Convex Optimization: Homework #3

Soo Min Kwon

March 4, 2020

1 Recovering Corrupted Guitar Image

Consider the following corrupted guitar image:

Figure 1: Corrupted Guitar Image

The dimensions of this corrupted guitar image is 511×205, but we transpose this image to 205×511, as i refers to the horizontal component and j refers to the vertical component (shown later). We are given that this corrupted image is created through a multiplication with an affine function. This relationship is mathematically formulated as the following:

$$e_c(i, j) = e(i, j)r(i, j),$$

where $e_c(i, j)$, $e(i, j)$ refers to the corrupted image and true image respectively, and $r(i, j) = ai + bj + c$.

1.1 Convex Formulation

To solve for the true image, we're given that the true image is uniform, $e(i, j) = 255$ from $i < 50$, $j < 250$. We can use this information and solve the following convex optimization problem:

$$
\begin{aligned}
\underset{a,b,c}{\text{minimize}} \quad & ||e_c(i,j) - e(i,j)r(i,j)||_F \\
\text{subject to} \quad & r(i,j) = ai + bj + c \\
& ai + bj + c \geq 0 \\
& ai + bj + c \leq 1 \\
& i = 1, ..., 49 \\
& j = 1, ..., 249
\end{aligned}
\tag{1}
$$

We can see that this problem is convex by looking at the objective function as well as the constraints. The objective function is the Frobenius norm, which is essentially the matrix form of the Euclidean norm, which is convex. The constraints are simply affine functions which is also convex.

The only modification made from the optimization formulation above is that it is not necessary to use all $0 \leq i \leq 49$ and $0 \leq j \leq 249$. We can use a fraction of those variables, to reduce run time and make computation more feasible. Instead consider the following problem:

$$
\begin{aligned}
\underset{a,b,c}{\text{minimize}} \quad & ||e_c(i,j) - e(i,j)r(i,j)||_F \\
\text{subject to} \quad & r(i,j) = ai + bj + c \\
& ai + bj + c \geq 0 \\
& ai + bj + c \leq 1 \\
& i = 1, ..., 20 \\
& j = 1, ..., 20
\end{aligned}
\tag{2}
$$

This is essentially the same as the problem above, except we use less i and j values. Since we know $e_c(i, j)$ and $e(i, j)$ from $0 \leq i \leq 20$ and $0 \leq j \leq 20$, we can throw it into the CVX solver and solve for a, b, and c from (2).

NOTE: Of course, if we consider more i and j values, the objective function value would converge to a smaller value. However, these values were chosen so that run time is reasonable and the solver converges to an optimal solution.

1.2 Results

The CVX solver returns the following a, b, and c values and objective function value:

```
Objective Function Value:  5.677822552505315
Solved A:  -0.0019722838587915613
Solved B:  -0.000999263068284205
Solved C:  0.9949551852220403
```

Figure 2: Returned objective function value, and solved a, b, c parameters

If we plug these solved parameters into the equation of $r(i, j)$ and divide the corrupted image by $r(i, j)$ to get back the true image, we get the following image:

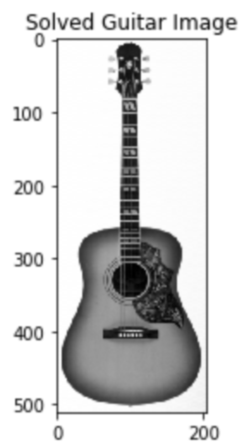

Figure 3: Solved true guitar image

To analyze if the solutions were correct, we show the first 5×5 matrix of the true guitar image:

```
[[255.28788007 255.54453104 254.79460527 255.05127641 255.3084652 ]
 [254.78787163 255.0445292  255.30170437 255.55939871 254.80647699]
 [255.2949439  254.54253516 254.799716   255.05741705 255.3156399 ]
 [254.79295537 255.05064274 255.30885186 255.56758434 254.81165586]
 [255.30206419 254.5466528  254.80486766 255.06360692 255.32287218]]
```

We can see that the true image has values close to 255, which is what we were given. In addition, the solution was tested to make sure that $ai + bj + c \geq 0$ and $ai + bj + c \leq 1$.

2 River Spline Fitting

Consider the following curved river image:

Figure 4: Curved River Picture

For this problem, we generate (x, y) data using Python's ginput. We take points along the contour of the river shown below:

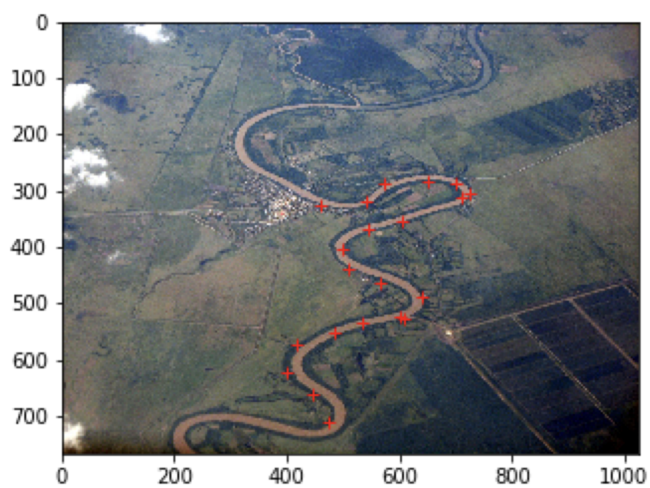

Figure 5: Red cross marks are the points taken along the river

There are a total of 21 points taken. We try to use three ($M = 3$) cubic splines to fit this curve, with an even distribution of 7 points per cubic.

2.1 Convex Formulation

To fit this curve, we can formulate the following convex optimization problem:

$$
\begin{aligned}
\underset{A,B}{\text{minimize}} \quad & ||x_i(t) - \hat{x_i}(t)||_2 + ||y_i(t) - \hat{y_i}(t)||_2 \\
\text{subject to} \quad & \hat{x_i(t)} - A_i \times T = 0 \\
& \hat{y_i(t)} - B_i \times T = 0 \\
& \hat{x_i(1)} - \hat{x_{i+1}}(0) = 0 \\
& \hat{y_i(1)} - \hat{y_{i+1}}(0) = 0 \\
& i = 1, ..., 3
\end{aligned} \tag{3}
$$

To summarize the convex optimization problem above:

- The variable i refers to the number of M used, where M is the number of cubic splines used to fit the curve.
- $x_i(t)$ and $\hat{x_i}(t)$ for $i = 1, ..., 3$ refers to the true x-coordinates and the measured x-coordinates, respectively.
- $y_i(t)$ and $\hat{y_i}(t)$ for $i = 1, ..., 3$ refers to the true y-coordinates and the measured y-coordinates, respectively.
- A_i is an array with entry values

$$
\begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix}
$$

- Likewise for B_i.
- T is a matrix of t values ranging from $[0, 1]$, where the dimensions of the row refers to the number of points taken per cubic (7 in this case), and the columns are t^3, ..., 1. For example, in this case, our T will be of dimensions 7×4.
- The first two constraints is simply that $\hat{x_i}(t)$ and $\hat{y_i}(t)$ have entries from the multiplication of A, B and T.

- The next two constraints is that the endpoints must meet from the the end of one cubic to the beginning of the other.
- **NOTE:** When solving the optimization problem, the constraints of continuity of the first and second derivative did not change the answer (answer being the plot of the curve), so I decided to omit those constraints.

2.2 Results

With the CVX solver, we get the following results:

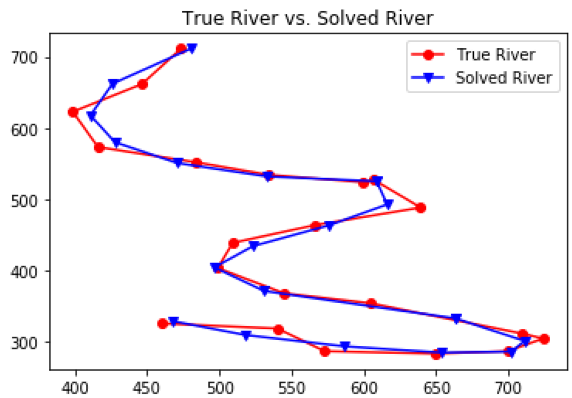

Figure 6: Red represents the true river plotted and blue represents the solved river through CVX

Some important notes:

- The plots are a reflection along the x-axis of the points taken from the contour of the river. I am not sure why ginput gave me the reflected points of what I took...
- In the code, I solve for a matrix of A and B values rather than column vectors of A_i and B_i. This would also mean that T would be replicated along the columns respectively.