exercise1-bisection (Score: 13.0 / 14.0)

Lab 2

1. 提交作業之前，建議可以先點選上方工具列的**Kernel**，再選擇**Restart & Run All**，檢查一下是否程式跑起來都沒有問題，最後記得儲存。
2. 請先填上下方的姓名(name)及學號(stduent_id)再開始作答，例如：

```
name = "我的名字"
student_id= "B06201000"
```

3. 四個求根演算法的實作可以參考lab-2 (https://yuanyuyuan.github.io/itcm/lab-2.html)，裡面有教學影片也有範例程式可以套用。
4. **Deadline: 10/9(Wed.)**

In [1]:

```
name = "鄭如芳"
student_id = "B05602020"
```

Exercise 1 - Bisection

Use the bisection method to find roots of

$$f(x) = cosh(x) + cos(x) - c, \text{for } c = 1, 2, 3,$$

Import libraries

In [2]:

```
import matplotlib.pyplot as plt
import numpy as np
```

1. Define a function $g(c)(x) = f(x) = cosh(x) + cos(x) - c$ with parameter $c = 1, 2, 3$.

In [3]:

(Top)

```
def g(c):
    assert c == 1 or c == 2 or c == 3
    def f(x):
        # Hint: return ...
        # ===== 請實做程式 =====
        return np.cosh(x)+np.cos(x)-c
        # ====================
    return f
```

Pass the following assertion.

In [4]:

cell-b59c94b754b1fc9e (Top)

```
assert g(1)(0) == np.cosh(0) + np.cos(0) - 1
### BEGIN HIDDEN TESTS
assert g(2)(0) == np.cosh(0) + np.cos(0) - 2
assert g(3)(0) == np.cosh(0) + np.cos(0) - 3
### END HIDDEN TESTS
```

2. Implement the algorithm

In [5]:

(Top)

```python
def bisection(
    func,
    interval,
    max_iterations=5,
    tolerance=1e-7,
    report_history=False,
):
    '''
    Parameters
    ----------
    func : function
        The target function
    interval: list
        The initial interval to search
    max_iterations: int
        One of the termination conditions. The amount of iterations allowed.
    tolerance: float
        One of the termination conditions. Error tolerance.
    report_history: bool
        Whether to return history.

    Returns
    -------
    result: float
        Approximation of the root.
    history: dict
        Return history of the solving process if report_history is True.
    '''

    # ===== 請實做程式 =====
    a, b=interval
    assert func(a)*func(b)<0
    num_iteration=0
    a_next,b_next=a,b

    if report_history:
        history = {'estimation': [], 'error': []}

    while True:
        c=(a_next+b_next)/2

        error=(b_next-a_next)/2

        if report_history:
            history['estimation'].append(c)
            history['error'].append(error)

        if error < tolerance:
            print('The approximation has satisfied the tolerance.')
            return (c, history) if report_history else c

        if num_iteration<max_iterations:
            num_iteration+=1

            value_of_func_c = func(c)
            if func(a_next)*value_of_func_c<0:
                a_next=a_next
                b_next=c
            elif func(b_next)*value_of_func_c<0:
                a_next=c
                b_next=b_next
            else:
                return (c, history) if report_history else c
        else:
            print('Terminate since reached the maximum iterations.')
            return (c, history) if report_history else c
    # ====================
```

Test your implementation with the assertion below.

In [6]:

```
cell-4d88293f2527c82d
```
(Top)

```
root = bisection(lambda x: x**2 - x - 1, [1.0, 2.0], max_iterations=100, tolerance=1e-7, report_history=False)
assert abs(root - ((1 + np.sqrt(5)) / 2)) < 1e-7
```

```
The approximation has satisfied the tolerance.
```

3. Answer the following questions under the case $c = 1$.

Plot the function to find an interval that contains the zero of f if possible.

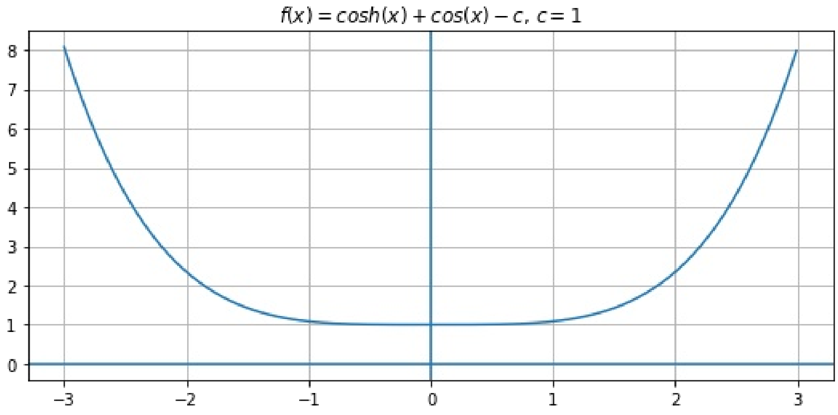
In [7]:

(Top)

```
c = 1
f = g(c)

# Hint: search_range = np.arange(左端點, 右端點, 點與點之間距),
# e.g. search_range = np.arange(0.0, 1.0, 0.01)
# ===== 請實做程式 =====
search_range=np.arange(-3.0,3.0,0.01)
# ====================

fig, ax = plt.subplots(figsize=(9, 4))
ax.plot(search_range, f(search_range))
ax.set_title(r'$f(x)=cosh(x)+cos(x)-c$, $c=$%d' % c)
ax.grid(True)
ax.axhline(y=0)
ax.axvline(x=0)
plt.show()
```



According to the figure above, estimate the zero of f.

For example,

```
root = 3        # 單根
root = -2, 1    # 多根
root = None     # 無解
```

In [8]:

(Top)

```
# Hint: root = ?
# ===== 請實做程式 =====
root=None
# ====================
```

In [9]:

cell-d872c7c57f11c968 (Top)

```
print('My estimation of root:', root)
### BEGIN HIDDEN TESTS
if root == None:
    print('Right answer!')
else:
    raise AssertionError('Wrong answer!')
### END HIDDEN TESTS
```

```
My estimation of root: None
Right answer!
```

Try to find the zero with a tolerance of 10^{-10}. If it works, plot the error and estimation of each step. Otherwise, state the reason why the method failed on this case.

(Top)

c = 1 f = g(c) print(f(0))

$$cannot find the zero with tolerance of 10^{-10} bcs even the point x = 0 which is the closest point to x - axis still have distance = 1 from x - axis$$

---

4. Answer the following questions under the case $c = 2$.

Plot the function to find an interval that contains the zero of f if possible.

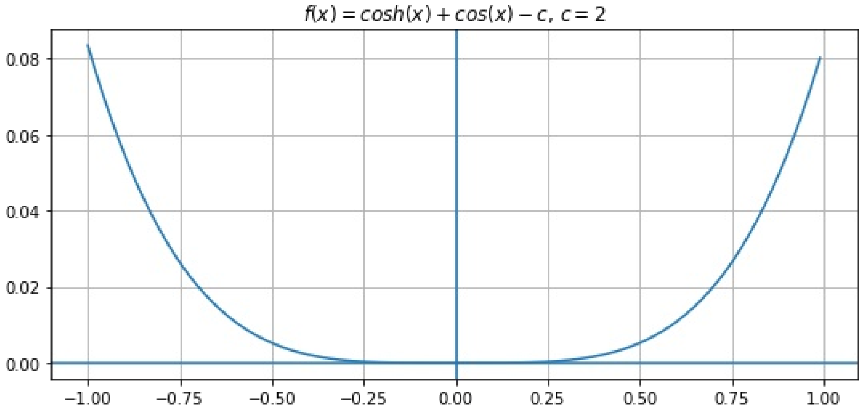
In [10]:

(Top)

```
c = 2
f = g(c)

# Hint: search_range = np.arange(左端點, 右端點, 點與點之間距),
# e.g. search_range = np.arange(0.0, 1.0, 0.01)
# ===== 請實做程式 =====
search_range=np.arange(-1.0,1.0,0.01)
# ====================

fig, ax = plt.subplots(figsize=(9, 4))
ax.plot(search_range, f(search_range))
ax.set_title(r'$f(x)=cosh(x)+cos(x)-c$, $c=$%d' % c)
ax.grid(True)
ax.axhline(y=0)
ax.axvline(x=0)
plt.show()
```



According to the figure above, estimate the zero of f.

For example,

```
root = 3        # 單根
root = -2, 1    # 多根
root = None     # 無解
```

In [11]:

(Top)

```
# Hint: root = ?
# ===== 請實做程式 =====
root=0.00
# ====================
```

In [12]:

cell-20fddbe6fa4c437b (Top)

```
print('My estimation of root:', root)

### BEGIN HIDDEN TESTS
assert type(root) is float or int, 'Wrong type!'
### END HIDDEN TESTS
```

```
My estimation of root: 0.0
```

Try to find the zero with a tolerance of 10^{-10}. If it works, plot the error and estimation of each step. Otherwise, state the reason why the method failed on this case.

(Top)

$the\ graph\ does\ not\ pass\ through\ the\ x-axis so\ there\ will\ be\ assertion\ error\ in\ our\ bisection\ function since\ we\ cannot\ satisfy f(a)*f(b)<0\ this\ condition$

---

5. Answer the following questions under the case $c = 3$.

Plot the function to find an interval that contains the zeros of f if possible.

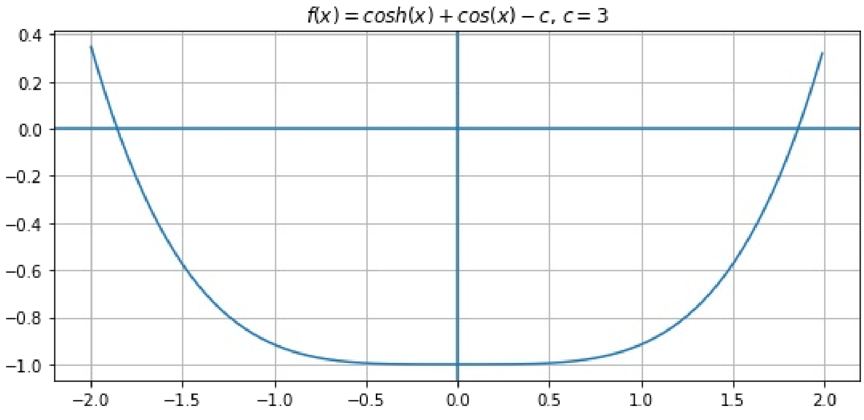
In [13]:

(Top)

```
c = 3
f = g(c)

# Hint: search_range = np.arange(左端點, 右端點, 點與點之間距),
# e.g. search_range = np.arange(0.0, 1.0, 0.01)
# ===== 請實做程式 =====
search_range=np.arange(-2.0,2.0,0.01)
# ====================

fig, ax = plt.subplots(figsize=(9, 4))
ax.plot(search_range, f(search_range))
ax.set_title(r'$f(x)=cosh(x)+cos(x)-c$, $c=$%d' % c)
ax.grid(True)
ax.axhline(y=0)
ax.axvline(x=0)
plt.show()
```



According to the figure above, estimate the zero of f.

For example,

```
root = 3         # 單根
root = -2, 1     # 多根
root = None      # 無解
```

In [14]:

(Top)

```
# Hint: root = ?
# ===== 請實做程式 =====
ans1_interval = [-2.0,0.0]
ans1 = bisection(
    f,
    ans1_interval,
)
ans2_interval=[0.0,2.0]
ans2 = bisection(
    f,
    ans2_interval,
)
root=ans1,ans2
# ===================
```

```
Terminate since reached the maximum iterations.
Terminate since reached the maximum iterations.
```

In [15]:

cell-06ec0b20844075c7 (Top)

```
print('My estimation of root:', root)

### BEGIN HIDDEN TESTS
assert type(root) == tuple, 'Should be multiple roots!'
### END HIDDEN TESTS
```

```
My estimation of root: (-1.84375, 1.84375)
```

Try to find the zero with a tolerance of 10^{-10}. If it works, plot the error and estimation of each step. Otherwise, state the reason why the method failed on this case.

In [16]:

(Top)

```
$$
since\,the\,graph\,of\,this\,case\,is\,symmetrical\,with\,y-axis\\
i\,discuss\,the\,positive\,root$$
```

Comments:
For case c=3, there are two roots to be found

```
  File "<ipython-input-16-ba6b949324ee>", line 1
    $$
    ^
SyntaxError: invalid syntax
```

In [17]:

```
my_initial_interval = [1.0, 2.0]

solution,history= bisection(
    f,
    my_initial_interval,
    max_iterations=33,
    tolerance=1e-10,
    report_history=True
)
print(solution)
exact_solution=1.8579208291484974
```

```
The approximation has satisfied the tolerance.
1.8579208291484974
```

In [18]:

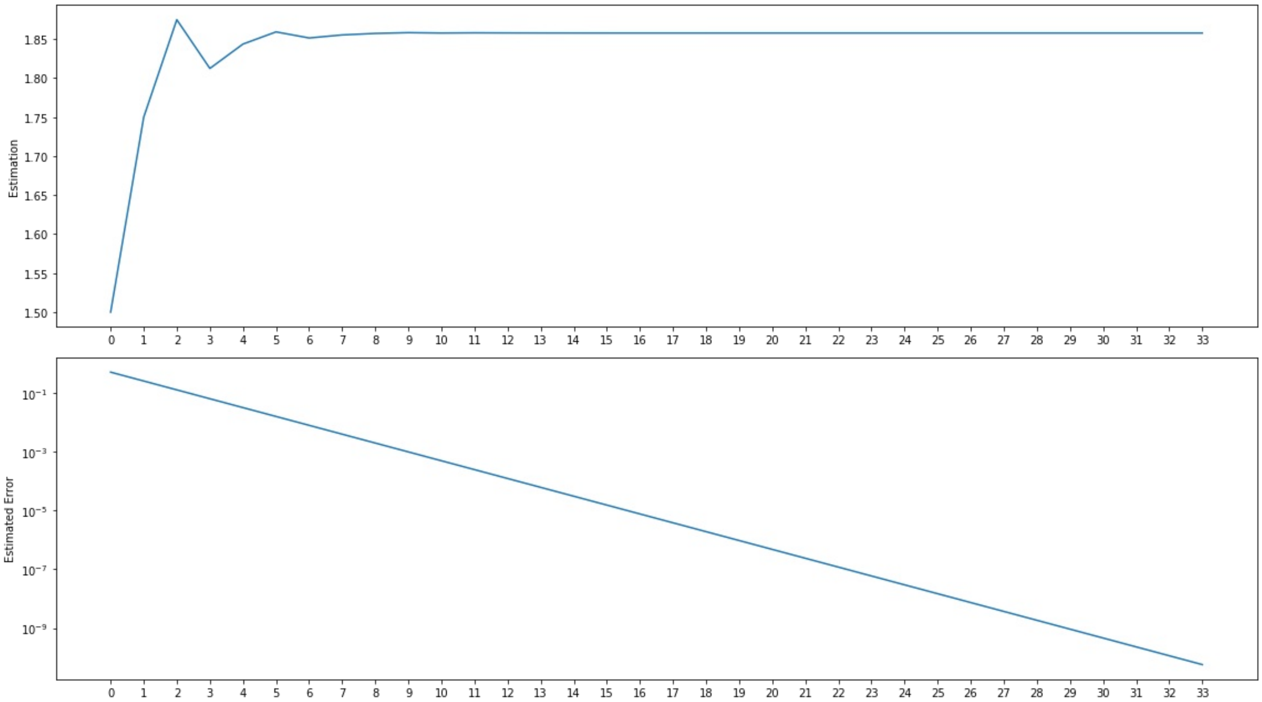
```
fig, axes = plt.subplots(2, 1, figsize=(16, 9))
ax1, ax2 = axes

num_iterations = len(history['estimation'])
iterations = range(num_iterations)
for ax in axes:
    ax.set_xticks(iterations)

ax1.plot(iterations, history['estimation'])
ax1.set_ylabel('Estimation')

ax2.plot(iterations, history['error'])
ax2.set_ylabel('Estimated Error')
ax2.set_yscale('log')

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



Discussion

For all cases above(c=1,2,3), do the results(e.g. error behaviors, estimations, etc) agree with the theoretical analysis?

Suppose given the tolerance $\epsilon = 10^{-10}$.

The minimal iterations n to converge started from the interval $[1, 2]$ can be derived by

$$|error| < \frac{|b-a|}{2^{n+1}} \implies 10^{-10} < \frac{2-1}{2^{n+1}} \implies n > \log_2(10^{10}) - 1 \implies n \geq 32.$$

In [19]:

```
cal=np.log2(1e10)
print(cal)
```

```
33.219280948873624
```

(Top)

The result calculate from the theoretical analysisis agree with what I have done aboveboth show that 33 *times, midpoint cut can satisfy the tolerance* 10^{-10}