Numerical Method

Lab 2

Nishad Bijukchhe

Roll no. 724

Class of 2021 'B'

Theory for Gauss Jordan Method

By eliminating the unknown variables from all the equations Gauss Jordan assists us to reduce the effort and reduces the time to perform back substitution and compute out the unknown constants .

Algorithm for Gauss Jordan Method

1. Start
2. Read the order of the matrix ‘n’ and read the coefficients of the linear equations.
3. Do for k=1 to n
   Do for l=k+1 to n+1
   a[k][l] = a[k][l] / a[k][k]
   End for l

Set a[k][k] = 1
Do for i=1 to n
if (i not equal to k) then,
Do for j=k+1 to n+1
a[i][j] = a[i][j] – (a[k][j] * a[i][k])
End for j
End for i
End for k

4. Do for m=1 to n
   x[m] = a[m][n+1]
   Display x[m]
   End for m

5. Stop

Program for Gauss Jordan Method

```
#include <stdio.h>

#include<stdlib.h>

void function_gauss(int, float[][10], float[], float[]);

int main()

{     int i, j, size;

      float matix1[10][10], matix2[10], x[10];

      printf("How many variables are there? ");

      scanf("%d", &size);


    if(size<=2){exit(0);}

      else{

      for (i = 1; i <= size; i++)

      {

            printf("\n%dth equation \n\n", i);
```

```
        for (j = 1; j <= size; j++)
        {     printf("Enter %dth number ", j);
    scanf("%f", &matix1[i][j]);
        }
        printf("Enter constant ");
        scanf("%f", &matix2[i]);
    }
      function_gauss(size, matix1, matix2, x);
    printf("\nSolution:\n ");
    for (i=1; i <= size; i++)
    {
        printf("\nx%d = %f",i,matix2[i]);
    }return 0;}}
void function_gauss(int n, float matix1[][10], float
matix2[], float x[10])
```

```
{	int i, j, k;
	float factor, sum, pivot;
	for (i=1; i<= n-1; i++)
	{
		for (j = i+1; j <= n; j++)
		{	factor = matix1[j][i] / matix1[i][i];
			for(k = 1; k<=n; k++)
			{matix1[j][k] = matix1[j][k]- factor*
matix1[i][k];
			}matix2[j] = matix2[j] - factor * matix2[i]
;
		}
	}printf("\nGauss Elimination has been
completed\n");
	for (i = 1; i<= n; i++)
	{
```

```
        for (j = 1; j <= n; j++)
        {printf("%f\t", matix1[i][j]);}
        printf("%f", matix2[i]);
        printf("\n");
    }printf("\nThe matrix  is divided my the pivot 
elements\n");
    for (i=1; i<= n; i++)
    {
        for (j = i; j <= n; j++)
        {pivot  = matix1[i][i];
            for(k = 1; k<=n; k++)
            {matix1[j][k] = matix1[j][k]/pivot; }
            matix2[j] = matix2[j]/pivot ;
        }
    }
```

```
for (i = n; i >= 1; i--)
{       for (j = i-1 ; j >= 1; j--)
        {
                factor = matix1[j][i] / matix1[i][i];


                for(k = n; k >= 1; k--)
                {       matix1[j][k] = matix1[j][k]- factor*
matix1[i][k];
                }
                        matix2[j] = matix2[j] - factor *
matix2[i] ;
        }
}printf("\nAppliying Gauss Jordan method\n");
for (i = 1; i<= n; i++)
{
        for (j = 1; j <= n; j++)
```

```
            printf("%f\t", matix1[i][j]);

          printf("%f", matix2[i]);    printf("\n");

    }return;

}
```

Output