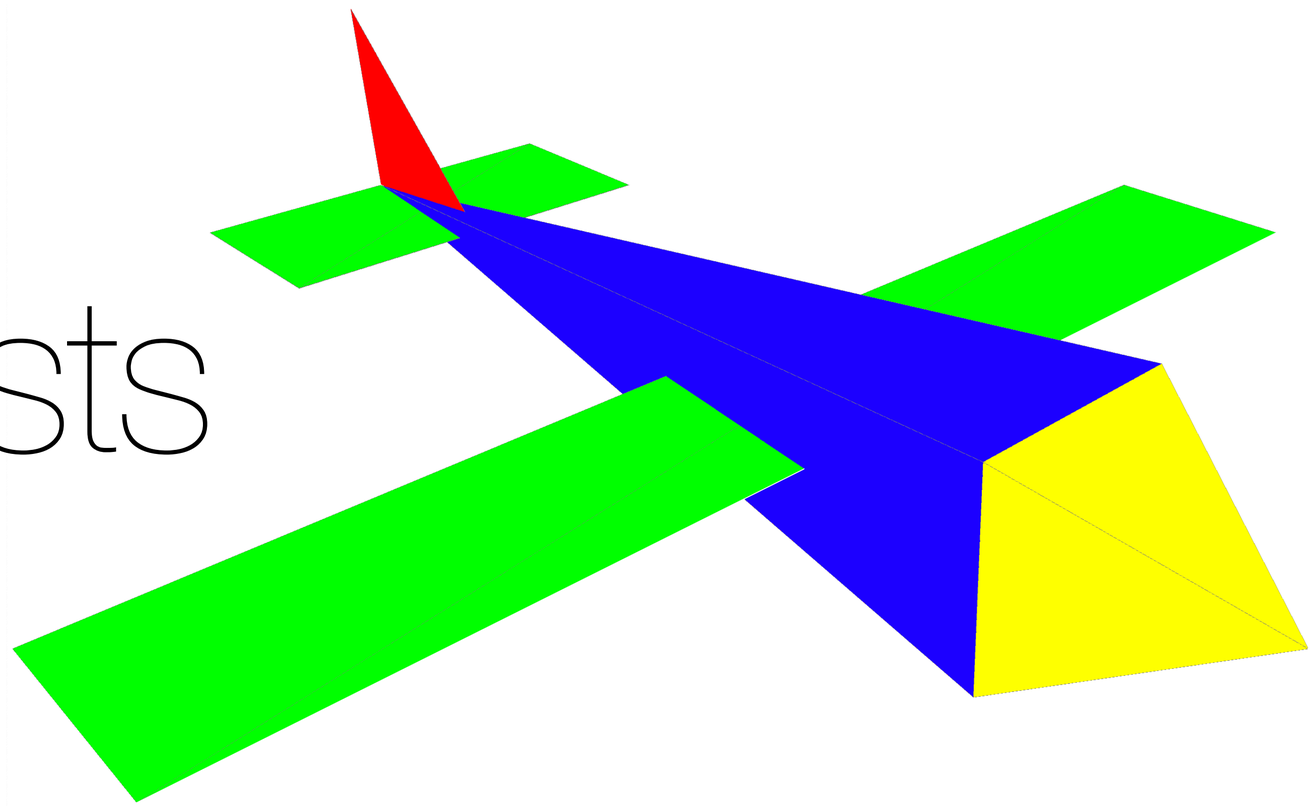
MAV Unit Tests

Final Project - EE 674



Dan Richards

Objectives

- Create unit tests for Chapter 3, 4, & 5
  - More efficient debugging & grading
  - Easy to update true values

Library

pytest

Format

- Unit Test folder
- Unit tests for each chapter
  - Test files—names contain “test”
  - True value files

Format

```python
def test_update_Mz():
    # initialize elements of the architecture
    mav = MavDynamics(SIM.ts_simulation)
    # set state
    mav._state = np.zeros((13,1),dtype=float)
    mav._state[3,0] = 25.0 # set u0
    mav._state[6,0] = 1.0 # set e0
    # set forces and moments
    forces_moments = np.zeros((6,1),dtype=float) # fx, fy, fz, Mx, My, Mz
    forces_moments[5,0] = 0.1 #set Mz
    # get update results
    mav.update(forces_moments)  # propagate the MAV dynamics
    #compare against true values
    assert np.allclose(mav._state, trueValues.state_Mz)
```

```python
state_Mz = np.array(    [[ 2.50000000e-01],
                         [-6.69747058e-20],
                         [-1.25415774e-14],
                         [ 2.50000000e+01],
                         [-7.17806614e-05],
                         [ 3.04437995e-11],
                         [ 1.00000000e+00],
                         [ 2.09665008e-07],
                         [ 3.07878132e-13],
                         [ 1.43561323e-06],
                         [ 8.38660032e-05],
                         [ 2.46302506e-10],
                         [ 5.74245291e-04]])
```

pytest Fixtures

```python
@pytest.fixture # create variables that can be used in each test function
def sim():
    # initialize mav
    mav = MavDynamics(SIM.ts_simulation)
    # set state
    mav._state = np.zeros((13,1),dtype=float)
    mav._state[3,0] = 25.0 # set u0
    mav._state[6,0] = 1.0 # set e0
    mav._Va = 25.0
    # initialize inputs
    delta = MsgDelta()
    delta.elevator =  0.0
    delta.aileron =   0.0
    delta.rudder =    0.0
    delta.throttle =  0.7

    # initialize wind
    wind = np.zeros((6,1))
    return mav, delta, wind
```

```python
def test_motor_thrust_torque(sim):
    mav, delta, wind = sim
    delta.throttle = 0.9
    thrust, torque = mav._motor_thrust_torque(delta.throttle)
    assert np.isclose(thrust, trueValues.thrust)
    assert np.isclose(torque, trueValues.torque)
```

Chapter 3 Tests

- chap3.MavDynamics.*_derivatives*
  - f_x, f_y, f_z, M_x, M_y, M_z
- chap3.MavDynamics.*update*
  - f_x, f_y, f_z, M_x, M_y, M_z

Chapter 4 Tests

- chap4.MavDynamics.*_motor_thrust_torque*
  - delta.throttle
- chap4.MavDynamics.*_forces_moments*
  - delta.aileron, delta.elevator, delta.rudder
- chap4.MavDynamics.*_update_velocity_data*
  - Wind - N, E, S, W

Chapter 5 Tests

- chap5.trim.***compute_trim***
- chap5.compute_model.***f_euler***
- chap5.compute_model.***df_dx***
- chap5.compute_model.***df_du***
- chap5.compute_model.***compute_ss_model***
- chap5.compute_model.***compute_tf_model***

How to Test

Run in terminal using commands:

```
cd /Users/mavsim_python
pytest -q unit_tests/unitTestsChap4.py
```

Pass

```
danada@dans-mbp-2 mavsim_python % pytest -q unit_tests/unitTestsChap5.py
......                                                                   [100%]
6 passed in 1.18s
```

Fail

```
_____________________________________________________________________________ test_df_du _____________________________________________________________________________
_________________

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)
>       assert np.allclose(B, trueValues.B)
E       assert False
E        +  where False = <function allclose at 0x10e02b8b0>(array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00,\n         0.00000000e+00],\n       [ 0.00000000e+00,  0.000
00...20e-06,\n        -9.87160320e-06],\n       [-3.62115122e-01,  0.00000000e+00,  5.01173501e+00,\n        -2.48813413e+01]]), array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],
\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n
 [0., 0., 0., 0.]]))
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0.
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.B

unit_tests/unitTestsChap5.py:122: AssertionError
------------------------------------------------------------------------------------ Captured stdout call ------------------------------------------------------------------------------------
------------------
Optimization terminated successfully    (Exit mode 0)
            Current function value: 2.586543198071751e-06
            Iterations: 13
            Function evaluations: 255
            Gradient evaluations: 13
elevator= -0.1247780701597401 aileron= 0.0018361809638628682 rudder= -0.000302608037876341 throttle= 0.6767522859047431
trim_state= [[-5.95888304e-15  0.00000000e+00 -1.00000000e+02  2.49687427e+01
   0.00000000e+00  1.24975516e+00  9.99687380e-01  0.00000000e+00
   2.50028494e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00]]
=================================================================================== short test summary info ===================================================================================
=================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

```
_____________________________________________________________________ test_df_du _____________________________________________________________________
-----------------

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)
>       assert np.allclose(B, trueValues.B)
E       assert False
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0.
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.B

unit_tests/unitTestsChap5.py:122: AssertionError
------------------------------------------------------------------------------------ Captured stdout call ------------------------------------------------------------------------------------
------------------
Optimization terminated successfully    (Exit mode 0)
            Current function value: 2.586543198071751e-06
            Iterations: 13
            Function evaluations: 255
            Gradient evaluations: 13
elevator= -0.1247780701597401 aileron= 0.0018361809638628682 rudder= -0.000302608037876341 throttle= 0.6767522859047431
trim_state= [[-5.95888304e-15  0.00000000e+00 -1.00000000e+02  2.49687427e+01
   0.00000000e+00  1.24975516e+00  9.99687380e-01  0.00000000e+00
   2.50028494e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00]]
================================================================================ short test summary info ================================================================================
==================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

```
----------------------------------------------------------------------------------
----------------

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)
>       assert np.allclose(B, trueValues.B)
E       assert False
```

```
_____________________________________________ test_df_du _____________________________________________
_________________

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)

E       assert False
E        +  where False = <function allclose at 0x10e02b8b0>(array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00,\n         0.00000000e+00],\n       [ 0.00000000e+00,  0.000
00...20e-06,\n        -9.87160320e-06],\n       [-3.62115122e-01,  0.00000000e+00,  5.01173501e+00,\n        -2.48813413e+01]]), array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],
\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n
 [0., 0., 0., 0.]]))
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0.
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.B

unit_tests/unitTestsChap5.py:122: AssertionError
_________________
Optimization terminated successfully    (Exit mode 0)
            Current function value: 2.586543198071751e-06
            Iterations: 13
            Function evaluations: 255
            Gradient evaluations: 13
elevator= -0.1247780701597401 aileron= 0.0018361809638628682 rudder= -0.000302608037876341 throttle= 0.6767522859047431
trim_state= [[-5.95888304e-15  0.00000000e+00 -1.00000000e+02  2.49687427e+01
   0.00000000e+00  1.24975516e+00  9.99687380e-01  0.00000000e+00
   2.50028494e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00]]
========================================================================== short test summary info ==========================================================================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

```
E       assert False
E        +  where False = <function allclose at 0x10e02b8b0>(array([[ 0.00000000e+00,  0.00000000e+00,
00...20e-06,\n        -9.87160320e-06],\n       [-3.62115122e-01,  0.00000000e+00,  5.01173501e+00,\n
\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [
 [0., 0., 0., 0.]]))
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.

unit_tests/unitTestsChap5.py:122: AssertionError
```

```
_____________________________________________ test_df_du _____________________________________________
_________________

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)
>       assert np.allclose(B, trueValues.B)
E       assert False
E        +  where False = <function allclose at 0x10e02b8b0>(array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00,\n         0.00000000e+00],\n       [ 0.00000000e+00,  0.000
00...20e-06,\n        -9.87160320e-06],\n       [-3.62115122e-01,  0.00000000e+00,  5.01173501e+00,\n        -2.48813413e+01]]), array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],
\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n
 [0., 0., 0., 0.]]))
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0.
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.B

unit_tests/unitTestsChap5.py:123: AssertionError
_________________ Captured stdout call _________________
Optimization terminated successfully    (Exit mode 0)
            Current function value: 2.586543198071751e-06
            Iterations: 13
            Function evaluations: 255
            Gradient evaluations: 13
elevator= -0.1247780701597401 aileron= 0.0018361809638628682 rudder= -0.000302608037876341 throttle= 0.6767522859047431
trim_state= [[-5.95888304e-15  0.00000000e+00 -1.00000000e+02  2.49687427e+01
   0.00000000e+00  1.24975516e+00  9.99687380e-01  0.00000000e+00
   2.50028494e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
   0.00000000e+00]]
=================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

```
_____________________________________________ test_df_du _____________________________________________
_________________

    def test_df_du():
        # initialize elements of the architecture
        mav = MavDynamics(SIM.ts_simulation)
        Va = 25.
        gamma = 0.*np.pi/180.
        trim_state, trim_input = compute_trim(mav, Va, gamma)
        mav._state = trim_state
        x_euler = np.array([[-5.95888304e-15],
                            [ 0.00000000e+00],
                            [-1.00000000e+02],
                            [ 2.49687427e+01],
                            [ 0.00000000e+00],
                            [ 1.24975516e+00],
                            [ 0.00000000e+00],
                            [ 5.00109104e-02],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00],
                            [ 0.00000000e+00]])
        B = cm.df_du(mav, x_euler, trim_input)
>       assert np.allclose(B, trueValues.B)
E       assert False
E        +  where False = <function allclose at 0x10e02b8b0>(array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00,\n         0.00000000e+00],\n       [ 0.00000000e+00,  0.000
00...20e-06,\n        -9.87160320e-06],\n       [-3.62115122e-01,  0.00000000e+00,  5.01173501e+00,\n        -2.48813413e+01]]), array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],
\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n
 [0., 0., 0., 0.]]))
E        +    where <function allclose at 0x10e02b8b0> = np.allclose
E        +    and   array([[0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0.... [0., 0., 0., 0.],\n       [0., 0.
, 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.],\n       [0., 0., 0., 0.]]) = trueValues.B

unit_tests/unitTestsChap5.py:122: AssertionError
_____________________________________ Captured stdout call _____________________________________
__________________
Optimization terminated successfully    (Exit mode 0)
            Current function value: 2.586543198071751e-06
            Iterations: 13
            Function evaluations: 255
            Gradient evaluations: 13
elevator= -0.1247780701597401 aileron= 0.0018361809638628682 rudder= -0.000302608037876341 throttle= 0.6767522859047431
trim_state= [[-5.95888304e-15  0.00000000e+00 -1.00000000e+02  2.49687427e+01
   0.00000000e+00  1.24975516e+00  9.99687380e-01  0.00000000e+00
   2.50028494e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00
```

```
=================================== short test summary info ===================================
=================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

```
========================================================================== short test summary info
==================
FAILED unit_tests/unitTestsChap5.py::test_df_dx - assert False
FAILED unit_tests/unitTestsChap5.py::test_df_du - assert False
2 failed, 4 passed in 1.32s
```

Update Values

- Debugger
- Breakpoint
- Copy Value
- Paste in true values file

```
mav: <chap3.mav_dynamics.MavDynamics object at 0x10fafc790>
  > special variables
  > function variables
  > true_state: <message_types.msg_state.MsgState object at 0x10fafc820>
    ts_simulation: 0.01
  > _derivatives: <bound method MavDynamics._derivatives of <chap3.mav_dynamics.…
  v _state: array([[ 0.],
    > special variables
    > [0:13] : [array([0.]), array([0 ray([25.]), array([0.]),…
    > dtype: dtype('float64')
      max: 25.0
      min: 0.0
    > shape: (13, 1)
      size: 13
```

View Value in Data Viewer
Set Value
Copy Value
Copy as Expression
Add to Watch

```
state_Mz = np.array(    [[ 2.50000000e-01],
                         [-6.69747058e-20],
                         [-1.25415774e-14],
                         [ 2.50000000e+01],
                         [-7.17806614e-05],
                         [ 3.04437995e-11],
                         [ 1.00000000e+00],
                         [ 2.09665008e-07],
                         [ 3.07878132e-13],
                         [ 1.43561323e-06],
                         [ 8.38660032e-05],
                         [ 2.46302506e-10],
                         [ 5.74245291e-04]])
```