16-811 hw 4

Erica Weng

October 30, 2020

1

1a

$$3y^2 \frac{dy}{dx} = 1$$

$$3y^2 dy = dx$$

integrate both sides

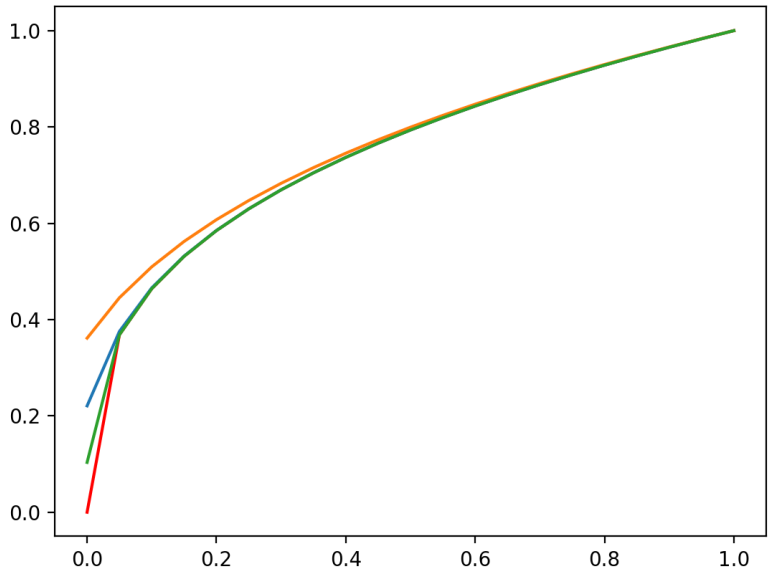
$$y^3 = x$$

$$y = \sqrt[3]{x}$$

since $y(1) = 1$ the final answer is $\boxed{y(x) = \sqrt[3]{x}}$

1b

code in `code/q1.py`



red is truth
note how we start on the right and error gets worse as we go left
orange is euler's method
green is rk4
blue is ab4

table for euler:

x_i	y_i	$y(x_i)$	$y(x_i) - y_i$
0.000	0.361	0.000	-0.3614
0.050	0.445	0.368	-0.0770
0.100	0.510	0.464	-0.0454
0.150	0.562	0.531	-0.0310
0.200	0.607	0.585	-0.0227
0.250	0.647	0.630	-0.0173
0.300	0.683	0.669	-0.0136
0.350	0.716	0.705	-0.0108
0.400	0.746	0.737	-0.0087
0.450	0.773	0.766	-0.0071
0.500	0.799	0.794	-0.0058
0.550	0.824	0.819	-0.0047
0.600	0.847	0.843	-0.0038
0.650	0.869	0.866	-0.0030
0.700	0.890	0.888	-0.0024
0.750	0.910	0.909	-0.0019
0.800	0.930	0.928	-0.0014
0.850	0.948	0.947	-0.0010
0.900	0.966	0.965	-0.0006
0.950	0.983	0.983	-0.0003
1.000	1.000	1.000	0.0000

1c

runge-kutta 4:

x_i	y_i	$y(x_i)$	$y(x_i) - y_i$
0.000	0.103	0.000	-0.1035
0.050	0.368	0.368	0.0000
0.100	0.464	0.464	0.0000
0.150	0.531	0.531	0.0000
0.200	0.585	0.585	0.0000
0.250	0.630	0.630	0.0000
0.300	0.669	0.669	0.0000
0.350	0.705	0.705	0.0000
0.400	0.737	0.737	0.0000
0.450	0.766	0.766	0.0000
0.500	0.794	0.794	0.0000
0.550	0.819	0.819	0.0000
0.600	0.843	0.843	0.0000
0.650	0.866	0.866	0.0000
0.700	0.888	0.888	0.0000
0.750	0.909	0.909	0.0000
0.800	0.928	0.928	0.0000
0.850	0.947	0.947	0.0000
0.900	0.965	0.965	0.0000
0.950	0.983	0.983	0.0000
1.000	1.000	1.000	0.0000

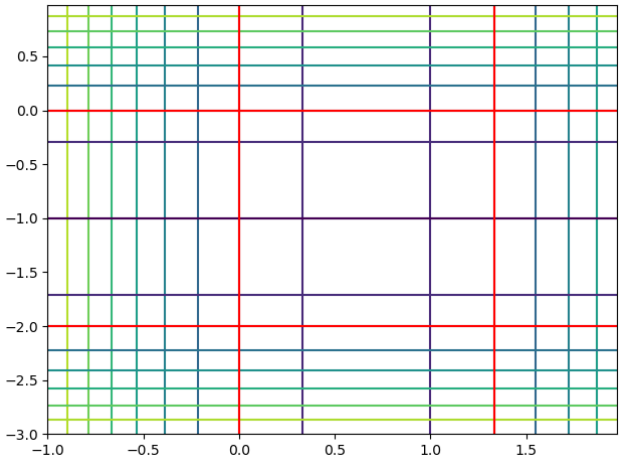
1d

adams-bashforth 4:

x_i	y_i	$y(x_i)$	$y(x_i) - y_i$
0.000	0.221	0.000	-0.2208
0.050	0.376	0.368	-0.0071
0.100	0.466	0.464	-0.0018
0.150	0.532	0.531	-0.0007
0.200	0.585	0.585	-0.0003
0.250	0.630	0.630	-0.0002
0.300	0.670	0.669	-0.0001
0.350	0.705	0.705	-0.0001
0.400	0.737	0.737	-0.0000
0.450	0.766	0.766	-0.0000
0.500	0.794	0.794	-0.0000
0.550	0.819	0.819	-0.0000
0.600	0.843	0.843	-0.0000
0.650	0.866	0.866	-0.0000
0.700	0.888	0.888	-0.0000
0.750	0.909	0.909	-0.0000
0.800	0.928	0.928	-0.0000
0.850	0.947	0.947	-0.0000
0.900	0.965	0.965	-0.0000
0.950	0.983	0.983	-0.0000
1.000	1.000	1.000	0.0000

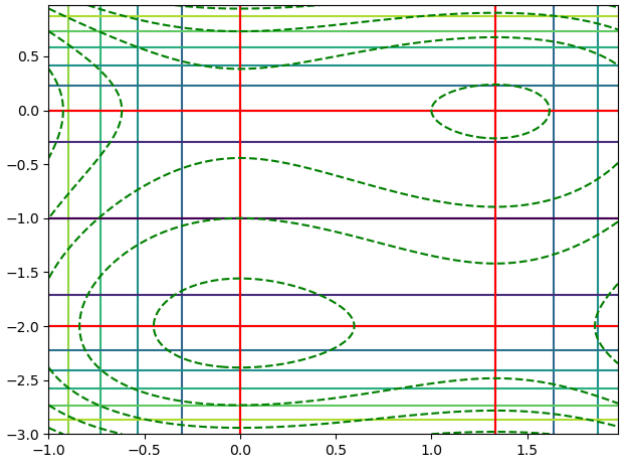
2

2a



purple is negative gradient, dark green, green, and yellow is positive gradient, red is 0 gradient (critical contour line). the horizontal lines are where $\frac{\partial f}{\partial y} = 0$, and the vertical lines are where $\frac{\partial f}{\partial x} = 0$. below is the same graph with the

contours lines of f sketched in dotted lines as well.



where the red lines intersect are critical points. there are four of them: $(0,0)$, $(\frac{4}{3},0)$, $(-2,0)$, and $(\frac{4}{3},-2)$.
$(0,0)$ and $(\frac{4}{3},-2)$ are saddle points because along the x-axis from small x to large x, the gradient wrt x (the vertical contour lines) goes in one direction (e.g. positive to negative for $(0,0)$) while along the y-axis from small y to large y, the gradient wrt y goes in the opposite direction (e.g. neg to pos for $(0,0)$.
$(-2,0)$ is a maximum bc along x and y axis, the grad goes from pos to neg.
$(\frac{4}{3},0)$ is a minimum because along both axes the grad goes from neg to pos.

2b

from starting point $(1,-1)$, only 1 steepest descent step is needed.

$$f(x,y) = x^3 + y^3 - 2x^2 + 3y^2 - 8$$

$$\nabla f(x,y) = \begin{bmatrix} 3x^2 - 4x \\ 3y^2 + 6y \end{bmatrix}$$

$$\nabla f(1,-1) = \begin{bmatrix} -1 \\ -3 \end{bmatrix}$$

plug into point-slope form to get the 2d line, i.e. y in terms of x on the negative grad line:

$$y - (-1) = \frac{-3}{-1}(x - (1))$$

$$y = 3x - 4$$

then the negative gradient line is given by:

$$f(x) = 28x^3 - 83x^2 + 72x - 24$$

set the derivative to 0 to get the zeros on this grad line:

$$\nabla f(x) = 84x^2 - 166x + 72 = 0$$

the zeros are at $x = (\frac{4}{3}, 0)$ and $(\frac{9}{14}, 3(\frac{9}{14}) - 4)$. the one closer to our starting point of $(-1, 1)$ is $(\frac{4}{3}, 0)$. which happens to be our local minimum. yay, we converged in 1 step!

3

3a

we show that 2 eigenvectors of the symmetric, positive-definite Q corresponding to distinct eigenvalues of Q are orthogonal. then we show that they are Q-orthogonal.

$$Qx_1 = \lambda_1 x_1$$
$$Qx_2 = \lambda_2 x_2$$

where x_1, x_2 are 2 eigenvectors of Q corresponding to distinct eigenvalues of Q, and λ_1, λ_2 are the 2 eigenvalues, respectively, that they correspond to.
take the first eigenvalue equation and multiply both sides by $x_2^\top$

$$x_2^\top Q x_1 = x_2^\top \lambda_1 x_1$$

substitute Q for $Q^\top$, because it is symmetric

$$(Qx_2)^\top x_1 = \lambda_1 x_2^\top x_1$$

use the second eigenvalue equation and substitute $(Qx_2)^\top$ for $\lambda_2 x_2^\top$

$$\lambda_2 x_2^\top x_1 = \lambda_2 x_2^\top x_1$$

as $\lambda_1 \neq \lambda_2$, the only way this can be true is if $x_2^\top x_1 = 0$; that is, if the two eigenvectors are orthogonal.

now, since $x_2^\top x_1 = 0$, then we can show that $x_2^\top Q x_1 = 0$:

$$x_2^\top Q x_1 = \lambda_1 x_2^\top x_1 = 0$$

thus completes our proof

3b

"one can find a basis of eigenvectors of Q that are pairwise orthogonal (in the usual sense), even if Q has repeated eigenvalues" – we can take this statement as a given. then, let's say x_1 and x_2 are two such eigenvectors that are orthogonal to each other and correspond to the same eigenvalue λ, i.e. that $Qx_1 = \lambda x_1$ and $Qx_2 = \lambda x_2$. Then we have:

$$x_2^\top Q x_1 = x_2^\top \lambda x_1$$
$$= \lambda x_2^\top x_1 = 0$$

thus any two such orthogonal basis vectors (correspond to the same eigenvalue) are in fact also Q-orthogonal.

4

4a

from step 2d of the CG algo, pg 17 of the optimization notes

$$d_k = -g_k + \beta_{k-1} d_{k-1}$$

multiply both sides by $d_k^\top Q$

$$d_k^\top Q d_k = d_k^\top Q(-g_k + \beta_{k-1} d_{k-1})$$

$$d_k^\top Q d_k = -d_k^\top Q g_k + \beta_{k-1} d_k^\top Q d_{k-1}$$

by the expanding subspace thm we know that all descent directions are Q-orthogonal to each other, so $d_k^\top Q d_{k-1} = 0$, so:

$$d_k^\top Q d_k = -d_k^\top Q g_k$$

thus completes our proof of the first statement

continuing, from 2a and 2b of the CG algo (pg 17) we have

$$x_{k+1} = x_k + \alpha_k d_k$$

$$\alpha_k = -\frac{g_k^\top d_k}{d_k^\top Q d_k}$$

plug some things in and get

$$x_{k+1} = x_k - \frac{g_k^\top d_k}{d_k^\top Q d_k} d_k$$

now substitute from the first statement $d_k^\top Q d_k = -d_k^\top Q g_k$

$$x_{k+1} = x_k + \frac{g_k^\top d_k}{d_k^\top Q g_k} d_k$$

thus completes our proof of the second statement

4b

given, we have

$$p_k = \nabla f(y_k) = \nabla f(x_k - g_k)$$

let's show that $g_k - p_k = Qg_k$.

$$g_k - p_k = g_k - \nabla f(x_k - g_k)$$

we have that $\nabla f(x_k) = g_k = Qx_k + b$ in a purely quadratic form. so, substituting:

$$\begin{aligned} &= Qx_k + b - (Q(x_k - g_k) + b) \\ &= Qx_k + b - Qx_k + Qg_k - b \\ &= Qg_k \end{aligned}$$

thus completes our proof

4c

algorithm: initialize:

$$g_0 = \nabla f(y_k)$$

$$d_0 = -g_0$$

@ starting point x_0

from part a we have:

$$x_{k+1} = x_k + \frac{g_k^\top d_k}{d_k^\top Q g_k} d_k$$

substituting part b in, we have:

$$x_{k+1} = x_k + \frac{g_k^\top d_k}{d_k^\top (g_k - \nabla f(x_k - g_k))} d_k$$

from e) on page 21 of optimization notes:

$$\beta_{k+1} = \frac{g_{k+1}^\top g_{k+1}}{g_k^\top g_k}$$

so then:

$$d_{k+1} = g_{k+1} + \beta_k d_k$$

note that $g_k = \nabla f(x_k)$.

iterate by updating d_k and x until convergence.

5

$$\begin{aligned} &\max \ hw \\ &\text{subject to } \ 2h + 2w = p \end{aligned}$$

$$\begin{aligned} f(h,w) &= hw \\ g(h,w) &= 2h + 2w = p \\ F(h,w,\lambda) &= hw + \lambda(2h + 2w) \\ &= hw + 2\lambda h + 2\lambda w \\ \nabla F &= \begin{bmatrix} w + 2\lambda \\ h + 2\lambda \\ 2h + 2w \end{bmatrix} = 0 \\ w &= -2\lambda = h \\ &\boxed{h = w = \frac{p}{4}} \\ \lambda = -\frac{w}{2} &= -\frac{p}{8} \end{aligned}$$

second-order necessary conditions: from page 34 of the optimization notes the 2nd order sufficiency conditions are: for $x*$ to be a local maximum, then for L defined by $L(x^*) = \nabla^2 f(x^*) + \lambda^\top \nabla^2 h(x^*)$, $y^\top L y < 0 \ \ \forall \ \ y \in \{y \mid \nabla h(x^*) y = 0\}$

$$\begin{aligned} \nabla f(h,w) &= \begin{bmatrix} w \\ h \end{bmatrix} \\ \nabla^2 f(h,w) &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \\ \nabla g(h,w) &= \begin{bmatrix} 2 \\ 2 \end{bmatrix} \\ \nabla^2 g(h,w) &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ L(x^*) = \nabla^2 f(x^*) + \lambda^\top \nabla^2 h(x^*) &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{aligned}$$

we need $y^\top L y < 0$ (L to be negative definite) for all $y \in \{y \mid \nabla h(x^*) y = 0\}$ which means

$$y = \alpha \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

for arbitrary scalar $\alpha \neq 0$

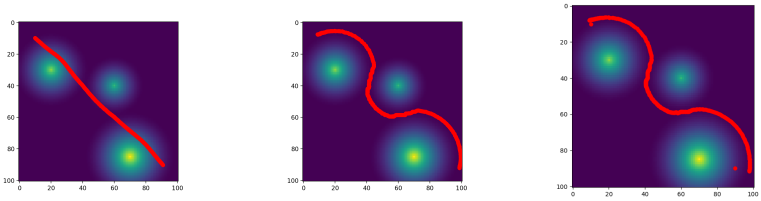
$$\begin{aligned} &\left(\alpha \begin{bmatrix} 1 & -1 \end{bmatrix}\right) \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \left(\alpha \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) \\ &\alpha^2 \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = -2 \cdot \alpha^2 < 0 \end{aligned}$$

so we see the 2nd order sufficient conditions hold

6

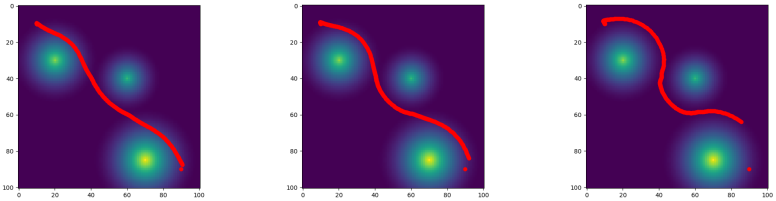
code is in `code/trajectory_optimization.py`

6a



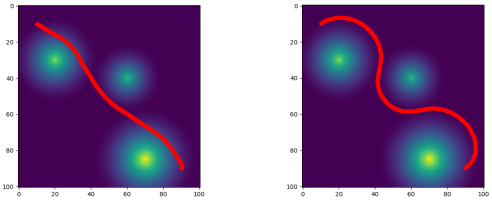
learning rate = 0.1; in order: 3 iters, 100 iters, 600 iters the points separated from the start and end points

6b



learning rate = 0.1; in order: 100, 200, and 500 iters the points gathered near the start bc the points are not tied to the last point

6c



learning rate = 0.1; in order: 100 and 5000 iters

6d

no it would not be the same. initialization is very important to gradient descent. the optimization trajectory depends on the gradients you encounter on the trajectory, and the gradients in turn influence the trajectory.

6e

you can try to alter the cost function to penalize the robot more for trying to go through those high cost regions i.e. square the cost function or take it to some higher power. this will make high cost regions even more expensive, thus discouraging the robot even more from passing through those regions