# Simulated Annealing Demo 4

April 7, 2019

## 1 Simulated annealing Demo 4

```python
In [153]: import numpy as np
          import matplotlib.pyplot as plt

          from IPython.display import clear_output, display
```

```python
In [154]: def _display_progress(current_progress, progress):
              idx = np.where(progress - current_progress >= 0)[0][0] + 1
              perc = int(100. * idx / progress.shape[0])
              clear_output(wait = True)
              prg = "["
              prg += "".join(["=" for i in range(int(perc / 5))])
              prg += "".join(["." for i in range(20 - int(perc / 5))])
              prg += "".join("]")
              display(prg + "        " + 'Progress: ' + str(perc) + "%")
```

```python
In [155]: import sys

          sys.path.insert(0, "../../src")
```

```python
In [156]: from env.light_grid import LightGrid
          from walker.light_walker import LightWalker
          from walker.simulated_annealing import SimulatedAnnealing
```
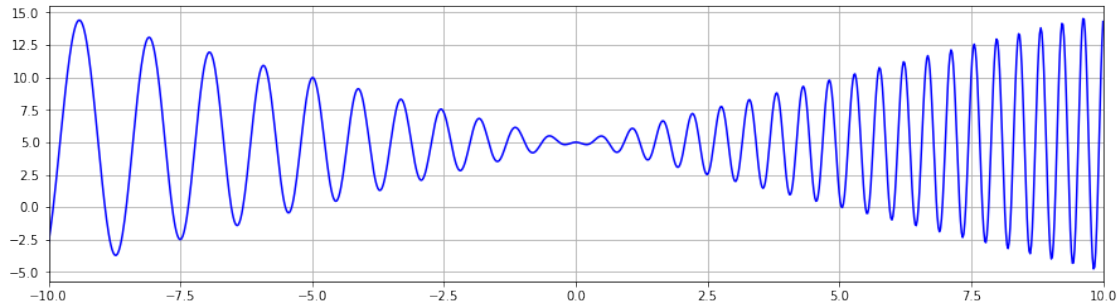
```python
In [157]: x_l, x_r = -10, 10
```

```python
In [158]: def global_optimizer(x):
              return 5 - x * np.sin(10 * x + .3 * x ** 2)
```

```python
In [159]: xx = np.linspace(x_l, x_r, 1000)
```

```python
In [227]: fig, ax = plt.subplots()

          fig.set_figwidth(15)

          ax.set_xlim(x_l, x_r)
          ax.plot(xx, global_optimizer(xx), c = 'b')
          ax.grid()
          plt.show()
```

```
In [228]: annealing = SimulatedAnnealing(initial_temperature = 10000, temperature_decay = 0.99
```

```
In [229]: grid = LightGrid(bounded = True, bounds = [(x_l, x_r)], dr = np.ones(2) * 1e-3, dim =
          grid.add_processor(global_optimizer, name = "optimizer")
```

```
In [230]: no_of_walkers = 10
          no_of_steps = int(5e3)

          def get_initial_coordinates(no, x_bounds):
              return np.random.uniform(low = x_bounds[0], high = x_bounds[1], size = no)
```
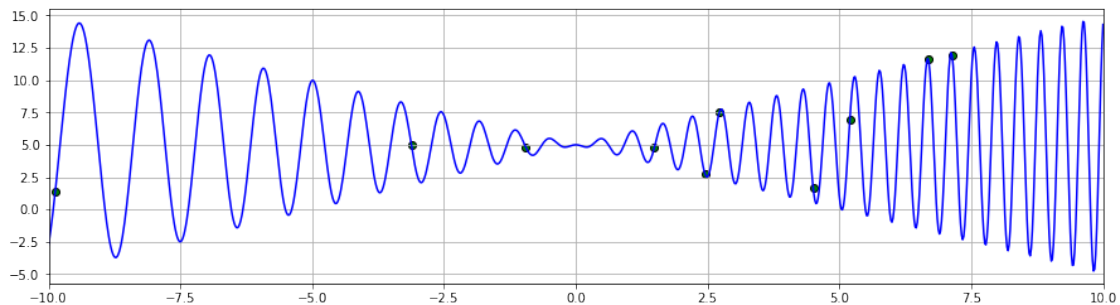
```
In [231]: walkers = []
```

```
In [232]: coordinates = get_initial_coordinates(no_of_walkers, (-10, 10))
```

```
In [233]: ax.scatter(coordinates, global_optimizer(coordinates), c = 'g', edgecolor = 'k')
          fig
```

Out[233]:



```
In [234]: for i in range(no_of_walkers):
              wlk = LightWalker(dim = 1, \
                      data_collection_keys = ['optimizer'], \
                      max_no_of_steps = no_of_steps, \
                      initial_position = np.array([coordinates[i]]), \
                      next_step_processor = annealing)
              walkers.append(wlk)
```

```
In [235]: drs = [np.ones(2, dtype = np.float64) * 1e-1, \
                 np.ones(2, dtype = np.float64) * 1e-2, \
                 np.ones(2, dtype = np.float64) * 1e-3, \
                 np.ones(2, dtype = np.float64) * 1e-4]

In [236]: from random import choice

In [237]: prg = np.linspace(0, no_of_steps, no_of_steps)

          global_min = 1e5
          coord_min = 0

          for i in range(no_of_steps):
              next_positions = 0
              for walker in walkers:
                  curr = walker.get_current_position()
                  data_collector = list(key for key in walker.data.keys())[0]
                  #dr = np.random.norma(low = 0, high = 1, size = 1)
                  dr = choice(drs)
                  next_positions = grid._next(curr, dr = dr)
                  eng_curr = grid.processors[data_collector](curr[0])
                  eng_next = np.array([grid.processors[data_collector](next_pos[0]) for next_po
                  walker.walk(possible_states = next_positions, energies = eng_next, current_en
                  tmp = walker.get_current_position()
                  if global_optimizer(tmp) < global_min:
                      coord_min = curr
                      global_min = global_optimizer(tmp)
              _display_progress(i, prg)

'[====================]         Progress: 100%'


In [238]: for wlk in walkers:
              ax.plot(wlk.visited[:, 0], global_optimizer(wlk.visited[:, 0]), c = 'r')
          ax.scatter(coord_min, global_min, c = 'k', s = 50)
          fig

Out[238]:
```
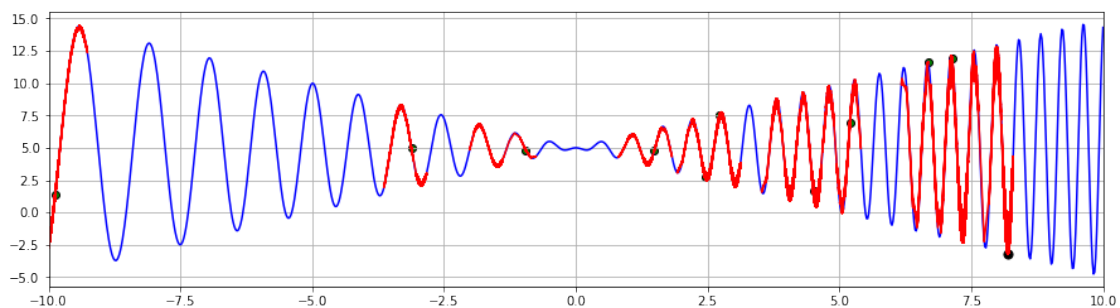
```
In [239]: global_min, coord_min

Out[239]: (array([-3.19556541]), array([8.19681433]))

In [ ]:
```