In [31]:
```
#          ##########################################################################################
#          #                                      IN THE NAME OF ALLAH                              #
#          #                                CABLE FOOT SUSPENSION BRIDGE 01                         #
#          #----------------------------------------------------------------------------------------#
#          #                  THIS PROGRAM WRITTEN BY SALAR DELAVAR GHASHGHAEI (QASHQAI)            #
#          #                           EMAIL: salar.d.ghashghaei@gmail.com                          #
#          ##########################################################################################
```

Suspension Foot Bridge:

A Suspension Foot Bridge is a type of bridge that uses the features of a suspension bridge but is specifically designed for pedestrian use. These bridges are typically used to cross natural obstacles like valleys, rivers, or difficult terrain, whereas regular footbridges are simpler in design and construction.

Features of a Suspension Foot Bridge:

1. Suspension Structure:
   - In a suspension footbridge, steel or wire cables are suspended to support the bridge deck. These cables are anchored to tall towers on each side of the bridge, transferring the load from the deck to the towers.
   - Unlike regular footbridges, which are usually made of simple beams (either concrete or steel), a suspension footbridge's primary structure consists of cables and towers.
2. Design Purpose:
   - These bridges are typically designed for pedestrian use, but compared to regular footbridges, they can span greater distances and are ideal for crossing deep valleys, rivers, or other natural barriers.
   - Suspension footbridges are often built in natural parks, hiking trails, or areas that require crossing significant natural obstacles.
3. Load Characteristics:
   - Suspension footbridges generally carry less load than full-scale suspension bridges (designed for vehicles or trains), as they are only built to support pedestrians. However, they still rely on suspended cables to carry the load.
   - The load on these bridges is primarily pedestrian traffic, but in some cases, they might also support bicycles or animals.
4. Advantages:
   - Longer Spans: Suspension footbridges can cover longer distances than regular pedestrian bridges without requiring additional supports in the middle of the span.
   - Crossing Natural Obstacles: These bridges are ideal for crossing rivers, valleys, and rugged terrain where building traditional bridges is difficult.
   - Aesthetic and Integration with Nature: Suspension footbridges often have a beautiful design that blends well with the natural surroundings, especially in tourist or recreational areas.
5. Construction and Installation:
   - Building a suspension footbridge requires specialized equipment for installing cables and towers. These bridges are commonly used in areas where the terrain is difficult, or access to traditional infrastructure is limited.
   - Installing cables and towers in such challenging locations requires precise engineering and safety considerations.

Examples: Suspension Foot Bridges in Mountainous Areas: Suspension footbridges are commonly found in mountainous regions or nature reserves where they allow pedestrians to cross valleys and rivers. These bridges provide access for both locals and tourists to difficult-to-reach areas.

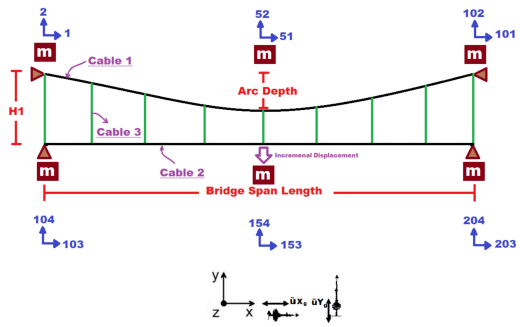
2. Suspension Footbridges in Parks: In some natural parks or recreational areas, suspension footbridges are used to cross rivers or valleys. These bridges serve both a practical purpose and act as tourist attractions.

Differences from Regular Suspension Bridges:

- Lower Load Capacity: While regular suspension bridges are designed to carry vehicles, trains, or heavy loads, and can bear significant stress, suspension footbridges are designed only for pedestrian use and therefore carry much lighter loads.
- Smaller Dimensions: Suspension footbridges are generally smaller in scale and have shorter spans than full-size suspension bridges. They are primarily intended for pedestrians and are rarely used for vehicle traffic.

Conclusion: A Suspension Foot Bridge is a type of suspension bridge specifically designed for pedestrian use. Due to the use of suspended cables, these bridges can span longer distances and are very useful in areas where crossing valleys, rivers, or difficult terrain is necessary.

In [32]:
```
# Load the image
def PLOT_IMAGE(image):
    import matplotlib.pyplot as plt
    import matplotlib.image as mpimg
    image = mpimg.imread(image_path)

    # Display the image
    plt.figure(figsize=(15, 8))
    plt.imshow(image)
    plt.axis('off')  # Hide axes
    plt.show()

image_path = 'OPENSEES_CABLE_SUSPENSION_BRIDGE_01.png'
PLOT_IMAGE(image_path)
```



In [33]:
```
# WIKIPEDIA: Simple suspension bridge
'https://en.wikipedia.org/wiki/Simple_suspension_bridge'
# IMAGE:
'https://www.waratahbridgeconstructions.com.au/pedestrian-bridges-north-east-victoria.html'
'https://www.moodie.com.au/?product=cable-bridges'
```

```
# PAPER: Study on Suspension Bridge: Analysis and Construction criteria
'https://www.ijsdr.org/papers/IJSDR1704077.pdf'
```

Out[33]:
```
'https://www.ijsdr.org/papers/IJSDR1704077.pdf'
```

In [34]:
```
#import the os module
import os
import time
import numpy as np
import openseespy.opensees as op
import matplotlib.pyplot as plt
```

In [35]:
```
#to create a directory at specified path with name "Data"
os.mkdir('C:\\OPENSEESPY_SALAR')
```

```
---------------------------------------------------------------------------
FileExistsError                           Traceback (most recent call last)
Cell In[35], line 2
      1 #to create a directory at specified path with name "Data"
----> 2 os.mkdir('C:\\OPENSEESPY_SALAR')

FileExistsError: [WinError 183] Cannot create a file when that file already exists: 'C:\\OPENSEESPY_SALAR'
```

In [36]:
```
FOLDER_NAME = 'OPENSEES_CABLE_SUSPENSION_BRIDGE'
dir = f"C:\\OPENSEESPY_SALAR\\{FOLDER_NAME}\\"
if not os.path.exists(dir):
    os.makedirs(dir)
```

In [37]:
```
# OUTPUT DATA ADDRESS:
SALAR_DIR = f'C://OPENSEESPY_SALAR//{FOLDER_NAME}//';
```

In [38]:
```
## DELETE ALL FILES IN DIRECTORY
def DELETE_FOLDER_CONTANTS(folder_path):
    import os
    for filename in os.listdir(folder_path):
        file_path = os.path.join(folder_path, filename)
        try:
            if os.path.isfile(file_path) or os.path.islink(file_path):
                os.unlink(file_path)
        except Exception as e:
            print(f"Failed to delete {file_path}. Reason: {e}")
    print("Deletion done")

FOLDER_PATH = f'C:\\OPENSEESPY_SALAR\\{FOLDER_NAME}'  # Specify the folder path
#DELETE_FOLDER_CONTANTS(FOLDER_PATH)
```

In [39]:
```
def CURRENT_TIME():
    import time
    t = time.localtime()
    current_time = time.strftime("%H:%M:%S", t)
    print(f"Current time (HH:MM:SS): {current_time}\n\n")

# ---------------------
def plot_shapes(initial_coords, displacements):
    import numpy as np
    import matplotlib.pyplot as plt
    """
    Plot the initial and deformed shapes of the arch.

    Parameters:
    - initial_coords: Initial coordinates of the nodes.
    - displacements: List of displacements for each node.
    """
    # Plot initial and deformed shape
    plt.figure(figsize=(12, 8))
    plt.plot(initial_coords[:num_nodes, 0], initial_coords[:num_nodes, 1], 'bo-', label='Initial Cable Shape')
    plt.plot(deformed_coords[:num_nodes, 0], deformed_coords[:num_nodes, 1], 'ro-', label='Deformed Cable Shape')
    plt.plot(initial_coords[num_nodes:, 0], initial_coords[num_nodes:, 1], 'go-', label='Initial Deck Shape')
    plt.plot(deformed_coords[num_nodes:, 0], deformed_coords[num_nodes:, 1], 'yo-', label='Deformed Deck Shape')

    # Plot connecting elements between the cable and the beam
    for i in range(num_nodes):
        plt.plot([initial_coords[i, 0], initial_coords[num_nodes + i, 0]],
                 [initial_coords[i, 1], initial_coords[num_nodes + i, 1]], 'c--', label='Initial Connecting Elements' if i == 0 else "")
        plt.plot([deformed_coords[i, 0], deformed_coords[num_nodes + i, 0]],
                 [deformed_coords[i, 1], deformed_coords[num_nodes + i, 1]], 'm--', label='Deformed Connecting Elements' if i == 0 else "")

    # Mark simply supported nodes with red triangles
    support_nodes = [0, num_nodes - 1, num_nodes, 2*num_nodes - 1]
    plt.plot(initial_coords[support_nodes, 0], initial_coords[support_nodes, 1], 'r^', markersize=20, label='Simply Supported Nodes')

    plt.xlabel('X [mm]')
    plt.ylabel('Y [mm]')
    plt.legend()
    plt.title('Pushover Analysis: Cable and Deck Deformed Shapes')
    plt.show()

# ---------------------

def plot_reactions(disp_x, reaction_x, disp_y, reaction_y):
    import matplotlib.pyplot as plt
    """
    Plot the base reaction versus displacement.

    Parameters:
    - disp_x: List of x displacements of the middle node.
    - reaction_x: List of base reactions in x.
    - disp_y: List of y displacements of the middle node.
    - reaction_y: List of base reactions in y.
    """
    plt.figure(figsize=(10, 6))
    plt.plot(disp_x, reaction_x, 'b-')
    plt.xlabel('Displacement X [mm]')
    plt.ylabel('Base Reaction X [N]')
    plt.title('Base Reaction X vs. Displacement X')
    plt.show()

    plt.figure(figsize=(10, 6))
    plt.plot(disp_y, reaction_y, 'r-')
    plt.xlabel('Displacement Y [mm]')
    plt.ylabel('Base Reaction Y [N]')
```

```
    plt.title('Base Reaction Y vs. Displacement Y')
    plt.show()

# --------------------
"""
When OK equals -1, it generally indicates that the command or operation was not executed
because it was already in progress or had already been completed. This can happen if you
try to run a command that is already running or has been completed in a previous step.

When OK equals -2, it typically indicates that the command or operation was not executed
because it was not recognized or not implemented. This could mean that the command
is either misspelled, not available in the current version of OpenSees, or not applicable to the current context.

When OK equals -3, it typically means that the command or operation failed.
This could be due to various reasons, such as incorrect input parameters,
syntax errors, or issues with the model setup.
"""
def ANALYSIS(OK, INCREMENT, TOLERANCE, MAX_ITERAIONS):
    import openseespy.opensees as op
    test = {1:'NormDispIncr', 2: 'RelativeEnergyIncr', 4: 'RelativeNormUnbalance',5: 'RelativeNormDispIncr', 6: 'NormUnbalance'}
    algorithm = {1:'KrylovNewton', 2: 'SecantNewton' , 4: 'RaphsonNewton',5: 'PeriodicNewton', 6: 'BFGS', 7: 'Broyden', 8: 'NewtonLineSearch'}

    for i in test:
        for j in algorithm:
            if OK != 0:
                if j < 4:
                    op.algorithm(algorithm[j], '-initial')

                else:
                    op.algorithm(algorithm[j])

                op.test(test[i], TOLERANCE, MAX_ITERAIONS)
                OK = op.analyze(INCREMENT)
                print(test[i], algorithm[j], OK)
                if OK == 0:
                    break
            else:
                continue
```

In [ ]:

```
#          ##########################################################################################
#          #                                          IN THE NAME OF ALLAH                          #
#          #                         PUSHOVER ANALYSIS OF CABLE SUSPENSION BRIDGE 01                #
#          #----------------------------------------------------------------------------------------#
#          #                  THIS PROGRAM WRITTEN BY SALAR DELAVAR GHASHGHAEI (QASHQAI)            #
#          #                            EMAIL: salar.d.ghashghaei@gmail.com                         #
#          ##########################################################################################
```

In [40]:

```
# --------------------
#   PUSHOVER ANALYSIS
# --------------------
def PUSHOVER_ANALYSIS(LINEAR, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, MAX_DISP, disp_incr, MAX_ITERATIONS, TOLERANCE):
    import openseespy.opensees as ops
    import numpy as np
    import matplotlib.pyplot as plt

    A_cable_01 = (np.pi * Cable_Dia_01 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Bottom
    A_cable_02 = (np.pi * Cable_Dia_02 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Top
    A_cable_03 = (np.pi * Cable_Dia_03 **2) / 4  # [mm^2] Vertical Longitudinal Cable Area Top

    # Define model builder
    ops.wipe()
    ops.model('basic', '-ndm', 2, '-ndf', 2)
    dx = L / (num_nodes - 1)

    # Create nodes
    for i in range(num_nodes):
        x = i * dx
        y = H1 - arc_depth * np.sin(np.pi * x / L)
        ops.node(i + 1, x, y)
        ops.node(num_nodes + i + 1, i * dx, 0.0)

    # Define boundary conditions (fixed at both ends)
    ops.fix(1, 1, 1)              # TOP CABLE
    ops.fix(num_nodes, 1, 1)      # TOP CABLE
    ops.fix(num_nodes + 1, 1, 1)  # BOTTOM CABLE - DECK
    ops.fix(2 * num_nodes, 1, 1)  # BOTTOM CABLE - DECK

    # Define material properties
    if LINEAR == True:
        ops.uniaxialMaterial('Elastic', 1, E_cable)
        #                                 TENSION    COMPRESSION
        #ops.uniaxialMaterial('Elastic', 1, E_cable, 0, 0.5 * E_cable)
    if LINEAR == False:
        Fy_cable = 355   # [N/mm^2] Yield strength of the cable
        b0 = 0.01
        ops.uniaxialMaterial('Steel01', 1, Fy_cable, E_cable, b0)

    # Define truss elements
    for i in range(num_nodes - 1):
        ops.element('corotTruss', i + 1, i + 1, i + 2, A_cable_02, 1)  # Cable element
        ops.element('corotTruss', num_nodes + i + 1, num_nodes + i + 1, num_nodes + i + 2, A_cable_01, 1)  # Deck element

    # Connect each node of the cable with the corresponding node of the deck using truss elements
    for i in range(num_nodes):
        ops.element('corotTruss', 2 * num_nodes + i + 1, i + 1, num_nodes + i + 1, A_cable_03, 1)

    #mid_node = num_nodes // 2 + 1
    mid_node = int(num_nodes + 0.5 * num_nodes)
    # Define load pattern with displacement at the middle span
    ops.timeSeries('Linear', 1)
    ops.pattern('Plain', 1, 1)
    ops.load(mid_node, 0.0, -1)  # Apply a horizontal load at node
    disp_incr = 0.01    # [mm] Incremental Displacement
    n_steps = int(np.abs(MAX_DISP / disp_incr))  # Analysis Steps

    # Define analysis parameters
    ops.system('BandGeneral')
    ops.numberer('Plain')
    ops.constraints('Plain')
    ops.integrator('DisplacementControl', mid_node, 2, -disp_incr)
    ops.test('EnergyIncr', TOLERANCE, MAX_ITERATIONS)
    ops.algorithm('ModifiedNewton')
```

```
    ops.analysis('Static')

    # OUTPUT DATA
    ops.recorder('Node', '-file', f"{SALAR_DIR}DTH_PUSH.txt",'-time', '-node', mid_node, '-dof', 1,2, 'disp')# Displacement Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_PUSH_01.txt",'-time', '-node', 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_PUSH_02.txt",'-time', '-node', num_nodes, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_PUSH_03.txt",'-time', '-node', num_nodes + 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_PUSH_04.txt",'-time', '-node', 2*num_nodes, '-dof', 1,2, 'reaction')# Base Shear Time History

    DISPLACEMENTS = []
    DISP_X , DISP_Y = [], []
    BASE_REACTION_X, BASE_REACTION_Y = [], []

    # Perform the analysis with increments
    for i in range(n_steps):
        OK = ops.analyze(1)
        ANALYSIS(OK, 1, TOLERANCE, MAX_ITERATIONS)
        DISPLACEMENTS.append([ops.nodeDisp(j + 1) for j in range(num_nodes * 2)])
        DISP_X.append(ops.nodeDisp(mid_node, 1))
        DISP_Y.append(ops.nodeDisp(mid_node, 2))
        x1 = ops.nodeResponse(1, 1, 6)
        x2 = ops.nodeResponse(num_nodes, 1, 6)
        x3 = ops.nodeResponse(num_nodes + 1, 1, 6)
        x4 = ops.nodeResponse(2*num_nodes, 1, 6)

        y1 = ops.nodeResponse(1, 2, 6)
        y2 = ops.nodeResponse(num_nodes, 2, 6)
        y3 = ops.nodeResponse(num_nodes + 1, 2, 6)
        y4 = ops.nodeResponse(2*num_nodes, 2, 6)
        BASE_REACTION_X.append(x1+x2+x3+x4) # CABLE REACION-X
        BASE_REACTION_Y.append(y1+y2+y3+y4) # CABLE REACION-Y
        #print(f'STEP: {i+1}')

    # Get initial and final node coordinates for plotting
    initial_coords = np.array([ops.nodeCoord(i + 1) for i in range(num_nodes * 2)])
    deformed_coords = initial_coords + np.array(DISPLACEMENTS[-1])

    # Output all unconstrained node displacements
    for i in range(num_nodes * 2):
        disp = ops.nodeDisp(i + 1)
        print(f"Node {i + 1}: X Displacement = {disp[0]}, Y Displacement = {disp[1]}")

    return initial_coords, deformed_coords, DISPLACEMENTS, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y
```

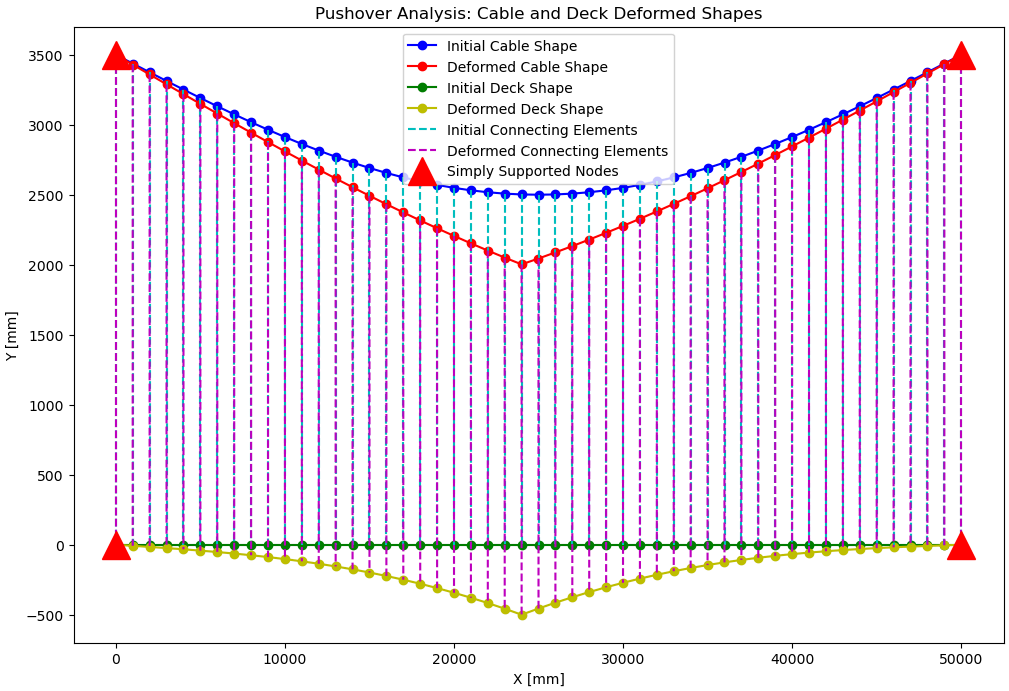
In [41]:

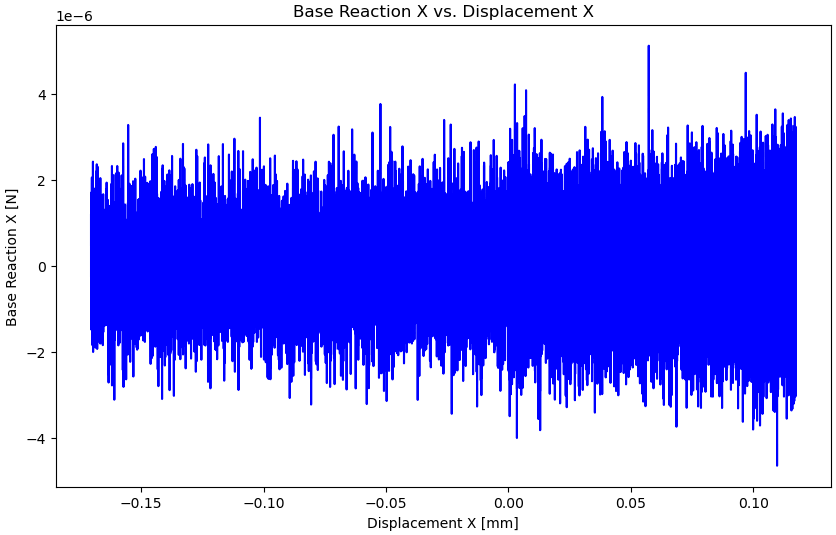
```
# --------------------
#   PUSHOVER ANALYSIS
# --------------------
# Parameters for the analysis
L = 50000.0        # [mm] Bridge span lentgh
H1 = 3500.0        # [mm] Height of Top Cable
arc_depth = 1000.0 # [mm]
E_cable = 210e5    # [N/mm^2] Modulus of elasticity Cable
Cable_Dia_01 = 25  # [mm] Horizontal Longitudinal Cable Diameter Bottom
Cable_Dia_02 = 18  # [mm] Horizontal Longitudinal Cable Diameter Top
Cable_Dia_03 = 10  # [mm] Vertical Longitudinal Cable Diameter Top
num_nodes = 51     # Cable Arc Number of nodes
MAX_DISP = -500.0  # [mm] Maximum Displacment
disp_incr = -0.01  # [mm] Incremental Displacement

MAX_ITERATIONS = 50000  # Maximum number of iterations
TOLERANCE = 1.0e-10     # Tolerance for convergence

LINEAR = True # False: Cable Nonlinear Materials Properties

starttime = time.process_time()
# Run the analysis
results = PUSHOVER_ANALYSIS(LINEAR, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, MAX_DISP, disp_incr, MAX_ITERATIONS, TOLERANCE)
initial_coords, deformed_coords, DISPLACEMENTS, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y = results
totaltime = time.process_time() - starttime
print(f'\nTotal time (s): {totaltime:.4f} \n\n')
```

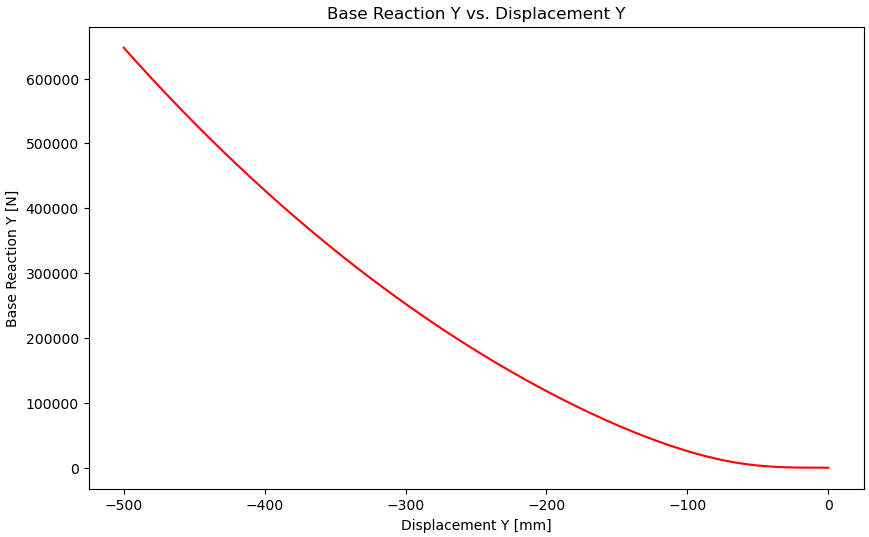
```
WARNING: CTestEnergyIncr::test() - failed to converge
after: 50000 iterations
 current EnergyIncr: -nan(ind) (max: 1e-10)     Norm deltaX: nan, Norm deltaR: nan
ModifiedNewton::solveCurrentStep() -the ConvergenceTest object failed in test()
StaticAnalysis::analyze() - the Algorithm failed at step: 0 with domain at load factor -nan(ind)
OpenSees > analyze failed, returned: -3 error flag
```

```
NormDispIncr KrylovNewton 0
Node 1: X Displacement = 0.0, Y Displacement = 0.0
Node 2: X Displacement = 0.3199093321564988, Y Displacement = -7.806389908584831
Node 3: X Displacement = 0.631027760133854, Y Displacement = -15.764664966975404
Node 4: X Displacement = 0.9247555455256964, Y Displacement = -24.026106496111915
Node 5: X Displacement = 1.1928702307870171, Y Displacement = -32.74081689215971
Node 6: X Displacement = 1.4277073923384542, Y Displacement = -42.057135691408995
Node 7: X Displacement = 1.622332110862012, Y Displacement = -52.12106165685267
Node 8: X Displacement = 1.770698284729665, Y Displacement = -63.07568290146361
Node 9: X Displacement = 1.8677931008152302, Y Displacement = -75.06061701609693
Node 10: X Displacement = 1.9097642032318738, Y Displacement = -88.21146317507215
Node 11: X Displacement = 1.8940273651560144, Y Displacement = -102.65926820367797
Node 12: X Displacement = 1.8193527683809405, Y Displacement = -118.53000860839629
Node 13: X Displacement = 1.6859283256916666, Y Displacement = -135.94409059115955
Node 14: X Displacement = 1.4953988381728838, Y Displacement = -155.01587009100749
Node 15: X Displacement = 1.250880158343515, Y Displacement = -175.85319491640485
Node 16: X Displacement = 0.956947925340774, Y Displacement = -198.556971044306
Node 17: X Displacement = 0.6196008446537753, Y Displacement = -223.2207551615983
Node 18: X Displacement = 0.2461988961689911, Y Displacement = -249.93037550380964
Node 19: X Displacement = -0.15462273573297428, Y Displacement = -278.7635829973928
Node 20: X Displacement = -0.5730628141799169, Y Displacement = -309.78973462838667
Node 21: X Displacement = -0.9982847005899349, Y Displacement = -343.0695108359222
Node 22: X Displacement = -1.4185665547848574, Y Displacement = -378.6546685562685
Node 23: X Displacement = -1.821468717721201, Y Displacement = -416.58782978268636
Node 24: X Displacement = -2.1939860062174326, Y Displacement = -456.9017142816989
Node 25: X Displacement = -2.511670554771266, Y Displacement = -499.3906115988735
Node 26: X Displacement = -2.547112242136458, Y Displacement = -455.62603297097127
Node 27: X Displacement = -2.657280625809106, Y Displacement = -414.0553153571602
Node 28: X Displacement = -2.8191839800275544, Y Displacement = -374.90342719373695
Node 29: X Displacement = -3.0193758337812246, Y Displacement = -338.1565908826925
Node 30: X Displacement = -3.244595967746345, Y Displacement = -303.79090227054826
Node 31: X Displacement = -3.4819469820560487, Y Displacement = -271.7730186968144
Node 32: X Displacement = -3.7190877861838976, Y Displacement = -242.06029416544825
Node 33: X Displacement = -3.9444152535021972, Y Displacement = -214.60095197313652
Node 34: X Displacement = -4.147231127818948, Y Displacement = -189.3342954123924
Node 35: X Displacement = -4.317891598800236, Y Displacement = -166.19095535900598
Node 36: X Displacement = -4.447937265183557, Y Displacement = -145.0931733086479
Node 37: X Displacement = -4.530201541491634, Y Displacement = -125.95511822781721
Node 38: X Displacement = -4.55889592913094, Y Displacement = -108.68323542832194
Node 39: X Displacement = -4.529670959493224, Y Displacement = -93.17662556083904
Node 40: X Displacement = -4.439652018139383, Y Displacement = -79.32745174773434
Node 41: X Displacement = -4.287449668585613, Y Displacement = -67.02137283210837
Node 42: X Displacement = -4.073144505121741, Y Displacement = -56.13800070193991
Node 43: X Displacement = -3.7982469701719217, Y Displacement = -46.55137964796155
Node 44: X Displacement = -3.4656329670143715, Y Displacement = -38.13048572492054
Node 45: X Displacement = -3.0794564775824953, Y Displacement = -30.739744102436582
Node 46: X Displacement = -2.6450407523323833, Y Displacement = -24.239562409355027
Node 47: X Displacement = -2.1687499699352424, Y Displacement = -18.486878091067872
Node 48: X Displacement = -1.6578435644342842, Y Displacement = -13.335717810695044
Node 49: X Displacement = -1.1203156825277303, Y Displacement = -8.637766931193015
Node 50: X Displacement = -0.5647224603638313, Y Displacement = -4.242947115834292
Node 51: X Displacement = 0.0, Y Displacement = 0.0
Node 52: X Displacement = 0.0, Y Displacement = 0.0
Node 53: X Displacement = 0.24112605465504822, Y Displacement = -7.8072747869822665
Node 54: X Displacement = 0.48105567210462, Y Displacement = -15.766397713668386
Node 55: X Displacement = 0.7185279680729767, Y Displacement = -24.02864811754546
Node 56: X Displacement = 0.9521548079230172, Y Displacement = -32.744127433061436
Node 57: X Displacement = 1.1803604652083914, Y Displacement = -42.06117471138081
Node 58: X Displacement = 1.4013251633564807, Y Displacement = -52.12578815303226
Node 59: X Displacement = 1.6129337000797217, Y Displacement = -63.081054631283514
Node 60: X Displacement = 1.8127302728272177, Y Displacement = -75.06658928193129
Node 61: X Displacement = 1.9978805244068158, Y Displacement = -88.21798725340128
Node 62: X Displacement = 2.1651417115164655, Y Displacement = -102.6662897199856
Node 63: X Displacement = 2.310841766928375, Y Displacement = -118.53746625716987
Node 64: X Displacement = 2.4308678803800152, Y Displacement = -135.95191565820377
Node 65: X Displacement = 2.520665066063016, Y Displacement = -155.02398723317057
Node 66: X Displacement = 2.575245018423436, Y Displacement = -175.86152457499296
Node 67: X Displacement = 2.5892053854758474, Y Displacement = -198.56543370202027
Node 68: X Displacement = 2.5567594127425624, Y Displacement = -223.22927739674802
Node 69: X Displacement = 2.4717757340734416, Y Displacement = -249.9388974592113
Node 70: X Displacement = 2.3278279106821866, Y Displacement = -278.7720664873375
Node 71: X Displacement = 2.1182531493437795, Y Displacement = -309.79817069104934
Node 72: X Displacement = 1.8362194672132621, Y Displacement = -343.07792514772893
Node 73: X Displacement = 1.4748004161407329, Y Displacement = -378.6631228239429
Node 74: X Displacement = 1.027056242454488, Y Displacement = -416.59642107811
Node 75: X Displacement = 0.4860821835797728, Y Displacement = -456.9121116977464
Node 76: X Displacement = -0.17053444703073659, Y Displacement = -499.999999969026
Node 77: X Displacement = -0.8828766685547187, Y Displacement = -455.63735237292235
Node 78: X Displacement = -1.4752667052045902, Y Displacement = -414.06533383497389
Node 79: X Displacement = -1.9699453163437748, Y Displacement = -374.91354199889855
Node 80: X Displacement = -2.37334110328229, Y Displacement = -338.1667019131928
Node 81: X Displacement = -2.6920666666837616, Y Displacement = -303.80094547678414
Node 82: X Displacement = -2.932865023031619, Y Displacement = -271.78294601563357
Node 83: X Displacement = -3.1025195899647424, Y Displacement = -242.07006185542613
Node 84: X Displacement = -3.207768280920555, Y Displacement = -214.61051514818658
Node 85: X Displacement = -3.255223075297706, Y Displacement = -189.34360697575613
Node 86: X Displacement = -3.2512962214099574, Y Displacement = -166.19996759809206
Node 87: X Displacement = -3.2021341003860795, Y Displacement = -145.10184063433705
Node 88: X Displacement = -3.1135596365658453, Y Displacement = -125.96339987017544
Node 89: X Displacement = -2.991023985646228, Y Displacement = -108.6910972796879
Node 90: X Displacement = -2.8395680679613786, Y Displacement = -93.18404073981579
Node 91: X Displacement = -2.6637943431529436, Y Displacement = -79.33439979813383
Node 92: X Displacement = -2.4678490466121494, Y Displacement = -67.02783774405411
Node 93: X Displacement = -2.255414930161023, Y Displacement = -56.14396812890243
Node 94: X Displacement = -2.029714372321433, Y Displacement = -46.55683379065218
Node 95: X Displacement = -1.7935225500832521, Y Displacement = -38.135406366880446
Node 96: X Displacement = -1.5491901971652293, Y Displacement = -30.744104226997216
Node 97: X Displacement = -1.29867531605593, Y Displacement = -24.24332672229692
Node 98: X Displacement = -1.043583065104162, Y Displacement = -18.490002638388983
Node 99: X Displacement = -0.7852129097940771, Y Displacement = -13.338150736543284
Node 100: X Displacement = -0.5246120109768905, Y Displacement = -8.639450284798523
Node 101: X Displacement = -0.2626337237928343, Y Displacement = -4.243819502893362
Node 102: X Displacement = 0.0, Y Displacement = 0.0

Total time (s): 89.4219
```

In [42]:
```
# Plot results
plot_shapes(initial_coords, DISPLACEMENTS)
plot_reactions(DISP_X, BASE_REACTION_X, DISP_Y, BASE_REACTION_Y)
```







In [24]:
```
#          ##########################################################################################
#          #                                    IN THE NAME OF ALLAH                                  #
#          #                     FREE VIBRATION ANALYSIS OF CABLE SUSPENSION BRIDGE 01               #
#          #-----------------------------------------------------------------------------------------#
#          #                THIS PROGRAM WRITTEN BY SALAR DELAVAR GHASHGHAEI (QASHQAI)               #
```

```
#         #                          EMAIL: salar.d.ghashghaei@gmail.com                          #
#         ########################################################################################
```

In [25]:

```
# -------------------------
#   FREE VIBRATION ANALYSIS
# -------------------------
def FREE_VIBRATION_ANALYSIS(damping, damping_ratio, LINEAR, duration, dt, TOTAL_MASS, u0, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, MAX_ITERATIONS
    import openseespy.opensees as ops
    import numpy as np
    import matplotlib.pyplot as plt

    A_cable_01 = (np.pi * Cable_Dia_01 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Bottom
    A_cable_02 = (np.pi * Cable_Dia_02 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Top
    A_cable_03 = (np.pi * Cable_Dia_03 **2) / 4  # [mm^2] Vertical Longitudinal Cable Area Top

    # Define model builder
    ops.wipe()
    ops.model('basic', '-ndm', 2, '-ndf', 2)
    KE = 1;
    dx = L / (num_nodes - 1)
    MASS = TOTAL_MASS / num_nodes
    # Create nodes
    for i in range(num_nodes):
        X = i * dx
        Y1 = H1 - arc_depth * np.sin(np.pi * X / L)
        ops.node(i + 1, X, Y1)
        ops.node(num_nodes + i + 1, i * dx, 0.0)
        # Define mass
        ops.mass(num_nodes + i + 1, MASS, MASS, 0)

    # Define boundary conditions (fixed at both ends)
    ops.fix(1, 1, 1)              # TOP CABLE
    ops.fix(num_nodes, 1, 1)      # TOP CABLE
    ops.fix(num_nodes + 1, 1, 1)  # BOTTOM CABLE
    ops.fix(2 * num_nodes, 1, 1)  # BOTTOM CABLE

    # Define material properties
    if LINEAR == True:
        ops.uniaxialMaterial('Elastic', 1, E_cable)
        #                                 TENSION    COMPRESSION
        #ops.uniaxialMaterial('Elastic', 1, E_cable, 0, 0.5 * E_cable)
    if LINEAR == False:
        Fy_cable = 355   # [N/mm^2] Yield strength of the cable
        b0 = 0.01
        ops.uniaxialMaterial('Steel01', 1, Fy_cable, E_cable, b0)

    # Define truss elements
    for i in range(num_nodes - 1):
        ops.element('corotTruss', i + 1, i + 1, i + 2, A_cable_02, 1)  # Top Cable element
        ops.element('corotTruss', num_nodes + i + 1, num_nodes + i + 1, num_nodes + i + 2, A_cable_01, 1)  # Bottom Cable Element

    # Connect each node of the cable with the corresponding node of the deck using truss elements
    for i in range(num_nodes):
        ops.element('corotTruss', 2 * num_nodes + i + 1, i + 1, num_nodes + i + 1, A_cable_03, 1) # Vertical Cable Element

    #mid_node = num_nodes // 2 + 1
    mid_node = int(num_nodes + 0.5 * num_nodes)
    # Define load pattern with displacement at the middle span
    ops.timeSeries('Linear', 1)
    ops.pattern('Plain', 1, 1)
    ops.load(mid_node, 0.0, -1)  # Apply a horizontal load at node
    ops.constraints('Transformation')
    ops.numberer('RCM')
    ops.system('BandGeneral')
    ops.algorithm('Newton')
    ops.test('EnergyIncr', TOLERANCE, MAX_ITERATIONS)
    ops.integrator('DisplacementControl', mid_node, 2, u0)
    ops.analysis('Static')
    ops.analyze(1)

    ops.setTime(0.0)

    # Wipe analysis and reset time
    ops.wipeAnalysis()
    ops.remove('loadPattern', 1)
    ops.system('UmfPack')

    # Dynamic analysis
    ops.constraints('Transformation')
    ops.numberer('RCM')
    ops.system('UmfPack')
    ops.test('EnergyIncr', TOLERANCE, MAX_ITERATIONS)
    ops.integrator('Newmark', 0.5, 0.25)
    ops.algorithm('Newton')

    if damping == True:
        # Calculate Rayleigh damping factors
        omega1 = np.sqrt(KE / TOTAL_MASS)
        omega2 = 2 * omega1  # Just an assumption for two modes
        a0 = damping_ratio * (2 * omega1 * omega2) / (omega1 + omega2)
        a1 = damping_ratio * 2 / (omega1 + omega2)
        # Apply Rayleigh damping
        ops.rayleigh(a0, a1, 0, 0)

    ops.analysis('Transient')

    # OUTPUT DATA
    ops.recorder('Node', '-file', f"{SALAR_DIR}DTH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2, 'disp')# Displacement Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}VTH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2,3, 'vel')  # Velocity Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}ATH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2,3, 'accel') # Acceleration Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_01.txt",'-time', '-node', 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_02.txt",'-time', '-node', num_nodes, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_03.txt",'-time', '-node', num_nodes + 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_04.txt",'-time', '-node', 2*num_nodes, '-dof', 1,2, 'reaction')# Base Shear

    DISPLACEMENTS = []
    DISP_X , DISP_Y, VELOCITY, ACCELERATION = [], [], [], []
    BASE_REACTION_X, BASE_REACTION_Y = [], []

    stable = 0
    current_time = 0.0
     # Perform the analysis with increments
```

```
    while stable == 0 and current_time < duration:
        OK = ops.analyze(1, dt)
        ANALYSIS(OK, 1, TOLERANCE, MAX_ITERATIONS)
        current_time = ops.getTime()
        DISPLACEMENTS.append([ops.nodeDisp(j + 1) for j in range(num_nodes * 2)])
        DISP_X.append(ops.nodeDisp(mid_node, 1))
        DISP_Y.append(ops.nodeDisp(mid_node, 2))
        VELOCITY.append(ops.nodeVel(mid_node, 2))
        ACCELERATION.append(ops.nodeAccel(mid_node, 2))
        x1 = ops.nodeResponse(1, 1, 6)
        x2 = ops.nodeResponse(num_nodes, 1, 6)
        x3 = ops.nodeResponse(num_nodes + 1, 1, 6)
        x4 = ops.nodeResponse(2*num_nodes, 1, 6)
        y1 = ops.nodeResponse(1, 2, 6)
        y2 = ops.nodeResponse(num_nodes, 2, 6)
        y3 = ops.nodeResponse(num_nodes + 1, 2, 6)
        y4 = ops.nodeResponse(2*num_nodes, 2, 6)
        BASE_REACTION_X.append(x1+x2+x3+x4) # CABLE REACION-X
        BASE_REACTION_Y.append(y1+y2+y3+y4) # CABLE REACION-Y
        KE =  BASE_REACTION_Y[-1] / DISP_Y[-1] # Effective Lateral Stiffness
        #print(f'STEP: {i+1}')

    # Get initial and final node coordinates for plotting
    initial_coords = np.array([ops.nodeCoord(i + 1) for i in range(num_nodes * 2)])
    deformed_coords = initial_coords + np.array(DISPLACEMENTS[-1])

    # Output all unconstrained node displacements
    for i in range(num_nodes * 2):
        disp = ops.nodeDisp(i + 1)
        print(f"Node {i + 1}: X Displacement = {disp[0]}, Y Displacement = {disp[1]}")

    return initial_coords, deformed_coords, DISPLACEMENTS, VELOCITY, ACCELERATION, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y
```

In [26]:

```
# Define the plotting function
def plot_time_history(TIME, DISP_X_undamped, DISP_X_damped, DISP_Y_undamped, DISP_Y_damped,
                      VELOCITY_undamped, VELOCITY_damped, ACCELERATION_undamped, ACCELERATION_damped,
                      BASE_REACTION_X_undamped, BASE_REACTION_X_damped, BASE_REACTION_Y_undamped, BASE_REACTION_Y_damped):
    import matplotlib.pyplot as plt
    import numpy as np

    plt.figure(figsize=(14, 26))

    # Displacement X
    plt.subplot(6, 1, 1)
    P1 = np.max(np.abs(np.array(DISP_X_undamped)))
    P2 = np.max(np.abs(np.array(DISP_X_damped)))
    plt.plot(TIME, DISP_X_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, DISP_X_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Displacement X [mm]')
    plt.legend()

    # Displacement Y
    plt.subplot(6, 1, 2)
    P1 = np.max(np.abs(np.array(DISP_Y_undamped)))
    P2 = np.max(np.abs(np.array(DISP_Y_damped)))
    plt.plot(TIME, DISP_Y_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, DISP_Y_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Displacement Y [mm]')
    plt.legend()

    # Velocity
    plt.subplot(6, 1, 3)
    P1 = np.max(np.abs(np.array(VELOCITY_damped)))
    P2 = np.max(np.abs(np.array(VELOCITY_undamped)))
    plt.plot(TIME, VELOCITY_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, VELOCITY_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Velocity [mm/s]')
    plt.legend()

    # Acceleration
    plt.subplot(6, 1, 4)
    P1 = np.max(np.abs(np.array(ACCELERATION_undamped)))
    P2 = np.max(np.abs(np.array(ACCELERATION_damped)))
    plt.plot(TIME, ACCELERATION_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, ACCELERATION_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Acceleration [mm/s^2]')
    plt.legend()

    # Base Reaction X
    plt.subplot(6, 1, 5)
    P1 = np.max(np.abs(np.array(BASE_REACTION_X_undamped)))
    P2 = np.max(np.abs(np.array(BASE_REACTION_X_damped)))
    plt.plot(TIME, BASE_REACTION_X_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, BASE_REACTION_X_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Base Reaction X [N]')
    plt.legend()

    # Base Reaction Y
    plt.subplot(6, 1, 6)
    P1 = np.max(np.abs(np.array(BASE_REACTION_Y_undamped)))
    P2 = np.max(np.abs(np.array(BASE_REACTION_Y_damped)))
    plt.plot(TIME, BASE_REACTION_Y_undamped, color='black', label=f'Undamped: {P1:.4e}')
    plt.plot(TIME, BASE_REACTION_Y_damped, color='red', label=f'Damped: {P2:.4e}')
    plt.xlabel('Time [s]')
    plt.ylabel('Base Reaction Y [N]')
    plt.legend()

    plt.suptitle('Time History of Free Vibration Analysis: Damped vs Undamped')
    plt.tight_layout(rect=[0, 0.03, 1, 0.95])
    plt.show()
```

In [27]:

```
# -------------------------
#   FREE VIBRATION ANALYSIS
# -------------------------
# Parameters for the analysis
L = 50000.0         # [mm] Bridge span lentgh
H1 = 3500.0         # [mm] Height of Top Cable
```

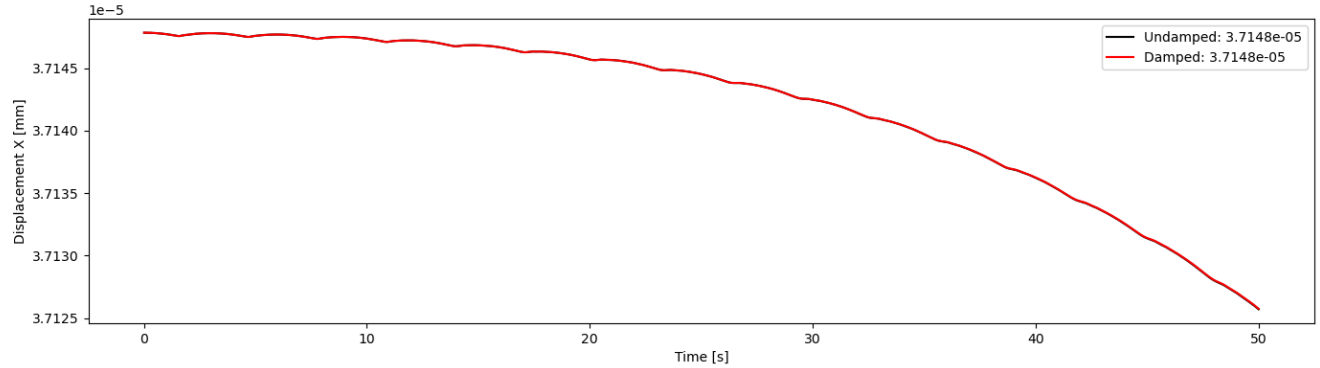
```
arc_depth = 1000.0  # [mm]
E_cable = 210e5     # [N/mm^2] Modulus of elasticity Cable
Cable_Dia_01 = 25   # [mm] Horizontal Longitudinal Cable Diameter Bottom
Cable_Dia_02 = 18   # [mm] Horizontal Longitudinal Cable Diameter Top
Cable_Dia_03 = 10   # [mm] Vertical Longitudinal Cable Diameter Top
num_nodes = 51      # Cable Arc Number of nodes

TOTAL_MASS = 500000.0  # [kg] Total Mass of Structure
u0 = 2.0               # [mm] Initial displacement
damping_ratio = 0.05   # Damping ratio
duration = 50.0        # [s] Duration of the analysis in seconds
dt = 0.01              # Time step in seconds

MAX_ITERATIONS = 10000  # Maximum number of iterations
TOLERANCE = 1.0e-14     # Tolerance for convergence

import time
starttime = time.process_time()

# Run the undamped analysis
damping = False
LINEAR = True # False: Cable NonLinear Materials Properties
results_undamped = FREE_VIBRATION_ANALYSIS(damping, damping_ratio, LINEAR, duration, dt, TOTAL_MASS, u0, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes,
initial_coords, deformed_coords, DISPLACEMENTS_undamped, VELOCITY_undamped, ACCELERATION_undamped, DISP_X_undamped, DISP_Y_undamped, BASE_REACTION_X_undamped, BASE_REACTION_Y_undamped

# Run the damped analysis
damping = True
LINEAR = True # False: Cable NonLinear Materials Properties
results_damped = FREE_VIBRATION_ANALYSIS(damping, damping_ratio, LINEAR, duration, dt, TOTAL_MASS, u0, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, N
initial_coords, deformed_coords, DISPLACEMENTS_damped, VELOCITY_damped, ACCELERATION_damped, DISP_X_damped, DISP_Y_damped, BASE_REACTION_X_damped, BASE_REACTION_Y_damped = results_damp

totaltime = time.process_time() - starttime
print(f'\nTotal time (s): {totaltime:.4f} \n\n')
```

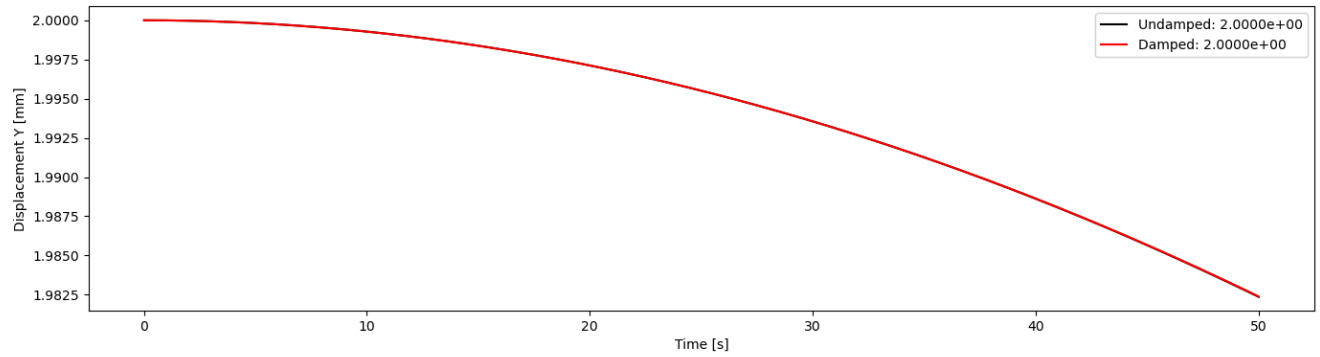
Node 1: X Displacement = 0.0, Y Displacement = 0.0
Node 2: X Displacement = -0.0060080285929951325, Y Displacement = -0.09561323463466348
Node 3: X Displacement = -0.011861368101459054, Y Displacement = -0.18913514782977003
Node 4: X Displacement = -0.017409051801990132, Y Displacement = -0.27848266326808296
Node 5: X Displacement = -0.022507495314215 94, Y Displacement = -0.3615891846792667
Node 6: X Displacement = -0.027024029350335637, Y Displacement = -0.43641275510935623
Node 7: X Displacement = -0.030840245147584534, Y Displacement = -0.5009441186472807
Node 8: X Displacement = -0.03385509486112691, Y Displacement = -0.5532146526426195
Node 9: X Displacement = -0.03598769314098611, Y Displacement = -0.5913041389681727
Node 10: X Displacement = -0.0371797709387947, Y Displacement = -0.613348343546968
Node 11: X Displacement = -0.03739773819661379, Y Displacement = -0.6175463741152317
Node 12: X Displacement = -0.0366343183848854 35, Y Displacement = -0.602167787123886
Node 13: X Displacement = -0.03490972476677465, Y Displacement = -0.5655594156872907
Node 14: X Displacement = -0.03227235567483926, Y Displacement = -0.5061518916333355
Node 15: X Displacement = -0.028798993862640412, Y Displacement = -0.4224658359676727
Node 16: X Displacement = -0.02459450302209153, Y Displacement = -0.31311769341571466
Node 17: X Displacement = -0.019791022705050173, Y Displacement = -0.17682518815795872
Node 18: X Displacement = -0.014546671033091976, Y Displacement = -0.012412379425652426
Node 19: X Displacement = -0.009043772574362046, Y Displacement = 0.18118570276721332
Node 20: X Displacement = -0.0034866365155758693, Y Displacement = 0.40491885979061 46
Node 21: X Displacement = 0.0019010824034236624, Y Displacement = 0.6596179807495808
Node 22: X Displacement = 0.006879400867173785, Y Displacement = 0.9459917674420825
Node 23: X Displacement = 0.011195355064763816, Y Displacement = 1.2646238465932016
Node 24: X Displacement = 0.014586019139360326, Y Displacement = 1.6159057115413942
Node 25: X Displacement = 0.016685525845849043, Y Displacement = 1.982360267064577
Node 26: X Displacement = 0.015817346337994476, Y Displacement = 1.5827670787783283
Node 27: X Displacement = 0.016501769131742814, Y Displacement = 1.1986093641051365
Node 28: X Displacement = 0.018515361022481722, Y Displacement = 0.8476257660667 09
Node 29: X Displacement = 0.021589531410905898, Y Displacement = 0.529684265642504
Node 30: X Displacement = 0.025457782500837776, Y Displacement = 0.24445761755015744
Node 31: X Displacement = 0.029859624654618 11, Y Displacement = -0.008510601089632145
Node 32: X Displacement = 0.034544424805533 63, Y Displacement = -0.22980409667014448
Node 33: X Displacement = 0.03927502792603657, Y Displacement = -0.4201315579629968
Node 34: X Displacement = 0.04383109525805321, Y Displacement = -0.5803238628298513
Node 35: X Displacement = 0.048012108263183036, Y Displacement = -0.7113308035047032
Node 36: X Displacement = 0.051639992787406504, Y Displacement = -0.8142173431104059
Node 37: X Displacement = 0.05456132417414063, Y Displacement = -0.8901594179074599
Node 38: X Displacement = 0.056649080917386734, Y Displacement = -0.9404393015632028
Node 39: X Displacement = 0.057803921801709776, Y Displacement = -0.9664405494805213
Node 40: X Displacement = 0.057954969224880 85, Y Displacement = -0.9696425429304212
Node 41: X Displacement = 0.057060089404370 34, Y Displacement = -0.9516146543227563
Node 42: X Displacement = 0.055105668321802935, Y Displacement = -0.9140100565182431
Node 43: X Displacement = 0.052105890410439865, Y Displacement = -0.8585592005267121
Node 44: X Displacement = 0.048101535026716 09, Y Displacement = -0.7870629872968078
Node 45: X Displacement = 0.043158313529581986, Y Displacement = -0.7013856605462575
Node 46: X Displacement = 0.037364777201634 67, Y Displacement = -0.6034474487364657
Node 47: X Displacement = 0.030829833163543596, Y Displacement = -0.49521698529723207
Node 48: X Displacement = 0.023679911749706 25, Y Displacement = -0.378703537129877 03
Node 49: X Displacement = 0.016055834424870626, Y Displacement = -0.255949072182558
Node 50: X Displacement = 0.008109436143016848, Y Displacement = -0.1290201975417752
Node 51: X Displacement = 0.0, Y Displacement = 0.0
Node 52: X Displacement = 0.0, Y Displacement = 0.0
Node 53: X Displacement = 1.1778779369701396e-05, Y Displacement = -0.0956132298201296
Node 54: X Displacement = 2.3755334300911662e-05, Y Displacement = -0.18913512779589212
Node 55: X Displacement = 3.6113567558451427e-05, Y Displacement = -0.2784826186464013
Node 56: X Displacement = 4.900994626475564e-05, Y Displacement = -0.36158910814575607
Node 57: X Displacement = 6.256038322755048e-05, Y Displacement = -0.43641264223408976
Node 58: X Displacement = 7.682795370523368e-05, Y Displacement = -0.5009439684845365
Node 59: X Displacement = 9.181156255459337e-05, Y Displacement = -0.5532144679840715
Node 60: X Displacement = 0.00010743586743282566, Y Displacement = -0.591303926215029
Node 61: X Displacement = 0.00012354259944439641, Y Displacement = -0.6133481121819705
Node 62: X Displacement = 0.00013988348886551048, Y Displacement = -0.6175461358125576
Node 63: X Displacement = 0.00015611493688544051, Y Displacement = -0.6021675545656181
Node 64: X Displacement = 0.00017179454259175263, Y Displacement = -0.5655592012015112
Node 65: X Displacement = 0.00018637960944957856, Y Displacement = -0.5061517058035979
Node 66: X Displacement = 0.00019922762232655233, Y Displacement = -0.42246568637908466
Node 67: X Displacement = 0.00020959880356322615, Y Displacement = -0.3131175837070456
Node 68: X Displacement = 0.00021666066180343373, Y Displacement = -0.17682511753430166
Node 69: X Displacement = 0.00021949454972532988, Y Displacement = -0.012412342738778013
Node 70: X Displacement = 0.00021710410765241 91, Y Displacement = 0.18118571432195796
Node 71: X Displacement = 0.00020842549969969 2, Y Displacement = 0.40491885738985145
Node 72: X Displacement = 0.00019233932136460965, Y Displacement = 0.6596179761922891
Node 73: X Displacement = 0.00016768398672673 55, Y Displacement = 0.9459917711512001
Node 74: X Displacement = 0.00013327041147741088, Y Displacement = 1.2646238656619513
Node 75: X Displacement = 8.792055786343145e-05, Y Displacement = 1.6159057482070747
Node 76: X Displacement = 3.7125703827647 46e-05, Y Displacement = 1.98236031823376
Node 77: X Displacement = -2.6362033427755 27e-05, Y Displacement = 1.582767123716312
Node 78: X Displacement = -8.380098141020887e-05, Y Displacement = 1.1986094137944079
Node 79: X Displacement = -0.00012904608444746208, Y Displacement = 0.8476258301061003
Node 80: X Displacement = -0.00016323983311232617, Y Displacement = 0.52968435437995
Node 81: X Displacement = -0.00018756729775372755, Y Displacement = 0.24445774226597608
Node 82: X Displacement = -0.00020321409371340396, Y Displacement = -0.008510428935229512
Node 83: X Displacement = -0.00021134982724928307, Y Displacement = -0.22980386673500042
Node 84: X Displacement = -0.00021311242272187622, Y Displacement = -0.42013126250646166
Node 85: X Displacement = -0.00020959352759318198, Y Displacement = -0.5803234980668742
Node 86: X Displacement = -0.00020182524876051413, Y Displacement = -0.7113303705852546
Node 87: X Displacement = -0.00019076837691 2784, Y Displacement = -0.8142168485634572
Node 88: X Displacement = -0.00017730228200561393, Y Displacement = -0.8901588734789692
Node 89: X Displacement = -0.00016221661747 15648, Y Displacement = -0.9404387234688801
Node 90: X Displacement = -0.00014620495126941297, Y Displacement = -0.9664399571685 68
Node 91: X Displacement = -0.00012986037470152193, Y Displacement = -0.9696419574891362
Node 92: X Displacement = -0.00011367317174651194, Y Displacement = -0.9516140967162735
Node 93: X Displacement = -9.803051443523192e-05, Y Displacement = -0.9140095458323928
Node 94: X Displacement = -8.321819901146637e-05, Y Displacement = -0.8585587523974632
Node 95: X Displacement = -6.942433362624153e-05, Y Displacement = -0.7870626126632295
Node 96: X Displacement = -5.674491264104725e-05, Y Displacement = -0.7013853648355397
Node 97: X Displacement = -4.519113455045774e-05, Y Displacement = -0.6034472315328188
Node 98: X Displacement = -3.469832340319181e-05, Y Displacement = -0.4952168405106385
Node 99: X Displacement = -2.5136285529064447e-05, Y Displacement = -0.378703453626818 34
Node 100: X Displacement = -1.6320882901463002e-05, Y Displacement = -0.25594903480514725
Node 101: X Displacement = -8.026623113259083e-06, Y Displacement = -0.12902018841351404
Node 102: X Displacement = 0.0, Y Displacement = 0.0
Node 1: X Displacement = 0.0, Y Displacement = 0.0
Node 2: X Displacement = -0.0060080331713695541, Y Displacement = -0.0956132762521068
Node 3: X Displacement = -0.011861374322185344, Y Displacement = -0.18913523090028908
Node 4: X Displacement = -0.017409061081553143, Y Displacement = -0.2784827874635279
Node 5: X Displacement = -0.022507507591817875, Y Displacement = -0.36158934950938326
Node 6: X Displacement = -0.027024044546343058, Y Displacement = -0.4364129599236035
Node 7: X Displacement = -0.030840263165012424, Y Displacement = -0.5009443626363449
Node 8: X Displacement = -0.033855111558674835, Y Displacement = -0.5532149348432227
Node 9: X Displacement = -0.035987716447573 12, Y Displacement = -0.591304458267009
Node 10: X Displacement = -0.037179796686783 75, Y Displacement = -0.6133486986828904

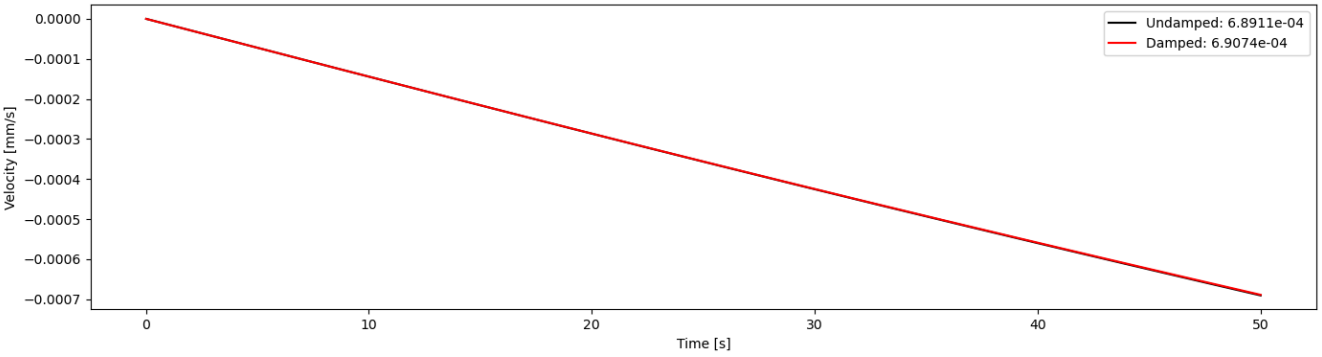
```
Node 11: X Displacement = -0.03739776623661078, Y Displacement = -0.617546763685933
Node 12: X Displacement = -0.03663434855982497, Y Displacement = -0.6021682095921316
Node 13: X Displacement = -0.03490975691450317, Y Displacement = -0.5655598693852096
Node 14: X Displacement = -0.03227238963060118, Y Displacement = -0.5061523747702859
Node 15: X Displacement = -0.02879902946191498 2, Y Displacement = -0.4224663466365517
Node 16: X Displacement = -0.02459454010269032 3, Y Displacement = -0.3131182296009271 7
Node 17: X Displacement = -0.01979106111034414 2, Y Displacement = -0.1768257477436469 3
Node 18: X Displacement = -0.01454671061385159 9, Y Displacement = -0.012412960203202717
Node 19: X Displacement = -0.00904381319170696, Y Displacement = 0.18118510308962835
Node 20: X Displacement = -0.00348667804278401 63, Y Displacement = 0.40491824357970513
Node 21: X Displacement = 0.00190104007864268 69, Y Displacement = 0.6596173504362032
Node 22: X Displacement = 0.00687935784129511 7, Y Displacement = 0.9459911255137
Node 23: X Displacement = 0.01119531141918354 7, Y Displacement = 1.2646231957744025
Node 24: X Displacement = 0.01458597609388319 9, Y Displacement = 1.6159051768845436
Node 25: X Displacement = 0.01668563807456626 2, Y Displacement = 1.9823878224901907
Node 26: X Displacement = 0.01581739139914461 3, Y Displacement = 1.5827665388915901
Node 27: X Displacement = 0.01650181388519425 5, Y Displacement = 1.1986087028521368
Node 28: X Displacement = 0.01851540525448878 3, Y Displacement = 0.8476251085418881
Node 29: X Displacement = 0.02158957508019629, Y Displacement = 0.5296836146301078
Node 30: X Displacement = 0.02545782554747551 6, Y Displacement = 0.24445697561946236
Node 31: X Displacement = 0.02985966699983633 5, Y Displacement = -0.008511231405330302
Node 32: X Displacement = 0.03454446635284037, Y Displacement = -0.22980471288441262
Node 33: X Displacement = 0.03927506856296576, Y Displacement = -0.42013215764358597
Node 34: X Displacement = 0.04383113485785760 5, Y Displacement = -0.5803244436100088
Node 35: X Displacement = 0.04801214668685263 6, Y Displacement = -0.7113313630925852
Node 36: X Displacement = 0.05164002988575461, Y Displacement = -0.8142178792987111
Node 37: X Displacement = 0.05456135979028566, Y Displacement = -0.8901599285793899
Node 38: X Displacement = 0.05664911488914014, Y Displacement = -0.9404397847026723
Node 39: X Displacement = 0.05780395396448986, Y Displacement = -0.9664410031808637
Node 40: X Displacement = 0.05795499941388425, Y Displacement = -0.9696429654013976
Node 41: X Displacement = 0.05706011745736784, Y Displacement = -0.9516150438960216
Node 42: X Displacement = 0.05510569408169014 4, Y Displacement = -0.9140104116565364
Node 43: X Displacement = 0.05210591372791159, Y Displacement = -0.8585595198277072
Node 44: X Displacement = 0.04810155576206049, Y Displacement = -0.7870632694988825
Node 45: X Displacement = 0.04315833155528795, Y Displacement = -0.7013859045369965
Node 46: X Displacement = 0.03736479240469782 5, Y Displacement = -0.6034476535521149
Node 47: X Displacement = 0.03082984544678917 7, Y Displacement = -0.4952171501284764
Node 48: X Displacement = 0.02367992103352621 6, Y Displacement = -0.378703661325713
Node 49: X Displacement = 0.01605584064836445 4, Y Displacement = -0.2559491552536373
Node 50: X Displacement = 0.00810943926505112 3, Y Displacement = -0.12902023915949598
Node 51: X Displacement = 0.0, Y Displacement = 0.0
Node 52: X Displacement = 0.0, Y Displacement = 0.0
Node 53: X Displacement = 1.17792080721669 7e-05, Y Displacement = -0.09561327143612255
Node 54: X Displacement = 2.37561887514325 3e-05, Y Displacement = -0.18913521086357388
Node 55: X Displacement = 3.61148545316573 9e-05, Y Displacement = -0.2784827428377992
Node 56: X Displacement = 4.90116598597391 44e-05, Y Displacement = -0.3615892729705327
Node 57: X Displacement = 6.25625254999624 8e-05, Y Displacement = -0.4364128470414856 4
Node 58: X Displacement = 7.68305244672577 7e-05, Y Displacement = -0.5009442124659488
Node 59: X Displacement = 9.18145647310835 4e-05, Y Displacement = -0.5532147501762615
Node 60: X Displacement = 0.00010743930044015775, Y Displacement = -0.5913042455039234
Node 61: X Displacement = 0.00012354646380703665, Y Displacement = -0.6133484673070918
Node 62: X Displacement = 0.00013988778709724865, Y Displacement = -0.6175465253720251
Node 63: X Displacement = 0.00015611966634921723, Y Displacement = -0.6021679770216563
Node 64: X Displacement = 0.00017179970553535227, Y Displacement = -0.565559654886562
Node 65: X Displacement = 0.00018638520728882983, Y Displacement = -0.5061521889269319
Node 66: X Displacement = 0.00019923365380315435, Y Displacement = -0.4224661970338402
Node 67: X Displacement = 0.00020960526964760546, Y Displacement = -0.3131181198777963
Node 68: X Displacement = 0.00021666756442791 9, Y Displacement = -0.1768256771050045
Node 69: X Displacement = 0.00021950188706562818, Y Displacement = -0.012412923501132172
Node 70: X Displacement = 0.00021711188175930495, Y Displacement = 0.18118511465979972
Node 71: X Displacement = 0.00020843370938293862, Y Displacement = 0.40491824119423575
Node 72: X Displacement = 0.00019234796894006 7, Y Displacement = 0.6596173458947433
Node 73: X Displacement = 0.00016769306688629743, Y Displacement = 0.9459911292389858
Node 74: X Displacement = 0.00013327992859015 83, Y Displacement = 1.2646232148584773
Node 75: X Displacement = 8.79304653963827 6e-05, Y Displacement = 1.6159052135657275
Node 76: X Displacement = 3.71257498902445 9e-05, Y Displacement = 1.9823878736733218
Node 77: X Displacement = -2.63727831384464 44e-05, Y Displacement = 1.58276658384612
Node 78: X Displacement = -8.381134221379375e-05, Y Displacement = 1.1986087525576101
Node 79: X Displacement = -0.00012905601384490694, Y Displacement = 0.8476251725972374
Node 80: X Displacement = -0.00016324932832792 77, Y Displacement = 0.5296837033839474
Node 81: X Displacement = -0.00018757635943548477, Y Displacement = 0.24445710035170293
Node 82: X Displacement = -0.00020322272072378057, Y Displacement = -0.008511059234856655
Node 83: X Displacement = -0.00021135801850044676, Y Displacement = -0.22980448293284775
Node 84: X Displacement = -0.00021312017553602 6, Y Displacement = -0.4201318621709831
Node 85: X Displacement = -0.00020960084463509927, Y Displacement = -0.5803240788316852
Node 86: X Displacement = -0.00020183213280484586, Y Displacement = -0.7113309301584803
Node 87: X Displacement = -0.00019077482607566076, Y Displacement = -0.8142173847367632
Node 88: X Displacement = -0.00017773082970689405, Y Displacement = -0.8901593841362829
Node 89: X Displacement = -0.00016222219939480054, Y Displacement = -0.9404392065947431
Node 90: X Displacement = -0.00014621009974732596, Y Displacement = -0.9664404108560914
Node 91: X Displacement = -0.00012986509121930363, Y Displacement = -0.969642379947546
Node 92: X Displacement = -0.00011367745458630401, Y Displacement = -0.9516144862779868
Node 93: X Displacement = -9.80343664205403 3e-05, Y Displacement = -0.9140099009596212
Node 94: X Displacement = -8.32216209712303 2e-05, Y Displacement = -0.8585590716882809
Node 95: X Displacement = -6.94273264449896 9e-05, Y Displacement = -0.7870628948571399
Node 96: X Displacement = -5.67474805133557 6e-05, Y Displacement = -0.7013856088184643
Node 97: X Displacement = -4.51932718498209 26e-05, Y Displacement = -0.6034474363414957
Node 98: X Displacement = -3.47000307088026 1e-05, Y Displacement = -0.4952170053364492
Node 99: X Displacement = -2.51375667843019 43e-05, Y Displacement = -0.3787035778189976
Node 100: X Displacement = -1.63217385237016 47e-05, Y Displacement = -0.2559491178733502
Node 101: X Displacement = -8.027050097114761e-06, Y Displacement = -0.12902023002976723
Node 102: X Displacement = 0.0, Y Displacement = 0.0

Total time (s): 7.9844
```

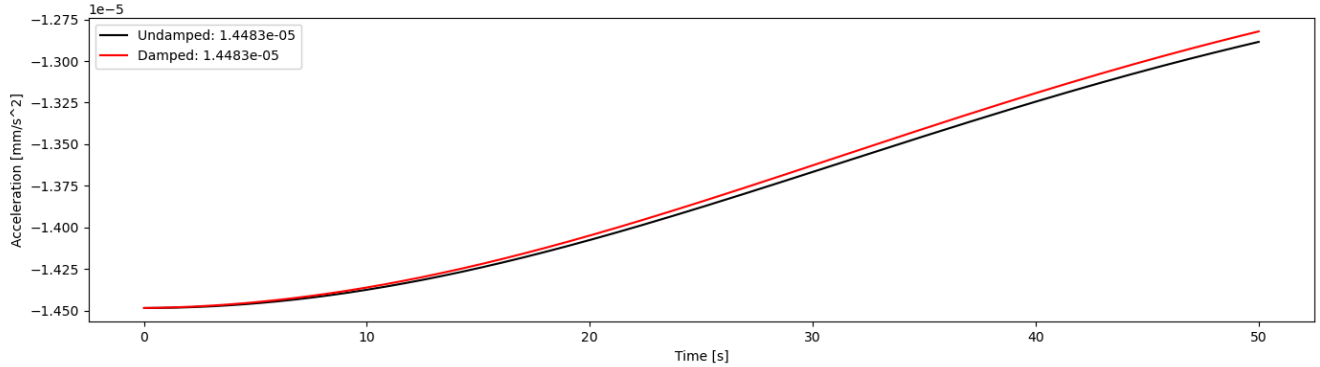
In [28]:
```
# Plotting the time history
TIME = np.arange(dt, duration+2*dt, dt)

plot_time_history(TIME, DISP_X_undamped, DISP_X_damped, DISP_Y_undamped, DISP_Y_damped,
                  VELOCITY_undamped, VELOCITY_damped, ACCELERATION_undamped, ACCELERATION_damped,
                  BASE_REACTION_X_undamped, BASE_REACTION_X_damped, BASE_REACTION_Y_undamped, BASE_REACTION_Y_damped)
```

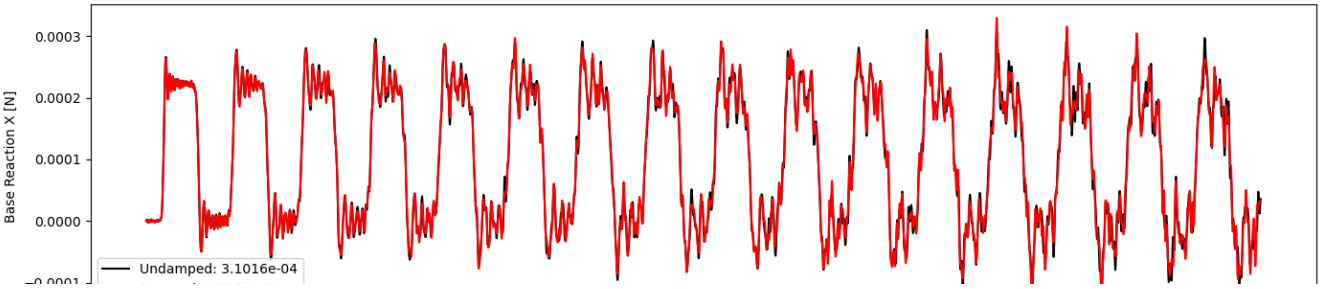
Time History of Free Vibration Analysis: Damped vs Undamped

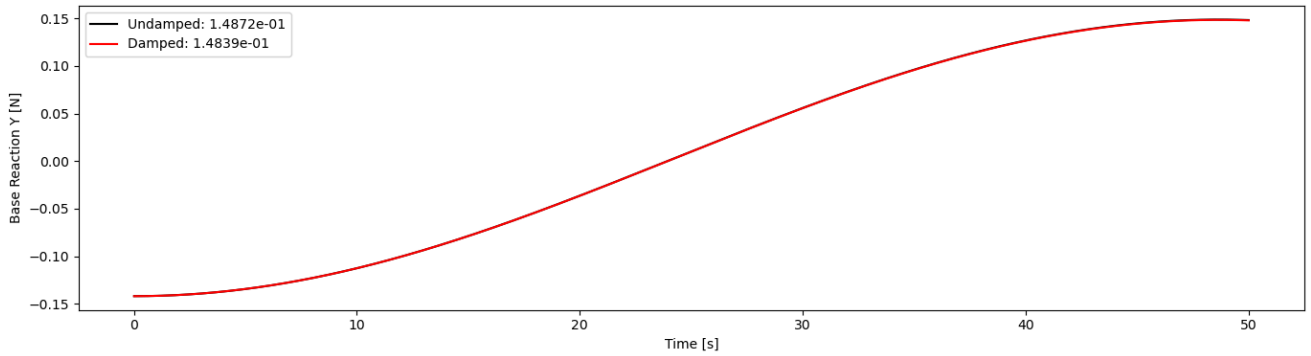




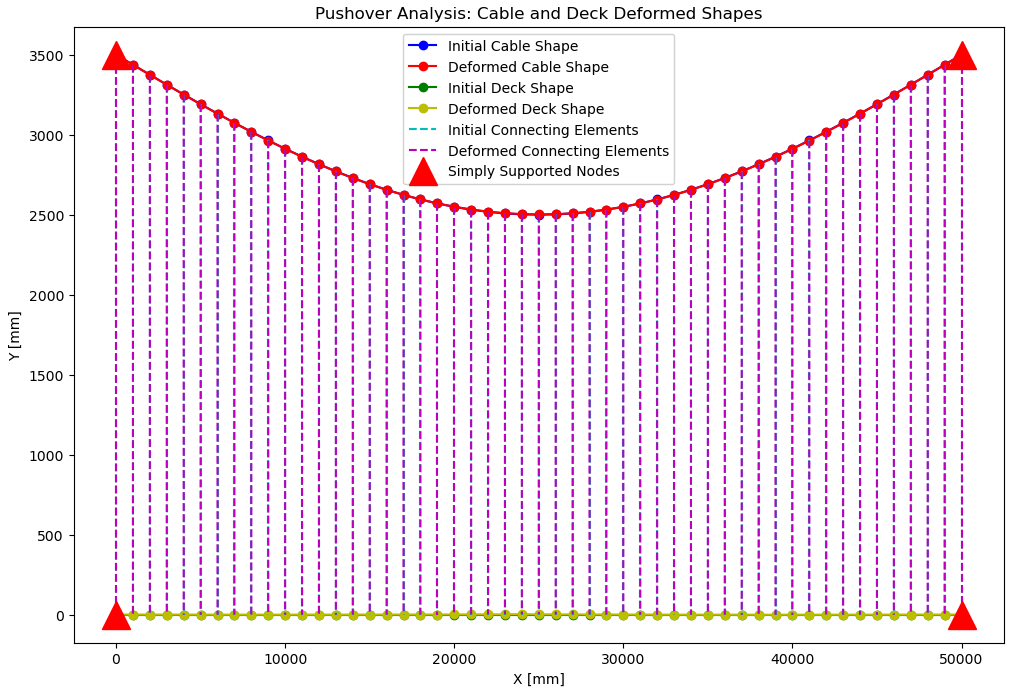


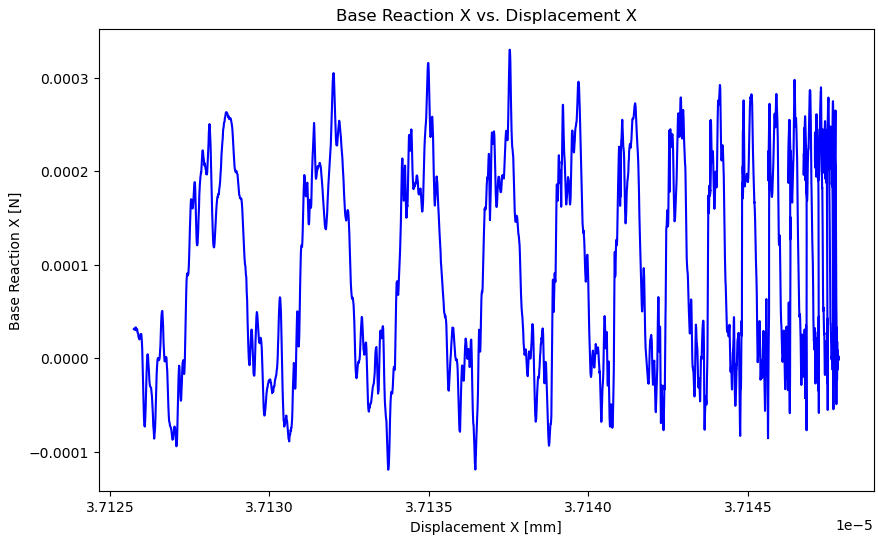


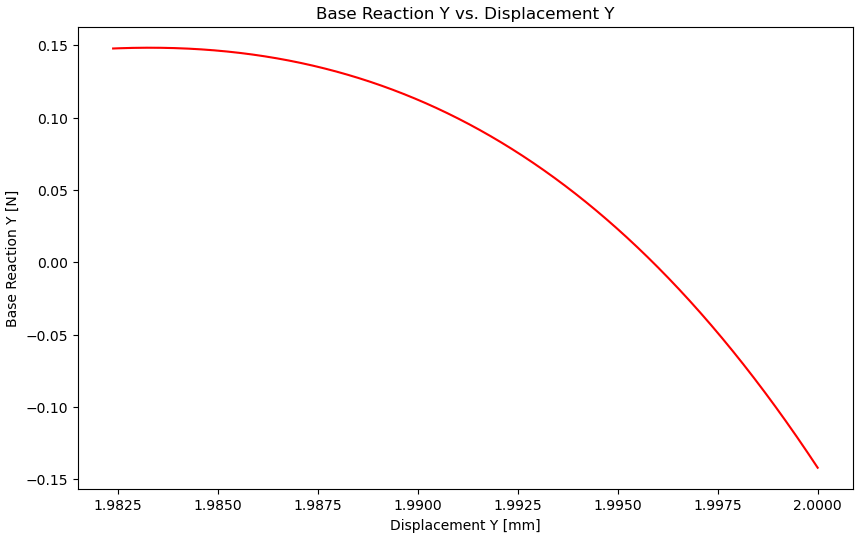




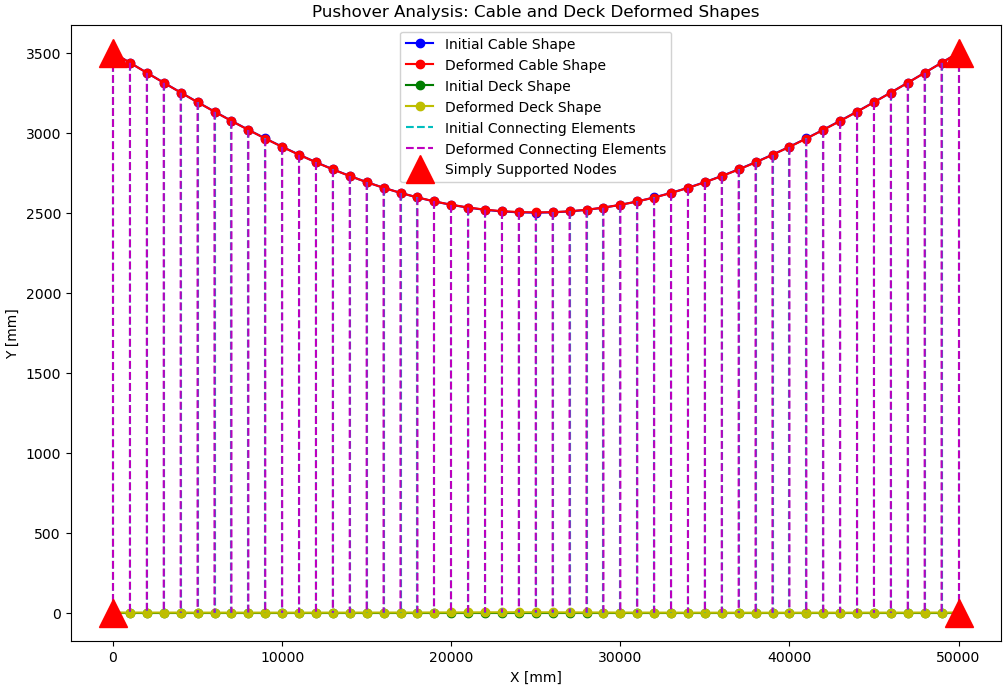
In [29]:
```
# Plot damped results
plot_shapes(initial_coords, DISPLACEMENTS_damped)
plot_reactions(DISP_X_damped, BASE_REACTION_X_damped, DISP_Y_damped, BASE_REACTION_Y_damped)
```

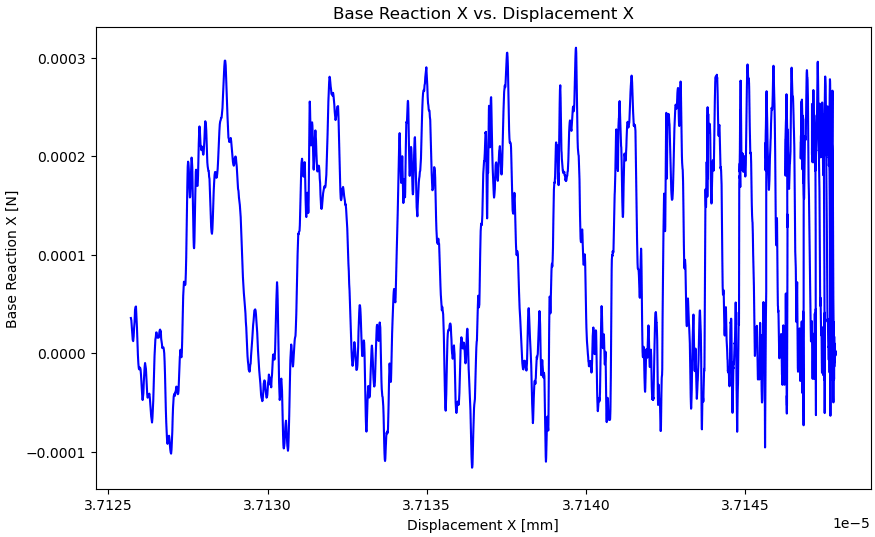


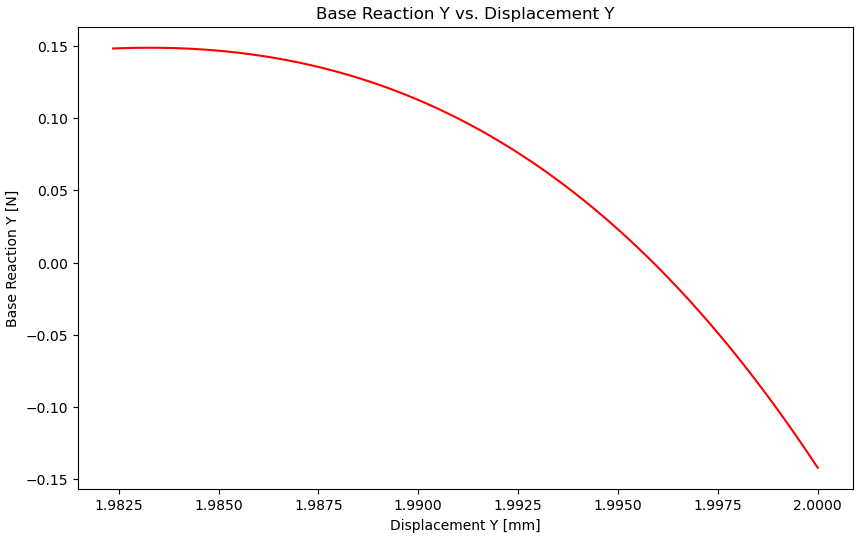




In [30]:
```
# Plot undamped results
plot_shapes(initial_coords, DISPLACEMENTS_undamped)
plot_reactions(DISP_X_undamped, BASE_REACTION_X_undamped, DISP_Y_undamped, BASE_REACTION_Y_undamped)
```







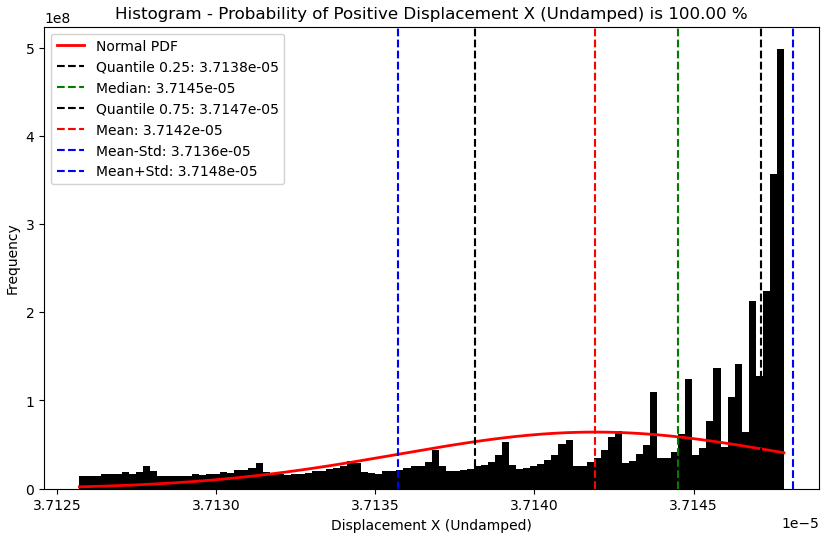
In [23]:

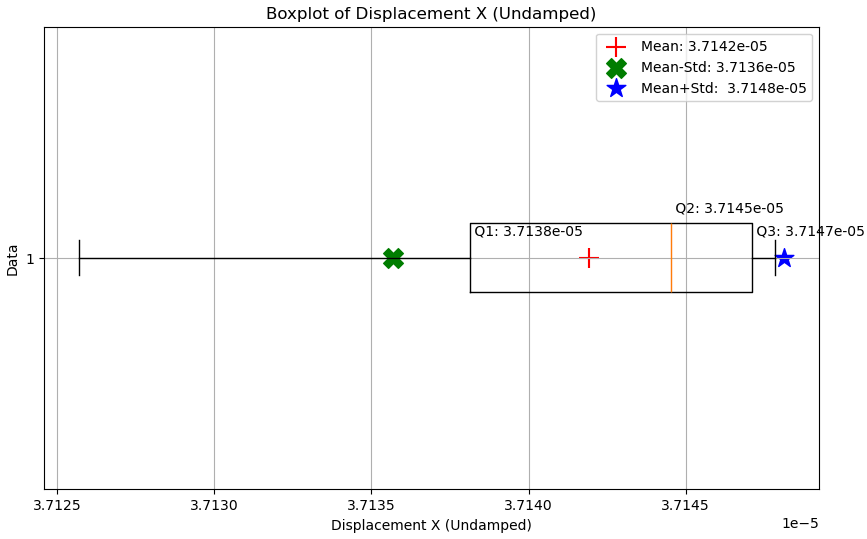
```python
def HISROGRAM_BOXPLOT(X, HISTO_COLOR, LABEL):
    import numpy as np
    import matplotlib.pyplot as plt
    X = np.array(X)
    print("-------------------------")
    from scipy.stats import skew, kurtosis
    MINIMUM = np.min(X)
    MAXIMUM = np.max(X)
    #MODE = max(set(X), key=list(X).count)
    MEDIAN = np.quantile(X, .50)#q2
    MEAN = np.mean(X)
    STD = np.std(X)
    q1 = np.quantile(X, .25)
    q3 = np.quantile(X, .75)
    SKEW = skew(X)
    KURT = kurtosis(X)
    #SKEW = (MEAN - MODE) / STD
    #KURT = (np.mean((X - MEAN)**4) / STD**4)
    # Estimate confidence intervals of the output variable
    lower_bound = np.quantile(X, .05)
    upper_bound = np.quantile(X, .95)
    print("Box-Chart Datas: ")
    print(f'Minimum: {MINIMUM:.4e}')
    print(f'First quartile: {q1:.4e}')
    #print(f'Mode: {MODE:.4e}')
    print(f'Median: {MEDIAN:.4e}')
    print(f'Mean: {MEAN:.4e}')
    print(f'Std: {STD:.4e}')
    print(f'Third quartile: {q3:.4e}')
    print(f'Maximum: {MAXIMUM :.4e}')
    print(f'Skewness: {skew(X) :.4e}')
    print(f'kurtosis: {kurtosis(X) :.4e}')
    print(f"90% Confidence Interval: ({lower_bound:.4e}, {upper_bound:.4e})")
    print("-------------------------")

    plt.figure(figsize=(10,6))
    # Plot histogram of data
    count, bins, ignored = plt.hist(X, bins=100, color=HISTO_COLOR, density=True, align='mid')#, edgecolor="black"

    # Plot lognormal PDF
    x = np.linspace(min(bins), max(bins), 10000)
    pdf = (np.exp(-(x - MEAN)**2 / (2 * STD**2)) / (STD * np.sqrt(2 * np.pi)))
    plt.plot(x, pdf, linewidth=2, color='r', label="Normal PDF")

    # Plot vertical lines for risk measures
    plt.axvline(q1, color="black", linestyle="--", label=f"Quantile 0.25: {q1:.4e}")
    plt.axvline(MEDIAN, color="green", linestyle="--", label=f"Median: {MEDIAN:.4e}")
    plt.axvline(q3, color="black", linestyle="--", label=f"Quantile 0.75: {q3:.4e}")
    #plt.axvline(MODE, color="purple", linestyle="--", label=f"Mode: {MODE:.4e}")
    plt.axvline(MEAN, color="red", linestyle="--", label=f"Mean: {MEAN:.4e}")
    plt.axvline(MEAN-STD, color="blue", linestyle="--", label=f"Mean-Std: {MEAN-STD:.4e}")
    plt.axvline(MEAN+STD, color="blue", linestyle="--", label=f"Mean+Std: {MEAN+STD:.4e}")
    plt.xlabel(LABEL)
    plt.ylabel("Frequency")
    prob = np.sum(X > 0) / len(X)
    plt.title(f"Histogram - Probability of Positive {LABEL} is {100*prob:.2f} %")
    plt.legend()
    #plt.grid()
    plt.show()

    #Plot boxplot with outliers
    plt.figure(figsize=(10,6))
    plt.boxplot(X, vert=0)
    # Write the quartile data on the chart
    plt.text(q1, 1.05, f" Q1: {q1:.4e}")
    plt.text(MEDIAN, 1.1, f" Q2: {MEDIAN:.4e}")
    plt.text(q3, 1.05, f" Q3: {q3:.4e}")
    #plt.text(MODE, 1.15, f" Mode: {MODE:.4e}")

    #plt.text(MEAN, 0.9, f" Mean: {MEAN:.4e}")
    #plt.text(MEAN-STD, 0.9, f" Mean-Std: {MEAN-STD:.4e}")
    #plt.text(MEAN+STD, 0.9, f" Mean+Std: {MEAN+STD:.4e}")
    plt.scatter(MEAN, 1, color="red", marker="+", s=200, label=f"Mean: {MEAN:.4e}")
    plt.scatter(MEAN-STD, 1, color="green", marker="X", s=200, label=f"Mean-Std: {MEAN-STD:.4e}")
    plt.scatter(MEAN+STD, 1, color="blue", marker="*", s=200, label=f"Mean+Std:  {MEAN+STD:.4e}")
    plt.xlabel(LABEL)
    plt.ylabel("Data")
    plt.title(f"Boxplot of {LABEL}")
    plt.legend()
```

```
    plt.grid()
    plt.show()
# ------------------------------------------------
def HISTOGRAM_BOXPLOT_PLOTLY( DATA, XLABEL='X', TITLE='A', COLOR='cyan'):
    # Plotting histogram and boxplot
    import plotly.express as px
    fig = px.histogram(x=DATA, marginal="box", color_discrete_sequence=[COLOR])
    fig.update_layout(title=TITLE, xaxis_title=XLABEL, yaxis_title="Frequency")
    fig.show()
```

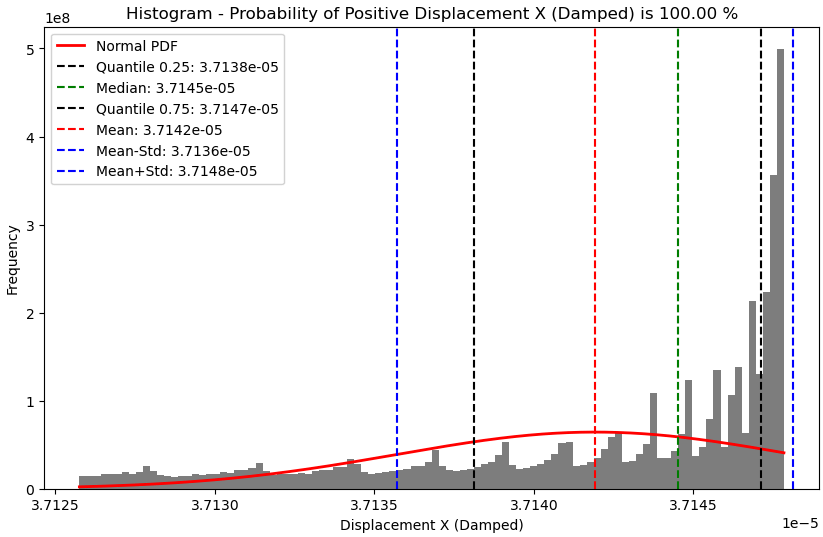
In [32]:
```
HISROGRAM_BOXPLOT(DISP_X_undamped, HISTO_COLOR='black', LABEL='Displacement X (Undamped)')
HISROGRAM_BOXPLOT(DISP_X_damped, HISTO_COLOR='grey', LABEL='Displacement X (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_X_undamped, XLABEL='Displacement X (Undamped)', TITLE='Displacement X (Undamped)', COLOR='black')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_X_damped, XLABEL='Displacement X (Damped)', TITLE='Displacement X (Damped)', COLOR='grey')
```

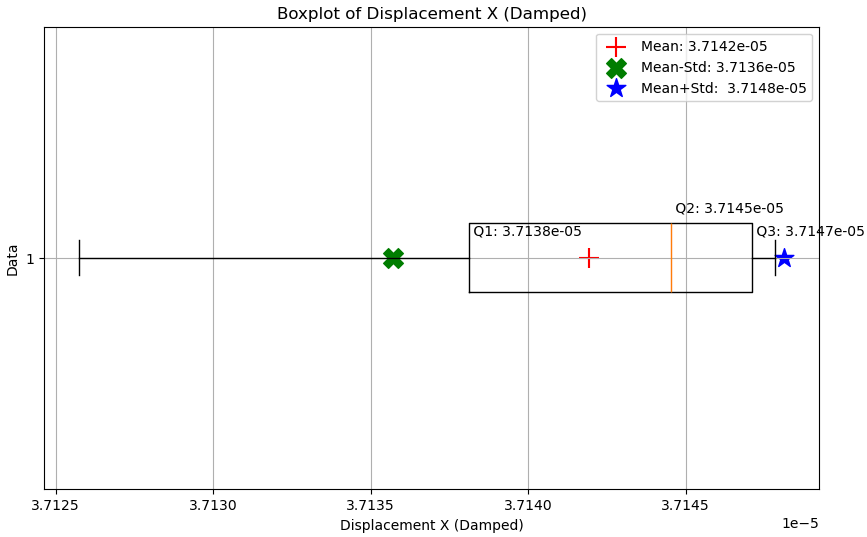
```
-------------------------
Box-Chart Datas:
Minimum: 3.7126e-05
First quartile: 3.7138e-05
Median: 3.7145e-05
Mean: 3.7142e-05
Std: 6.2137e-09
Third quartile: 3.7147e-05
Maximum: 3.7148e-05
Skewness: -1.0058e+00
kurtosis: -1.8075e-01
90% Confidence Interval: (3.7129e-05, 3.7148e-05)
-------------------------
```





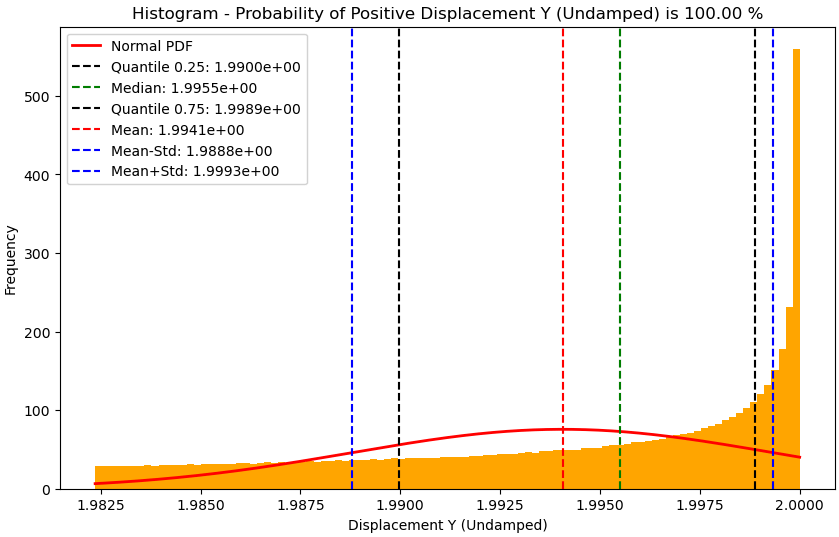

```
-------------------------
Box-Chart Datas:
Minimum: 3.7126e-05
First quartile: 3.7138e-05
Median: 3.7145e-05
Mean: 3.7142e-05
Std: 6.2016e-09
Third quartile: 3.7147e-05
Maximum: 3.7148e-05
Skewness: -1.0050e+00
kurtosis: -1.8262e-01
90% Confidence Interval: (3.7129e-05, 3.7148e-05)
-------------------------
```

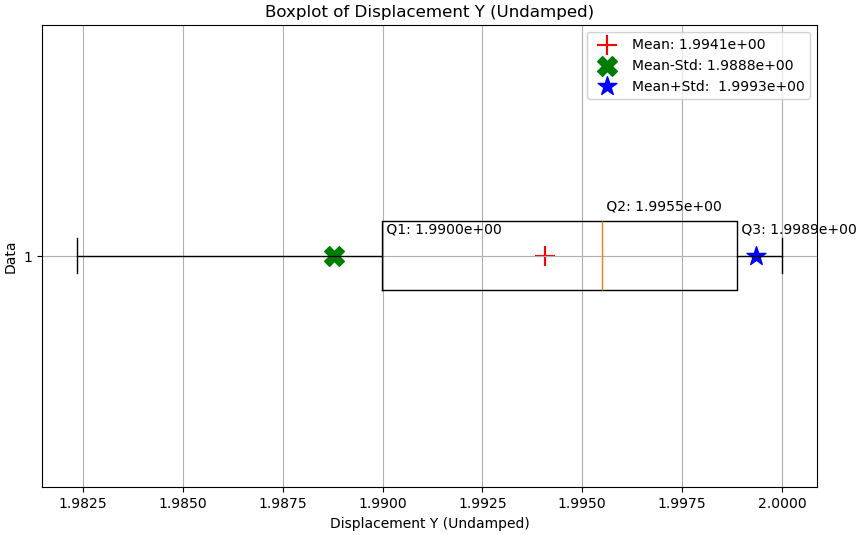




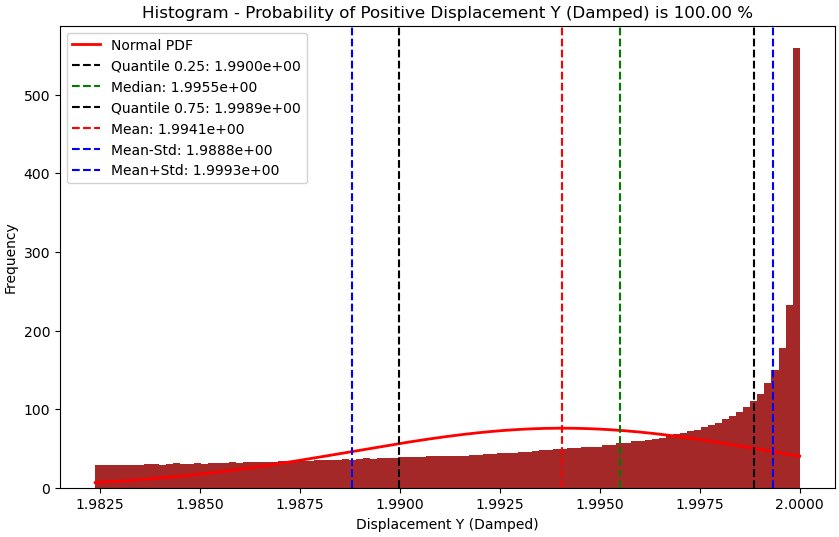
In [33]:
```
HISROGRAM_BOXPLOT(DISP_Y_undamped, HISTO_COLOR='orange', LABEL='Displacement Y (Undamped)')
HISROGRAM_BOXPLOT(DISP_Y_damped, HISTO_COLOR='brown', LABEL='Displacement Y (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_Y_undamped, XLABEL='Displacement Y (Undamped)', TITLE='Displacement Y (Undamped)', COLOR='orange')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_Y_damped, XLABEL='Displacement Y (Damped)', TITLE='Displacement Y (Damped)', COLOR='brown')
```

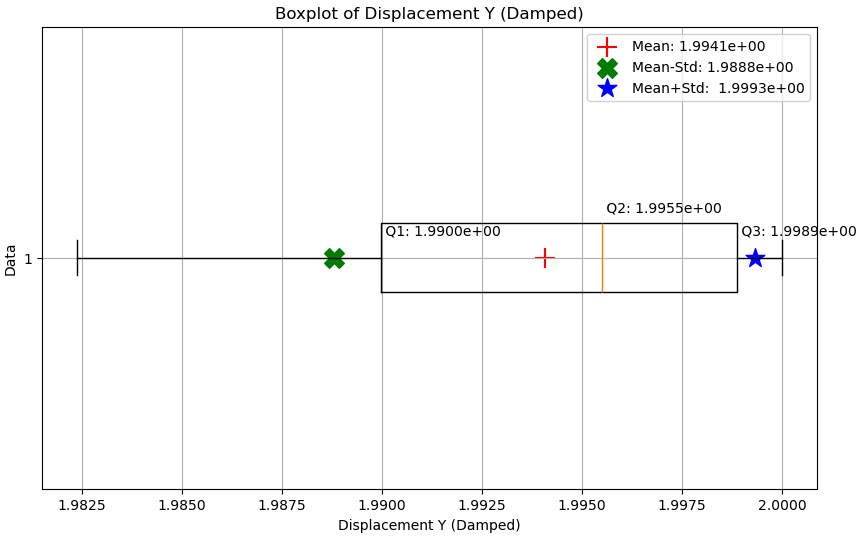
```
-------------------------
Box-Chart Datas:
Minimum: 1.9824e+00
First quartile: 1.9900e+00
Median: 1.9955e+00
Mean: 1.9941e+00
Std: 5.2758e-03
Third quartile: 1.9989e+00
Maximum: 2.0000e+00
Skewness: -6.2305e-01
kurtosis: -8.8164e-01
90% Confidence Interval: (1.9840e+00, 2.0000e+00)
-------------------------
```





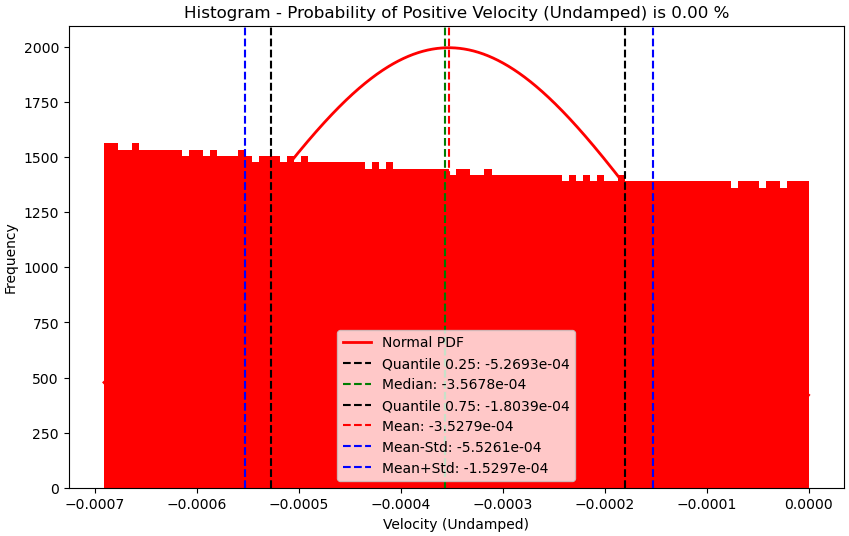

```
-------------------------
Box-Chart Datas:
Minimum: 1.9824e+00
First quartile: 1.9900e+00
Median: 1.9955e+00
Mean: 1.9941e+00
Std: 5.2681e-03
Third quartile: 1.9989e+00
Maximum: 2.0000e+00
Skewness: -6.2241e-01
kurtosis: -8.8257e-01
90% Confidence Interval: (1.9841e+00, 2.0000e+00)
-------------------------
```

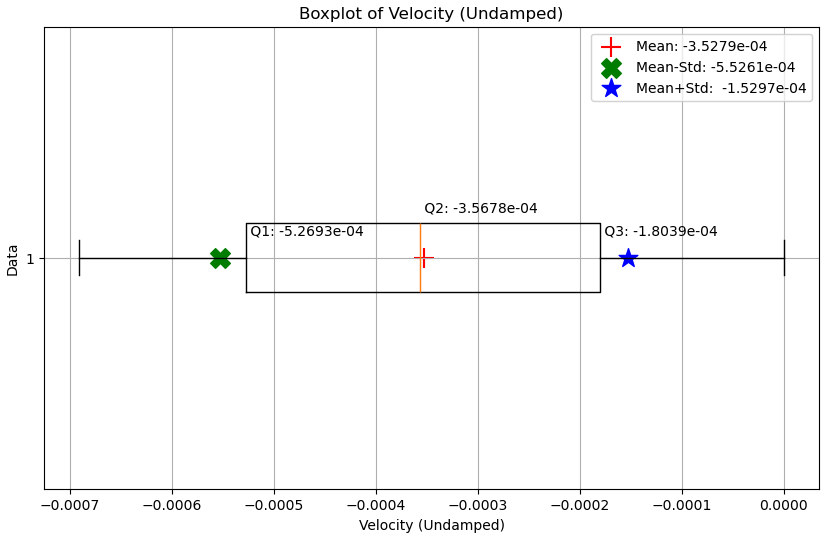




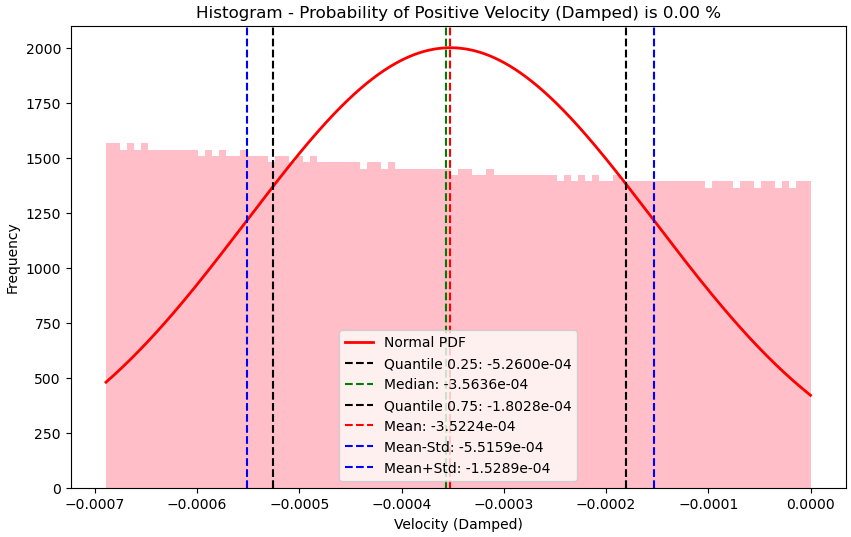
In [34]:
```
HISROGRAM_BOXPLOT(VELOCITY_undamped, HISTO_COLOR='red', LABEL='Velocity (Undamped)')
HISROGRAM_BOXPLOT(VELOCITY_damped, HISTO_COLOR='pink', LABEL='Velocity (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(VELOCITY_undamped, XLABEL='Velocity (Undamped)', TITLE='Velocity (Undamped)', COLOR='red')
#HISTOGRAM_BOXPLOT_PLOTLY(VELOCITY_damped, XLABEL='Velocity (Damped)', TITLE='Velocity (Damped)', COLOR='pink')
```

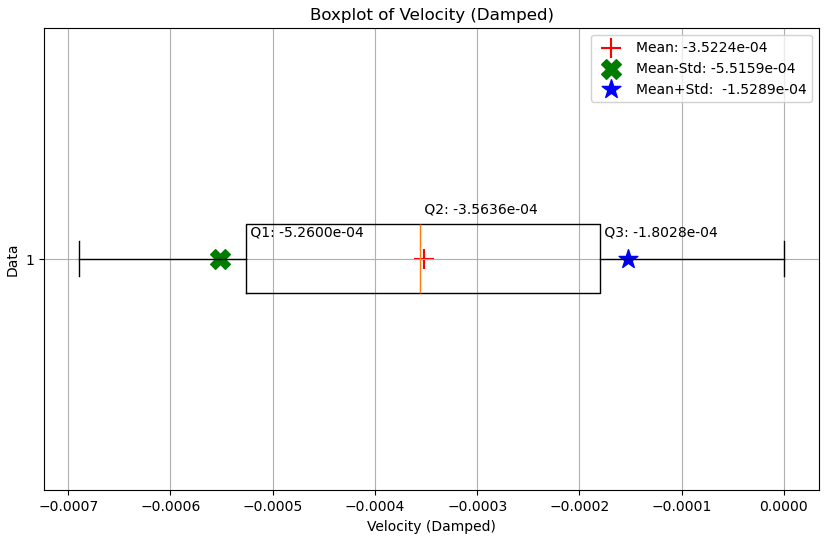
```
-------------------------
Box-Chart Datas:
Minimum: -6.9074e-04
First quartile: -5.2693e-04
Median: -3.5678e-04
Mean: -3.5279e-04
Std: 1.9982e-04
Third quartile: -1.8039e-04
Maximum: -7.2413e-08
Skewness: 4.6243e-02
kurtosis: -1.2022e+00
90% Confidence Interval: (-6.5842e-04, -3.6273e-05)
-------------------------
```






```
-------------------------
Box-Chart Datas:
Minimum: -6.8911e-04
First quartile: -5.2600e-04
Median: -3.5636e-04
Mean: -3.5224e-04
Std: 1.9935e-04
Third quartile: -1.8028e-04
Maximum: -7.2413e-08
Skewness: 4.7925e-02
kurtosis: -1.2020e+00
90% Confidence Interval: (-6.5695e-04, -3.6269e-05)
-------------------------
```

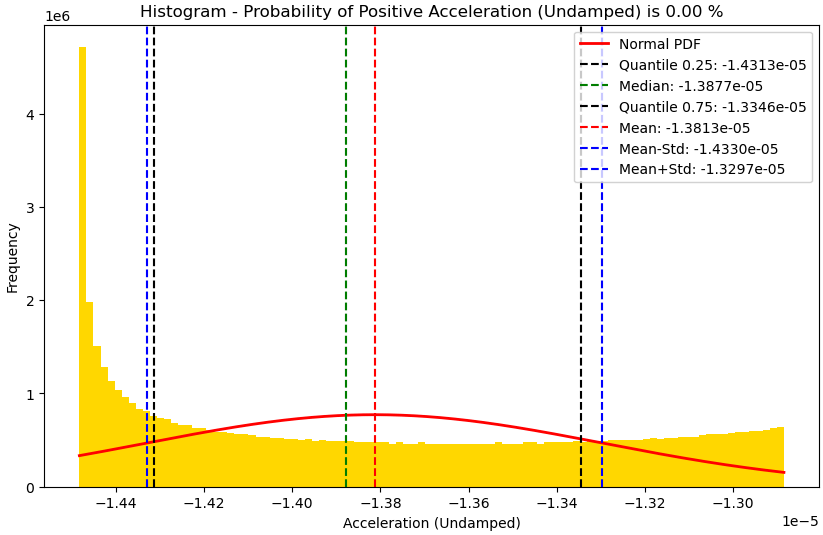


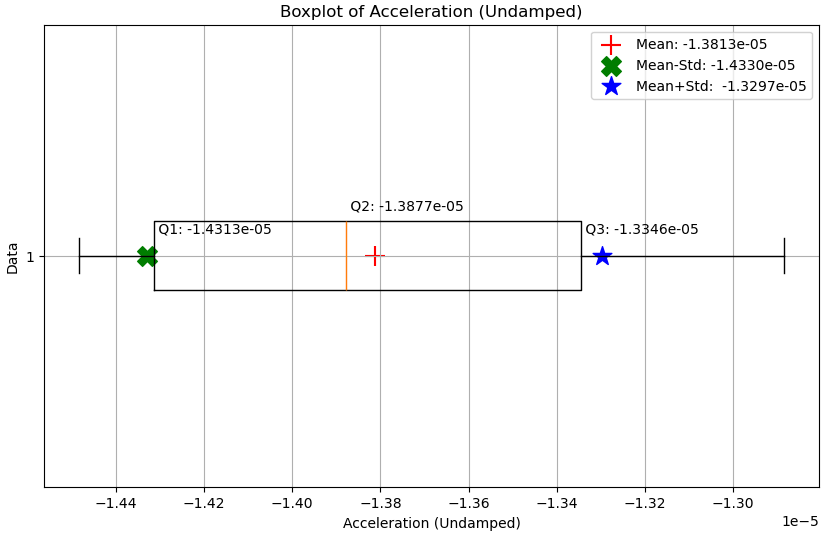


In [35]:
```
HISROGRAM_BOXPLOT(ACCELERATION_undamped, HISTO_COLOR='gold', LABEL='Acceleration (Undamped)')
HISROGRAM_BOXPLOT(ACCELERATION_damped, HISTO_COLOR='brown', LABEL='Acceleration (Damped)')
```

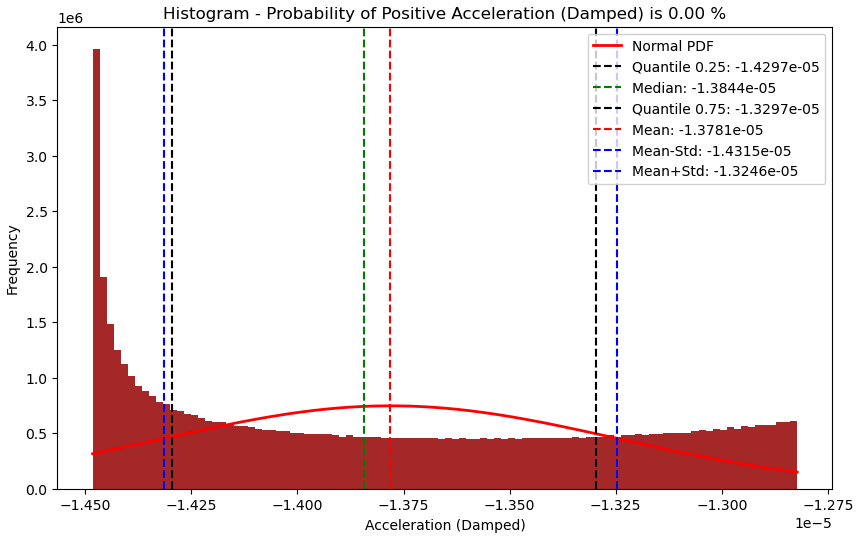
```
#HISTOGRAM_BOXPLOT_PLOTLY(ACCELERATION_undamped, XLABEL='Acceleration (Undamped)', TITLE='Acceleration (Undamped)', COLOR='gold')
#HISTOGRAM_BOXPLOT_PLOTLY(ACCELERATION_damped, XLABEL='Acceleration (Damped)', TITLE='Acceleration (Damped)', COLOR='brown')
```

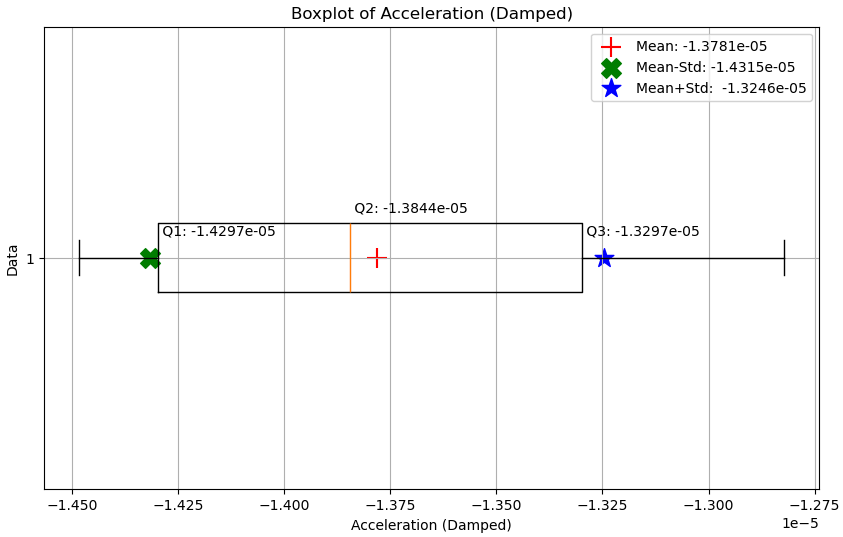
```
-------------------------
Box-Chart Datas:
Minimum: -1.4483e-05
First quartile: -1.4313e-05
Median: -1.3877e-05
Mean: -1.3813e-05
Std: 5.1642e-07
Third quartile: -1.3346e-05
Maximum: -1.2885e-05
Skewness: 2.5677e-01
kurtosis: -1.3241e+00
90% Confidence Interval: (-1.4476e-05, -1.2967e-05)
-------------------------
```





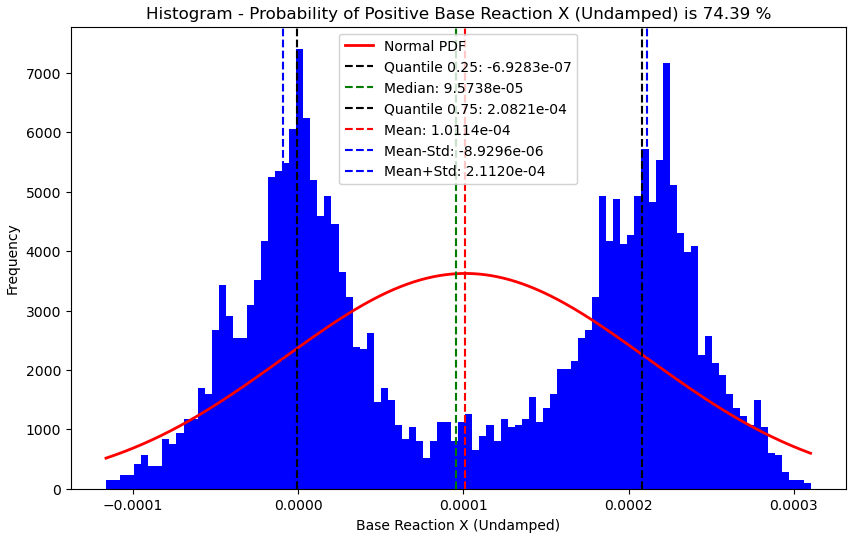

```
-------------------------
Box-Chart Datas:
Minimum: -1.4483e-05
First quartile: -1.4297e-05
Median: -1.3844e-05
Mean: -1.3781e-05
Std: 5.3434e-07
Third quartile: -1.3297e-05
Maximum: -1.2822e-05
Skewness: 2.4734e-01
kurtosis: -1.3235e+00
90% Confidence Interval: (-1.4472e-05, -1.2907e-05)
-------------------------
```

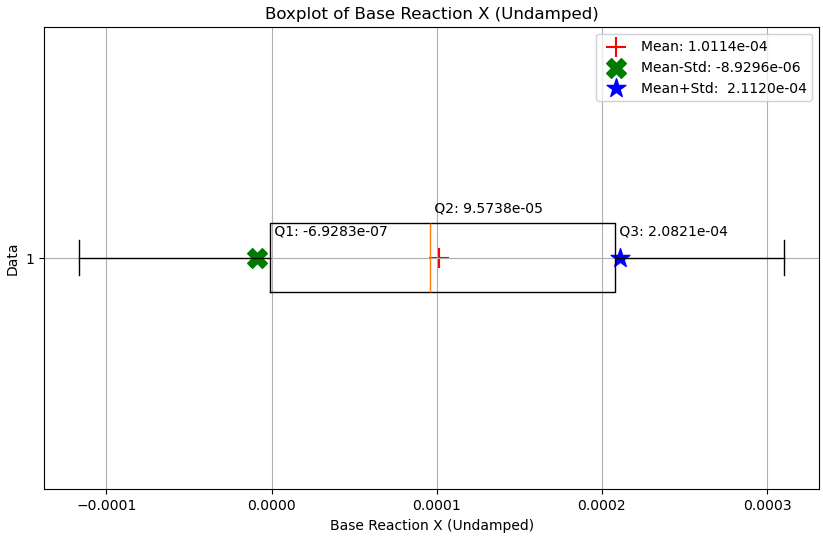




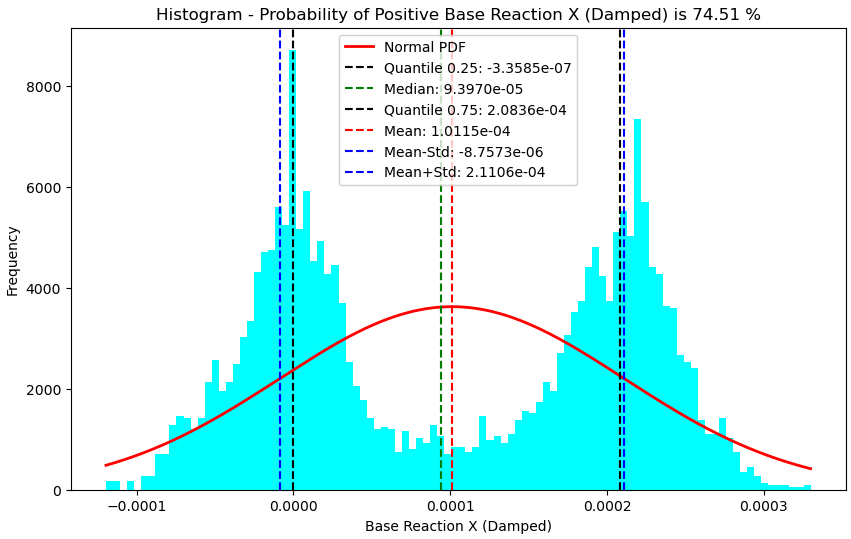
In [36]:
```
HISROGRAM_BOXPLOT(BASE_REACTION_X_undamped, HISTO_COLOR='blue', LABEL='Base Reaction X (Undamped)')
HISROGRAM_BOXPLOT(BASE_REACTION_X_damped, HISTO_COLOR='cyan', LABEL='Base Reaction X (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_X_undamped, XLABEL='Base Reaction X (Undamped)', TITLE='Base Reaction X (Undamped)', COLOR='blue')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_X_damped, XLABEL='Base Reaction X (Damped)', TITLE='Base Reaction X (Damped)', COLOR='cyan')
```

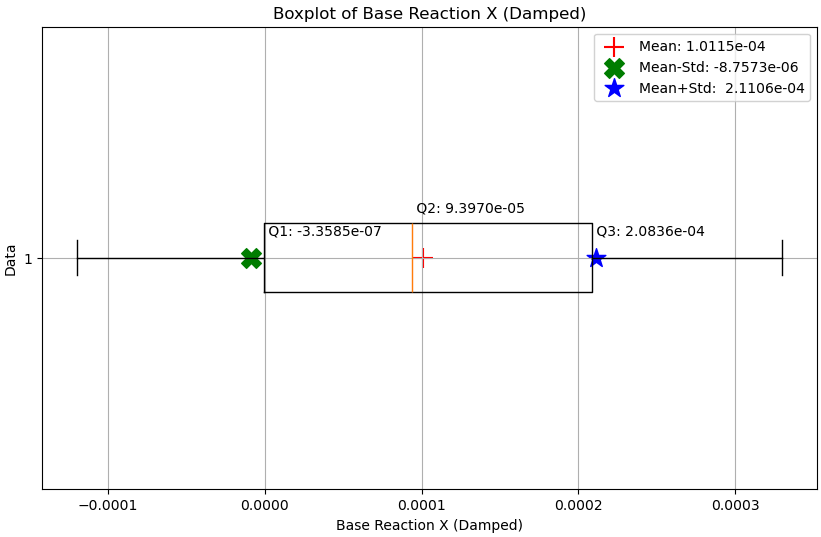
```
-------------------------
Box-Chart Datas:
Minimum: -1.1630e-04
First quartile: -6.9283e-07
Median: 9.5738e-05
Mean: 1.0114e-04
Std: 1.1007e-04
Third quartile: 2.0821e-04
Maximum: 3.1016e-04
Skewness: 1.9720e-02
kurtosis: -1.5392e+00
90% Confidence Interval: (-5.0007e-05, 2.5391e-04)
-------------------------
```





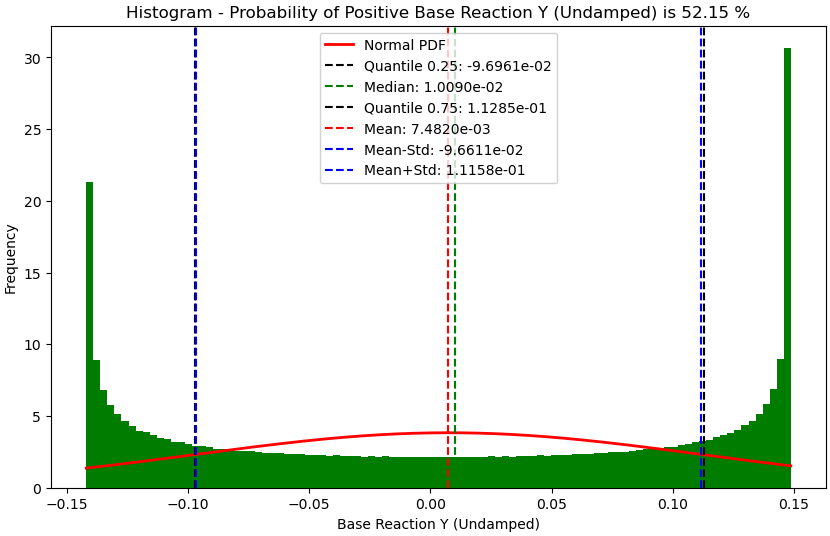

```
-------------------------
Box-Chart Datas:
Minimum: -1.1947e-04
First quartile: -3.3585e-07
Median: 9.3970e-05
Mean: 1.0115e-04
Std: 1.0991e-04
Third quartile: 2.0836e-04
Maximum: 3.2975e-04
Skewness: 1.8111e-02
kurtosis: -1.5364e+00
90% Confidence Interval: (-5.2773e-05, 2.5308e-04)
-------------------------
```

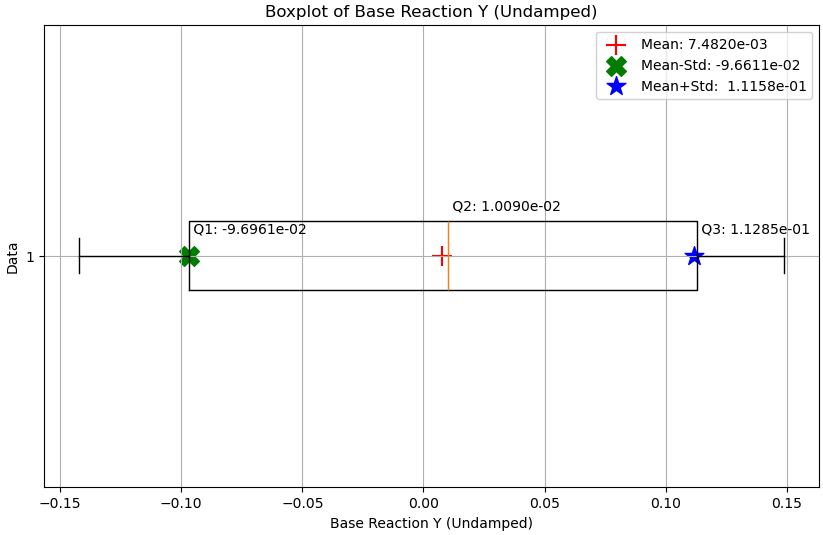




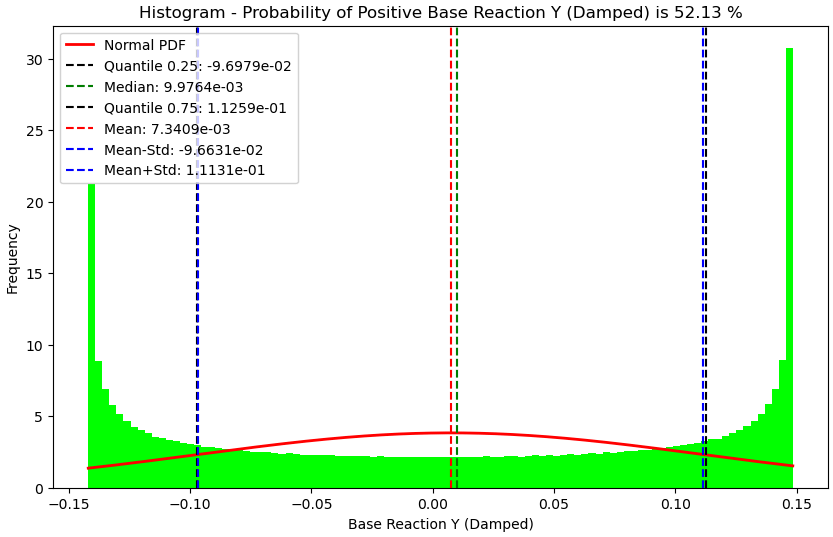
In [37]:
```
HISROGRAM_BOXPLOT(BASE_REACTION_Y_undamped, HISTO_COLOR='green', LABEL='Base Reaction Y (Undamped)')
HISROGRAM_BOXPLOT(BASE_REACTION_Y_damped, HISTO_COLOR='lime', LABEL='Base Reaction Y (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_Y_undamped, XLABEL='Base Reaction Y (Undamped)', TITLE='Base Reaction Y (Undamped)', COLOR='green')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_Y_damped, XLABEL='Base Reaction Y (Damped)', TITLE='Base Reaction Y (Damped)', COLOR='lime')
```

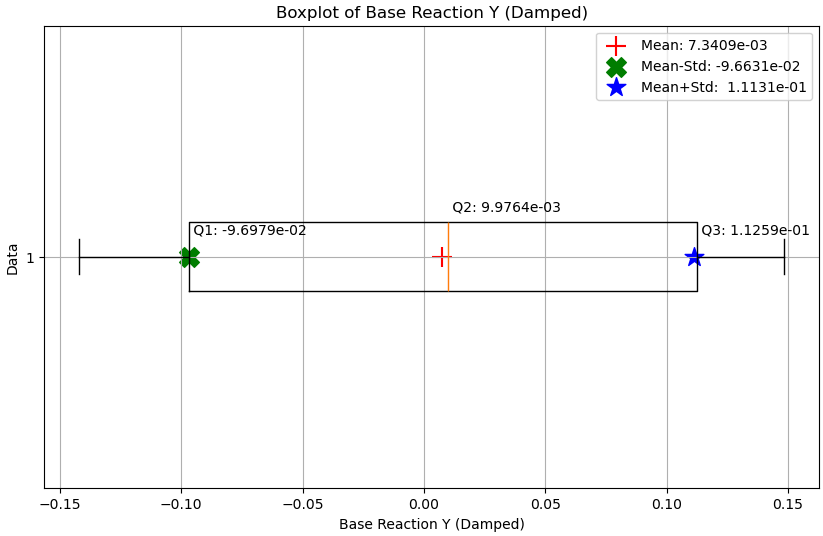
```
-------------------------
Box-Chart Datas:
Minimum: -1.4199e-01
First quartile: -9.6961e-02
Median: 1.0090e-02
Mean: 7.4820e-03
Std: 1.0409e-01
Third quartile: 1.1285e-01
Maximum: 1.4872e-01
Skewness: -4.2545e-02
kurtosis: -1.5124e+00
90% Confidence Interval: (-1.4008e-01, 1.4825e-01)
-------------------------
```






```
-------------------------
Box-Chart Datas:
Minimum: -1.4199e-01
First quartile: -9.6979e-02
Median: 9.9764e-03
Mean: 7.3409e-03
Std: 1.0397e-01
Third quartile: 1.1259e-01
Maximum: 1.4839e-01
Skewness: -4.2987e-02
kurtosis: -1.5123e+00
90% Confidence Interval: (-1.4008e-01, 1.4792e-01)
-------------------------
```





In [15]:
```
#          ##########################################################################################
#          #                                    IN THE NAME OF ALLAH                                  #
```

```
#         #                          DYNAMIC ANALYSIS OF CABLE SUSPENSION BRIDGE 01                 #
#         #-------------------------------------------------------------------------------------#
#         #                  THIS PROGRAM WRITTEN BY SALAR DELAVAR GHASHGHAEI (QASHQAI)             #
#         #                              EMAIL: salar.d.ghashghaei@gmail.com                    #
#         ##########################################################################################
```

In [47]:

```
# -------------------
#   DYNAMIC ANALYSIS
# -------------------
def DYNAMIC_ANALYSIS(damping, damping_ratio, LINEAR, periodTF, duration, dt, TOTAL_MASS, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, MAX_ITERATIONS,
    import openseespy.opensees as ops
    import numpy as np
    import matplotlib.pyplot as plt

    #GMfact = 9810              # standard acceleration of gravity or standard acceleration
    GMfact = 1;
    iv0 = 0.0                  # [mm/s] Initial velocity applied to the node
    st_iv0 = 0.0               # [s] Initial velocity applied starting time
    KE = 1000;                 # [N/mm] Lateral column Effective Stiffness
    A_cable_01 = (np.pi * Cable_Dia_01 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Bottom
    A_cable_02 = (np.pi * Cable_Dia_02 **2) / 4  # [mm^2] Horizontal Longitudinal Cable Area Top
    A_cable_03 = (np.pi * Cable_Dia_03 **2) / 4  # [mm^2] Vertical Longitudinal Cable Area Top

    # Define model builder
    ops.wipe()
    ops.model('basic', '-ndm', 2, '-ndf', 2)
    dx = L / (num_nodes - 1)
    MASS = TOTAL_MASS / num_nodes
    # Create nodes
    for i in range(num_nodes):
        X = i * dx
        Y1 = H1 - arc_depth * np.sin(np.pi * X / L)
        ops.node(i + 1, X, Y1)
        ops.node(num_nodes + i + 1, i * dx, 0.0)
        # Define mass
        ops.mass(num_nodes + i + 1, MASS, MASS, 0)

    # Define boundary conditions (fixed at both ends)
    ops.fix(1, 1, 1)             # TOP CABLE
    ops.fix(num_nodes, 1, 1)     # TOP CABLE
    ops.fix(num_nodes + 1, 1, 1)  # BOTTOM CABLE
    ops.fix(2 * num_nodes, 1, 1)  # BOTTOM CABLE

    # Define material properties
    if LINEAR == True:
        ops.uniaxialMaterial('Elastic', 1, E_cable)
        #                                TENSION    COMPRESSION
        #ops.uniaxialMaterial('Elastic', 1, E_cable, 0, 0.5 * E_cable)
    if LINEAR == False:
        Fy_cable = 355   # [N/mm^2] Yield strength of the cable
        b0 = 0.01
        ops.uniaxialMaterial('Steel01', 1, Fy_cable, E_cable, b0)

    # Define truss elements
    for i in range(num_nodes - 1):
        ops.element('corotTruss', i + 1, i + 1, i + 2, A_cable_02, 1)  # Top Cable element
        ops.element('corotTruss', num_nodes + i + 1, num_nodes + i + 1, num_nodes + i + 2, A_cable_01, 1)  # Bottom Cable Element

    # Connect each node of the cable with the corresponding node of the deck using truss elements
    for i in range(num_nodes):
        ops.element('corotTruss', 2 * num_nodes + i + 1, i + 1, num_nodes + i + 1, A_cable_03, 1) # Vertical Cable Element

    #mid_node = num_nodes // 2 + 1
    mid_node = int(num_nodes + 0.5 * num_nodes)

    # Dynamic analysis setup
    ops.constraints('Transformation')
    ops.numberer('RCM')
    ops.system('UmfPack')
    ops.test('EnergyIncr', TOLERANCE, MAX_ITERATIONS)
    #ops.integrator('CentralDifference')
    ops.integrator('HHT', 0.9)
    #ops.integrator('Newmark', 0.5, 0.25)
    ops.algorithm('ModifiedNewton')

    # Define analysis type
    ops.analysis('Transient')

    # Define time series for input motion (Acceleration time history)
    ops.timeSeries('Path', 1, '-dt', 0.01, '-filePath', 'OPENSEES_SPRING_SEISMIC_01.txt', '-factor', GMfact, '-startTime', st_iv0) # SEISMIC-X
    ops.timeSeries('Path', 2, '-dt', 0.01, '-filePath', 'OPENSEES_SPRING_SEISMIC_02.txt', '-factor', GMfact) # SEISMIC-Y

    # Define load patterns
    # pattern UniformExcitation $patternTag $dof -accel $tsTag <-vel0 $vel0> <-fact $cFact>
    ops.pattern('UniformExcitation', 1, 1, '-accel', 1, '-vel0', iv0, '-fact', 1.0) # SEISMIC-X
    ops.pattern('UniformExcitation', 2, 2, '-accel', 2)                             # SEISMIC-Y

    # Perform eigenvalue analysis to determine modal periods
    if periodTF == True:
        eigenvalues01 = ops.eigen('-genBandArpack', 1) # eigenvalue mode 1
        #eigenvalues01 = ops.eigen('-fullGenLapack', 1) # eigenvalue mode 1
        #eigenvalues01 = ops.eigen('-symmBandLapack', 1) # eigenvalue mode 1
        #Omega = np.power(eigenvalues01, 0.5)
        #Omega = np.power(max(min(eigenvalues01), min(eigenvalues02), min(eigenvalues03)), 0.5)
        Omega = np.power(min(eigenvalues01), 0.5)
        modal_period = 2 * np.pi / np.sqrt(Omega)  # [Second]
        frequency = 1 / modal_period               # [Hertz]

    if periodTF == False:
        modal_period = 0.0            # [Second]
        frequency = 0.0               # [Hertz]

    if damping == True:
        # Calculate Rayleigh damping factors
        omega1 = np.sqrt(KE / TOTAL_MASS)
        omega2 = 2 * omega1  # Just an assumption for two modes
        a0 = damping_ratio * (2 * omega1 * omega2) / (omega1 + omega2)
        a1 = damping_ratio * 2 / (omega1 + omega2)
        # Apply Rayleigh damping
        ops.rayleigh(a0, a1, 0, 0)

    # OUTPUT DATA
```

```
    ops.recorder('Node', '-file', f"{SALAR_DIR}DTH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2, 'disp')# Displacement Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}VTH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2,3, 'vel')  # Velocity Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}ATH_DYN.txt",'-time', '-node', mid_node, '-dof', 1,2,3, 'accel') # Acceleration Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_01.txt",'-time', '-node', 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_02.txt",'-time', '-node', num_nodes, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_03.txt",'-time', '-node', num_nodes + 1, '-dof', 1,2, 'reaction')# Base Shear Time History
    ops.recorder('Node', '-file', f"{SALAR_DIR}BTH_DYN_04.txt",'-time', '-node', 2*num_nodes, '-dof', 1,2, 'reaction')# Base Shear

    DISPLACEMENTS = []
    DISP_X , DISP_Y, VELOCITY, ACCELERATION = [], [], [], []
    BASE_REACTION_X, BASE_REACTION_Y = [], []

    stable = 0
    current_time = 0.0
     # Perform the analysis with increments
    while stable == 0 and current_time < duration:
        OK = ops.analyze(1, dt)
        ANALYSIS(OK, 1, TOLERANCE, MAX_ITERATIONS)
        current_time = ops.getTime()
        DISPLACEMENTS.append([ops.nodeDisp(j + 1) for j in range(num_nodes * 2)])
        DISP_X.append(ops.nodeDisp(mid_node, 1))
        DISP_Y.append(ops.nodeDisp(mid_node, 2))
        VELOCITY.append(ops.nodeVel(mid_node, 2))
        ACCELERATION.append(ops.nodeAccel(mid_node, 2))
        x1 = ops.nodeResponse(1, 1, 6)
        x2 = ops.nodeResponse(num_nodes, 1, 6)
        x3 = ops.nodeResponse(num_nodes + 1, 1, 6)
        x4 = ops.nodeResponse(2*num_nodes, 1, 6)
        y1 = ops.nodeResponse(1, 2, 6)
        y2 = ops.nodeResponse(num_nodes, 2, 6)
        y3 = ops.nodeResponse(num_nodes + 1, 2, 6)
        y4 = ops.nodeResponse(2*num_nodes, 2, 6)
        BASE_REACTION_X.append(x1+x2+x3+x4) # CABLE REACION-X
        BASE_REACTION_Y.append(y1+y2+y3+y4) # CABLE REACION-Y
        KE =  BASE_REACTION_Y[-1] / DISP_Y[-1] # Effective Lateral Stiffness
        #print(f'STEP: {i+1}')

    # Get initial and final node coordinates for plotting
    initial_coords = np.array([ops.nodeCoord(i + 1) for i in range(num_nodes * 2)])
    deformed_coords = initial_coords + np.array(DISPLACEMENTS[-1])

    # Output all unconstrained node displacements
    for i in range(num_nodes * 2):
        disp = ops.nodeDisp(i + 1)
        print(f"Node {i + 1}: X Displacement = {disp[0]}, Y Displacement = {disp[1]}")

    ops.wipe()
    print("Period: ", modal_period)
    print("Frequency: ", frequency)
    if damping == False:
        print(f' Undamping Structure Dynamic Analysis Done.')
    if damping == True:
        print(f' Damping Structure Dynamic Analysis Done.')

    return initial_coords, deformed_coords, DISPLACEMENTS, VELOCITY, ACCELERATION, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y
```

In [48]:

```
def plot_time_history(DISP_X_undamped, DISP_X_damped, DISP_Y_undamped, DISP_Y_damped,
                      VELOCITY_undamped, VELOCITY_damped, ACCELERATION_undamped, ACCELERATION_damped,
                      BASE_REACTION_X_undamped, BASE_REACTION_X_damped, BASE_REACTION_Y_undamped, BASE_REACTION_Y_damped):
    import matplotlib.pyplot as plt
    import numpy as np
    # Assuming you have 'time' array representing the time steps
    time = [i * dt for i in range(len(DISPLACEMENTS_undamped))]
    plt.figure(figsize=(14, 20))

    # Plot Displacements in X direction
    plt.subplot(6, 1, 1)
    P1 = DISP_X_undamped
    P2 = DISP_X_damped
    P11 = np.max(np.abs(P1))
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped X: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped X: {P22:.5e}')
    plt.title(f'Displacement Time History (X direction) - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Displacement [m] (X direction)')
    plt.legend()

    # Plot Displacements in Y direction
    plt.subplot(6, 1, 2)
    P1 = DISP_Y_undamped
    P2 = DISP_Y_damped
    P11 = np.max(np.abs(P1))
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped Y: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped Y: {P22:.5e}')
    plt.title(f'Displacement Time History (Y direction) - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Displacement [m] (Y direction)')
    plt.legend()

    # Plot Velocities
    plt.subplot(6, 1, 3)
    P1 = VELOCITY_undamped
    P2 = VELOCITY_damped
    P11 = np.max(np.abs(P1))
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped: {P22:.5e}')
    plt.title(f'Velocity Time History - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Velocity [m/s] (Y direction)')
    plt.legend()

    # Plot Accelerations
    plt.subplot(6, 1, 4)
    P1 = ACCELERATION_undamped
    P2 = ACCELERATION_damped
    P11 = np.max(np.abs(P1))
```

```
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped: {P22:.5e}')
    plt.title(f'Acceleration Time History - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Acceleration [m/s²] (Y direction)')
    plt.legend()

    # Plot Base Reactions in X direction
    plt.subplot(6, 1, 5)
    P1 = BASE_REACTION_X_undamped
    P2 = BASE_REACTION_X_damped
    P11 = np.max(np.abs(P1))
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped: {P22:.5e}')
    plt.title(f'Base Reaction Time History (X direction) - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Base Reaction [N] (X direction)')
    plt.legend()

    # Plot Base Reactions in Y direction
    plt.subplot(6, 1, 6)
    P1 = BASE_REACTION_Y_undamped
    P2 = BASE_REACTION_Y_damped
    P11 = np.max(np.abs(P1))
    P22 = np.max(np.abs(P2))
    P3 = P11 / P22
    plt.plot(time, P1, color='black', label=f'Undamped: {P11:.5e}')
    plt.plot(time, P2, color='red', label=f'Damped: {P22:.5e}')
    plt.title(f'Base Reaction Time History (Y direction) - Amplification Factor: {P3:.3f}')
    plt.xlabel('Time [s]')
    plt.ylabel('Base Reaction [N] (Y direction)')
    plt.legend()

    plt.tight_layout()
    plt.show()
```

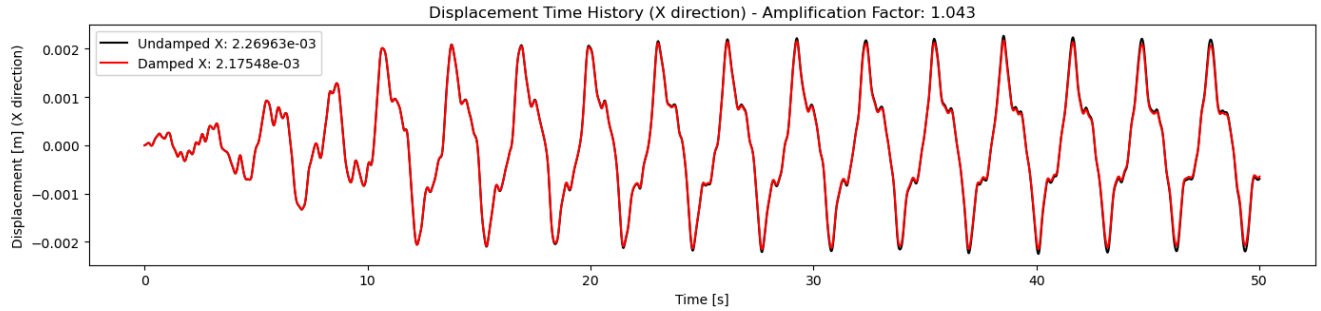
In [49]:

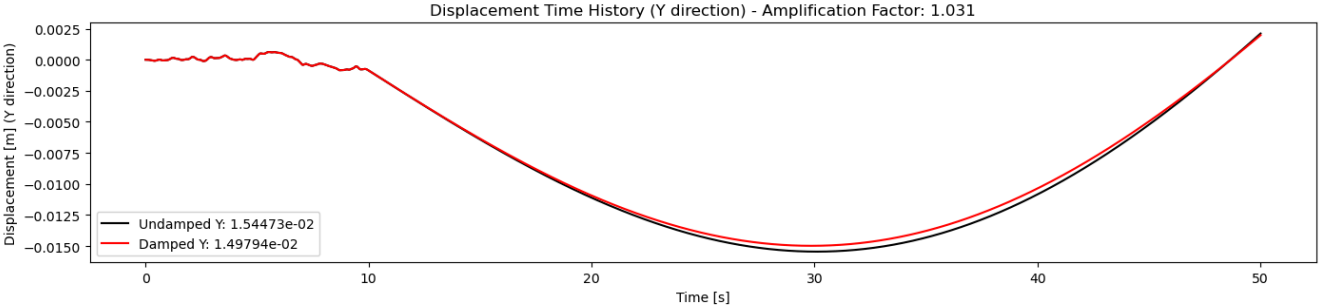
```
# -------------------
#   DYNAMIC ANALYSIS
# -------------------
# Parameters for the analysis
L = 50000.0         # [mm] Bridge span lentgh
H1 = 3500.0         # [mm] Height of Top Cable
arc_depth = 1000.0  # [mm]
E_cable = 210e5     # [N/mm^2] Modulus of elasticity Cable
Cable_Dia_01 = 25   # [mm] Horizontal Longitudinal Cable Diameter Bottom
Cable_Dia_02 = 18   # [mm] Horizontal Longitudinal Cable Diameter Top
Cable_Dia_03 = 10   # [mm] Vertical Longitudinal Cable Diameter Top
num_nodes = 51      # Cable Arc Number of nodes

TOTAL_MASS = 500000.0  # [kg] Total Mass of Structure
damping_ratio = 0.05   # Damping ratio
duration = 50.0        # [s] Duration of the analysis in seconds
dt = 0.01              # Time step in seconds

MAX_ITERATIONS = 10000  # Maximum number of iterations
TOLERANCE = 1.0e-14     # Tolerance for convergence

import time
starttime = time.process_time()

# Run the undamped analysis
damping = False
LINEAR = True # False: Cable NonLinear Materials Properties
periodTF = False
results_undamped = DYNAMIC_ANALYSIS(damping, damping_ratio, LINEAR, periodTF, duration, dt, TOTAL_MASS, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes,
initial_coords, deformed_coords, DISPLACEMENTS_undamped, VELOCITY_undamped, ACCELERATION_undamped, DISP_X_undamped, DISP_Y_undamped, BASE_REACTION_X_undamped, BASE_REACTION_Y_undamped

# Run the damped analysis
damping = True
LINEAR = True # False: Cable NonLinear Materials Properties
periodTF = False
results_damped = DYNAMIC_ANALYSIS(damping, damping_ratio, LINEAR, periodTF, duration, dt, TOTAL_MASS, L, H1, arc_depth, E_cable, Cable_Dia_01, Cable_Dia_02, Cable_Dia_03, num_nodes, MA
initial_coords, deformed_coords, DISPLACEMENTS_damped, VELOCITY_damped, ACCELERATION_damped, DISP_X_damped, DISP_Y_damped, BASE_REACTION_X_damped, BASE_REACTION_Y_damped = results_damp

totaltime = time.process_time() - starttime
print(f'\nTotal time (s): {totaltime:.4f} \n\n')
```

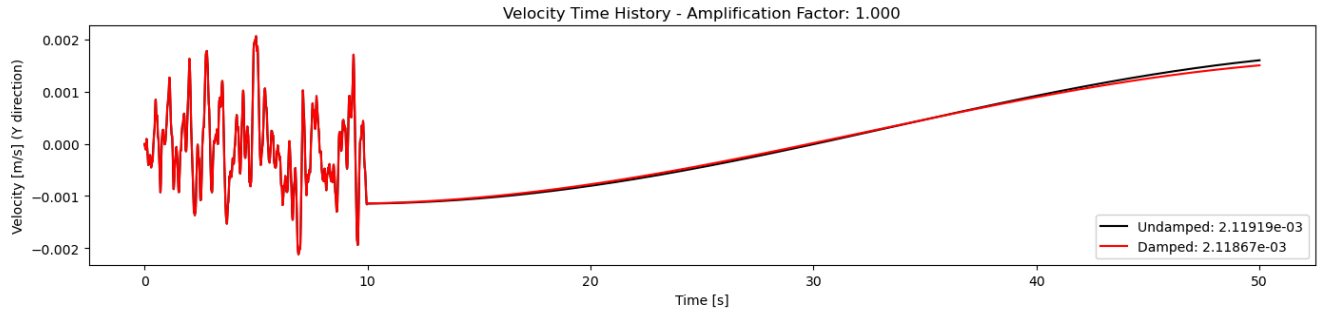
Node 1: X Displacement = 0.0, Y Displacement = 0.0
Node 2: X Displacement = -0.0027149136862820965, Y Displacement = -0.0435567618976172 65
Node 3: X Displacement = -0.0025178038686 10918, Y Displacement = -0.04074069392452289
Node 4: X Displacement = -0.0023078010755906183, Y Displacement = -0.03769434475642839
Node 5: X Displacement = -0.0021022577713170247, Y Displacement = -0.0346839438557570 16
Node 6: X Displacement = -0.0019025800731 2154, Y Displacement = -0.03172173693899921
Node 7: X Displacement = -0.0017100589546797569, Y Displacement = -0.028819414637991293
Node 8: X Displacement = -0.0015258725111800653, Y Displacement = -0.025988431017593523
Node 9: X Displacement = -0.0013510673945118 63, Y Displacement = -0.023239958541443787
Node 10: X Displacement = -0.0011865423010942 54, Y Displacement = -0.02058484398233741
Node 11: X Displacement = -0.0010330338083749206, Y Displacement = -0.0180335656565965 9602
Node 12: X Displacement = -0.0008911047567153665, Y Displacement = -0.0155961921360386 56
Node 13: X Displacement = -0.0007611353728797475, Y Displacement = -0.013282342411244052
Node 14: X Displacement = -0.0006433172743087112, Y Displacement = -0.01110114802549537
Node 15: X Displacement = -0.0005376504469161407, Y Displacement = -0.009061216983481717
Node 16: X Displacement = -0.0004439432453345 0236, Y Displacement = -0.007170599788990378
Node 17: X Displacement = -0.0003618154151548279, Y Displacement = -0.0054367576631647796
Node 18: X Displacement = -0.0002907040898931 3273, Y Displacement = -0.0038665331026337168
Node 19: X Displacement = -0.0002298726678750 8908, Y Displacement = -0.002466122873132193
Node 20: X Displacement = -0.0001784224293464 7, Y Displacement = -0.0012410535613773465
Node 21: X Displacement = -0.0001353067108874096, Y Displacement = -0.0001961597891846 7798
Node 22: X Displacement = -9.934741091696255e-05, Y Displacement = 0.0006644348445130573
Node 23: X Displacement = -6.925356940710637e-05, Y Displacement = 0.001337334043009977
Node 24: X Displacement = -4.364172567769229e-05, Y Displacement = 0.0018198822407185639
Node 25: X Displacement = -2.1057735753783584e-05, Y Displacement = 0.002110175095884294
Node 26: X Displacement = 2.94380322819366e-10, Y Displacement = 0.002207066992857453
Node 27: X Displacement = 2.1058323641873623e-05, Y Displacement = 0.0021101755513325705
Node 28: X Displacement = 4.3642311016199895e-05, Y Displacement = 0.0018198831359133487
Node 29: X Displacement = 6.925415081969192e-05, Y Displacement = 0.001337335362573402
Node 30: X Displacement = 9.934798687644573e-05, Y Displacement = 0.0006644365660649108
Node 31: X Displacement = 0.0001353072809927246, Y Displacement = -0.0001961577222 4865642
Node 32: X Displacement = 0.0001784229945714 2814, Y Displacement = -0.0012410512531490384
Node 33: X Displacement = 0.0002298732302743 8874, Y Displacement = -0.0024661204395333134
Node 34: X Displacement = 0.0002907046511523 3667, Y Displacement = -0.0038665306143188774
Node 35: X Displacement = 0.0003618159750354 6467, Y Displacement = -0.005436755122124962
Node 36: X Displacement = 0.0004439438020535313, Y Displacement = -0.007170597146398329
Node 37: X Displacement = 0.0005376509975136179, Y Displacement = -0.009061214176103454
Node 38: X Displacement = 0.0006433178163030123, Y Displacement = -0.011101145000879761
Node 39: X Displacement = 0.0007611359049838775, Y Displacement = -0.013282339156164441
Node 40: X Displacement = 0.0008911052798726417, Y Displacement = -0.015596188681085067
Node 41: X Displacement = 0.0010330343255030613, Y Displacement = -0.018033562081321914
Node 42: X Displacement = 0.0011865428165084733, Y Displacement = -0.020584840358167617
Node 43: X Displacement = 0.0013510679130990279, Y Displacement = -0.0232399549668865
Node 44: X Displacement = 0.0015258730362813 9, Y Displacement = -0.025988427538239194
Node 45: X Displacement = 0.0017100594845515 24, Y Displacement = -0.028819411243644642
Node 46: X Displacement = 0.0019025806093422896, Y Displacement = -0.03172173364237419
Node 47: X Displacement = 0.0021022583208094325, Y Displacement = -0.034683940770847285
Node 48: X Displacement = 0.0023078016531050305, Y Displacement = -0.037694342118875 99
Node 49: X Displacement = 0.0025178052250249 57, Y Displacement = -0.040740703830169374
Node 50: X Displacement = 0.0027150712166049812, Y Displacement = -0.043559268985599364
Node 51: X Displacement = 0.0, Y Displacement = 0.0
Node 52: X Displacement = 0.0, Y Displacement = 0.0
Node 53: X Displacement = -0.0001940723461311307, Y Displacement = -0.04355682937789629
Node 54: X Displacement = -0.0003532254538407 0214, Y Displacement = -0.040740806136131 65
Node 55: X Displacement = -0.0004602120583865 7085, Y Displacement = -0.037694509853533 97
Node 56: X Displacement = -0.0005148654723768144, Y Displacement = -0.034684159198332405
Node 57: X Displacement = -0.0005300225173402687, Y Displacement = -0.03172199973178854
Node 58: X Displacement = -0.0005281932649093066, Y Displacement = -0.028819722019852 88
Node 59: X Displacement = -0.0005312300747954734, Y Displacement = -0.025988780088674 08
Node 60: X Displacement = -0.0005438302028678317, Y Displacement = -0.023240346381786507
Node 61: X Displacement = -0.0005545718896268799, Y Displacement = -0.020585267677346045
Node 62: X Displacement = -0.0005528216593636215, Y Displacement = -0.018034022332272 62
Node 63: X Displacement = -0.0005359569473013373, Y Displacement = -0.015596678946843776
Node 64: X Displacement = -0.0005061714423206904, Y Displacement = -0.013282856616581592
Node 65: X Displacement = -0.0004698101059146344, Y Displacement = -0.011101686965037376
Node 66: X Displacement = -0.0004375345379444 0425, Y Displacement = -0.009061778101441176
Node 67: X Displacement = -0.0004204981500914 95, Y Displacement = -0.007171180637701807
Node 68: X Displacement = -0.0004293226560952 2475, Y Displacement = -0.005437355909065973
Node 69: X Displacement = -0.0004670377802642559, Y Displacement = -0.0038671465252536836
Node 70: X Displacement = -0.0005283738993768 87, Y Displacement = -0.0024667493637266823
Node 71: X Displacement = -0.0005949140103311411, Y Displacement = -0.0012416911213205312
Node 72: X Displacement = -0.0006447337857305187, Y Displacement = -0.0001968065252596669 8
Node 73: X Displacement = -0.0006686206346432638, Y Displacement = 0.0006637807314895457
Node 74: X Displacement = -0.0006762045438035985, Y Displacement = 0.001336674275233328
Node 75: X Displacement = -0.0006833363447603132, Y Displacement = 0.0018192184811626018
Node 76: X Displacement = -0.0006942676248390545, Y Displacement = 0.002109508963307 56
Node 77: X Displacement = -0.0007000297267974719, Y Displacement = 0.0022064000771452 22
Node 78: X Displacement = -0.0006941875662797712, Y Displacement = 0.0021095094303728978
Node 79: X Displacement = -0.0006831763101202605, Y Displacement = 0.0018192194000125038
Node 80: X Displacement = -0.0006759646823408 92, Y Displacement = 0.0013366756313165262
Node 81: X Displacement = -0.0006683011488860102, Y Displacement = 0.0006637825053253928
Node 82: X Displacement = -0.0006443349283218977, Y Displacement = -0.0001968043900707211 6
Node 83: X Displacement = -0.0005944360850894 3, Y Displacement = -0.001241688731158735
Node 84: X Displacement = -0.0005278172626361464, Y Displacement = -0.0024667468370994576
Node 85: X Displacement = -0.0004664028452766556, Y Displacement = -0.0038671439338608922
Node 86: X Displacement = -0.0004286099012797 6104, Y Displacement = -0.005437353251224657
Node 87: X Displacement = -0.0004201581208241161, Y Displacement = -0.007171177857113065
Node 88: X Displacement = -0.0004366678223657 85, Y Displacement = -0.009061775122254544
Node 89: X Displacement = -0.0004688673090943 2184, Y Displacement = -0.011101683723477022
Node 90: X Displacement = -0.0004505153176457 1674, Y Displacement = -0.013282853088076552
Node 91: X Displacement = -0.0004534863822099768, Y Displacement = -0.015596675158955072
Node 92: X Displacement = -0.0004551654312 3403237, Y Displacement = -0.018034018355884772
Node 93: X Displacement = -0.0004553330908 1952005, Y Displacement = -0.020585263610173 88
Node 94: X Displacement = -0.0004542516136 6228113, Y Displacement = -0.023240342321557052
Node 95: X Displacement = -0.0004529843409 5821718, Y Displacement = -0.025988776082765702
Node 96: X Displacement = -0.0004526734416 1857082, Y Displacement = -0.028819718050549024
Node 97: X Displacement = -0.0004528491832 4151104, Y Displacement = -0.03172199580454852
Node 98: X Displacement = -0.0004513263239 4332898, Y Displacement = -0.034684155449197056
Node 99: X Displacement = -0.0004585385253049 0497, Y Displacement = -0.037694506576390875
Node 100: X Displacement = -0.0003514808388 6170784, Y Displacement = -0.040740815515426916
Node 101: X Displacement = -0.0001922553391 9643305, Y Displacement = -0.043559336162014 04
Node 102: X Displacement = 0.0, Y Displacement = 0.0
Period: 0.0
Frequency: 0.0
 Undamping Structure Dynamic Analysis Done.
Node 1: X Displacement = 0.0, Y Displacement = 0.0
Node 2: X Displacement = -0.0025569140376183546, Y Displacement = -0.041022975892697475
Node 3: X Displacement = -0.0023708530619 48677, Y Displacement = -0.03836429918416687
Node 4: X Displacement = -0.0021731277131 50373, Y Displacement = -0.03549640013525909
Node 5: X Displacement = -0.0019796019457837197, Y Displacement = -0.032662364981026315
Node 6: X Displacement = -0.0017915981593881864, Y Displacement = -0.029873700369789844
Node 7: X Displacement = -0.0016103316963372 55, Y Displacement = -0.027141412013271242

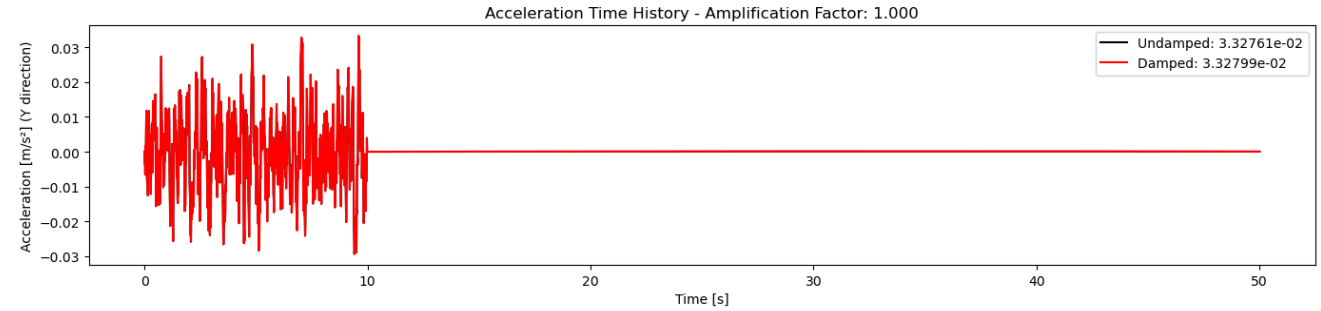
```
Node 8: X Displacement = -0.001436911635693057, Y Displacement = -0.02447628293043563
Node 9: X Displacement = -0.0012723233106233984, Y Displacement = -0.02188883103683384
Node 10: X Displacement = -0.0011174127711962413, Y Displacement = -0.019389267646462757
Node 11: X Displacement = -0.0009728734430402791, Y Displacement = -0.016987457204534123
Node 12: X Displacement = -0.00083923519312216, Y Displacement = -0.014692878389439164
Node 13: X Displacement = -0.0007168559732250078, Y Displacement = -0.012514586681515266
Node 14: X Displacement = -0.0006059161754545716, Y Displacement = -0.010461178636891043
Node 15: X Displacement = -0.0005064157862075425, Y Displacement = -0.008540757957718706
Node 16: X Displacement = -0.0004181743872359723 4, Y Displacement = -0.006760903501870243
Node 17: X Displacement = -0.0003408340007512906 7, Y Displacement = -0.005128639370167705
Node 18: X Displacement = -0.0002738647347316124 3, Y Displacement = -0.003650407186561504 6
Node 19: X Displacement = -0.0002165731401600472 7, Y Displacement = -0.002332040679440304 4
Node 20: X Displacement = -0.0001681131477385179, Y Displacement = -0.001178742659910787
Node 21: X Displacement = -0.0001274994107745303, Y Displacement = -0.000195064511702434 22
Node 22: X Displacement = -9.362284508219386e-05, Y Displacement = 0.000615111749732055
Node 23: X Displacement = -6.52681191747468e-05, Y Displacement = 0.001248588803275402 7
Node 24: X Displacement = -4.1132820226334286e-05, Y Displacement = 0.001702866663427162
Node 25: X Displacement = -1.9847993004395675e-05, Y Displacement = 0.001976152553737724
Node 26: X Displacement = 2.702035099005731e-10, Y Displacement = 0.002067367977039868
Node 27: X Displacement = 1.9848532689375934e-05, Y Displacement = 0.00197615295355034
Node 28: X Displacement = 4.1133357535386694e-05, Y Displacement = 0.001702867449942442 3
Node 29: X Displacement = 6.526865297277432e-05, Y Displacement = 0.001248589962738949 8
Node 30: X Displacement = 9.362337416068156e-05, Y Displacement = 0.000615113262124040 3
Node 31: X Displacement = 0.0001274999346836967 5, Y Displacement = -0.000195062696247706 27
Node 32: X Displacement = 0.0001681136673638203 3, Y Displacement = -0.001178740633718306 7
Node 33: X Displacement = 0.0002165736572976020 5, Y Displacement = -0.002332038543135007 7
Node 34: X Displacement = 0.0002738652506268945, Y Displacement = -0.003650405001275991 7
Node 35: X Displacement = 0.0003408345154589 99, Y Displacement = -0.005128637136688779
Node 36: X Displacement = 0.0004181748990895 4955, Y Displacement = -0.006760901178576752
Node 37: X Displacement = 0.0005064162927426626, Y Displacement = -0.008540755489329269
Node 38: X Displacement = 0.0006059166744545991, Y Displacement = -0.010461175976835973
Node 39: X Displacement = 0.0007168564635541571, Y Displacement = -0.012514583814799233
Node 40: X Displacement = 0.0008392356753320268, Y Displacement = -0.014692875345417128
Node 41: X Displacement = 0.0009728739198835688, Y Displacement = -0.016987454045287054
Node 42: X Displacement = 0.0011174132466261054, Y Displacement = -0.019389264449029324
Node 43: X Displacement = 0.0012723237887083702, Y Displacement = -0.02188882788321088
Node 44: X Displacement = 0.0014369121186329 12, Y Displacement = -0.024476279859852363
Node 45: X Displacement = 0.0016103321842471444, Y Displacement = -0.027141409021319254
Node 46: X Displacement = 0.0017915986527885 85, Y Displacement = -0.029873697464928257
Node 47: X Displacement = 0.0019796024508582696, Y Displacement = -0.03266236226098648
Node 48: X Displacement = 0.0021731282429322133, Y Displacement = -0.03549639780943433
Node 49: X Displacement = 0.0023708542832691676, Y Displacement = -0.03836430799713927
Node 50: X Displacement = 0.0025570586090367204, Y Displacement = -0.04102527683065716
Node 51: X Displacement = 0.0, Y Displacement = 0.0
Node 52: X Displacement = 0.0, Y Displacement = 0.0
Node 53: X Displacement = -0.0001819424928785935 3, Y Displacement = -0.041023038956003004
Node 54: X Displacement = -0.0003312178283286201, Y Displacement = -0.038364404984026323
Node 55: X Displacement = -0.0004315820656338785 6, Y Displacement = -0.035496555777753657
Node 56: X Displacement = -0.0004828403576687509, Y Displacement = -0.032662567976111269
Node 57: X Displacement = -0.0004970398333040806, Y Displacement = -0.029873948084985794
Node 58: X Displacement = -0.0004953412554510297, Y Displacement = -0.027141701752421865
Node 59: X Displacement = -0.0004982576800652811, Y Displacement = -0.024476611960365217
Node 60: X Displacement = -0.0005101423142716671, Y Displacement = -0.021889196606676584
Node 61: X Displacement = -0.0005202118254625154, Y Displacement = -0.01938966700776908
Node 62: X Displacement = -0.0005184549635173894, Y Displacement = -0.016987887643564153
Node 63: X Displacement = -0.0005023978084679694, Y Displacement = -0.014693337240848769
Node 64: X Displacement = -0.0004741097561593149, Y Displacement = -0.01251507135341186
Node 65: X Displacement = -0.0004396030345747897, Y Displacement = -0.01046168662285111
Node 66: X Displacement = -0.0004089560372055562, Y Displacement = -0.00854128684786756
Node 67: X Displacement = -0.0003930775633635509 4, Y Displacement = -0.006761450989261663
Node 68: X Displacement = -0.0003400627636620061 6, Y Displacement = -0.005129203254861813
Node 69: X Displacement = -0.0003456807285014761 3, Y Displacement = -0.003650985377188666 4
Node 70: X Displacement = -0.0003492886628329365, Y Displacement = -0.0023326311884335 88
Node 71: X Displacement = -0.0003555017384790007 1, Y Displacement = -0.001179343603440454 5
Node 72: X Displacement = -0.0003601556571447209 2, Y Displacement = -0.000195674105671209 96
Node 73: X Displacement = -0.0003623897687655797 1, Y Displacement = 0.000614495201468982
Node 74: X Displacement = -0.0003631079461810718 3, Y Displacement = 0.001247966924683665
Node 75: X Displacement = -0.0003637927389904601 4, Y Displacement = 0.001702241021351700 3
Node 76: X Displacement = -0.0003648340877801738 6, Y Displacement = 0.001975524674962739 5
Node 77: X Displacement = -0.0003653813885659349, Y Displacement = 0.002066739359469745
Node 78: X Displacement = -0.0003648269870244141 2, Y Displacement = 0.001975525085560078
Node 79: X Displacement = -0.0003637785448202113 5, Y Displacement = 0.001702241828791504 6
Node 80: X Displacement = -0.0003630866718363739 2, Y Displacement = 0.001247968116379750 4
Node 81: X Displacement = -0.0003623614322330143, Y Displacement = 0.000614496760073589 3
Node 82: X Displacement = -0.0003601202807413303 2, Y Displacement = -0.000195672230411402 1
Node 83: X Displacement = -0.0003554593486858638 8, Y Displacement = -0.001179341504916821
Node 84: X Displacement = -0.0003492329264698922 9, Y Displacement = -0.002332628969990926 8
Node 85: X Displacement = -0.0003435117552344116 9, Y Displacement = -0.003650983100866264
Node 86: X Displacement = -0.0003399954278720541 3, Y Displacement = -0.005129200918689402
Node 87: X Displacement = -0.0003923768050041158, Y Displacement = -0.006761448544112257
Node 88: X Displacement = -0.0004081872515214466, Y Displacement = -0.008541284227677505
Node 89: X Displacement = -0.0004387667572802236, Y Displacement = -0.010461683770608335
Node 90: X Displacement = -0.0004732065316129817 7, Y Displacement = -0.012515068245382977
Node 91: X Displacement = -0.0005014281869814478, Y Displacement = -0.014693333902355893
Node 92: X Displacement = -0.0005174194949689991, Y Displacement = -0.016987884138461 89
Node 93: X Displacement = -0.0005191110402934358, Y Displacement = -0.019389663420850 49
Node 94: X Displacement = -0.0005089766993323514, Y Displacement = -0.021889193023734762
Node 95: X Displacement = -0.0004970276653674196, Y Displacement = -0.024476608425267964
Node 96: X Displacement = -0.0004940472086333195, Y Displacement = -0.027141698251925972
Node 97: X Displacement = -0.0004956820580765575, Y Displacement = -0.029873944622439 87
Node 98: X Displacement = -0.0004814191062033677 3, Y Displacement = -0.032662564668748 12
Node 99: X Displacement = -0.0004430097556975048 9, Y Displacement = -0.035496552886740 25
Node 100: X Displacement = -0.0003296702565367218, Y Displacement = -0.038364413331991 71
Node 101: X Displacement = -0.0001803307385195948 7, Y Displacement = -0.041025339625236 56
Node 102: X Displacement = 0.0, Y Displacement = 0.0
Period:  0.0
Frequency:  0.0
 Damping Structure Dynamic Analysis Done.

Total time (s): 10.8594
```

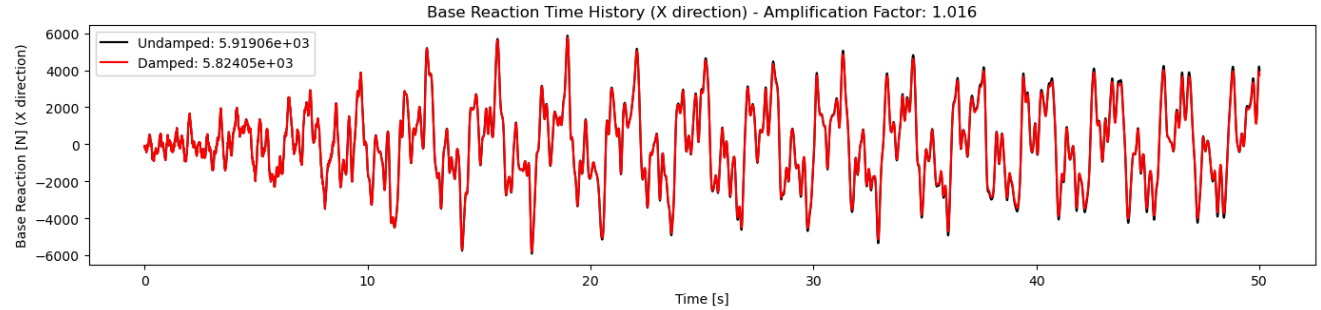
In [50]:
```
plot_time_history(DISP_X_undamped, DISP_X_damped, DISP_Y_undamped, DISP_Y_damped, VELOCITY_undamped, VELOCITY_damped, ACCELERATION_undamped, ACCELERATION_damped, BASE_REACTION_X_undamp
```

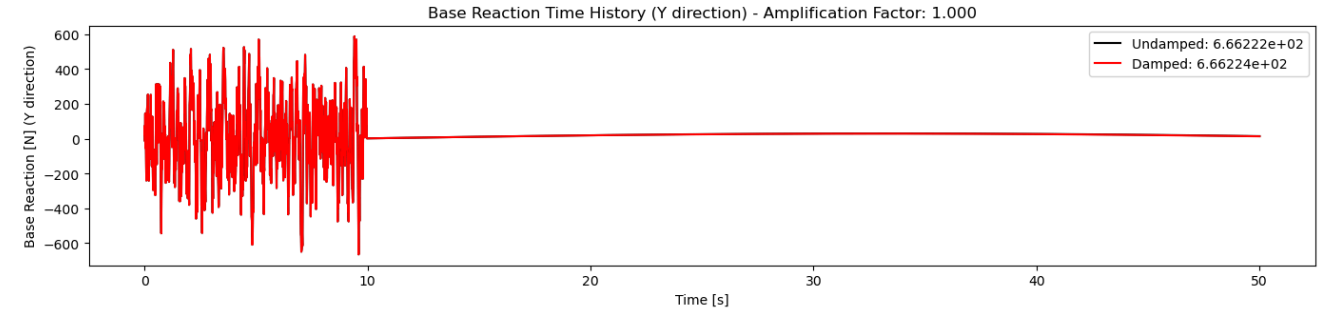

Displacement Time History (X direction) - Amplification Factor: 1.043
Undamped X: 2.26963e-03
Damped X: 2.17548e-03
Displacement [m] (X direction)
Time [s]


Displacement Time History (Y direction) - Amplification Factor: 1.031
Undamped Y: 1.54473e-02
Damped Y: 1.49794e-02
Displacement [m] (Y direction)
Time [s]


Velocity Time History - Amplification Factor: 1.000
Undamped: 2.11919e-03
Damped: 2.11867e-03
Velocity [m/s] (Y direction)
Time [s]

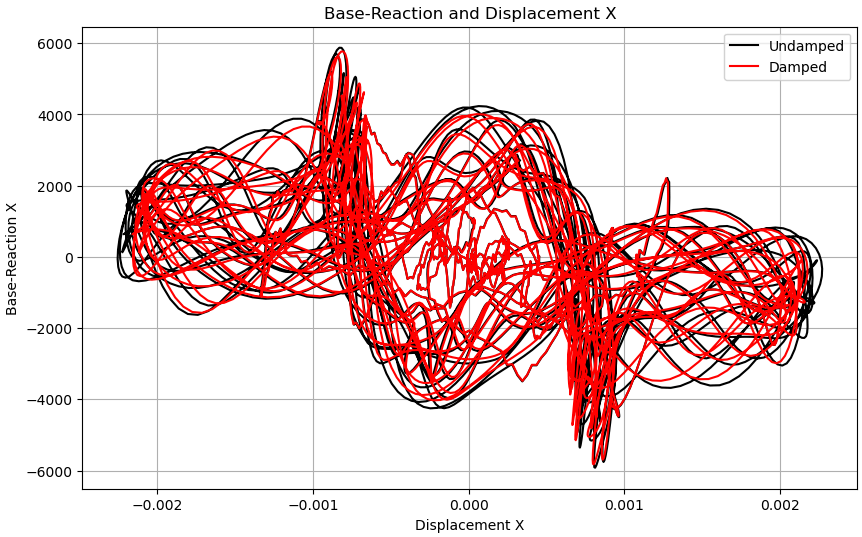

Acceleration Time History - Amplification Factor: 1.000
Undamped: 3.32761e-02
Damped: 3.32799e-02
Acceleration [m/s²] (Y direction)
Time [s]


Base Reaction Time History (X direction) - Amplification Factor: 1.016
Undamped: 5.91906e+03
Damped: 5.82405e+03
Base Reaction [N] (X direction)
Time [s]

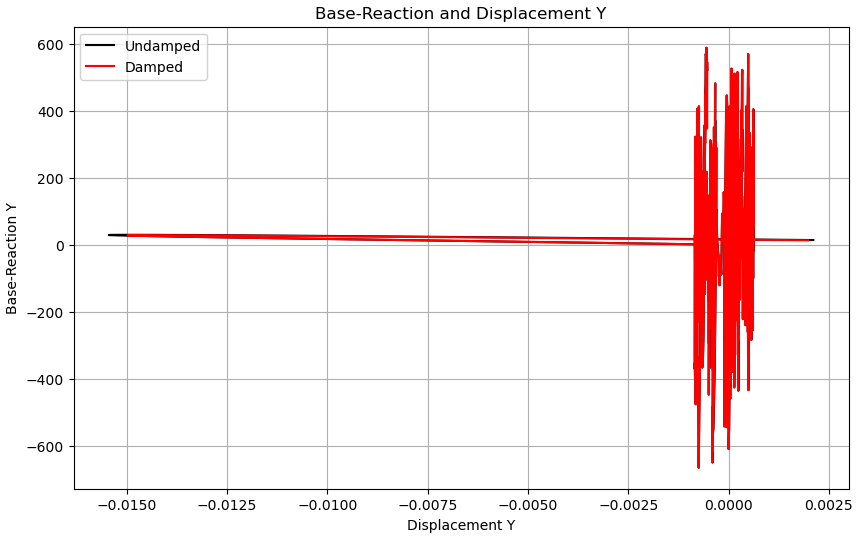

Base Reaction Time History (Y direction) - Amplification Factor: 1.000
Undamped: 6.66222e+02
Damped: 6.66224e+02
Base Reaction [N] (Y direction)
Time [s]

In [51]:
```
def PLOT_2D(X1, Y1, X2, Y2, XLABEL, YLABEL, TITLE):
    plt.figure(figsize=(10, 6))
    plt.plot(X1, Y1, label='Undamped', color='black')
    plt.plot(X2, Y2, label='Damped', color='red')
    plt.xlabel(XLABEL)
    plt.ylabel(YLABEL)
    plt.title(TITLE)
    plt.grid(True)
    #plt.semilogy()
    plt.legend()
    plt.show()
```

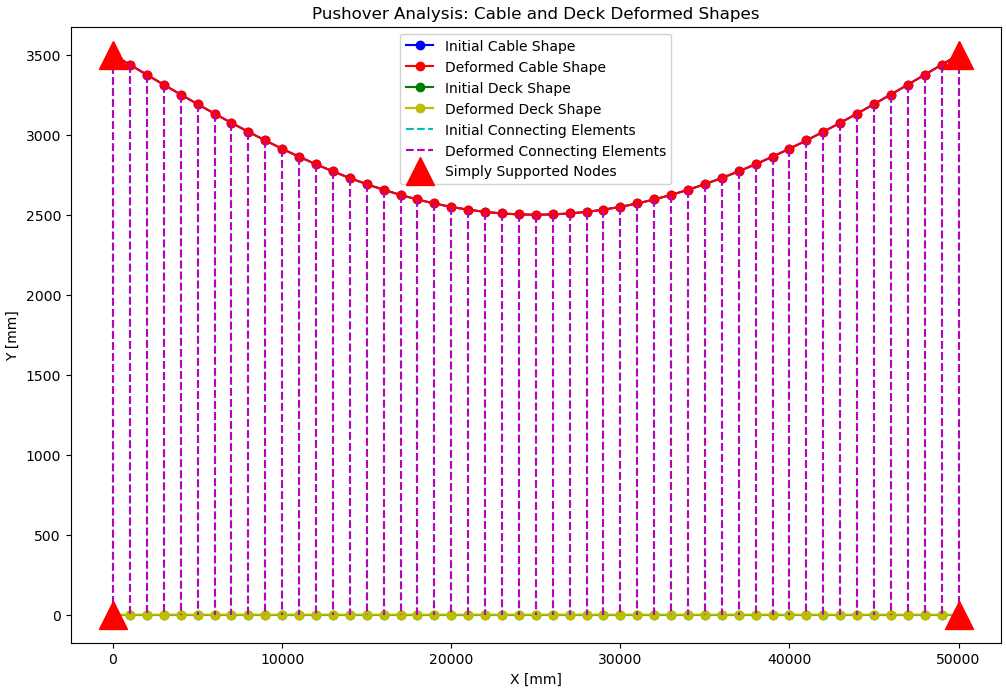
In [53]:
```
### BASE REACTION & DISPALCEMENT IN X:
X1 =  DISP_X_undamped
X2 =  DISP_X_damped
Y1 = BASE_REACTION_X_undamped
Y2 = BASE_REACTION_X_damped
XLABEL = 'Displacement X'
YLABEL = 'Base-Reaction X'
TITLE = 'Base-Reaction and Displacement X'

PLOT_2D(X1, Y1, X2, Y2, XLABEL, YLABEL, TITLE)
```



In [54]:
```
### BASE REACTION & DISPALCEMENT IN Y:
X1 =  DISP_Y_undamped
X2 =  DISP_Y_damped
Y1 = BASE_REACTION_Y_undamped
Y2 = BASE_REACTION_Y_damped
XLABEL = 'Displacement Y'
YLABEL = 'Base-Reaction Y'
TITLE = 'Base-Reaction and Displacement Y'

PLOT_2D(X1, Y1, X2, Y2, XLABEL, YLABEL, TITLE)
```



In [55]:
```
# Plot damped results
plot_shapes(initial_coords, DISPLACEMENTS_damped)
```



Plotting a histogram for the results of dynamic analysis of a structure, such as displacements, accelerations, and support reactions, helps analyze the statistical distribution and trends in the data. The following insights can be drawn from these histograms:

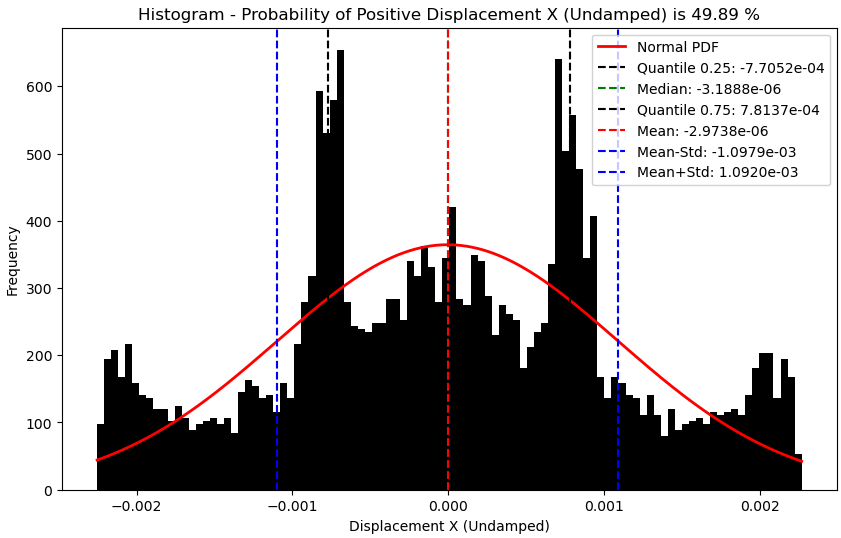
1. Range Identification

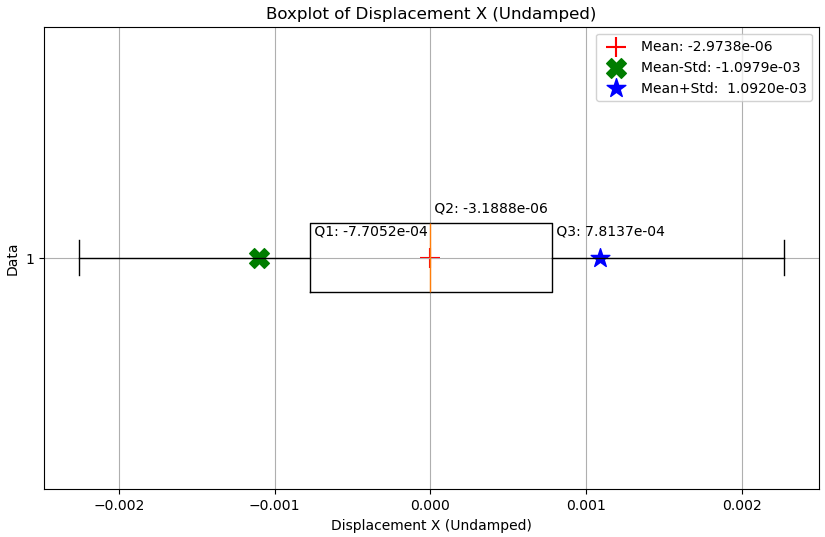
The histogram shows the range within which displacements, accelerations, or support reactions vary. This is crucial for assessing the stability of the structure and determining the allowable design limits. 2. Central Tendency From the histogram, measures like mean, median, or mode can be extracted. This helps identify the most likely values and provides an overview of the dominant behavior of the structure. 3. Data Dispersion The histogram illustrates the spread of the data (e.g., through its width and shape). A wide histogram may indicate nonlinear behavior or unusual effects in the structure. 4. Anomaly Detection Unexpected peaks or asymmetric distributions may indicate abnormal structural behavior (e.g., weaknesses or inelastic behavior). It can also reveal specific phenomena like resonances or amplification effects. 5. Probability of Specific Values The histogram's shape can indicate the probability distribution of the data (e.g., normal, uniform, or Poisson). This is useful for simulations and predicting future structural behavior. 6. Behavior Under Specific Conditions Analyzing histograms of acceleration or displacement can reveal dominant modes of the structure and their impact on overall response. These insights are helpful in designing vibration control systems (e.g., dampers). 7. Impact of External Forces or Loading An unusual histogram shape might suggest the influence of extraordinary dynamic loads (e.g., severe earthquakes or impact forces). Conclusion: Histograms are a simple yet effective tool for analyzing dynamic data. They help engineers and designers:

Understand the overall behavior of a structure. Assess the likelihood of extreme or unexpected responses. Define appropriate design or retrofitting criteria. For deeper analysis, histograms can be combined with other statistical tools, such as standard deviation or spectral analysis.

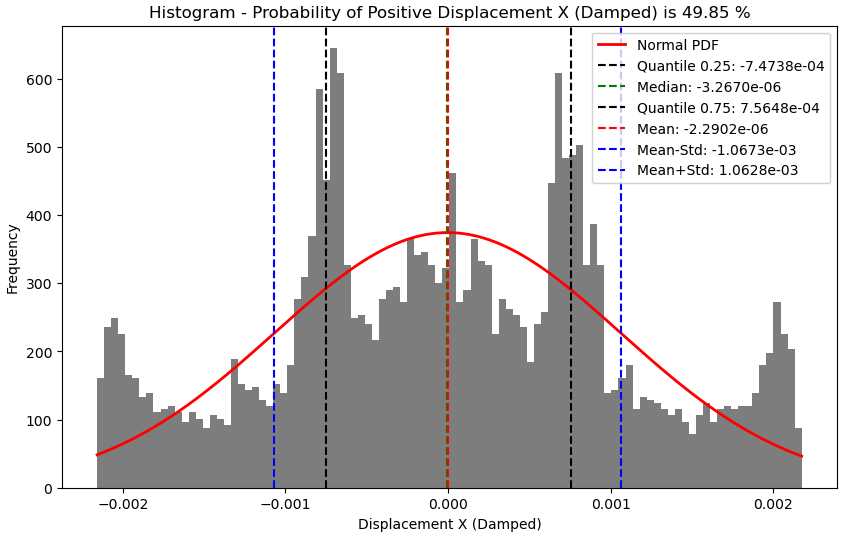
In [56]:
```
HISROGRAM_BOXPLOT(DISP_X_undamped, HISTO_COLOR='black', LABEL='Displacement X (Undamped)')
HISROGRAM_BOXPLOT(DISP_X_damped, HISTO_COLOR='grey', LABEL='Displacement X (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_X_undamped, XLABEL='Displacement X (Undamped)', TITLE='Displacement X (Undamped)', COLOR='black')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_X_damped, XLABEL='Displacement X (Damped)', TITLE='Displacement X (Damped)', COLOR='grey')
```

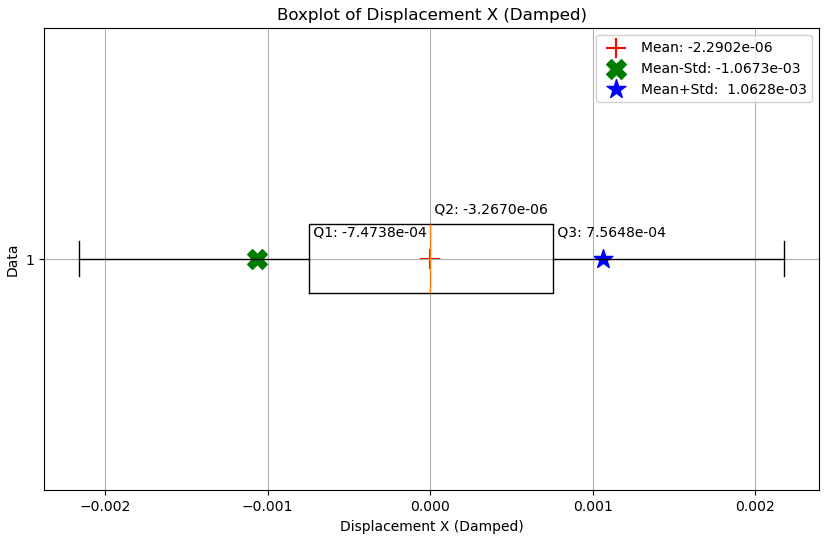
```
-------------------------
Box-Chart Datas:
Minimum: -2.2531e-03
First quartile: -7.7052e-04
Median: -3.1888e-06
Mean: -2.9738e-06
Std: 1.0949e-03
Third quartile: 7.8137e-04
Maximum: 2.2696e-03
Skewness: 4.0488e-03
kurtosis: -6.1986e-01
90% Confidence Interval: (-1.9638e-03, 1.9613e-03)
-------------------------
```






```
-------------------------
Box-Chart Datas:
Minimum: -2.1582e-03
First quartile: -7.4738e-04
Median: -3.2670e-06
Mean: -2.2902e-06
Std: 1.0650e-03
Third quartile: 7.5648e-04
Maximum: 2.1755e-03
Skewness: 6.7665e-03
kurtosis: -6.2866e-01
90% Confidence Interval: (-1.9084e-03, 1.9175e-03)
-------------------------
```

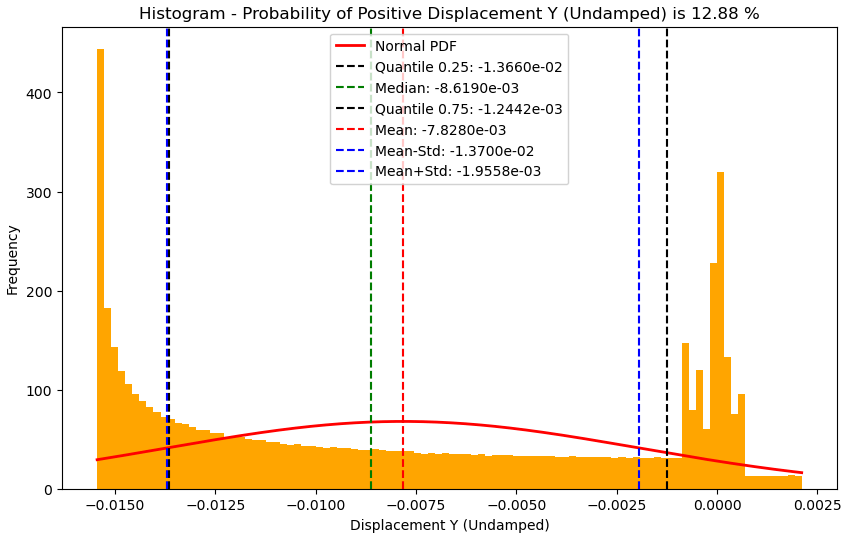


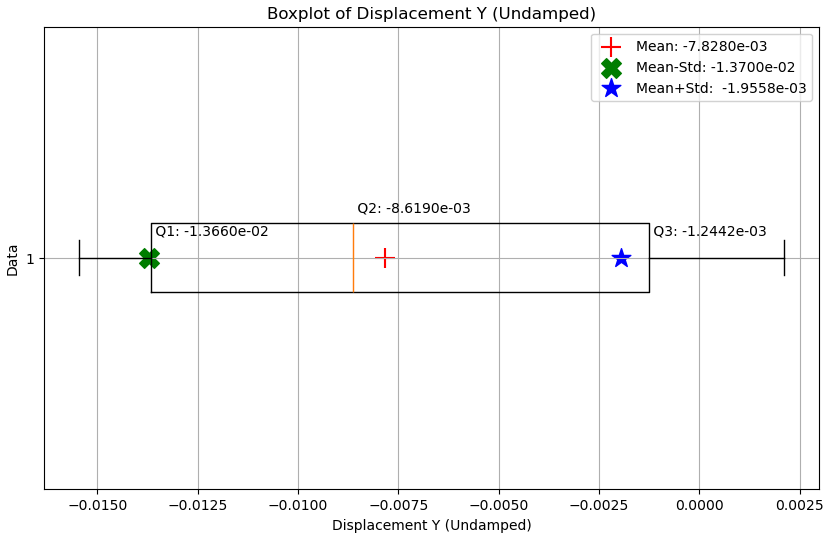


In [57]:
```
HISROGRAM_BOXPLOT(DISP_Y_undamped, HISTO_COLOR='orange', LABEL='Displacement Y (Undamped)')
HISROGRAM_BOXPLOT(DISP_Y_damped, HISTO_COLOR='brown', LABEL='Displacement Y (Damped)')
```

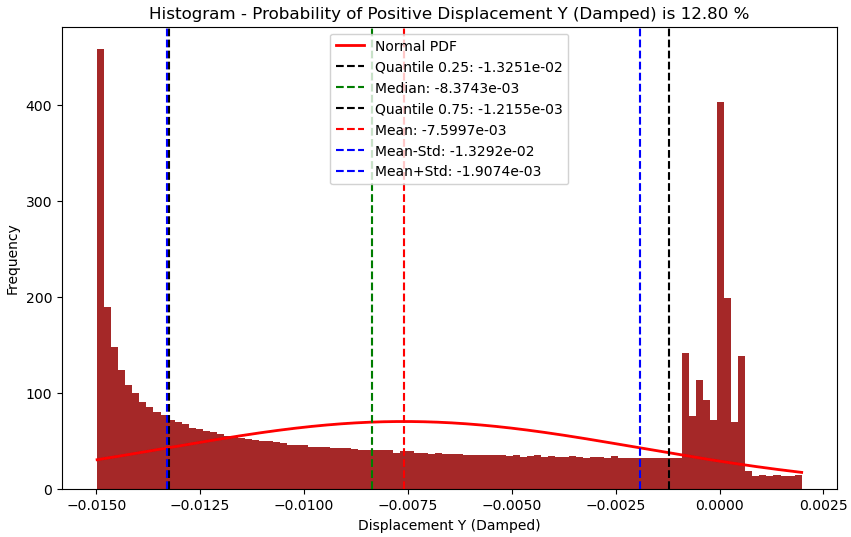
```
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_Y_undamped, XLABEL='Displacement Y (Undamped)', TITLE='Displacement Y (Undamped)', COLOR='orange')
#HISTOGRAM_BOXPLOT_PLOTLY(DISP_Y_damped, XLABEL='Displacement Y (Damped)', TITLE='Displacement Y (Damped)', COLOR='brown')
```

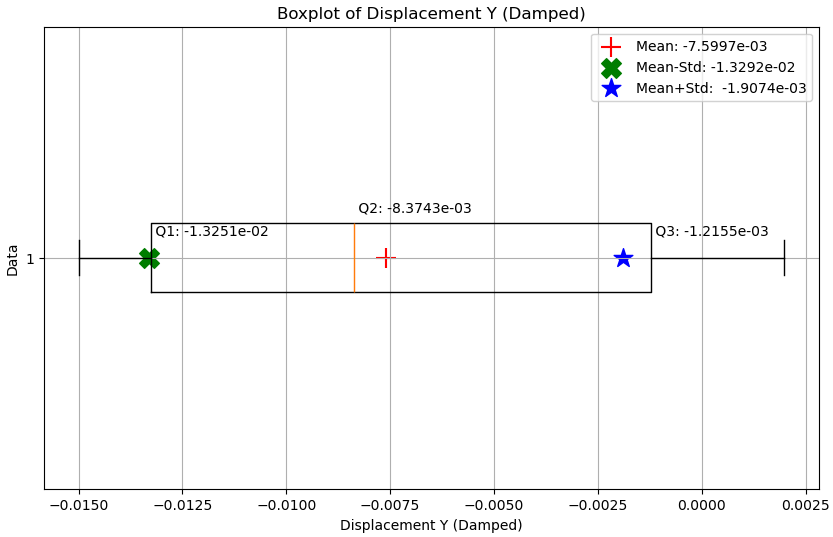
```
-------------------------
Box-Chart Datas:
Minimum: -1.5447e-02
First quartile: -1.3660e-02
Median: -8.6190e-03
Mean: -7.8280e-03
Std: 5.8722e-03
Third quartile: -1.2442e-03
Maximum: 2.1095e-03
Skewness: 1.5596e-01
kurtosis: -1.5264e+00
90% Confidence Interval: (-1.5375e-02, 3.3226e-04)
-------------------------
```





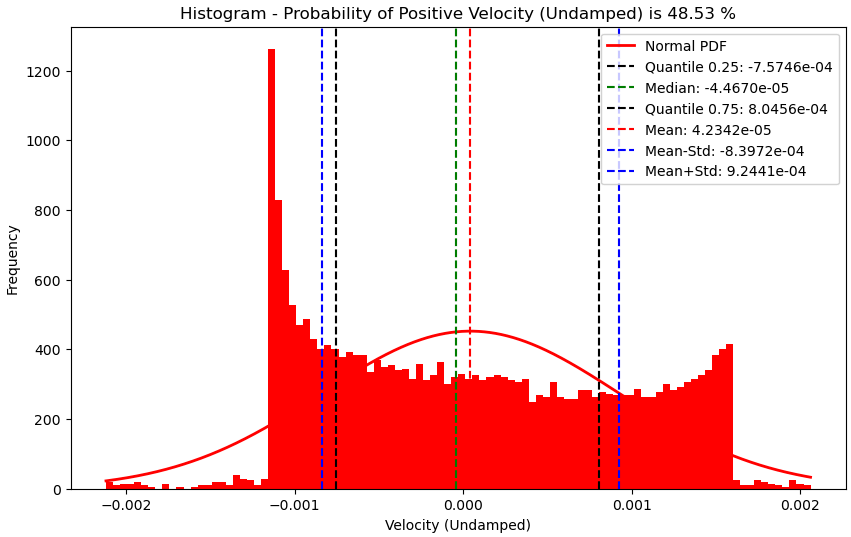

```
-------------------------
Box-Chart Datas:
Minimum: -1.4979e-02
First quartile: -1.3251e-02
Median: -8.3743e-03
Mean: -7.5997e-03
Std: 5.6923e-03
Third quartile: -1.2155e-03
Maximum: 1.9755e-03
Skewness: 1.5760e-01
kurtosis: -1.5264e+00
90% Confidence Interval: (-1.4909e-02, 3.2601e-04)
-------------------------
```

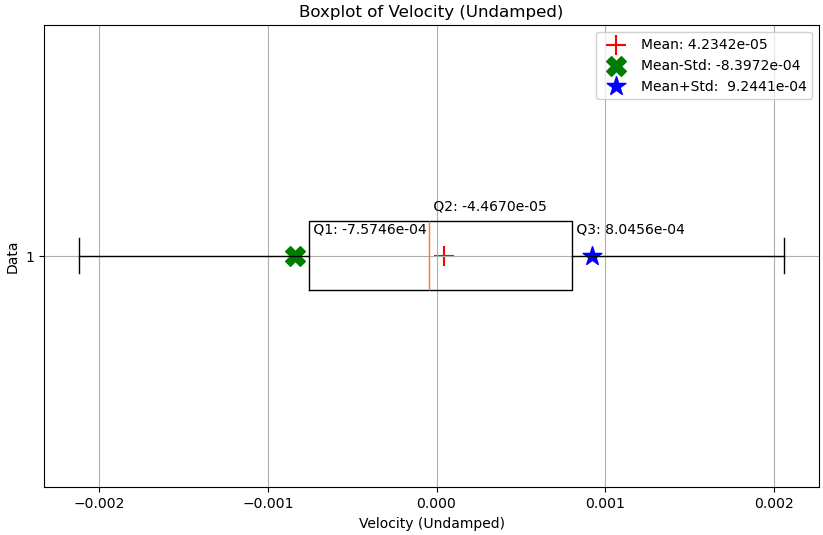




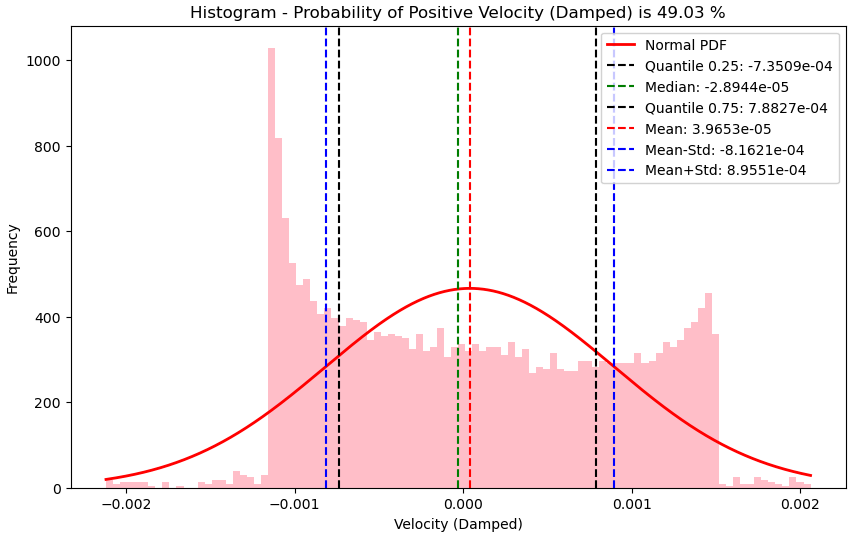
In [58]:
```
HISROGRAM_BOXPLOT(VELOCITY_undamped, HISTO_COLOR='red', LABEL='Velocity (Undamped)')
HISROGRAM_BOXPLOT(VELOCITY_damped, HISTO_COLOR='pink', LABEL='Velocity (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(VELOCITY_undamped, XLABEL='Velocity (Undamped)', TITLE='Velocity (Undamped)', COLOR='red')
#HISTOGRAM_BOXPLOT_PLOTLY(VELOCITY_damped, XLABEL='Velocity (Damped)', TITLE='Velocity (Damped)', COLOR='pink')
```

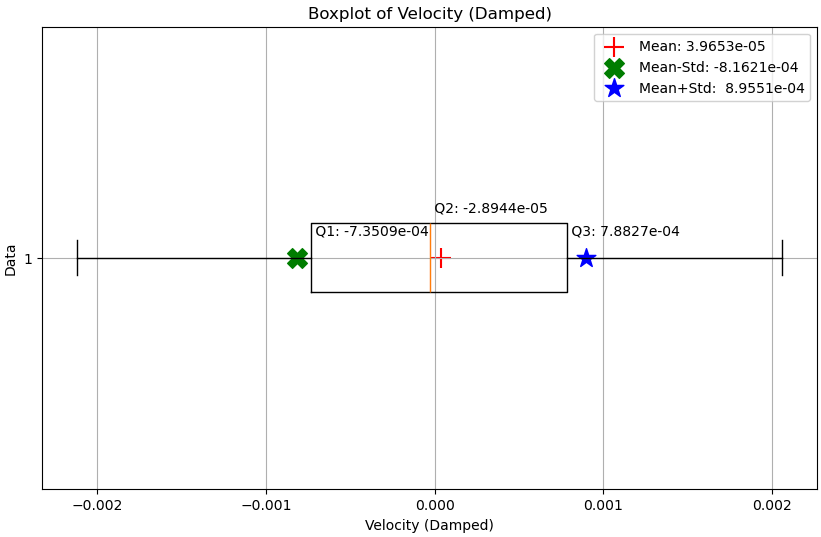
```
-------------------------
Box-Chart Datas:
Minimum: -2.1192e-03
First quartile: -7.5746e-04
Median: -4.4670e-05
Mean: 4.2342e-05
Std: 8.8206e-04
Third quartile: 8.0456e-04
Maximum: 2.0610e-03
Skewness: 2.0954e-01
kurtosis: -1.1562e+00
90% Confidence Interval: (-1.1298e-03, 1.4930e-03)
-------------------------
```





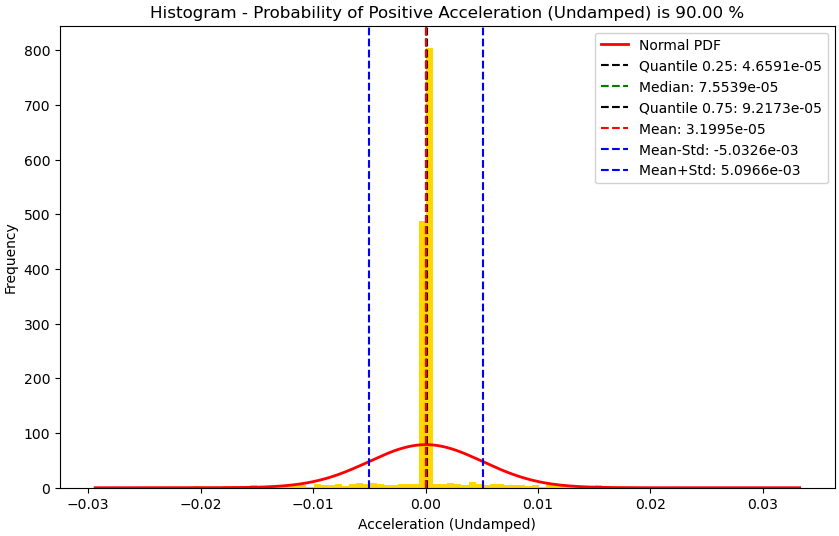

```
-------------------------
Box-Chart Datas:
Minimum: -2.1187e-03
First quartile: -7.3509e-04
Median: -2.8944e-05
Mean: 3.9653e-05
Std: 8.5586e-04
Third quartile: 7.8827e-04
Maximum: 2.0602e-03
Skewness: 1.6260e-01
kurtosis: -1.1619e+00
90% Confidence Interval: (-1.1213e-03, 1.4132e-03)
-------------------------
```

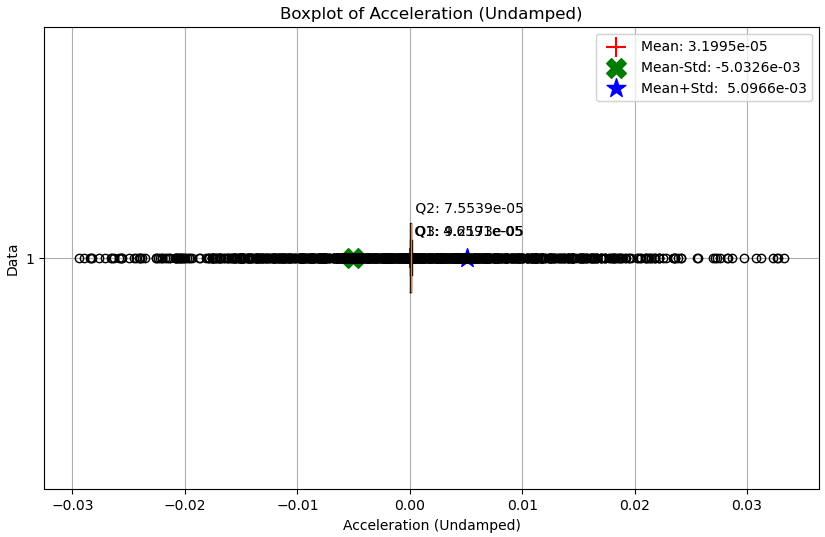




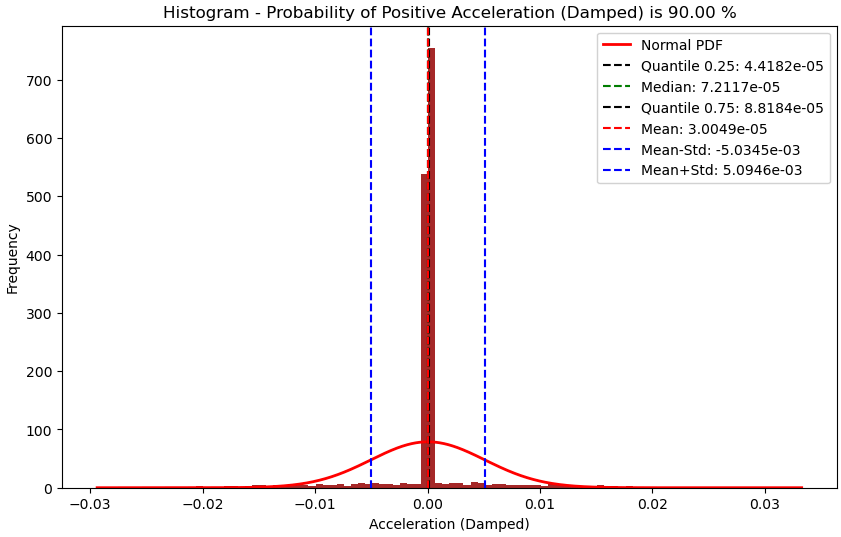
In [59]:
```
HISROGRAM_BOXPLOT(ACCELERATION_undamped, HISTO_COLOR='gold', LABEL='Acceleration (Undamped)')
HISROGRAM_BOXPLOT(ACCELERATION_damped, HISTO_COLOR='brown', LABEL='Acceleration (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(ACCELERATION_undamped, XLABEL='Acceleration (Undamped)', TITLE='Acceleration (Undamped)', COLOR='gold')
#HISTOGRAM_BOXPLOT_PLOTLY(ACCELERATION_damped, XLABEL='Acceleration (Damped)', TITLE='Acceleration (Damped)', COLOR='brown')
```

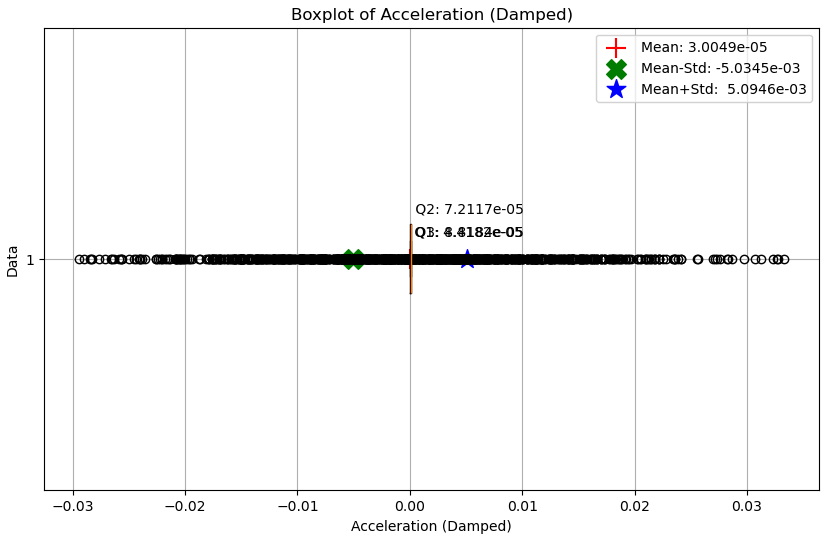
```
-------------------------
Box-Chart Datas:
Minimum: -2.9386e-02
First quartile: 4.6591e-05
Median: 7.5539e-05
Mean: 3.1995e-05
Std: 5.0646e-03
Third quartile: 9.2173e-05
Maximum: 3.3276e-02
Skewness: 7.3052e-02
kurtosis: 1.1642e+01
90% Confidence Interval: (-7.6494e-03, 6.9992e-03)
-------------------------
```






```
-------------------------
Box-Chart Datas:
Minimum: -2.9389e-02
First quartile: 4.4182e-05
Median: 7.2117e-05
Mean: 3.0049e-05
Std: 5.0646e-03
Third quartile: 8.8184e-05
Maximum: 3.3280e-02
Skewness: 7.4283e-02
kurtosis: 1.1643e+01
90% Confidence Interval: (-7.6465e-03, 6.9998e-03)
-------------------------
```

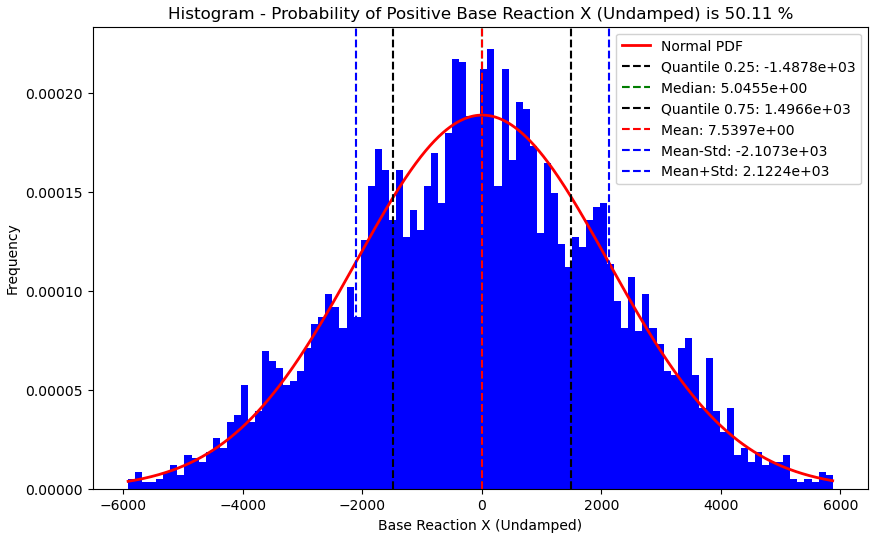


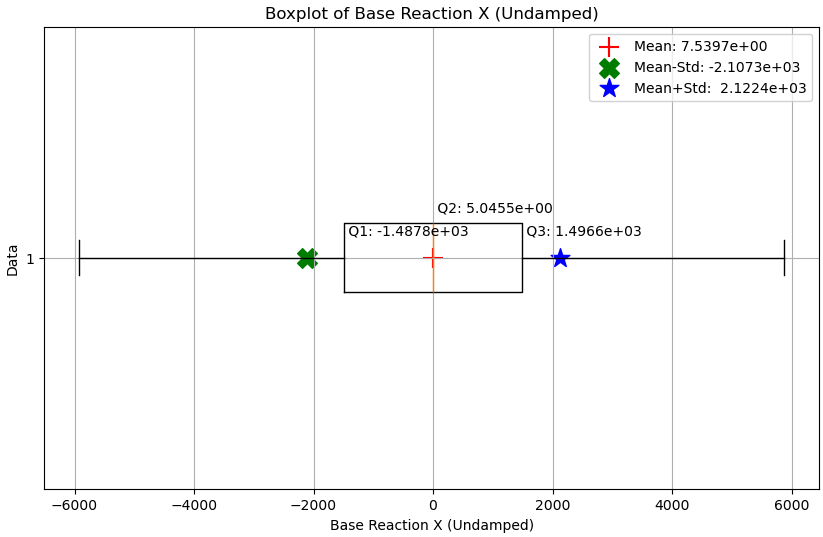


In [60]:
```
HISROGRAM_BOXPLOT(BASE_REACTION_X_undamped, HISTO_COLOR='blue', LABEL='Base Reaction X (Undamped)')
HISROGRAM_BOXPLOT(BASE_REACTION_X_damped, HISTO_COLOR='cyan', LABEL='Base Reaction X (Damped)')
```

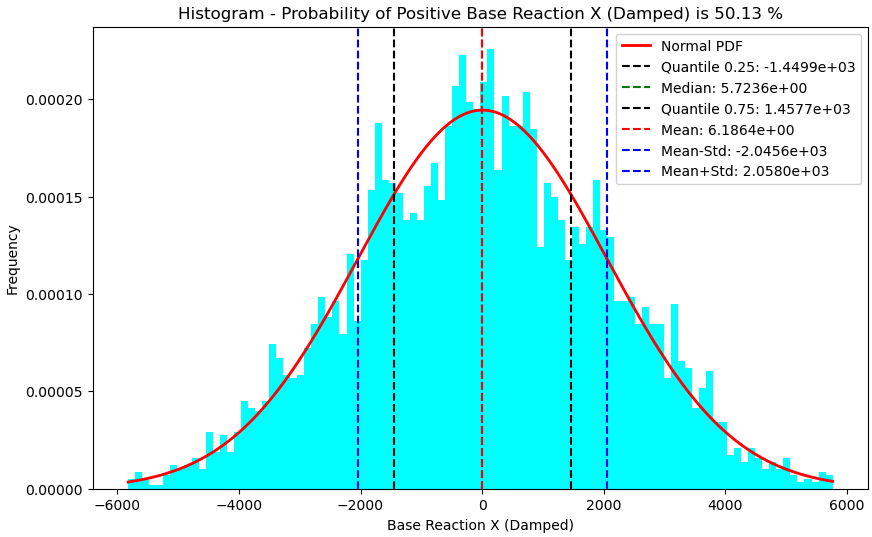
```
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_X_undamped, XLABEL='Base Reaction X (Undamped)', TITLE='Base Reaction X (Undamped)', COLOR='blue')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_X_damped, XLABEL='Base Reaction X (Damped)', TITLE='Base Reaction X (Damped)', COLOR='cyan')
```

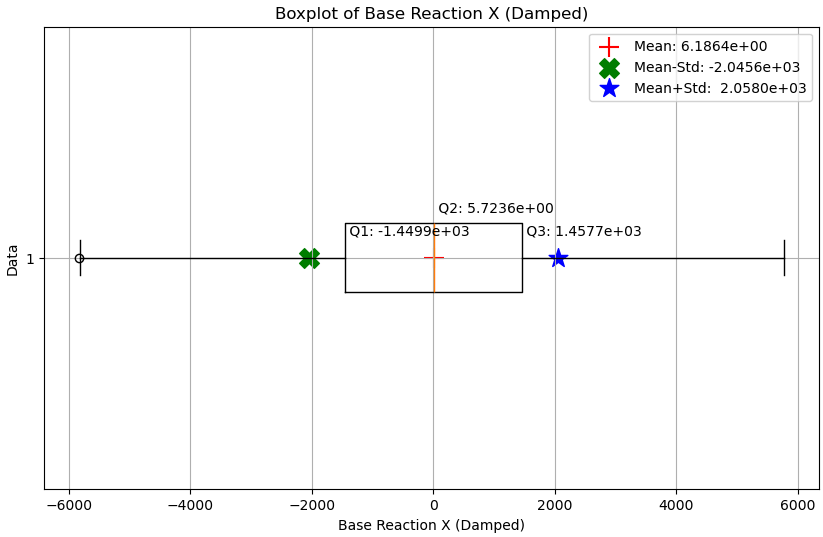
```
-------------------------
Box-Chart Datas:
Minimum: -5.9191e+03
First quartile: -1.4878e+03
Median: 5.0455e+00
Mean: 7.5397e+00
Std: 2.1149e+03
Third quartile: 1.4966e+03
Maximum: 5.8726e+03
Skewness: -1.0914e-02
kurtosis: -3.3292e-01
90% Confidence Interval: (-3.5753e+03, 3.5248e+03)
-------------------------
```





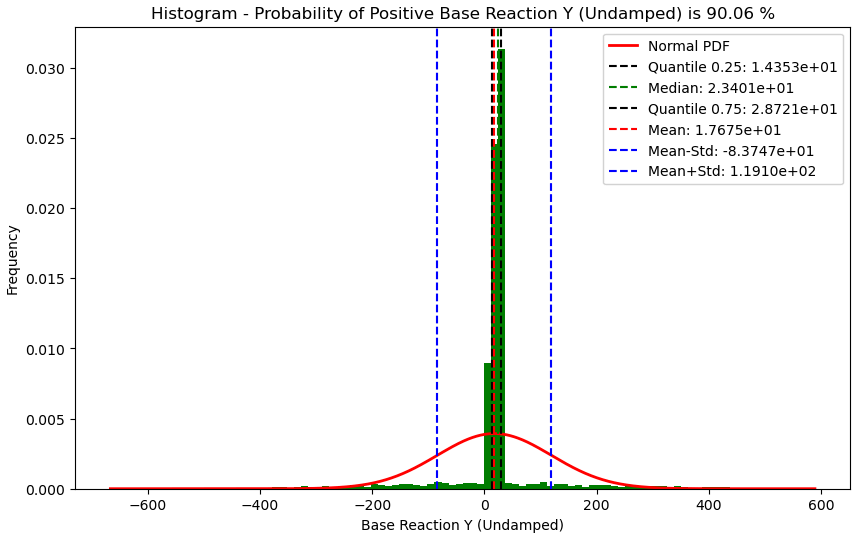

```
-------------------------
Box-Chart Datas:
Minimum: -5.8241e+03
First quartile: -1.4499e+03
Median: 5.7236e+00
Mean: 6.1864e+00
Std: 2.0518e+03
Third quartile: 1.4577e+03
Maximum: 5.7685e+03
Skewness: -1.2907e-02
kurtosis: -3.2647e-01
90% Confidence Interval: (-3.4306e+03, 3.3849e+03)
-------------------------
```

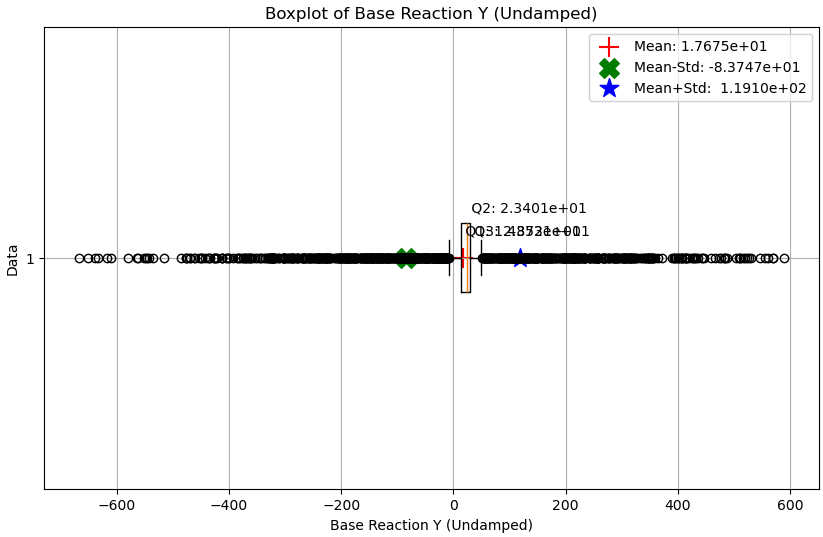




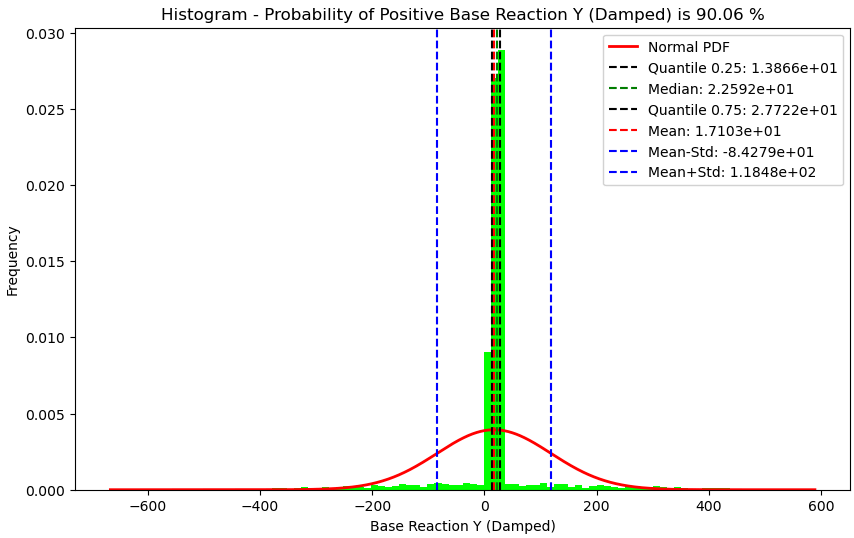
In [61]:
```
HISROGRAM_BOXPLOT(BASE_REACTION_Y_undamped, HISTO_COLOR='green', LABEL='Base Reaction Y (Undamped)')
HISROGRAM_BOXPLOT(BASE_REACTION_Y_damped, HISTO_COLOR='lime', LABEL='Base Reaction Y (Damped)')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_Y_undamped, XLABEL='Base Reaction Y (Undamped)', TITLE='Base Reaction Y (Undamped)', COLOR='green')
#HISTOGRAM_BOXPLOT_PLOTLY(BASE_REACTION_Y_damped, XLABEL='Base Reaction Y (Damped)', TITLE='Base Reaction Y (Damped)', COLOR='lime')
```

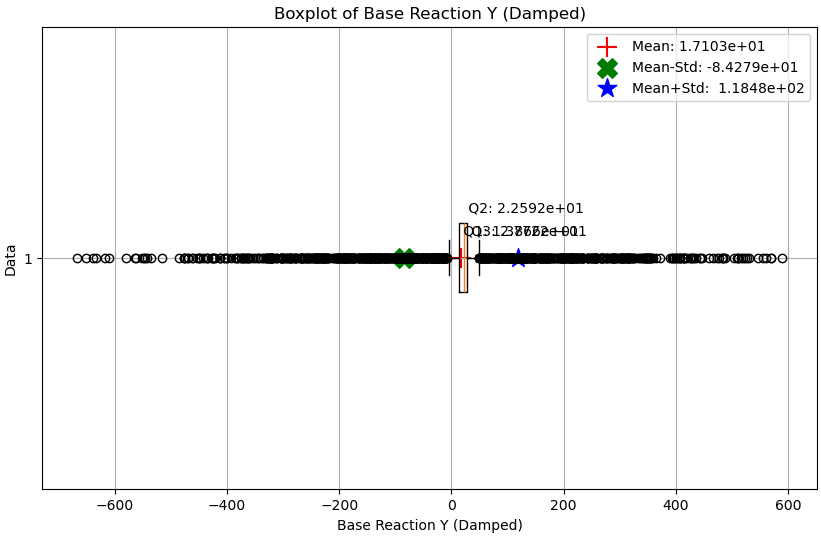
```
-------------------------
Box-Chart Datas:
Minimum: -6.6622e+02
First quartile: 1.4353e+01
Median: 2.3401e+01
Mean: 1.7675e+01
Std: 1.0142e+02
Third quartile: 2.8721e+01
Maximum: 5.8865e+02
Skewness: -5.9676e-01
kurtosis: 1.1418e+01
90% Confidence Interval: (-1.4462e+02, 1.5239e+02)
-------------------------
```






```
-------------------------
Box-Chart Datas:
Minimum: -6.6622e+02
First quartile: 1.3866e+01
Median: 2.2592e+01
Mean: 1.7103e+01
Std: 1.0138e+02
Third quartile: 2.7722e+01
Maximum: 5.8865e+02
Skewness: -5.8063e-01
kurtosis: 1.1428e+01
90% Confidence Interval: (-1.4462e+02, 1.5239e+02)
-------------------------
```





In [ ]:

In [ ]:

In [ ]:

```
###### BEAM ######

def PUSHOVER_ANALYSIS_BEAM(span, height, A_cable, num_nodes, MAX_DISP, MAX_ITERATIONS, TOLERANCE):
    import openseespy.opensees as ops
    import numpy as np
    import matplotlib.pyplot as plt

    # Define model builder
    ops.wipe()
    ops.model('basic', '-ndm', 2, '-ndf', 3)
    dx = span / (num_nodes - 1)

    # Create nodes
    for i in range(num_nodes):
        x = i * dx
        y = -height * np.sin(np.pi * x / span)
        ops.node(i + 1, x, y)
    for j in range(num_nodes):
        ops.node(num_nodes + j + 1, j * dx, -7000)

    # Define boundary conditions (fixed at both ends)
    ops.fix(1, 1, 1)              # TOP CABLE
    ops.fix(num_nodes, 1, 1)      # TOP CABLE
    ops.fix(num_nodes + 1, 1, 1)  # BOTTOM CABLE - DECK
    ops.fix(2 * num_nodes, 1, 1)  # BOTTOM CABLE - DECK

    # Define material properties
    Fy_cable = 355   # [N/mm^2] Yield strength of cable
    E_cable = 210e5  # [N/mm^2] Modulus of elasticity Cable
    E_beam = 210e5   # [N/mm^2] Modulus of elasticity Beam
    I_beam = 100000  # [mm^4] Moment of inertia of the beam
    A_beam = 2000    # [mm^2] Cross-sectional area of the beam
    b0 = 0.01
    ops.uniaxialMaterial('Elastic', 1, E_cable)
    ops.uniaxialMaterial('Elastic', 2, E_beam)

    # Define truss elements for the cable
    for i in range(num_nodes - 1):
        ops.element('corotTruss', i + 1, i + 1, i + 2, A_cable, 1)  # Cable element

    # Define geometric transformation
    ops.geomTransf('Linear', 1)
    # Define beam elements for the beam
    for i in range(num_nodes - 1):
        # element elasticBeamColumn $eleTag $iNode $jNode $A $E $Iz $transfTag <-mass $massDens> <-cMass>
        ops.element('elasticBeamColumn', num_nodes + i + 1, num_nodes + i + 1, num_nodes + i + 2, A_beam, E_beam, I_beam, 1)  # Beam element

    # Connect each node of the cable with the corresponding node of the beam using truss elements
    for i in range(num_nodes):
        ops.element('corotTruss', 2 * num_nodes + i + 1, i + 1, num_nodes + i + 1, A_cable, 1)

    mid_node = num_nodes // 2 + 1
    # Define load pattern with displacement at the middle span
    ops.timeSeries('Linear', 1)
    ops.pattern('Plain', 1, 1)
    ops.load(mid_node, 0.0, 1)  # Apply a horizontal load at node
    disp_incr = 0.01    # [mm] Incremental Displacement
    n_steps = int(np.abs(MAX_DISP / disp_incr))  # Analysis Steps

    # Define analysis parameters
    ops.system('BandGeneral')
    ops.numberer('Plain')
    ops.constraints('Plain')
    ops.integrator('DisplacementControl', mid_node, 2, disp_incr)
    ops.test('EnergyIncr', TOLERANCE, MAX_ITERATIONS)
    ops.algorithm('ModifiedNewton')
    ops.analysis('Static')

    DISPLACEMENTS = []
    DISP_X , DISP_Y = [], []
    BASE_REACTION_X, BASE_REACTION_Y = [], []

    # Perform the analysis with increments
    for i in range(n_steps):
        OK = ops.analyze(1)
```

```
        ANALYSIS(OK, 1, TOLERANCE, MAX_ITERATIONS)
        displacements = [ops.nodeDisp(j + 1) for j in range(num_nodes * 2)]
        DISPLACEMENTS.append([disp[:2] for disp in displacements])  # Remove the rotational component
        DISP_X.append(ops.nodeDisp(mid_node, 1))
        DISP_Y.append(ops.nodeDisp(mid_node, 2))
        BASE_REACTION_X.append(ops.eleResponse(1, 'force')[0])
        BASE_REACTION_Y.append(ops.eleResponse(1, 'force')[1])
        print(f'STEP: {i+1}')

    # Get initial and final node coordinates for plotting
    initial_coords = np.array([ops.nodeCoord(i + 1) for i in range(num_nodes * 2)])
    deformed_coords = initial_coords + np.array(DISPLACEMENTS[-1])

    # Output all unconstrained node displacements
    for i in range(num_nodes * 2):
        disp = ops.nodeDisp(i + 1)
        print(f"Node {i + 1}: X Displacement = {disp[0]}, Y Displacement = {disp[1]}")

    # Plot initial and deformed shape
    plt.figure(figsize=(12, 8))
    plt.plot(initial_coords[:num_nodes, 0], initial_coords[:num_nodes, 1], 'bo-', label='Initial Cable Shape')
    plt.plot(deformed_coords[:num_nodes, 0], deformed_coords[:num_nodes, 1], 'ro-', label='Deformed Cable Shape')
    plt.plot(initial_coords[num_nodes:, 0], initial_coords[num_nodes:, 1], 'go-', label='Initial Beam Shape')
    plt.plot(deformed_coords[num_nodes:, 0], deformed_coords[num_nodes:, 1], 'yo-', label='Deformed Beam Shape')

    # Plot connecting elements between the cable and the beam
    for i in range(num_nodes):
        plt.plot([initial_coords[i, 0], initial_coords[num_nodes + i, 0]],
                 [initial_coords[i, 1], initial_coords[num_nodes + i, 1]], 'c--', label='Initial Connecting Elements' if i == 0 else "")
        plt.plot([deformed_coords[i, 0], deformed_coords[num_nodes + i, 0]],
                 [deformed_coords[i, 1], deformed_coords[num_nodes + i, 1]], 'm--', label='Deformed Connecting Elements' if i == 0 else "")

    # Mark simply supported nodes with red triangles
    support_nodes = [0, num_nodes - 1, num_nodes, BEAM_NODE - 1]
    plt.plot(initial_coords[support_nodes, 0], initial_coords[support_nodes, 1], 'r^', markersize=20, label='Simply Supported Nodes')

    plt.xlabel('X [mm]')
    plt.ylabel('Y [mm]')
    plt.legend()
    plt.title('Pushover Analysis: Cable and Beam Deformed Shapes')
    plt.show()

    return initial_coords, deformed_coords, DISPLACEMENTS, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y
```

In [ ]:
```
# Define geometry and nodes
span = 10000.0  # [mm]
height = 5000.0  # [mm]
A_cable = 200  # [mm^2]
num_nodes = 51
MAX_DISP = -10.0  # [mm]
MAX_ITERATIONS = 5000
TOLERANCE = 1.0e-10

# Run the analysis and plot results
initial_coords, deformed_coords, DISPLACEMENTS, DISP_X, DISP_Y, BASE_REACTION_X, BASE_REACTION_Y = PUSHOVER_ANALYSIS_BEAM(span, height, A_cable, num_nodes, MAX_DISP, MAX_ITERATIONS, TO
plot_reactions(DISP_X, BASE_REACTION_X, DISP_Y, BASE_REACTION_Y)
```

In [ ]:
```

```

In [ ]:
```

```