**Name:** Okoye Adunife Kizito
**Student ID:** 100611918
**CSCI 4140U**

# Laboratory Seven

### Laboratory Activity

For your laboratory activity repeat the model fitting with 3 qubits. This is a small change to the code. Your test circuit will now need to have three qubits. Start with the previous circuit that produced a Bell state and just add a X gate to the third qubit. Cut and paste the resulting histogram and submit it as your laboratory report. The report must be a PDF or PNG file.

1. Start with the previous circuit that produced a Bell state and just add a X gate to the third qubit.

In [38]:
```
qc = QuantumCircuit(3,3)
qc.h(0)
qc.cx(0,1)
qc.x(2)
qc.measure(qc.qregs[0],qc.cregs[0])
print(execute(qc, Aer.get_backend('qasm_simulator'),noise_model=noise_model,shots=10000).result().get_counts())
```

```
{'000': 399, '001': 90, '010': 86, '011': 408, '100': 3742, '101': 796, '110': 805, '111': 3674}
```

2. Use the complete_meas_cal procedure to construct the circuits that we need to collect the data required for constructing the matrix.

In [31]:
```
qr = QuantumRegister(3)
meas_calibs, state_labels = complete_meas_cal(qr=qr, circlabel='mcal')

for circuit in meas_calibs:
    print('Circuit',circuit.name)
    print(circuit)
    print()
```

```
Circuit mcalcal_000
q42_0: |M|
q42_1: |M|
q42_2: |M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_001
q42_0: |X|M|
q42_1: |M|
q42_2: |M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_010
q42_0: |M|
q42_1: |X|M|
q42_2: |M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_011
q42_0: |X|M|
q42_1: |X|M|
q42_2: |M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_100
q42_0: |M|
q42_1: |M|
q42_2: |X|M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_101
q42_0: |X|M|
q42_1: |M|
q42_2: |X|M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_110
q42_0: |M|
q42_1: |X|M|
q42_2: |X|M|
 c2: 3/=0 1 2
```

```
Circuit mcalcal_111
q42_0: |X|M|
q42_1: |X|M|
q42_2: |X|M|
 c2: 3/=0 1 2
```

3. Run these circuits and fit the model. The CompleteMeasFitter procedure does the fitting

In [32]:
```
noise_model = get_noise(0.1)

backend = Aer.get_backend('qasm_simulator')
job = execute(meas_calibs, backend=backend, shots=1000, noise_model=noise_model)
cal_results = job.result()

meas_fitter = CompleteMeasFitter(cal_results, state_labels, circlabel='mcal')
print(meas_fitter.cal_matrix)
```

```
[[0.751 0.08  0.098 0.016 0.078 0.012 0.014 0.003]
 [0.078 0.728 0.015 0.087 0.005 0.072 0.    0.005]
 [0.075 0.004 0.716 0.079 0.008 0.002 0.077 0.01 ]
 [0.004 0.08  0.086 0.724 0.002 0.016 0.009 0.086]
 [0.078 0.009 0.008 0.    0.724 0.081 0.09  0.005]
 [0.01  0.091 0.    0.01  0.088 0.725 0.007 0.082]
 [0.003 0.    0.063 0.003 0.079 0.011 0.726 0.083]
 [0.001 0.008 0.014 0.081 0.016 0.081 0.077 0.726]]
```

4. Repeat the first section

In [35]:
```
qc = QuantumCircuit(3,3)
qc.h(0)
qc.cx(0,1)
qc.x(2)
qc.measure(qc.qregs[0],qc.cregs[0])

results = execute(qc, backend=backend, shots=10000, noise_model=noise_model).result()

noisy_counts = results.get_counts()
print(noisy_counts)
```

```
{'000': 391, '001': 79, '010': 95, '011': 446, '100': 3669, '101': 805, '110': 787, '111': 3728}
```

5. Error Mitigation Model histogram

In [36]:
```
# Get the filter object
meas_filter = meas_fitter.filter

# Results with mitigation
mitigated_results = meas_filter.apply(results)
mitigated_counts = mitigated_results.get_counts(0)

plot_histogram([noisy_counts, mitigated_counts], legend=['noisy', 'mitigated'])
```

Out[36]: