Homework 8

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Problem 1 Consider the following linear program

$$\begin{aligned} \min \quad & -2x_1 + 4x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 4 \\ & -x_1 + x_2 \leq 2 \\ & x_1 - x_2 \leq 2 \\ & x_1 \geq 0, x_2 \geq 0. \end{aligned}$$

Graph the constraints of this linear program, and indicate the feasible region.

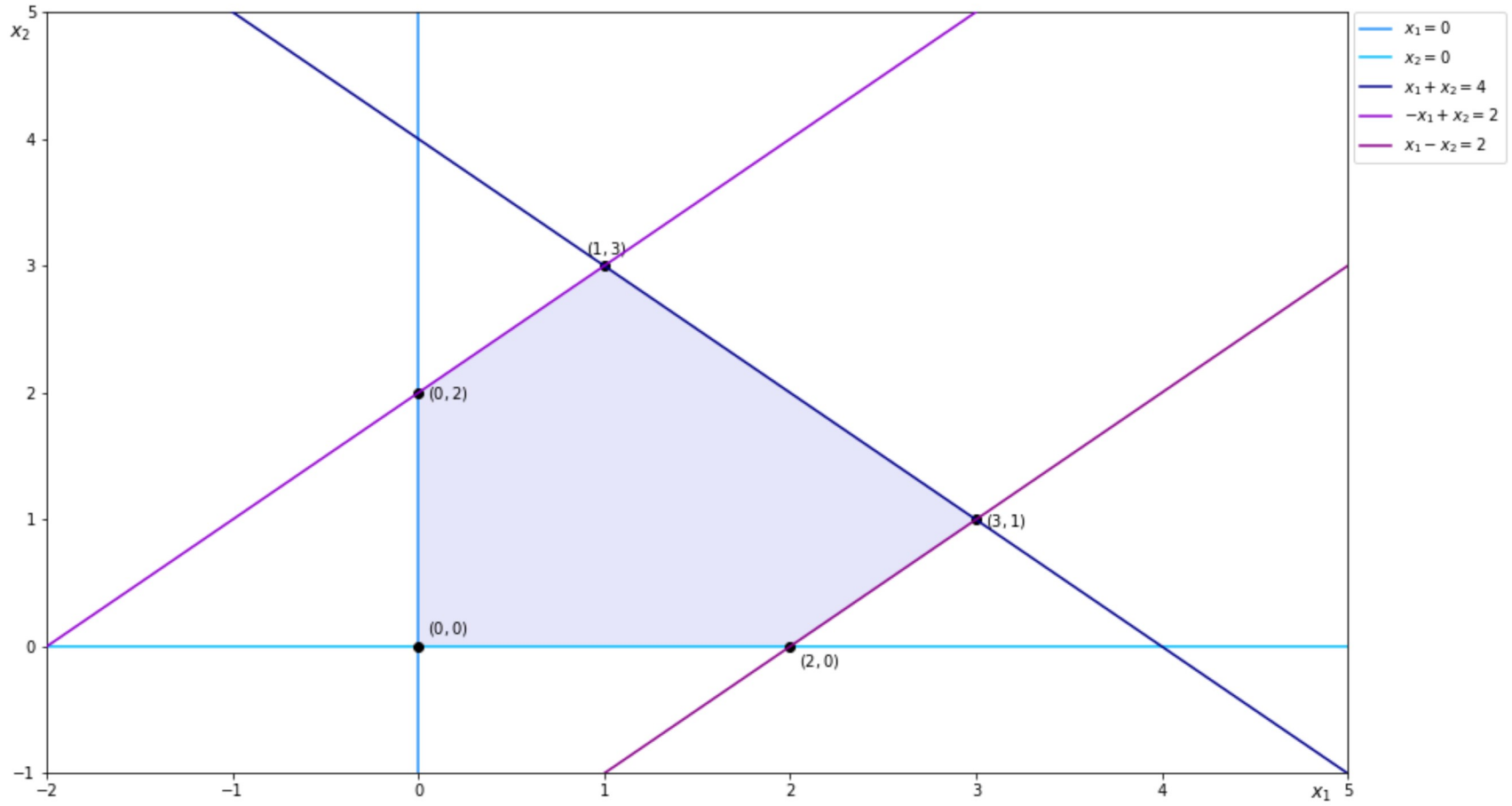
Answer: Using matplotlib we can graph constraints and the feasible region as follows. Green points represent basic feasible solutions used in simplex method.

In [3]:
```
# construct lines
x = np.linspace(-2, 5, 2000)
# x_2 = 0
y1 = 0*x
# x_1 + x_2 = 4
y2 = 4 - x
# -x_1 + x_2 = 2
y3 = 2 + x
# x_1 - x_2 = 2
y4 = -2 + x

# make plot - draw the lines and corner points
fig = plt.figure(figsize = (15, 9))
ax = fig.add_subplot()
plt.plot([0,0],[-2,5],label = r'$x_1=0$', zorder = 1, color = 'dodgerblue')
plt.plot(x, y1, label = r'$x_2=0$', zorder = 2, color = 'deepskyblue')
plt.plot(x, y2, label = r'$x_1+x_2=4$', zorder = 3, color = 'darkblue')
plt.plot(x, y3, label = r'$-x_1+x_2=2$', zorder = 4, color = 'darkviolet')
plt.plot(x, y4, label = r'$x_1-x_2=2$', zorder = 5, color = 'darkmagenta')
plt.plot([0, 2, 3, 1, 0],[0, 0, 1, 3, 2], 'o', color = 'black')

# set axis limits
plt.xlim((-2, 5))
plt.ylim((-1, 5))
# label axis
ax.annotate(r'$x_2$', (-2, 5), (-2.2, 4.8), color = 'black', fontsize = 12)
ax.annotate(r'$x_1$', (5, 0), (4.8, -1.2), color = 'black', fontsize = 12)
#label extreme points
ax.annotate(r'$(0, 0)$', (0, 0), (0.05, 0.1))
ax.annotate(r'$(2, 0)$', (2, 0), (2.05, -0.15))
ax.annotate(r'$(3, 1)$', (3, 1), (3.05, 0.95))
ax.annotate(r'$\left(1, 3\right)$', (1, 3), (0.9, 3.1))
ax.annotate(r'$(0, 2)$', (0, 2), (0.05, 1.95))

# fill feasible region
plt.fill([0, 2, 3, 1, 0], [0, 0, 1, 3, 2], color = 'lavender')
plt.legend(bbox_to_anchor = (1.005, 1), loc = 2, borderaxespad=0.)
#plt.show()
plt.savefig("HW8Q1.jpg", dpi = 600)
```



Problem 2 To solve the problem using the simplex method, firts transform it into a standard form LP. Denote $x = [x_1, x_2, x_3, x_4, x_5]^T$ as the vector of variables, and use the standard form notation:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0, \end{aligned}$$

specify c, A, b for the above problem.

Answer: We can rewrite the problem as standard form LP:

$$\begin{aligned} \min \quad & -2x_1 + 4x_2 \\ \text{s.t.} \quad & x_1 + x_2 + x_3 = 4 \\ & -x_1 + x_2 + x_4 = 2 \\ & x_1 - x_2 + x_5 = 2 \\ & x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0. \end{aligned}$$

or using matrix notation as:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0, \end{aligned}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \; c = \begin{bmatrix} -2 \\ 4 \\ 0 \end{bmatrix}, \; A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 1 & 0 \end{bmatrix}, \; b = \begin{bmatrix} 4 \\ 2 \end{bmatrix}.$$

In [4]:
```
c = np.array([-2, 4, 0, 0, 0])
A = np.array([[1, 1, 1, 0, 0], [-1, 1, 0, 1, 0], [1, -1, 0, 0, 1]])
b = np.array([4, 2, 2])
```

Problem 3 Now we want to solve the above standard-form linear program by the simplex method. If in an iteration of the Simplex method, there is any ambiguity about which nonbasic variable to enter the basis or whih basic variable to exit the basis, use Bland's rule.

For each iteration of the Simplex method, write down the following information in the format given below:

- Iteration $k = \underline{\text{numerical value, e.g. } 1, 2, \ldots}$;
- Basis $B = [A_i, A_j, A_k] = \underline{\text{numerical value}}$ (i.e. you need to specify i, j, k as well as the numerical values of the columns);
- Basis inverse $B^{-1} = \underline{\text{numerical value}}$;
- Basis variable $x_B = [x_i, x_j, x_k] = \underline{\text{numerical value}}$ (you need to specify i, j, k and numerical values of x_i, x_j, x_k);
- Nonbasic variable $x_N = [x_p, x_q] = \underline{\text{numerical value}}$ (you need to specify p, q and numerical values x_p, x_q);
- Reduced cost for each nonbasic variable $\bar{c}_p = c_p - c_B^T B^{-1} A_? = \underline{\text{numerical value}}$; Same for $\bar{c}_q$; (you need index for "?" for $A_?$);
- Is the current solution optimal? If not, which nonbasic variable to enter the basis?

- Direction $d_B = B^{-1}A_? = \underline{\text{numerical value}}$. Does the simplex method terminate with an unbounded optimum?
- Min-ratio test $\theta^* = \min\{x_{B(i)}/(-d_{B(i)}) \mid i : d_{B(i)} < 0\} = \min\{\underline{\text{numerical values of the ratios}}\} = \underline{\text{numerical value of } \theta^*}$.
- Which basic varible to exit the basis?

Start at iteration $k = 1$ with the basis $B^1 = [A_2, A_3, A_4]$. Solve the above linear program with simplex method and write down all the information required above for each iteration. Also indicate the basic feasible solution at each step on the picture in (x_1, x_2). What is the optimal solution of the above LP in $(x_1, \ldots, x_5)$? What is the optimal cost?

Answer

Iteration 1:

- $k = 1$
- $B = [A_1, A_2, A_5] = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & 1 \end{bmatrix}$

In [5]:
```
# find B1^(-1)
B = A[:, [0, 1, 4]]
invB = np.linalg.inv(B)
print(invB)
xB = np.dot(invB, b)
print(xB)
```

```
[[ 0.5 -0.5  0. ]
 [ 0.5  0.5  0. ]
 [ 0.   1.   1. ]]
[1. 3. 4.]
```

- $B^{-1} = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 & 1 \end{bmatrix}$
- f$x_B = \begin{bmatrix} x_1 \\ x_2 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}$

$\begin{bmatrix} x_3 \end{bmatrix} \quad \begin{bmatrix} 0 \end{bmatrix}$

In [6]:
```
# redused cost for two nonbasic variables
c3 = c[2] - np.dot(np.dot(c[[0, 1, 4]].T, invB), A[:, 2])
print(c3)
c4 = c[3] - np.dot(np.dot(c[[0, 1, 4]].T, invB), A[:, 3])
print(c4)
```

```
-1.0
-3.0
```

- We have two reduced cost values $\bar{c}_3 = c_3 - c_B^T B^{-1} A_3 = -1$, and $\bar{c}_4 = c_4 - c_B^T B^{-1} A_4 = -3$
- Since we do have nonbasic variable with negative reduced cost, current solution is not optimal. Using Bland's rule we choose x_3 to enter the basis.

In [7]:
```
# dB direction
dB = np.dot(-invB, A[:, 2])
print(dB)
```

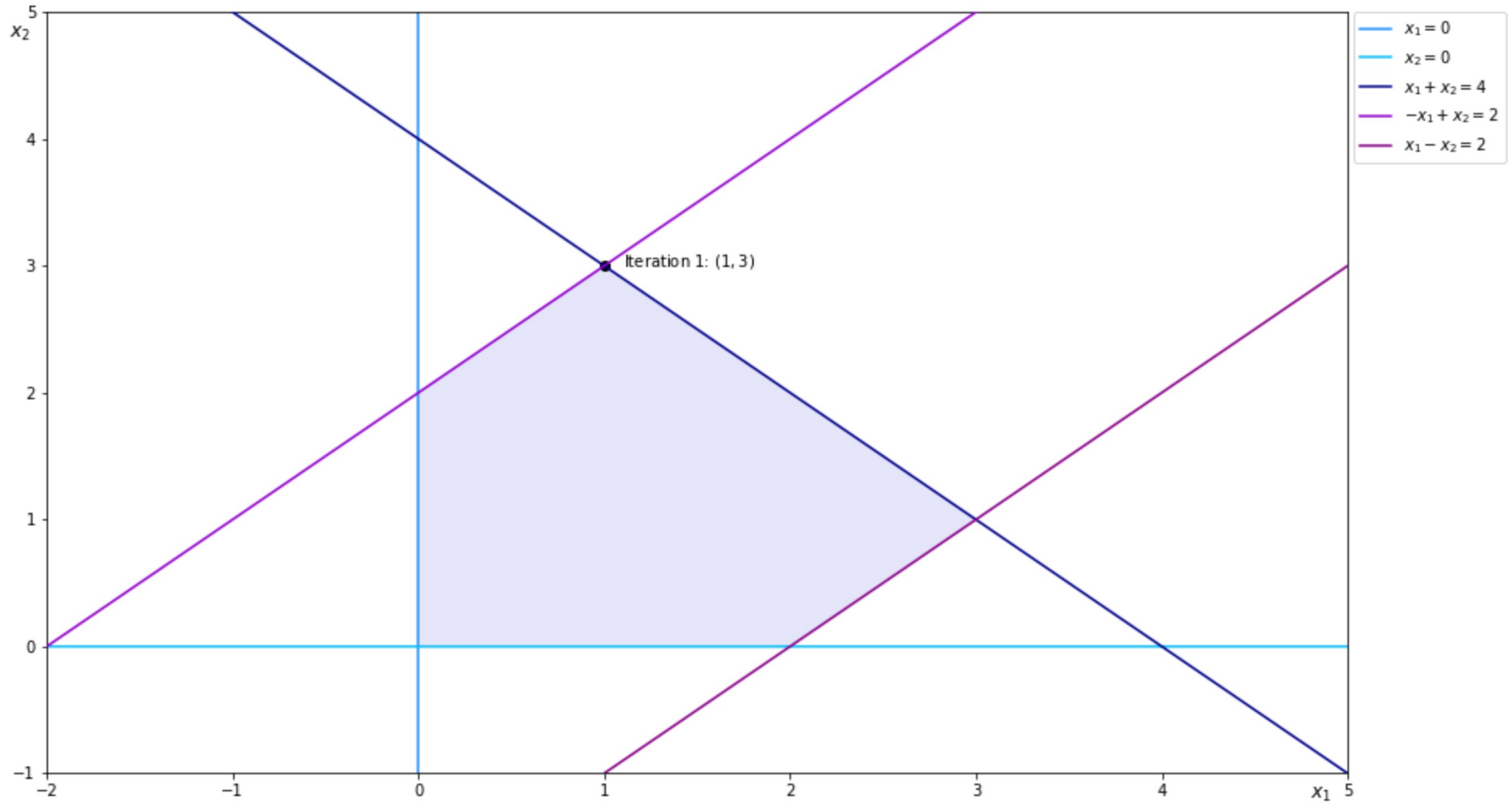
```
[-0.5 -0.5  0. ]
```

- We have $d_B = -B^{-1} A_2 = \begin{bmatrix} d_1 \\ d_2 \\ d_5 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ 0 \end{bmatrix}$ and $d_N = \begin{bmatrix} d_3 \\ d_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ so $d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ 1 \\ 0 \\ 0 \end{bmatrix}$. Since direction d has two negative values we continue.
- $\theta^* = \min\{\frac{x_1}{-d_1}, \frac{x_2}{-d_2}\} = \min\{\frac{1}{0.5}, \frac{3}{0.5}\} = 2$
- Variable x_1 will exit the basis.

In [8]:
```
# construct lines
x = np.linspace(-2, 5, 2000)
# x_2 = 0
y1 = 0*x
# x_1 + x_2 = 4
y2 = 4 - x
# -x_1 + x_2 = 2
y3 = 2 + x
# x_1 - x_2 = 2
y4 = -2 + x

# make plot - draw the lines and corner points
fig = plt.figure(figsize = (15, 9))
ax = fig.add_subplot()
plt.plot([0,0],[-2,5],label = r'$x_1=0$', zorder = 1, color = 'dodgerblue')
plt.plot(x, y1, label = r'$x_2=0$', zorder = 2, color = 'deepskyblue')
plt.plot(x, y2, label = r'$x_1+x_2=4$', zorder = 3, color = 'darkblue')
plt.plot(x, y3, label = r'$-x_1+x_2=2$', zorder = 4, color = 'darkviolet')
plt.plot(x, y4, label = r'$x_1-x_2=2$', zorder = 5, color = 'darkmagenta')
plt.plot([1],[3], 'o', color = 'black')
# set axis limits
plt.xlim((-2, 5))
plt.ylim((-1, 5))
# label axis
ax.annotate(r'$x_2$', (-2, 5), (-2.2, 4.8), color = 'black', fontsize = 12)
ax.annotate(r'$x_1$', (5, 0), (4.8, -1.2), color = 'black', fontsize = 12)
#label extreme points
ax.annotate(r'Iteration 1: $(1, 3)$', (1, 3), (1.1, 3))


# fill feasible region
plt.fill([0, 2, 3, 1, 0], [0, 0, 1, 3, 2], color = 'lavender')
plt.legend(bbox_to_anchor = (1.005, 1), loc = 2, borderaxespad=0.)
#plt.show()
plt.savefig("HW8Q3It1.jpg", dpi = 600)
```



Iteration 2:

- $k = 2$
- current feasible solution is $x = \begin{bmatrix} 1 \\ 3 \\ 0 \\ 0 \\ 4 \end{bmatrix} + 2 \cdot \begin{bmatrix} -\frac{1}{2} \\ -\frac{1}{2} \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 2 \\ 0 \\ 4 \end{bmatrix}$

$$\begin{bmatrix} 1 & 1 & 0 \end{bmatrix}$$

In [9]:
```
# find B1^(-1)
B = A[:, [1, 2, 4]]
invB = np.linalg.inv(B)
print(invB)
```

```
[[ 0.  1.  0.]
 [ 1. -1. -0.]
 [ 0.  1.  1.]]
```

- $B^{-1} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$
- $x_B = \begin{bmatrix} x_2 \\ x_3 \\ x_5 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix}$
- $x_N = \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

In [10]:
```
# redused cost for two nonbasic variables
c1 = c[0] - np.dot(np.dot(c[[1, 2, 4]].T, invB), A[:, 0])
print(c1)
c4 = c[3] - np.dot(np.dot(c[[1, 2, 4]].T, invB), A[:, 3])
print(c4)
```

```
2.0
-4.0
```

- We have one negative reduced cost values $\bar{c}_4 = c_4 - c_B^T B^{-1} A_4 = -4$
- Since we do have nonbasic variable with negative reduced cost, current solution is not optimal. Variable to enter the basis is x_4.

In [11]:
```
# dB direction
dB = np.dot(-invB, A[:, 3])
print(dB)
```

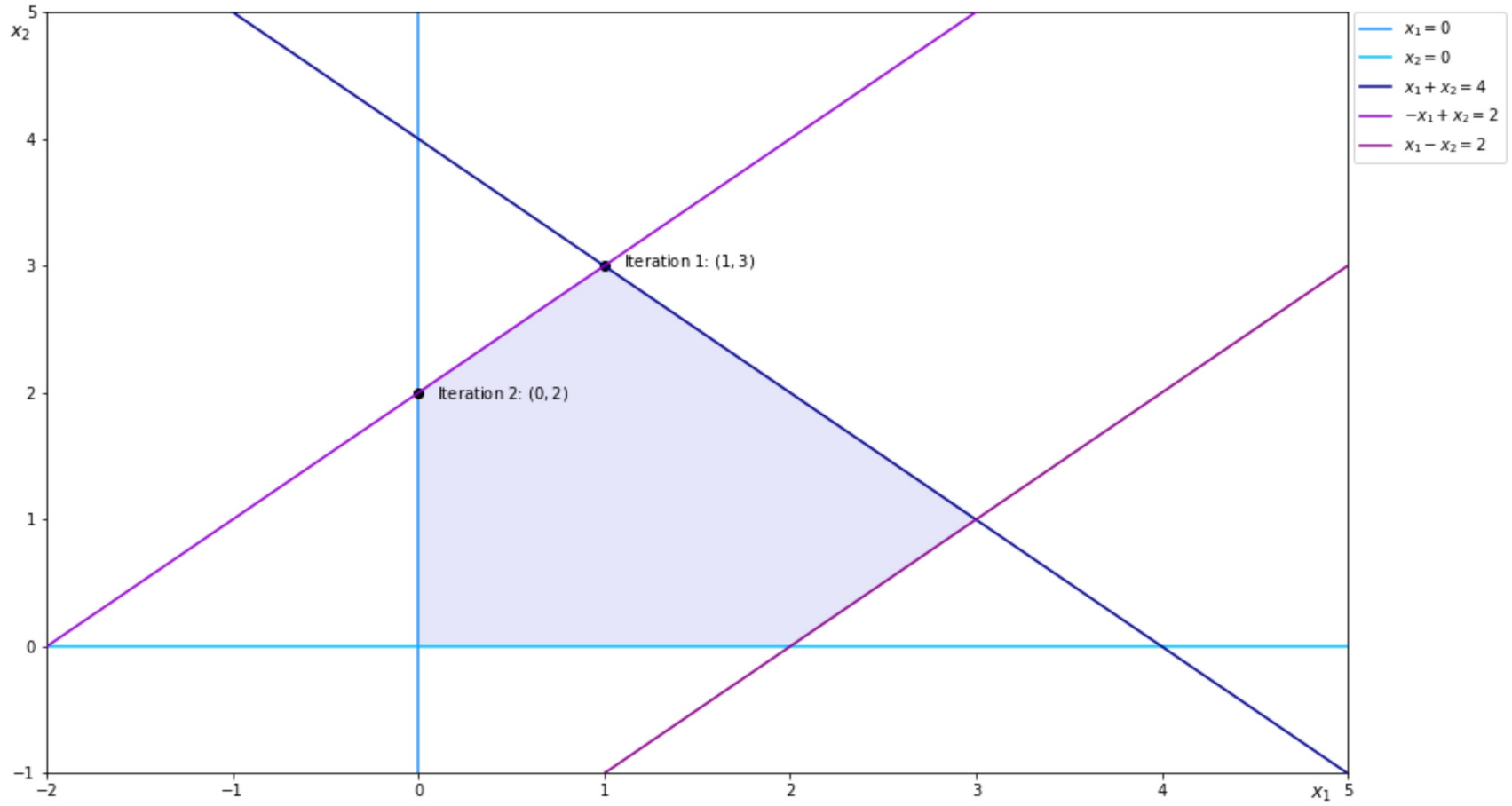
```
[-1.  1. -1.]
```

- We have $d_B = -B^{-1}A_4 = \begin{bmatrix} d_2 \\ d_3 \\ d_5 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$ and $d_N = \begin{bmatrix} d_1 \\ d_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ so $d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}$. Since direction d has at least one negative value we continue.
- $\theta^* = \min\{\frac{x_2}{-d_2}, \frac{x_5}{-d_5}\} = \min\{\frac{2}{1}, \frac{4}{1}\} = 2$
- Variable x_2 will exit the basis.

In [12]:
```
# construct lines
x = np.linspace(-2, 5, 2000)
# x_2 = 0
y1 = 0*x
# x_1 + x_2 = 4
y2 = 4 - x
# -x_1 + x_2 = 2
y3 = 2 + x
# x_1 - x_2 = 2
y4 = -2 + x

# make plot - draw the lines and corner points
fig = plt.figure(figsize = (15, 9))
ax = fig.add_subplot()
plt.plot([0,0],[-2,5],label = r'$x_1=0$', zorder = 1, color = 'dodgerblue')
plt.plot(x, y1, label = r'$x_2=0$', zorder = 2, color = 'deepskyblue')
plt.plot(x, y2, label = r'$x_1+x_2=4$', zorder = 3, color = 'darkblue')
plt.plot(x, y3, label = r'$-x_1+x_2=2$', zorder = 4, color = 'darkviolet')
plt.plot(x, y4, label = r'$x_1-x_2=2$', zorder = 5, color = 'darkmagenta')
plt.plot([1, 0],[3, 2], 'o', color = 'black')
# set axis limits
plt.xlim((-2, 5))
plt.ylim((-1, 5))
# label axis
ax.annotate(r'$x_2$', (-2, 5), (-2.2, 4.8), color = 'black', fontsize = 12)
ax.annotate(r'$x_1$', (5, 0), (4.8, -1.2), color = 'black', fontsize = 12)
#label extreme points
ax.annotate(r'Iteration 1: $(1, 3)$', (1, 3), (1.1, 3))
ax.annotate(r'Iteration 2: $(0, 2)$', (0, 2), (0.1, 1.95))


# fill feasible region
plt.fill([0, 2, 3, 1, 0], [0, 0, 1, 3, 2], color = 'lavender')
plt.legend(bbox_to_anchor = (1.005, 1), loc = 2, borderaxespad=0.)
#plt.show()
plt.savefig("HW8Q3It2.jpg", dpi = 600)
```



Iteration 3:

- $k = 3$
- current feasible solution is $x = \begin{bmatrix} 0 \\ 2 \\ 2 \\ 0 \\ 4 \end{bmatrix} + 2 \cdot \begin{bmatrix} 0 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 4 \\ 2 \\ 2 \end{bmatrix}$

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

In [13]:
```
# find B1^(-1)
B = A[:, [2, 3, 4]]
invB = np.linalg.inv(B)
print(invB)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

- $B^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$
- $x_B = \begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 2 \end{bmatrix}$
- $x_N = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

In [20]:
```
# redused cost for two nonbasic variables
c1 = c[0] - np.dot(np.dot(c[[2, 3, 4]].T, invB), A[:, 0])
print(c1)
c2 = c[1] - np.dot(np.dot(c[[2, 3, 4]].T, invB), A[:, 1])
print(c2)
```

```
-2.0
4.0
```

- We have one negative reduced cost values $\bar{c}_1 = c_1 - c_B^T B^{-1} A_1 = -1$
- Since we do have nonbasic variable with negative reduced cost, current solution is not optimal. Variable to enter the basis is x_1.

In [15]:
```
# dB direction
dB = np.dot(-invB, A[:, 0])
print(dB)
```

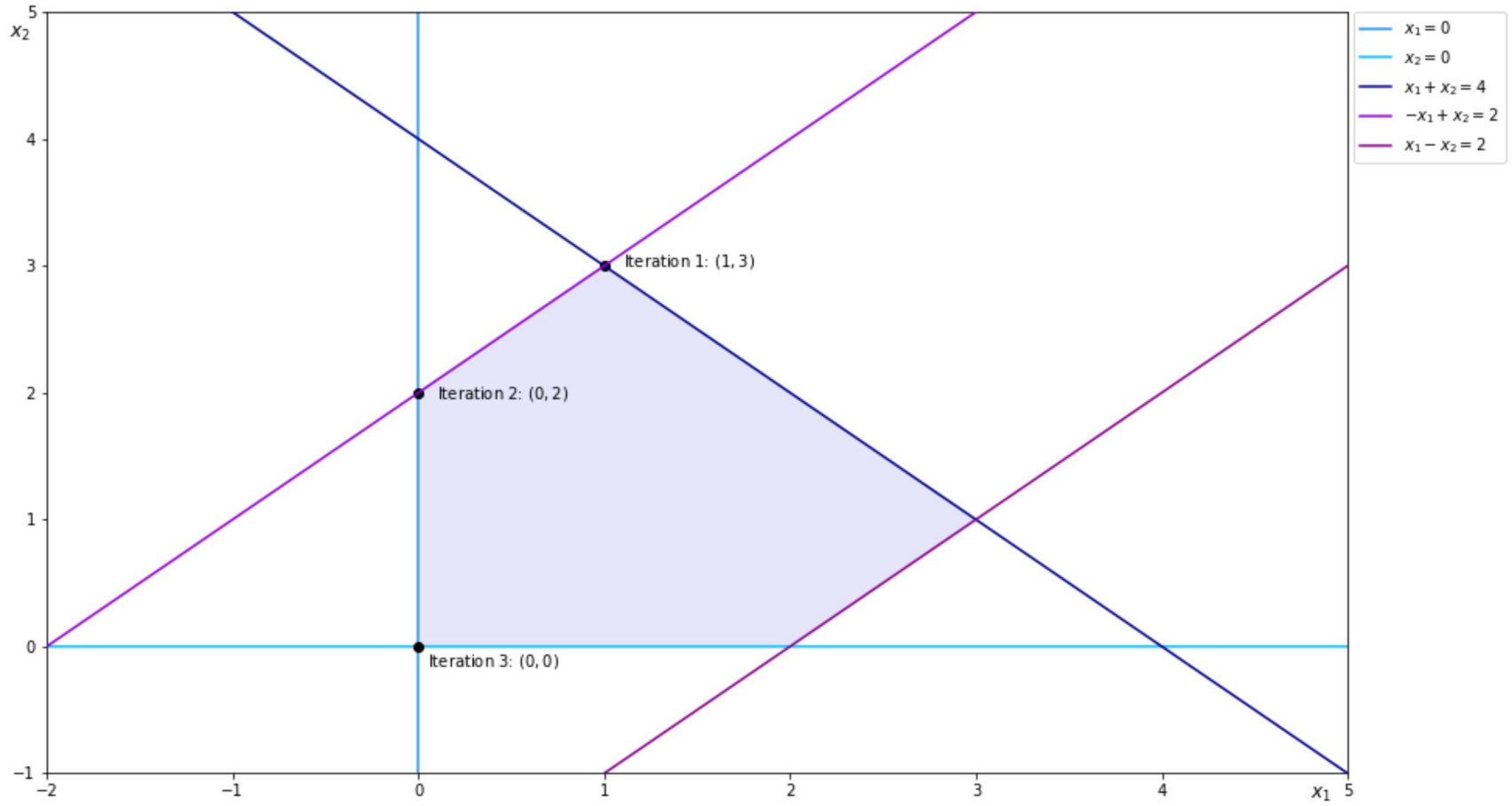
```
[-1.  1. -1.]
```

- We have $d_B = -B^{-1}A_1 = \begin{bmatrix} d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$ and $d_N = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ so $d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 1 \\ -1 \end{bmatrix}$. Since direction d has at least one negative value we continue.
- $\theta^* = \min\{\frac{x_3}{-d_3}, \frac{x_5}{-d_5}\} = \min\{\frac{4}{1}, \frac{2}{1}\} = 2$
- Variable x_5 will exit the basis.

In [16]:
```
# construct lines
x = np.linspace(-2, 5, 2000)
# x_2 = 0
y1 = 0*x
# x_1 + x_2 = 4
y2 = 4 - x
# -x_1 + x_2 = 2
y3 = 2 + x
# x_1 - x_2 = 2
y4 = -2 + x

# make plot - draw the lines and corner points
fig = plt.figure(figsize = (15, 9))
ax = fig.add_subplot()
plt.plot([0,0],[-2,5],label = r'$x_1=0$', zorder = 1, color = 'dodgerblue')
plt.plot(x, y1, label = r'$x_2=0$', zorder = 2, color = 'deepskyblue')
plt.plot(x, y2, label = r'$x_1+x_2=4$', zorder = 3, color = 'darkblue')
plt.plot(x, y3, label = r'$-x_1+x_2=2$', zorder = 4, color = 'darkviolet')
plt.plot(x, y4, label = r'$x_1-x_2=2$', zorder = 5, color = 'darkmagenta')
plt.plot([1, 0, 0],[3, 2, 0], 'o', color = 'black')
# set axis limits
plt.xlim((-2, 5))
plt.ylim((-1, 5))
# label axis
ax.annotate(r'$x_2$', (-2, 5), (-2.2, 4.8), color = 'black', fontsize = 12)
ax.annotate(r'$x_1$', (5, 0), (4.8, -1.2), color = 'black', fontsize = 12)
#label extreme points
ax.annotate(r'Iteration 1: $(1, 3)$', (1, 3), (1.1, 3))
ax.annotate(r'Iteration 2: $(0, 2)$', (0, 2), (0.1, 1.95))
ax.annotate(r'Iteration 3: $(0, 0)$', (0, 0), (0.05, -0.15))


# fill feasible region
plt.fill([0, 2, 3, 1, 0], [0, 0, 1, 3, 2], color = 'lavender')
plt.legend(bbox_to_anchor = (1.005, 1), loc = 2, borderaxespad=0.)
#plt.show()
plt.savefig("HW8Q3It3.jpg", dpi = 600)
```



Iteration 4:

- $k = 4$
- current feasible solution is $x = \begin{bmatrix} 0 \\ 0 \\ 4 \\ 2 \\ 2 \end{bmatrix} + 2 \cdot \begin{bmatrix} 1 \\ 0 \\ -1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 4 \\ 0 \end{bmatrix}$

$$\begin{bmatrix} 1 & 1 & 0 \end{bmatrix}$$

In [17]:
```
# find B1^(-1)
B = A[:, [0, 2, 3]]
invB = np.linalg.inv(B)
print(invB)
```

```
[[ 0.  0.  1.]
 [ 1.  0. -1.]
 [ 0.  1.  1.]]
```

- $B^{-1} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{bmatrix}$
  - $x_B = \begin{bmatrix} x_1 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix}$
  - $x_N = \begin{bmatrix} x_2 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

In [18]:
```
# redused cost for two nonbasic variables
c2 = c[1] - np.dot(np.dot(c[[0, 2, 3]].T, invB), A[:, 1])
print(c2)
c5 = c[4] - np.dot(np.dot(c[[0, 2, 3]].T, invB), A[:, 4])
print(c5)
```

```
2.0
2.0
```

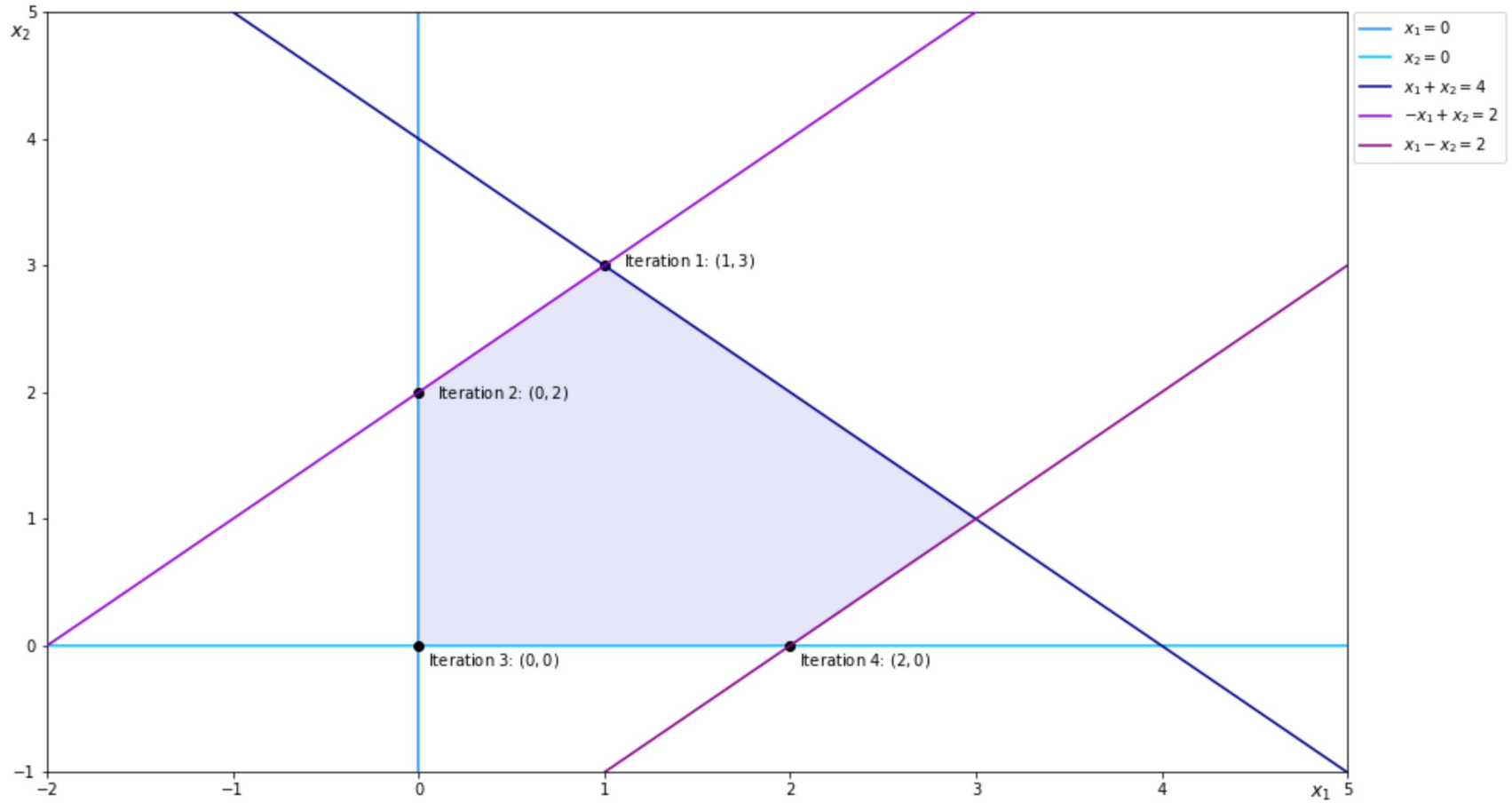
- All reduced cost values are positive, which means that current solution is optimal.

Optimal Solution: $x = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 4 \\ 0 \end{bmatrix}$. **Optimal value:** $c^T x = -4$

In [19]:
```
# construct lines
x = np.linspace(-2, 5, 2000)
# x_2 = 0
y1 = 0*x
# x_1 + x_2 = 4
y2 = 4 - x
# -x_1 + x_2 = 2
y3 = 2 + x
# x_1 - x_2 = 2
y4 = -2 + x

# make plot - draw the lines and corner points
fig = plt.figure(figsize = (15, 9))
ax = fig.add_subplot()
plt.plot([0,0],[-2,5],label = r'$x_1=0$', zorder = 1, color = 'dodgerblue')
plt.plot(x, y1, label = r'$x_2=0$', zorder = 2, color = 'deepskyblue')
plt.plot(x, y2, label = r'$x_1+x_2=4$', zorder = 3, color = 'darkblue')
plt.plot(x, y3, label = r'$-x_1+x_2=2$', zorder = 4, color = 'darkviolet')
plt.plot(x, y4, label = r'$x_1-x_2=2$', zorder = 5, color = 'darkmagenta')
plt.plot([1, 0, 0, 2],[3, 2, 0, 0], 'o', color = 'black')
# set axis limits
plt.xlim((-2, 5))
plt.ylim((-1, 5))
# label axis
ax.annotate(r'$x_2$', (-2, 5), (-2.2, 4.8), color = 'black', fontsize = 12)
ax.annotate(r'$x_1$', (5, 0), (4.8, -1.2), color = 'black', fontsize = 12)
#label extreme points
ax.annotate(r'Iteration 1: $(1, 3)$', (1, 3), (1.1, 3))
ax.annotate(r'Iteration 2: $(0, 2)$', (0, 2), (0.1, 1.95))
ax.annotate(r'Iteration 3: $(0, 0)$', (0, 0), (0.05, -0.15))
ax.annotate(r'Iteration 4: $(2, 0)$', (2, 0), (2.05, -0.15))


# fill feasible region
plt.fill([0, 2, 3, 1, 0], [0, 0, 1, 3, 2], color = 'lavender')
plt.legend(bbox_to_anchor = (1.005, 1), loc = 2, borderaxespad=0.)
#plt.show()
plt.savefig("HW8Q3It4.jpg", dpi = 600)
```


x_2
x_1
$x_1 = 0$
$x_2 = 0$
$x_1 + x_2 = 4$
$-x_1 + x_2 = 2$
$x_1 - x_2 = 2$
Iteration 1: (1, 3)
Iteration 2: (0, 2)
Iteration 3: (0, 0)
Iteration 4: (2, 0)

In [ ]: