

```
1: import lib
2: import numpy as np
3: import json
4:
5:
6: def iterate(self):
7:     self._x_value = self._start
8:     self._old_x_value = None
9:     self._iteration = 0
10:    self._m = np.zeros(self._x_value.shape, dtype=np.float64)
11:    self._v = np.zeros(self._x_value.shape, dtype=np.float64)
12:    self._converged_value = False
13:    self._grad_value = self._gradient(self._x_value)
14:
15:    yield self.state_dict()
16:
17:    while not self._converged_value:
18:        if self._max_iter > 0 and self._iteration > self._max_iter:
19:            break
20:        self._grad_value = self._gradient(self._x_value)
21:        self._m = self._beta * self._m + (1-self._beta)*self._grad_value
22:        # grad_value * grad_value gives element-wise product of np array
23:        self._v = self._beta2 * self._v + (1-self._beta2) * (self._grad_value*self._grad_value)
24:        self._old_x_value = self._x_value
25:        self._iteration += 1
26:        m_hat = self._m / (1-(self._beta ** self._iteration))
27:        v_hat = np.array(self._v / (1-(self._beta2 ** self._iteration)))
28:        v_hat_aug = v_hat**(0.5) + self._epsilon
29:        self._adam_grad = m_hat / v_hat_aug
30:        self._x_value = self._x_value - self._step_size * self._adam_grad
31:        self._converged_value = self._converged(self._x_value, self._old_x_value)
32:        yield self.state_dict()
```