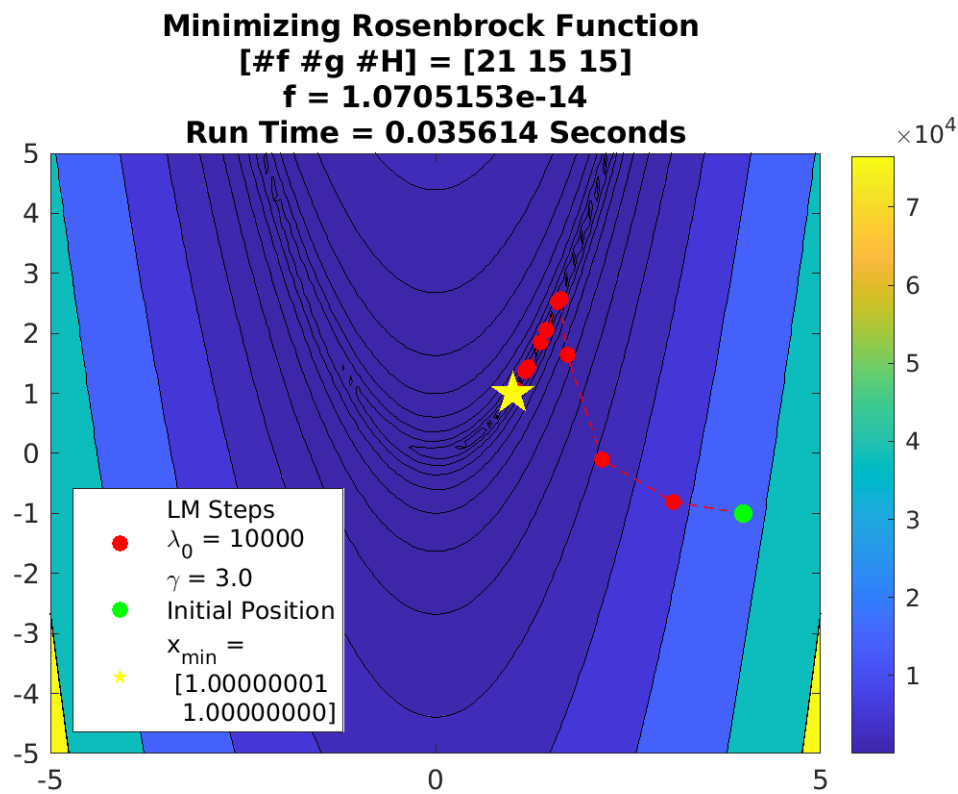
Figure 8: Finding minimum of Rosenbrock function using Levenberg-Marquart algorithm. Minimum successfully found within bounds of problem.

Part 3.2

Problem Statement Repeat Part 3.1 but instead minimize the function below with the two initial conditions: $\mathbf{x} = [4, \, -1]$ and $\mathbf{x} = [2, \, 3]$.

$$f = e^{u^2} + \sin(4x_1 - 3x_2)^4 + 0.5(2x_1 + x_2 - 10)^2$$

where

$$u = \frac{1}{2}(x_1^2 + x_2^2 + 25)$$

Problem 4: Orbits Problem

Problem Statement Find the initial velocity that solves the planar J_2-Lambert problem described below:

Input Constants:

$$J_2 = 0.1 \quad R = 0.9 \quad \mu = 1 \quad \text{TOF} = 83$$

$$\mathbf{r}_f = \begin{bmatrix} -2.891216226058043 \\ -1.254145998446107 \\ 0 \end{bmatrix}$$

Performance Index:

$$f = (\mathbf{r}_* - \mathbf{r}_f)^T(\mathbf{r}_* - \mathbf{r}_f) + 10^{10}(N - N_*)$$

Where N_* and $\mathbf{r}_*$ are the number of revolutions and final position of the propagated orbit, respectively.

Equations of Motion:

$$\ddot{\mathbf{r}} = \nabla U = \left[\frac{dU}{dx}, \frac{dU}{dy}, \frac{dU}{dz} \right]$$

$$U = \frac{\mu}{\sqrt{x^2 + y^2 + z^2}} \left[1 - \frac{R^2}{x^2 + y^2 + z^2} J_2 \left(\frac{3z^2}{2(x^2 + y^2 + z^2)} - \frac{1}{2} \right) \right]$$

Initial Guesses:

$$\mathbf{r}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{v}_0 = \begin{bmatrix} 0.05 \\ 1.305 \\ 0 \end{bmatrix} \quad \textbf{or} \quad \mathbf{v}_0 = \begin{bmatrix} -0.2 \\ 1.27 \\ 0 \end{bmatrix}$$

Show the solution space, and path for any method implemented successfully. Discuss the method used for computing derivatives, and solving the equations of motion. Discuss the approach including any failed or stalled efforts and generally discuss results of success. Plot the initial condition of the trajectory, the targeted position, and the propagated solution.

Equations of Motion   For computing the equations of motion, the equations were symbolically solved using MATLAB's symbolic toolbox. Starting with the original U equation, the derivatives were first taken with respect to x and y as their equations will yield very similar answers where only the numerators have their x terms swapped with y terms. Since z requires an additional quotient rule it will yield different results. All equations are shown below:

$$\frac{dU}{dx} = \frac{\frac{2\,J_2\,R^2\,x\,\sigma_2}{\sigma_1{}^2} + \frac{12\,J_2\,R^2\,x\,z^2}{\left(2\,x^2 + 2\,y^2 + 2\,z^2\right)^2\,\sigma_1}}{\sqrt{\sigma_1}} + \frac{x\left(\frac{J_2\,R^2\,\sigma_2}{\sigma_1} - 1\right)}{\sigma_1{}^{3/2}}$$

$$\frac{dU}{dy} = \frac{\frac{2\,J_2\,R^2\,y\,\sigma_2}{\sigma_1{}^2} + \frac{12\,J_2\,R^2\,y\,z^2}{\left(2\,x^2 + 2\,y^2 + 2\,z^2\right)^2\,\sigma_1}}{\sqrt{\sigma_1}} + \frac{y\left(\frac{J_2\,R^2\,\sigma_2}{\sigma_1} - 1\right)}{\sigma_1{}^{3/2}}$$

$$\frac{dU}{dz} = \frac{z\left(\frac{J_2\,R^2\,\sigma_2}{\sigma_1} - 1\right)}{\sigma_1{}^{3/2}} - \frac{\frac{J_2\,R^2\left(\frac{6\,z}{\sigma_3} - \frac{12\,z^3}{\sigma_3{}^2}\right)}{\sigma_1} - \frac{2\,J_2\,R^2\,z\,\sigma_2}{\sigma_1{}^2}}{\sqrt{\sigma_1}}$$

where

$$\sigma_1 = x^2 + y^2 + z^2$$

$$\sigma_2 = \frac{3\,z^2}{\sigma_3} - \frac{1}{2}$$

$$\sigma_3 = 2*x^2 + 2*y^2 + 2*z^2$$

Derivatives   To calculate the derivatives of the performance index, the integration of the trajectory is passed through a complex step derivative function. This incurs 3 additional functions for each gradient computed when minizing.

Minimization Methods   Searching the space and descending to the minima was left to the BFGS (Broyden-Fletcher-Goldfarb-Shanno Algorithm) to approximate the Hessian without requiring finite differencing the complex step method used previously. After using BFGS to find the direction of travel, a line search is conducted and the Golden Ratio method was used to minimize the line search. The number of iterations to find the solution showed high sensitivity to the initial stride length, t_0, which had to be tweaks from the value used earlier in the problem set.
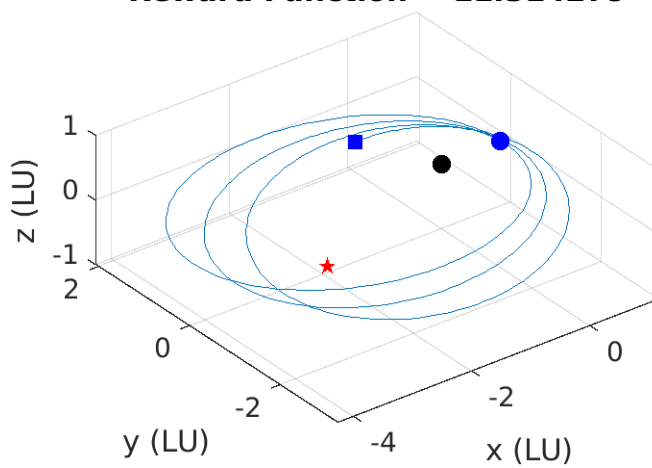
t_0	f Calls	g Calls	Run Time (sec)
0.1			
0.05			
0.01			
0.005			

Part 4.1

Problem Statement Solving the trajectory using the first initial condition for velocity:

$$\mathbf{v}_0 = \begin{bmatrix} 0.05 \\ 1.305 \\ 0 \end{bmatrix}$$

Propagating Initial State
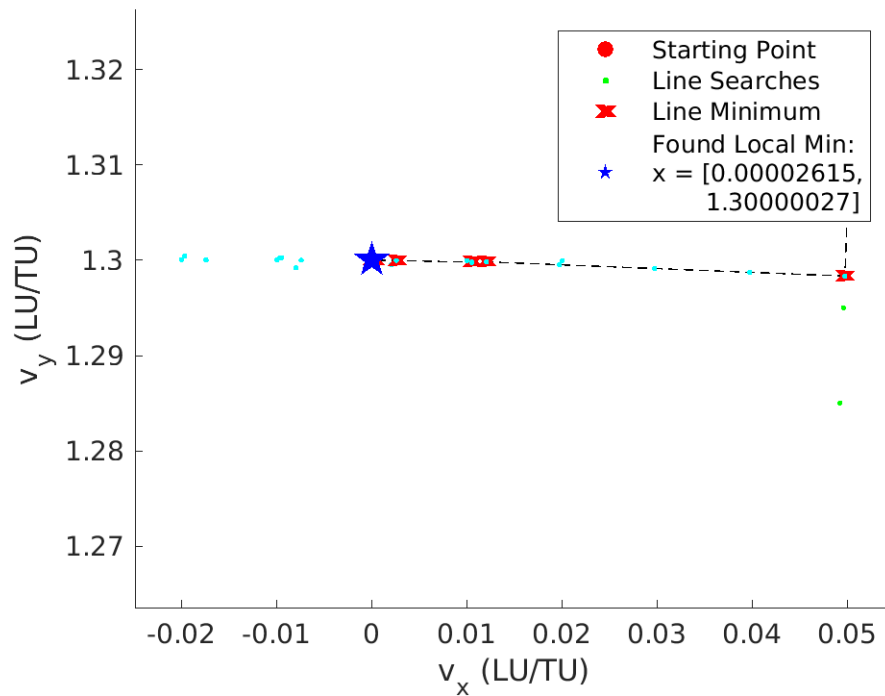**|r*-r$_f$| = 3.364 :: N* = 3**
Reward Function = 11.314279

Finding Minimum Using BFGS
Initial Stride = 0.01
[#f #g] = [229 8]
Run Time = 1.5216 seconds

Propagating Initial State
|r*-r$_f$| = 1.716e-06 :: N* = 3
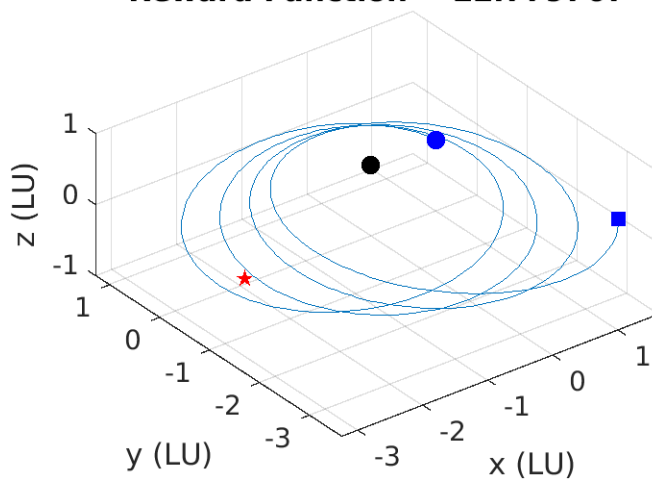**Reward Function = 2.9461498e-12**
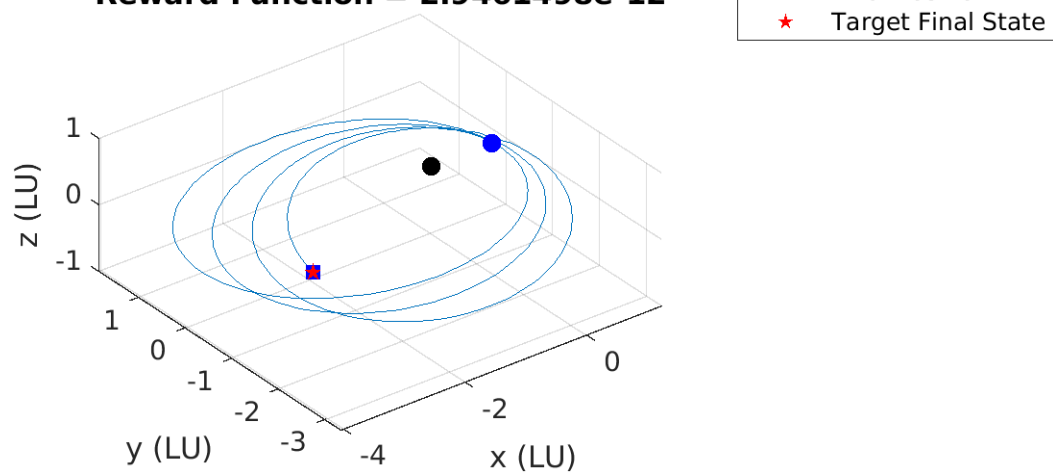
Part 4.2

Problem Statement Solving the trajectory using the second initial condition for velocity:

$$\mathbf{v}_0 = \begin{bmatrix} -0.2 \\ 1.27 \\ 0 \end{bmatrix}$$

Propagating Initial State
|r*-r$_f$| = 4.773 :: N* = 3
Reward Function = 22.779707

Finding Minimum Using BFGS
Initial Stride = 0.01
[#f #g] = [458 16]
Run Time = 3.4341 seconds
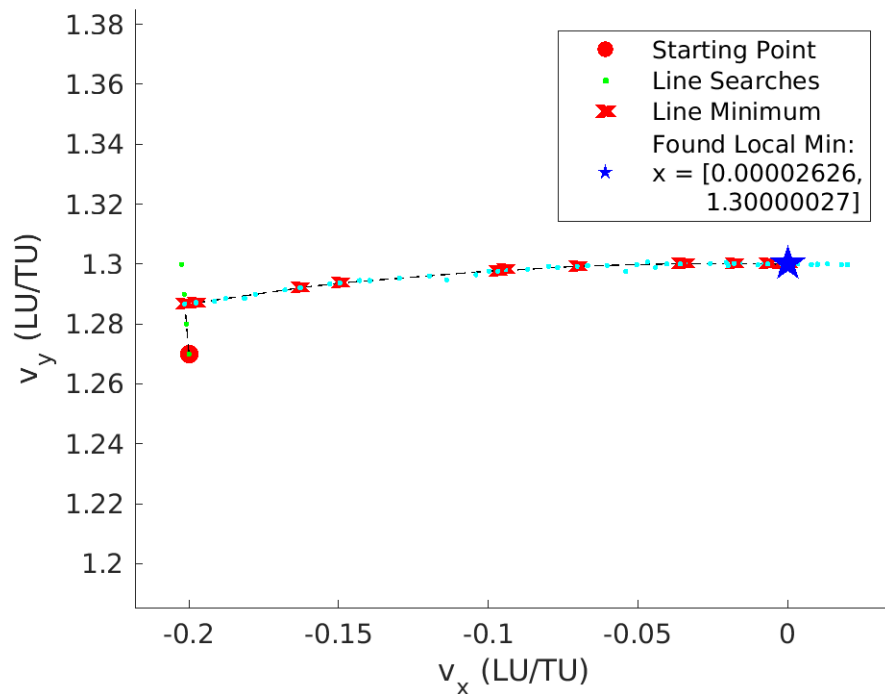
Propagating Initial State
|r*-r$_f$| = 8.848e-07 :: N* = 3
Reward Function = 7.8285533e-13