In [ ]:
```
## Load Libraries
import pandas as pd
import numpy as np
import random
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
%matplotlib inline
```

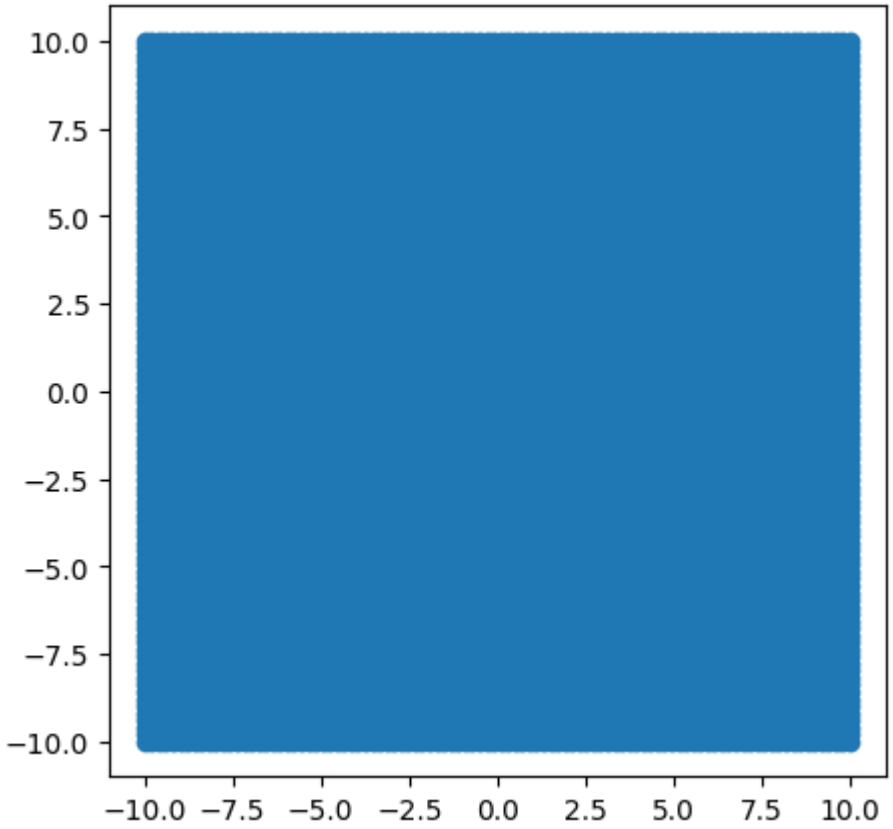
Cell-1

In [ ]:
```
## Solve for the unknown vector x by creating a grid
x2, x3 = np.mgrid[-10.0:10.0:100j, -10.0:10.0:100j] # free variables values
x1 = 4 - 2*x2 - 3*x3 # pivot variable values
x1
```

Out[ ]:
```
array([[ 54.        ,  53.39393939,  52.78787879, ...,  -4.78787879,
         -5.39393939,  -6.        ],
       [ 53.5959596 ,  52.98989899,  52.38383838, ...,  -5.19191919,
         -5.7979798 ,  -6.4040404 ],
       [ 53.19191919,  52.58585859,  51.97979798, ...,  -5.5959596 ,
         -6.2020202 ,  -6.80808081],
       ...,
       [ 14.80808081,  14.2020202 ,  13.5959596 , ..., -43.97979798,
        -44.58585859, -45.19191919],
       [ 14.4040404 ,  13.7979798 ,  13.19191919, ..., -44.38383838,
        -44.98989899, -45.5959596 ],
       [ 14.        ,  13.39393939,  12.78787879, ..., -44.78787879,
        -45.39393939, -46.        ]])
```

In [ ]:
```
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(x2, x3)
```

Out[ ]:
```
<matplotlib.collections.PathCollection at 0x16ddf028730>
```

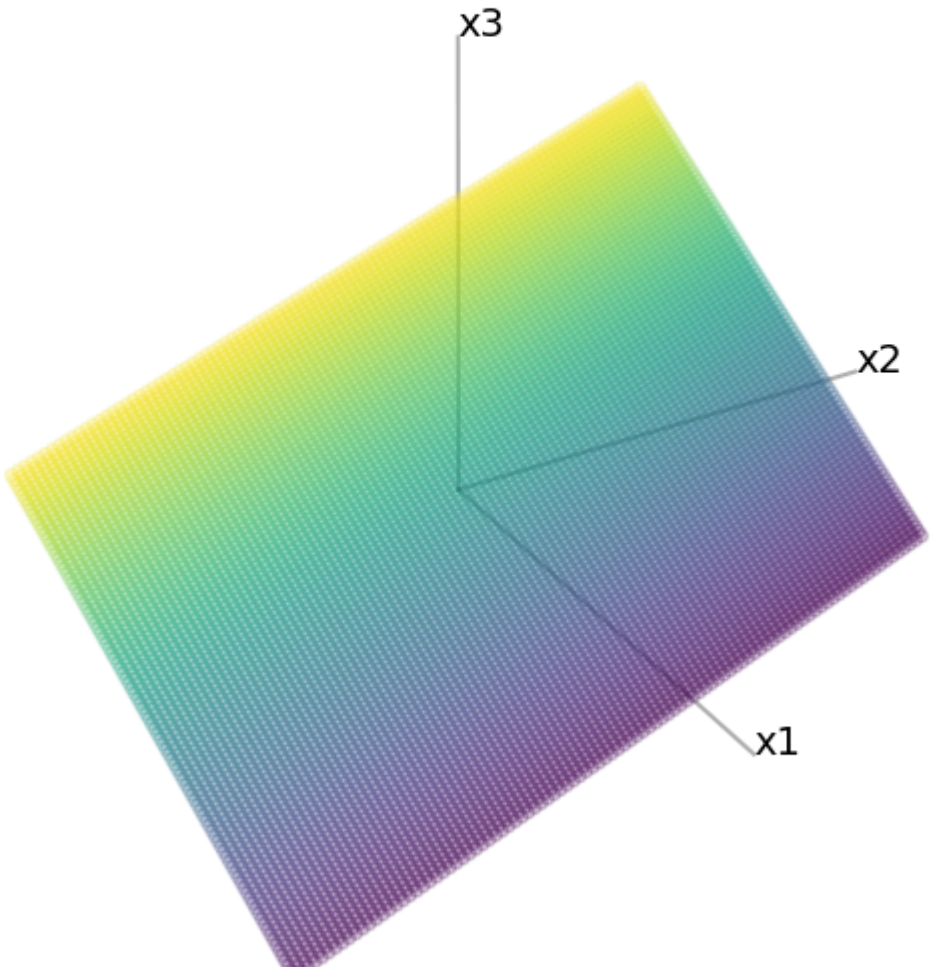


Cell-2

In [ ]:
```
## Make a scatter plot of points represented by vector x calculated
## in the previous cell.

fig = plt.figure(figsize = (12, 10))
ax = fig.add_subplot(111, projection='3d')
ax.view_init(30, -30)
ax.set_axis_off()
ax.grid(False)
ax.plot([0, 60], [0, 0], [0, 0], color = 'k', alpha = 0.3)  # x1-axis line
ax.plot([0, 0], [0, 10], [0, 0], color = 'k', alpha = 0.3)  # x2-axis line
ax.plot([0, 0], [0, 0], [0, 40], color = 'k', alpha = 0.3)  # x3-axis line
```

```
ax.text(60, 0, 0, 'x1', fontsize = 14) # put text for x1-axis
ax.text(0, 10, 0, 'x2', fontsize = 14) # put text for x2-axis
ax.text(0, 0, 40, 'x3', fontsize = 14) # put text for x3-axis

ax.set_xlim(x1.min(), x1.max()) # set limit for x1-axis
ax.set_ylim(x2.min(), x2.max()) # set limit for x2-axis

ax.scatter(x1, x2, x3, c = x3, alpha = 0.2)
```

Out[ ]: <mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x16de2117970>


x3
x2
x1

In [ ]:
```
## Create flattened versions of the grid points
x1f = x1.flatten()
x2f = x2.flatten()
x3f = x3.flatten()

# Print the shapes to understand the underlying array structure
print(x1.shape)
print(x1f.shape)
```

```
(100, 100)
(10000,)
```

Cell-3

In [ ]:
```
# Confirm that the vector w is orthogonal to any vector on the plane
w = np.array([x1f, x2f, x3f])

# Get two random indices
idx = random.sample(np.arange(len(x1)).tolist(), 2)

# Get two vectors representing two points on the plane in standard positions
vector1 = np.array([x1f[idx[0]], x2f[idx[0]], x3f[idx[0]]])
vector2 = np.array([x1f[idx[1]], x2f[idx[1]], x3f[idx[1]]])

# Get a vector lying on the plane
vector_on_plane = vector2 - vector1

# Calculate dot product of w and the vector on plane
np.dot(w.T, vector_on_plane)
```

Out[ ]:
```
array([-972.92929293, -961.50188756, -950.0744822 , ...,  814.31690644,
        825.7443118 ,  837.17171717])
```