9.4.24

BEERAM MADHURI - EE25BTECH11012

October 2025

Question

A cottage industry produces a certain number of toys in a day. The cost of production of each toy (in rupees) was found to be 55 minus the number of toys produced in a day. On a particular day, the total cost of production was $750Rs$. We would like to find out the number of toys produced on that day.

finding the number of toys produced on that day:

Let number of toys produced per day $= x$
cost of each toy$= 55 - x$
Total Cost of toys $= x(55 - x)$

On a particular day cost $= 750$

$$(55 - x)x = 750 \tag{1}$$
$$x^2 - 55x + 750 = 0 \tag{2}$$

this can be compared with the characteristic equation of a Matrix

$$A^2 - tr(A)A + (\det(A))I = 0 \tag{3}$$

$$tr(A) = 55 \tag{4}$$

$$\det(A) = 750 \tag{5}$$

from this Matrix A can be:

$$A = \begin{bmatrix} 0 & 1 \\ -750 & 55 \end{bmatrix} \tag{6}$$

Finding Eigenvalues of A

$$|A - \lambda I| = 0 \tag{7}$$

$$\begin{bmatrix} 0 & 1 \\ -750 & 55 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = 0 \tag{8}$$

$$\begin{bmatrix} -\lambda & 1 \\ -750 & 55 - \lambda \end{bmatrix} = 0 \tag{9}$$

$$-\lambda(55 - \lambda) + 750 = 0 \tag{10}$$

By solving this equation, eigenvalues are:

$$\lambda_1 = 25, \quad \lambda_2 = 30 \tag{11}$$

$\therefore$ no. of toys produced that day can be either 25 or 30.

Python Code

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

# --- 1. Solve the Quadratic Equation ---

# The equation is x^2 - 55x + 750 = 0
# Coefficients for ax^2 + bx + c = 0
a = 1
b = -55
c = 750
```

Python Code

```
# Calculate the roots (solutions) of the equation
solutions = np.roots([a, b, c])
solution1, solution2 = solutions[0], solutions[1]

print(f"--- Problem Solution ---")
print(f"The quadratic equation is: {a}x^2 + ({b})x + {c} = 0")
print(f"The possible number of toys produced are: {int(solution1)
    } and {int(solution2)}")
```

Python Code

```
# Create a range of x-values (number of toys) to plot
# We'll plot from 0 to 60 to get a good view of the parabola
x_values = np.linspace(0, 60, 400)
# Calculate the corresponding y-values using the quadratic
    function
y_values = a * x_values**2 + b * x_values + c
```

Python Code

```python
# --- 3. Plot the Graph ---
plt.style.use('seaborn-v0_8-whitegrid')
fig, ax = plt.subplots(figsize=(10, 6))
# Plot the quadratic function (parabola)
ax.plot(x_values, y_values, label='Cost Function: $y = x^2 - 55x
    + 750$', color='dodgerblue')
```

Python Code

```
# Draw a horizontal line at y=0 (x-axis) to highlight where the
    roots are
ax.axhline(0, color='gray', linestyle='--')
# Plot the solutions (roots) on the graph as distinct points
ax.scatter(solutions, [0, 0], color='red', zorder=5, s=100, label
    =f'Solutions: x={int(solution1)}, x={int(solution2)}')
```

Python Code

```python
# --- 4. Customize and Show the Plot ---
# Adding titles and labels
ax.set_title('Graph of the Toy Production Cost Function',
    fontsize=16)
ax.set_xlabel('Number of Toys Produced (x)', fontsize=12)
ax.set_ylabel('Total Cost Equation (y)', fontsize=12)
```

```python
# Set plot limits to focus on the relevant area
ax.set_xlim(0, 60)
ax.set_ylim(-100, 800)

# Add annotations for the solution points
ax.annotate(f'({int(solution1)}, 0)', xy=(solution1, 0), xytext=(
    solution1 - 8, -70),
```

Python Code

```
ax.annotate(f'({int(solution2)}, 0)', xy=(solution2, 0), xytext=(
    solution2 + 2, -70),
arrowprops=dict(facecolor='black', shrink=0.05))

# Add legend and display the graph
ax.legend()
plt.show()
```

C Code

```c
#include <stdio.h>
#include <math.h> // Required for the square root function sqrt()

int main() {
    // Coefficients for the quadratic equation: x^2 - 55x + 750 =
        0
    double a = 1.0;
    double b = -55.0;
    double c = 750.0;
    double discriminant, root1, root2;
```

C Code

```
// Calculate the discriminant (b^2 - 4ac)
discriminant = b * b - 4 * a * c;

// Check if real solutions exist
if (discriminant >= 0) {
    // Calculate the two possible roots using the quadratic
        formula
    root1 = (-b + sqrt(discriminant)) / (2 * a);
    root2 = (-b - sqrt(discriminant)) / (2 * a);
```

C Code

```
    printf("The problem translates to the quadratic equation:
         x^2 - 55x + 750 = 0\n");
    printf("Solving for x, we find two possible solutions.\n\
        n");

    // Print the final answer in the context of the problem
    printf("The number of toys produced on that day was
        either %.0f or %.0f.\n\n", root1, root2);
```

C Code

```
// Verification for both cases
printf("Verification:\n");
printf("Case 1: If %.0f toys were produced, the cost per toy is (55 - %.0f) = %.0f. Total cost = %.0f * %.0f = %.0f\n", root1, root1, (55-root1), root1, (55-root1), root1*(55-root1));
printf("Case 2: If %.0f toys were produced, the cost per toy is (55 - %.0f) = %.0f. Total cost = %.0f * %.0f = %.0f\n", root2, root2, (55-root2), root2, (55-root2), root2*(55-root2));
```

C Code

```
    } else {
        // This case will not occur for the given problem numbers
        printf("The equation has no real solutions, which means
            there is an error in the problem's premises.\n");
    }
    return 0;
}
```

Python and C Code

```
from ctypes import c_double
from math import sqrt
def main():
    # Coefficients for the quadratic equation: x^2 - 55x + 750 =
        0
    a = c_double(1.0)
    b = c_double(-55.0)
    c = c_double(750.0)
    # Calculate the discriminant (b^2 - 4ac)
    discriminant = c_double(b.value ** 2 - 4 * a.value * c.value)
```

Python and C Code

```
# Check if real solutions exist
if discriminant.value >= 0:
    # Calculate the two possible roots using the quadratic
        formula
    root1 = c_double((-b.value + sqrt(discriminant.value)) /
        (2 * a.value))
    root2 = c_double((-b.value - sqrt(discriminant.value)) /
        (2 * a.value))
```

Python and C Code

```
print("The problem translates to the quadratic equation:
    x^2 - 55x + 750 = 0")
print("Solving for x, we find two possible solutions.\n")

print(f"The number of toys produced on that day was
    either {root1.value:.0f} or {root2.value:.0f}.\n")

# Verification for both cases
print("Verification:")
```

Python and C Code

```
cost1 = c_double(55 - root1.value)
total_cost1 = c_double(root1.value * cost1.value)
print(f"Case 1: If {root1.value:.0f} toys were produced,
    the cost per toy is "
      f"(55 - {root1.value:.0f}) = {cost1.value:.0f}.
          Total cost = "
      f"{root1.value:.0f} * {cost1.value:.0f} = {
          total_cost1.value:.0f}")
```

Python and C Code

```
cost2 = c_double(55 - root2.value)
total_cost2 = c_double(root2.value * cost2.value)
print(f"Case 2: If {root2.value:.0f} toys were produced,
    the cost per toy is "
      f"(55 - {root2.value:.0f}) = {cost2.value:.0f}.
         Total cost = "
      f"{root2.value:.0f} * {cost2.value:.0f} = {
         total_cost2.value:.0f}")
```

Python and C Code

```python
    else:
        print("The equation has no real solutions, which means
            there is an error in the problem's premises.")

if __name__ == "__main__":
    main()
```

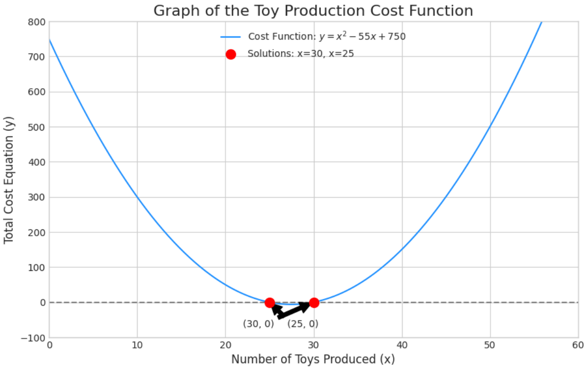

Figure: plot 9.4.24