

# CMPS 351

## Assignment 1

```
In [1]: import numpy as np
        from numpy import linalg as la
        import matplotlib.pyplot as plt
```

### Rosenbrock

```
In [2]: def rosenbrock(x):
        f = 10 * (x[1] - x[0]**2)**2 + (1 - x[0])**2
        return f
```

### Rosenbrock Gradient

```
In [3]: def rosenbrock_gradient(x):
        g = np.zeros(2)
        g[0] = -40*x[0]*(x[1] - x[0]**2) + 2*x[0] - 2
        g[1] = 20*x[1] - 20*x[0]**2
        return g
```

### Backtrack Line Search

```
In [4]: def backtrack_linesearch(f, gk, pk, xk, alpha = 0.01, beta = 0.6):
        t = 1
        while ( f(xk + t*pk) > f(xk) + alpha*t*gk@pk):
            t *= beta
        return t
```

### Steepest Descent Backtrack

```
In [5]: def steepest_descent_backtrack(f, grad, x0, tol = 1e-5):
        x = x0
        history = np.array([x0])
        while (la.norm(grad(x)) > tol):
            p = -grad(x)
            t = backtrack_linesearch(f, grad(x), p, x)
            x += t * p
            history = np.vstack((history, x))
        return x, history
```

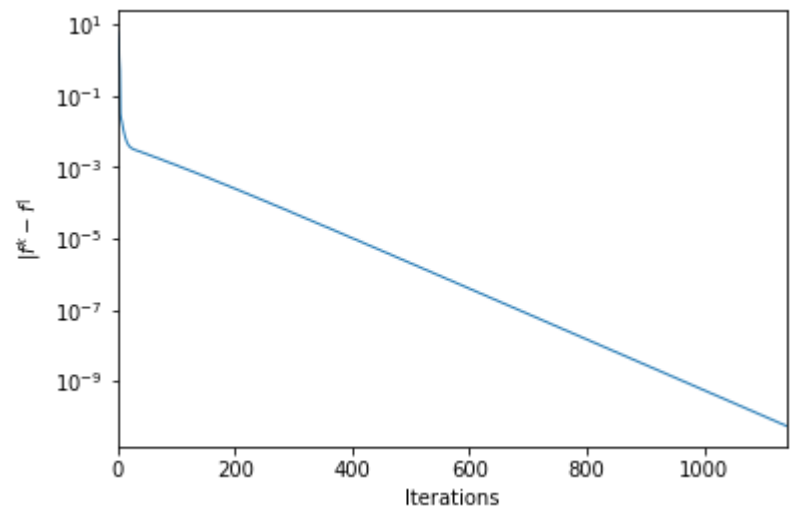
### Plotting the performance

```
In [6]: x0 = np.array([-1.2, 1.0])
xstar, history = steepest_descent_backtrack(rosenbrock, rosenbrock_gradient, x0)

nsteps = history.shape[0]
fhist = np.zeros(nsteps)

for i in range(nsteps):
    fhist[i] = rosenbrock(history[i,:])

plt.figure()
plt.autoscale(enable=True, axis='x', tight=True)
plt.semilogy(np.arange(0, nsteps), fhist, linewidth=1)
plt.xlabel('Iterations')
plt.ylabel(r'$|f^k - f|$')
plt.show()
```



We see here that initially within the first 50 iterations the error decreases very rapidly, but then the progress becomes very slow. This is due to the contours being very elongated causing the countour lines to be nearly parallel.

Because we are moving in the direction of the steepest descent we find ourselves almost zigzagging in place and making very slow progress towards the minimum.

### Plotting the contours

```
In [7]: x0 = np.arange(-1.2, 1.5, 0.01)
x1 = np.arange(-0.5, 1.5, 0.01)

F = np.zeros((x1.shape[0], x0.shape[0]))

for i in range(F.shape[0]):
    for j in range(F.shape[1]):
        x = [x0[j], x1[i]]
        F[i, j] = rosenbrock(x)

plt.figure('Contours')
plt.plot(history[:,0], history[:,1], linewidth=0.5, color='red', marker='o', markersize=2)
plt.plot([1],[1], color='green', marker='x', markersize=8)
plt.contour(x0, x1, F, 50, linewidths=0.5)
plt.axis('equal')
plt.colorbar()
```

Out[7]: <matplotlib.colorbar.Colorbar at 0x13918e6d860>

