HW3 - Henrique Gasparini Fiuza do Nascimento

November 12, 2018

1 Lasso regression

Given $x_1, ..., x_n \in \mathbb{R}^d$ data vectors and $y_1, ..., y_n \in \mathbb{R}$ observations, we are searching for regression parameters $w \in \mathbb{R}^d$ which fit data inputs to observations y by minimizing their squared difference. In a high dimensional setting (when $n \ll d$) a l_1 norm penalty is often used on the regression coefficients w in order to enforce sparsity of the solution (so that w will only have a few non-zeros entries). Such penalization has well known statistical properties, and makes the model both more interpretable, and faster at test time.

From an optimization point of view we want to solve the following problem called LASSO (which stands for Least Absolute Shrinkage Operator and Selection Operator)

minimize $\frac{1}{2}||Xw - y||_2^2 + \lambda||w||_1$

in the variable $w \in \mathbb{R}$, where $X = (x_1^T, ..., x_n^T) \in \mathbb{R}^{n \times d}$, $y = (y_1, ..., y_n) \in \mathbb{R}^n$ and $\lambda > 0$ is a regularization parameter.

1.1 Exercise 1

Derive the dual problem of LASSO and format it as a general Quadratic Problem as follows

minimize $v^T Q v + p^T v$

subject to $Av \preceq b$

in variable $v \in \mathbb{R}^n$, where $Q \succeq 0$.

Solution: the Lasso problem described above has the following dual problem:

minimize $v^T v + 2y^T v$

subject to $X^T v \preceq 1$ and $X^T v \succeq -1$

which is equivalent to

minimize $v^T v + 2y^T v$

subject to $[X, -X]^T v \preceq 1$

The handwritten computations are shown in the photo below:

```
In [2]: %matplotlib inline
```

```
In [3]: import matplotlib.pyplot as plt
```

```
In [4]: exercise_1_path = 'Exercise 1 - Homework 3 - Convex Optimization - MVA.jpg'
        plt.figure(figsize=(28,21))
        plt.imshow(plt.imread(exercise_1_path))
```

```
Out[4]: <matplotlib.image.AxesImage at 0x7fd789878b00>
```

1. Let $d = Xw - y$. We compute the dual function

$$g(\nu) = \inf_{w,d} \frac{1}{2} d^T d + \lambda \|w\|_1 + \nu^T(Xw - y - d)$$

$$= \inf_{w,d} \left(\frac{1}{2} d^T d - \nu^T d\right) + \left(\lambda \|w\|_1 + \nu^T X w\right) - \nu^T y$$

$$= \underbrace{\inf_{d} \left(\frac{1}{2} d^T d - \nu^T d\right)}_{-\frac{\|\nu\|_2^2}{2},\ \text{for } d^* = -\nu} + \underbrace{\inf_{w} \left(\lambda \|w\|_1 + (X^T\nu) w\right)}_{\begin{cases} 0 & \text{if } (X^T\nu)_{(i)} \in [-1,1]\ \forall i,\ \text{for } w^* = 0 \\ -\infty & \text{otherwise} \end{cases}} - \nu^T y$$

$$\Rightarrow g(\nu) = \begin{cases} -\frac{\|\nu\|_2^2}{2} - \nu^T y & \text{if } \|X^T\nu\|_\infty \le 1 \\ -\infty & \text{otherwise} \end{cases}$$

The dual problem is:

$$\begin{cases} \text{maximize} & -\frac{\|\nu\|_2^2}{2} - \nu^T y \\ \text{subject to} & \|X^T \nu\|_\infty \le 1 \end{cases},$$

which is equivalent to:

$$\text{minimize} \quad \nu^T \nu + 2y^T \nu$$

$$\text{subject to} \quad \|X^T \nu\|_\infty \le 1$$

$\hookrightarrow$ can be rewritten as $X^T\nu \le 1$ and $X^T\nu \ge -1$

1.2 Exercise 2

Implement the barrier method to solve QP.

- Write a function `v_seq = centering_step(Q,p,A,b,t,v_0,eps)` which implements the Newton method to solve the centering step given the inputs (Q,p,A,b), the barrier method parameter t (see lectures), initial variable v_0 and a target precision ϵ. The function outputs the sequence of variables iterates $(v_i)_{i=1...n(\epsilon)}$, where $n(\epsilon)$ is the number of iterations to obtain the ϵ precision. Use a backtracking line search with appropriate parameters.

- Write a function `v_seq = barr_method(Q,p,A,b,v0,eps)` which implements the barrier method to solve QP using precedent function given the data inputs (Q,p,A,b), a feasible point v_0, a precision criterion ϵ. The function outputs the sequence of variables iterates $(v_i)_{i=1,...,n_\epsilon}$, where n_ϵ is the number of iterations to obtain the ϵ precision.

```
In [5]: import numpy as np
```

```
In [529]: class Differentiator:
              def __init__(self, Q, p, A, b, t, eps):
                  self.Q, self.p, self.A, self.b, self.t, self.eps = Q, p, A, b, t, eps

              def standard_loss(self, v):
                  return (np.matrix(v) * np.matrix(self.Q) * np.matrix(v).T + np.matrix(self.p)

              def phi_i(self, v, i):
                  return np.dot(self.A[i], np.squeeze(v)) - self.b[i]

              def phi_loss(self, v):
                  return np.sum([-np.log(-self.phi_i(v, i)) for i in range(len(self.A))])

              def t_loss(self, v):
                  return self.t * self.standard_loss(v) + self.phi_loss(v)

              def standard_gradient(self, v):
                  return 2 * np.dot(self.Q, v) + self.p

              def phi_gradient(self, v):
                  return np.sum([-1/self.phi_i(v, i) * A[i].T for i in range(len(self.A))], axis

              def t_gradient(self, v):
                  return self.t * self.standard_gradient(v) + self.phi_gradient(v)

              def standard_hessian(self, v):
                  return 2 * self.Q

              def phi_hessian(self, v):
                  phi_is = [self.phi_i(v, i) for i in range(len(self.A))]
                  return np.sum([1/phi_is[i]**2 * np.array(np.matrix(A[i]).T * np.matrix(A[i]))
```

```
    def t_hessian(self, v):
        return self.t * self.standard_hessian(v) + self.phi_hessian(v)


def line_search(v, step, solver, t0=1., alpha=0.25, beta=0.9):
    print('Performing linear search:')
    t = t0
    while solver.t_loss(v + t * step) > solver.t_loss(v) + alpha * t * np.dot(solver.t
        print('{} > {} + {}'.format(solver.t_loss(v + t * step), solver.t_loss(v), alp
        t = t * beta
    return v + t * step

def centering_step(Q, p, A, b, t, v0, eps):
    print('Centering step:')
    solver = Differentiator(Q, p, A, b, t, eps)
    v = v0
    while True:
#         print('v:')
#         print(v)
        grad, hessian = np.matrix(solver.t_gradient(v)).T, np.matrix(solver.t_hessian(
        newton_step = -np.linalg.inv(hessian) * grad
        squared_newton_decrement = (- grad.T * newton_step).item(0, 0)
        newton_step = newton_step.getA().T[0]

        print('squared newton decrement: ')
        print(squared_newton_decrement)

#         print('\nNewton step: ')
#         print(newton_step)

        if squared_newton_decrement < 2 * eps:
            break
        v = line_search(v, newton_step, solver)
    return v


def barr_method(Q, p, A, b, v0, eps, mu=2):
    t = 0.1
    v = v0
    while True:
        v = centering_step(Q, p, A, b, t, v, eps)
        if t > len(A)/eps:
            break
        t = mu * t
    return v, t
```

1.3 Exercise 3

Test your function on randomly generated matrices X and observations y with $\lambda = 10$. Plot precision criterion and gap $f(v_t) - f^*$ in semilog scale (using the best value found for f as a surrogate for f). Repeat for different values of the barrier method parameter $\mu = 2, 15, 50, 100, ...$ and check the impact on w. What would be an appropriate choice for *mu*?

1.3.1 Generating the data

```
In [530]: def generate(n, d, eps = 0.01):
              X = np.random.randn(d, n)
              w = np.random.randn(1, d)
              errors = np.random.randn(1, n)
              y = (np.dot(w, X) + errors > 0)

              return X, y, w
```

```
In [531]: n, d = 100, 10
```

```
In [532]: X, y, truth_w = generate(100, 10)
```

1.3.2 Construct the dual problem

```
In [533]: Q = np.identity(n)
          p = 2 * y[0]
          A = np.concatenate((X, -X))
          b = np.ones(n)
          v0 = np.zeros(n)
          eps = 1e-5
```

1.3.3 Find the optimal solution for the dual problem

```
In [534]: hat_v, hat_t = barr_method(Q, p, A, b, v0, eps, mu=2)
```

```
Centering step:
squared newton decrement:
5.961612271523791
Performing linear search:
squared newton decrement:
4.862234818168254e-09
Centering step:
squared newton decrement:
0.005048465394669288
Performing linear search:
squared newton decrement:
7.517853485304915e-08
Centering step:
squared newton decrement:
0.01991517344983214
```

```
Performing linear search:
squared newton decrement:
4.445666130688041e-06
Centering step:
squared newton decrement:
0.07595613947508348
Performing linear search:
squared newton decrement:
0.00021884007010569874
Performing linear search:
squared newton decrement:
1.1599440639363917e-08
Centering step:
squared newton decrement:
0.27475196789357526
Performing linear search:
squared newton decrement:
0.007220121281321462
Performing linear search:
squared newton decrement:
2.6851404532102028e-05
Performing linear search:
squared newton decrement:
4.1871223705181983e-10
Centering step:
squared newton decrement:
0.8236944094289637
Performing linear search:
squared newton decrement:
0.09609452712649263
Performing linear search:
squared newton decrement:
0.005423791999877515
Performing linear search:
squared newton decrement:
2.4017370001795313e-05
Performing linear search:
squared newton decrement:
4.80947699006057e-10
Centering step:
squared newton decrement:
1.915502235753106
Performing linear search:
-196.48895807555584 > -196.1216778028373 + -0.4788755589382765
squared newton decrement:
0.2714031628695348
Performing linear search:
squared newton decrement:
```

```
0.03304729456198913
Performing linear search:
squared newton decrement:
0.0008547240569439492
Performing linear search:
squared newton decrement:
6.811310750394926e-07
Centering step:
squared newton decrement:
3.4492309825011422
Performing linear search:
-396.93646403274 > -397.0517945552553 + -0.8623077456252856
-397.6502521020384 > -397.0517945552553 + -0.776076971062757
squared newton decrement:
0.4464568205870718
Performing linear search:
squared newton decrement:
0.056601472646633816
Performing linear search:
squared newton decrement:
0.0016412614947128716
Performing linear search:
squared newton decrement:
2.2846490909985562e-06
Centering step:
squared newton decrement:
5.065639696618139
Performing linear search:
-801.394791332403 > -802.9628294147345 + -1.2664099241545348
-803.2202342282102 > -802.9628294147345 + -1.1397689317390813
squared newton decrement:
0.9626811508774143
Performing linear search:
squared newton decrement:
0.18166346627012528
Performing linear search:
squared newton decrement:
0.009263602239158633
Performing linear search:
squared newton decrement:
4.2884828233544856e-05
Performing linear search:
squared newton decrement:
1.556461262695049e-09
Centering step:
squared newton decrement:
6.49578809190868
Performing linear search:
```

```
-1615.8179883127998 > -1619.9496143752276 + -1.62394702297717
-1619.580208460762 > -1619.9496143752276 + -1.461552320679453
-1620.9671823035517 > -1619.9496143752276 + -1.3153970886115078
squared newton decrement:
0.7606118323010608
Performing linear search:
squared newton decrement:
0.09907662772590325
Performing linear search:
squared newton decrement:
0.002112544555275155
Performing linear search:
squared newton decrement:
1.4660581806220393e-06
Centering step:
squared newton decrement:
7.620217490078926
Performing linear search:
-3252.5028904675114 > -3259.8752739832016 + -1.9050543725197315
-3258.8072144185253 > -3259.8752739832016 + -1.7145489352677583
-3260.7736202435126 > -3259.8752739832016 + -1.5430940417409826
squared newton decrement:
1.0218598954921634
Performing linear search:
squared newton decrement:
0.15617522471651263
Performing linear search:
squared newton decrement:
0.004270598545103989
Performing linear search:
squared newton decrement:
3.998674963445732e-06
Centering step:
squared newton decrement:
8.458462416528265
Performing linear search:
-6534.98617173191 > -6546.152167628915 + -2.1146156041320663
-6544.361964650502 > -6546.152167628915 + -1.9031540437188597
-6546.8681089587635 > -6546.152167628915 + -1.712838639346974
squared newton decrement:
1.2640583569623012
Performing linear search:
squared newton decrement:
0.20784045202729062
Performing linear search:
squared newton decrement:
0.006596323496914178
Performing linear search:
```

```
squared newton decrement:
7.771047704199464e-06
Centering step:
squared newton decrement:
9.049069262666224
Performing linear search:
-13110.161796565179 > -13125.400330905919 + -2.262267315666556
-13122.884204602315 > -13125.400330905919 + -2.0360405840999003
-13125.883408342715 > -13125.400330905919 + -1.8324365256899104
squared newton decrement:
1.4939174888898932
Performing linear search:
squared newton decrement:
0.2561576373441549
Performing linear search:
squared newton decrement:
0.008721540356773022
Performing linear search:
squared newton decrement:
1.1805562926428444e-05
Centering step:
squared newton decrement:
9.436800041932383
Performing linear search:
-26271.351537818944 > -26290.728623680432 + -2.3592000104830957
-26287.52038226799 > -26290.728623680432 + -2.123280009434786
-26290.96293023072 > -26290.728623680432 + -1.9109520084913076
squared newton decrement:
1.695219396075285
Performing linear search:
squared newton decrement:
0.3038270234572019
Performing linear search:
squared newton decrement:
0.010759805393525109
Performing linear search:
squared newton decrement:
1.5579082661733867e-05
Centering step:
squared newton decrement:
9.670045256862522
Performing linear search:
-52604.93681014501 > -52628.27970030972 + -2.4175113142156306
-52624.48544710137 > -52628.27970030972 + -2.175760182794068
-52628.297159323556 > -52628.27970030972 + -1.958184164514661
-52629.959017799585 > -52628.27970030972 + -1.762365748063195
squared newton decrement:
0.8327962223855289
```

```
Performing linear search:
squared newton decrement:
0.07131288538351656
Performing linear search:
squared newton decrement:
0.0005570229101799309
Performing linear search:
squared newton decrement:
3.815034568157461e-08
Centering step:
squared newton decrement:
9.84701022710538
Performing linear search:
-105276.20546331687 > -105310.28816256313 + -2.461752556776345
-105305.92632305455 > -105310.28816256313 + -2.2155773010987105
-105310.1081183777 > -105310.28816256313 + -1.9940195709888398
-105311.89810635745 > -105310.28816256313 + -1.7946176138899559
squared newton decrement:
0.9015164582707981
Performing linear search:
squared newton decrement:
0.08189555168673283
Performing linear search:
squared newton decrement:
0.0006936219776729942
Performing linear search:
squared newton decrement:
5.3260711450158625e-08
Centering step:
squared newton decrement:
9.91988168190661
Performing linear search:
-210641.38450898346 > -210681.2538100845 + -2.4799704204766524
-210676.60090494965 > -210681.2538100845 + -2.2319733784289872
-210680.97165838603 > -210681.2538100845 + -2.0087760405860884
-210682.82503826267 > -210681.2538100845 + -1.8078984365274797
squared newton decrement:
0.9352770274750329
Performing linear search:
squared newton decrement:
0.08758291731277833
Performing linear search:
squared newton decrement:
0.0007741328232903635
Performing linear search:
squared newton decrement:
6.21485067391564e-08
Centering step:
```

```
squared newton decrement:
9.957712487084478
Performing linear search:
-421384.75300640403 > -421430.1155383104 + -2.4894281217711196
-421425.2924337431 > -421430.1155383104 + -2.240485309594008
-421429.7752554583 > -421430.1155383104 + -2.016436778634607
-421431.66453996365 > -421430.1155383104 + -1.8147931007711464
squared newton decrement:
0.953827651674636
Performing linear search:
squared newton decrement:
0.09096624989642388
Performing linear search:
squared newton decrement:
0.0008290477279907853
Performing linear search:
squared newton decrement:
6.945734251376247e-08
Centering step:
squared newton decrement:
9.976955120418442
Performing linear search:
-842884.4654182543 > -842934.7701398308 + -2.4942387801046104
-842929.8542659093 > -842934.7701398308 + -2.2448149020941495
-842934.3987382024 > -842934.7701398308 + -2.0203334118847347
-842936.3072067635 > -842934.7701398308 + -1.8183000706962613
squared newton decrement:
0.9635564168935645
Performing linear search:
squared newton decrement:
0.09281870991475775
Performing linear search:
squared newton decrement:
0.0008616620104882098
Performing linear search:
squared newton decrement:
7.443470499874035e-08
Centering step:
squared newton decrement:
9.986655400550223
Performing linear search:
-1685896.453536442 > -1685951.0106805037 + -2.4966638501375558
-1685946.0462514258 > -1685951.0106805037 + -2.2469974651238003
-1685950.6231633516 > -1685951.0106805037 + -2.02229771861142
-1685952.5415596326 > -1685951.0106805037 + -1.8200679467502783
squared newton decrement:
0.9685391096241939
Performing linear search:
```

```
squared newton decrement:
0.093788982458123
Performing linear search:
squared newton decrement:
0.0008795161079397847
Performing linear search:
squared newton decrement:
7.739134840985944e-08
Centering step:
squared newton decrement:
9.991524627335442
Performing linear search:
-3371932.3715591403 > -3371990.4231714425 + -2.4978811568338606
-3371985.4338860326 > -3371990.4231714425 + -2.2480930411504745
-3371990.0274519096 > -3371990.4231714425 + -2.023283737035427
-3371991.950899623 > -3371990.4231714425 + -1.8209553633318847
squared newton decrement:
0.9710605848572827
Performing linear search:
squared newton decrement:
0.09428563177437908
Performing linear search:
squared newton decrement:
0.0008888674833411343
Performing linear search:
squared newton decrement:
7.901021263409442e-08
Centering step:
squared newton decrement:
9.993963906746195
Performing linear search:
-6744015.391628095 > -6744076.179594794 + -2.498490976686549
-6744071.1777316965 > -6744076.179594794 + -2.248641879017894
-6744075.779737206 > -6744076.179594794 + -2.0237776911161047
-6744077.7057329975 > -6744076.179594794 + -1.8213999220044943
squared newton decrement:
0.9723289172474955
Performing linear search:
squared newton decrement:
0.09453689843962296
Performing linear search:
squared newton decrement:
0.0008936543175197343
Performing linear search:
squared newton decrement:
7.985820036159783e-08
Centering step:
squared newton decrement:
```

```
9.995184698825106
Performing linear search:
-13488191.810507992 > -13488254.623898266 + -2.4987961747062766
-13488249.615708323 > -13488254.623898266 + -2.248916557235649
-13488254.221962346 > -13488254.623898266 + -2.0240249015120844
-13488256.149237825 > -13488254.623898266 + -1.8216224113608759
squared newton decrement:
0.9729649893557116
Performing linear search:
squared newton decrement:
0.0946632751934209
Performing linear search:
squared newton decrement:
0.0008960761899672777
Performing linear search:
squared newton decrement:
8.029221521478473e-08
Centering step:
squared newton decrement:
9.995795383085005
Performing linear search:
-26976554.231290035 > -26976618.443969518 + -2.498948845771251
-26976613.432606544 > -26976618.443969518 + -2.249053961194126
-26976618.04099209 > -26976618.443969518 + -2.0241485650747135
-26976619.96890883 > -26976618.443969518 + -1.8217337085672423
squared newton decrement:
0.9732835037131584
Performing linear search:
squared newton decrement:
0.09472665053267924
Performing linear search:
squared newton decrement:
0.0008972943149119473
Performing linear search:
squared newton decrement:
8.051202078518721e-08
Centering step:
squared newton decrement:
9.99610079316616
Performing linear search:
-53953287.90902988 > -53953353.015580095 + -2.49902519829154
-53953348.00262822 > -53953353.015580095 + -2.249122678462386
-53953352.61208133 > -53953353.015580095 + -2.024210410616148
-53953354.54031903 > -53953353.015580095 + -1.821789369554533
squared newton decrement:
0.9734428796019077
Performing linear search:
squared newton decrement:
```

```
0.09475838470945003
Performing linear search:
squared newton decrement:
0.0008979051865041495
Performing linear search:
squared newton decrement:
8.062282039869414e-08
Centering step:
squared newton decrement:
9.996253510402234
Performing linear search:
-107906763.4549505 > -107906829.09027112 + -2.4990633776005584
-107906824.07652423 > -107906829.09027112 + -2.2491570398405027
-107906828.68651156 > -107906829.09027112 + -2.0242413358564524
-107906830.61490986 > -107906829.09027112 + -1.8218172022708072
squared newton decrement:
0.9735225946195277
Performing linear search:
squared newton decrement:
0.09477426400960676
Performing linear search:
squared newton decrement:
0.0008982110828294321
Performing linear search:
squared newton decrement:
8.067778081323333e-08
```

In [536]:
```python
print('For t = {}, estimated v = {}'.format(hat_t, hat_v))
```

```
For t = 3355443.2, estimated v = [-0.29769446 -0.42638632 -0.74557201 -0.20524725  0.23191902 -0
  0.04431438 -0.9092493  -0.69344959 -0.16914976 -1.10533629 -0.50539352
 -0.2820204  -0.08659938 -0.38324512 -0.84222879  0.1644654  -0.49239971
 -0.41979449 -0.61016285  0.1436708  -0.42187794 -0.44498955 -0.37435205
 -0.79038386 -0.34427991 -0.14015162 -0.21718662 -0.38519728 -0.5276036
 -0.04999381  0.16830248 -0.75303872 -0.70626894  0.15177644 -0.13880239
 -0.89064123 -0.56805602 -0.65733595 -0.80183313 -0.07537877 -0.35376797
 -0.84754465 -0.78591256 -0.03522196 -0.95057251 -0.28834724 -0.19455472
 -0.17496384 -1.07562819 -0.22744126 -0.33807814 -0.36977053 -0.14404294
 -0.71288038 -0.78294752 -0.10361717 -0.36191762 -0.77053197  0.00424833
 -0.72376468  0.160863   -0.60053468 -0.87763478 -0.82076279 -0.27599629
  0.12255754 -0.86166363  0.12951415 -0.43432758 -0.39650022 -0.74757218
 -0.96502182 -0.50944506 -0.98630614 -0.65959771  0.45442377 -0.65531457
 -0.8096264  -0.44407562 -0.59701379 -0.11397101 -0.27026805 -0.84470356
 -0.79996554 -0.86105944  0.00364943 -0.53812693 -0.501526    0.02802527
 -0.39310645 -0.08089398 -0.83213003 -0.11782457  0.13570512 -0.41406252
 -0.67306097 -0.13590495 -0.3922528  -0.2493648 ]
```

1.3.4 Obtaining the optimal solution for the original problem using the solution for the dual problem

In [ ]:

1.3.5 Comparing with the expected solution w, and varying the hyperparameters, in particular μ

In [ ]: