Simulated Annealing Algorithm

In [53]:
```
import francium.algorithms.simulated_annealing as sa
import francium.core.eval_functions as eval_functions
from francium.core import State
```

using an environment with z = x^2 + y^2

In [54]:
```
agent = sa.Agent(step_size=1e-1)
env = sa.Environment(x_bounds=(-5.0, 5.0), y_bounds=(-5.0, 5.0), eval_func=eval_functions.convex_x_square)
solver = sa.Solver(agent=agent, environment=env, initial_temp=100.0, final_temp=0.0, iters_per_temp=100, temp_reduction="linea
r")
```

In [55]:
```
solver.init_solver(
    init_state=State({
        'x': 4.0,
        'y': 2.0,
        'z': env.evaluation_func(4.0, 2.0)
    })
)
```

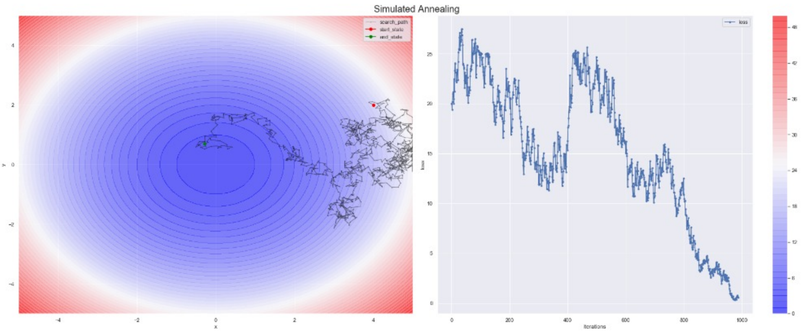
```
[ 2020-12-06 19:57:23,948 - francium.algorithms.simulated_annealing.solver ] INFO: => Initialized Solver with State: {'x': 4.0,
'y': 2.0, 'z': 20.0}
```

In [56]:
```
for episode in range(10):
    trainable = solver.train_step()
    if not trainable:
        break
```

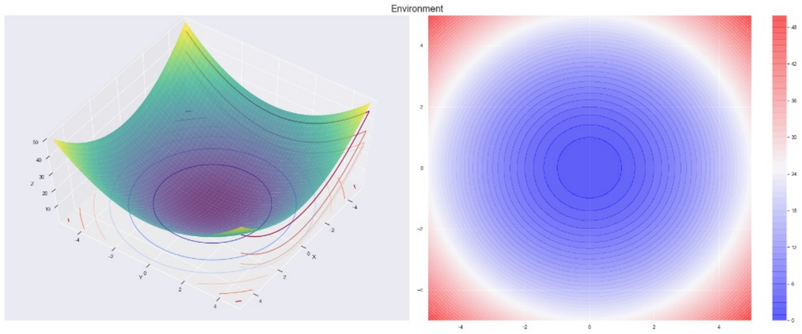
In [57]:
```
solver.memory.best_episode
```

Out[57]:
```
{'x': 0.048191555495727595, 'y': 0.5528596995811312, 'z': 0.3079762734420365}
```

In [58]:
```
solver.plot_history()
```



In [59]:
```
env.plot_environment()
```



using an environment with z = 5 * sin(x^2 + y^2) + x^2 + y^2

In [60]:
```
agent = sa.Agent(step_size=1e-1)
env = sa.Environment(x_bounds=(-5.0, 5.0), y_bounds=(-5.0, 5.0), eval_func=eval_functions.sinx_plus_x)
solver = sa.Solver(agent=agent, environment=env, initial_temp=100.0, final_temp=0.0, iters_per_temp=100, temp_reduction="linea
r")
```

In [61]:
```
solver.init_solver(
    init_state=State({
        'x': 4.0,
        'y': 2.0,
        'z': env.evaluation_func(4.0, 2.0)
    })
)
```

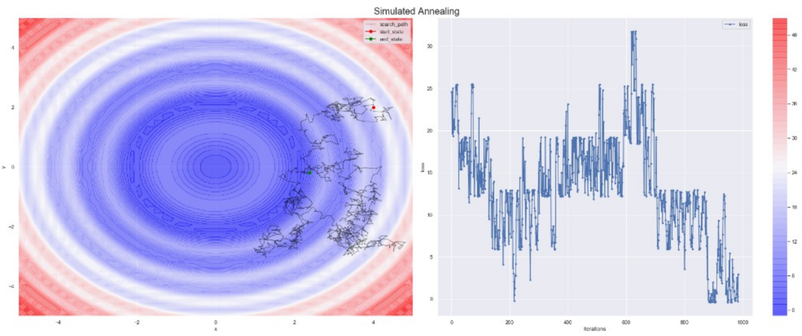
```
[ 2020-12-06 19:57:25,510 - francium.algorithms.simulated_annealing.solver ] INFO: => Initialized Solver with State: {'x': 4.0,
'y': 2.0, 'z': 24.56472625363814}
```

In [62]:
```
for episode in range(10):
    trainable = solver.train_step()
    if not trainable:
        break
```

In [63]:
```
solver.memory.best_episode
```

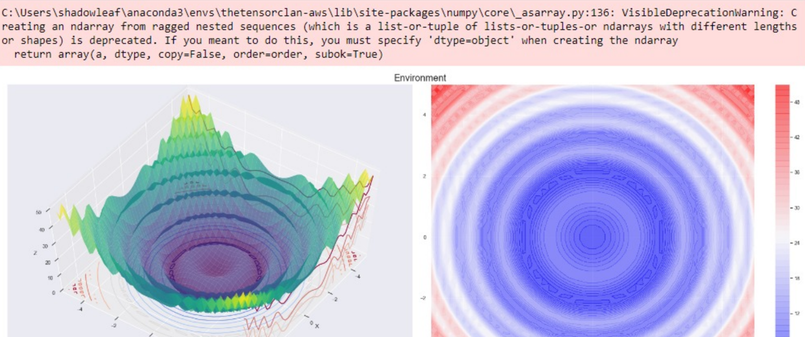
Out[63]:
```
{'x': 1.8416088700771585, 'y': -1.0496854976975198, 'z': -0.38718472068411014}
```

In [64]:
```
solver.plot_history()
```



In [65]:
```
env.plot_environment()
```

```
C:\Users\shadowleaf\anaconda3\envs\thetensorclan-aws\lib\site-packages\numpy\core\_asarray.py:136: VisibleDeprecationWarning: C
reating an ndarray from ragged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths
or shapes) is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order, subok=True)
```



In [65]:
```

```