MatLab

continuation

Gábor Borbély

Budapest University of Technology and Economics

2019.11.25

System of Linear Equations – I.

$$x + y - z = 3$$
$$-x + y - 2z = 1$$

```
>> A = [1 1 -1; -1 1 -2];
>> b = [3; 1];
>> A\b
ans = 1.6667
           0
     -1.3333
```

```
>> rref(A)
ans =
    1.0000         0    0.5000
         0    1.0000   -1.5000

>> X = rref([A b])
ans =
    1.0000         0    0.5000    1.0000
         0    1.0000   -1.5000    2.0000

>> rank(X(:,1:end-1))
ans = 2
>> rank(X)
ans = 2
```

System of Linear Equations – III.

All of the solutions

```
>> null(A)
ans =
   -0.2673
    0.8018
    0.5345
>> size(ans,2)
ans = 1

>> c = 3.14;
>> A\b + null(A)*c
ans =
    0.8275
    2.5176
    0.3451
```

Multidimensional arrays

- vector (row/column)

```
>> ones(1, 4)
ans =
       1       1       1       1
```

- matrix

```
>> ones(2, 4)
ans =
       1       1       1       1
       1       1       1       1
```

- 3-dimensional

```
>> ones(2, 4, 2)
ans(:,:,1) =
       1       1       1       1
       1       1       1       1
ans(:,:,2) =
       1       1       1       1
       1       1       1       1
```

Reductions

```
>> A = [1 2 3; 4 5 6];
```

- row-wise-sum: sum up the column indices

```
>> sum(A, 2)
ans =
     6
    15
```

- column-wise-sum: sum up the row indices

```
>> sum(A, 1)
ans =
     5     7     9
```

- in the third dimension

```
>> sum(rand(2, 4, 2), 3)
ans =
    1.7722    0.2846    1.5895    1.0788
    1.8707    1.8840    0.5829    0.6888
```

More dimension, more reduction

- sum up all the entries: `sum(A, 'all')`
- summation reduces the current dimension:

```
>> size(sum(rand(2, 4, 3),1))
ans = 1      4      3
>> size(sum(rand(2, 4, 3),3))
ans = 2      4      1
```

- you can calculate `mean`, `product`, `max` ...

```
>> mean(rand(2, 4, 30),3)
    0.4342    0.5122    0.4937    0.5416
    0.5033    0.5849    0.5580    0.4245
```

- generalized transpose (dim-shuffle):

```
>> size(permute(rand(2, 4, 3),[3,2,1]))
ans =
     3      4      2
```

- the ordinary transpose: `permute(A, [2,1])`

Plot (2D)

- `plot(x, y)`, where `x` and `y` are two vectors of the same length
- list of line segments between the points (x_i, y_i)

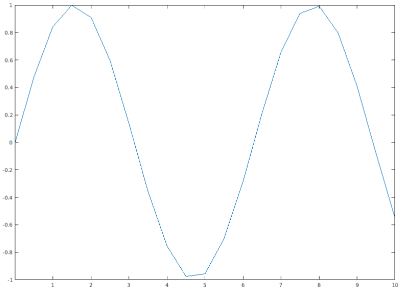

Figure: `plot(0:0.5:10, sin(0:0.5:10))`

Plot (3D)

- `plot3(x, y, z)`, where `x`, `y` and `z` are vectors of the same length
- list of line segments between the points (x_i, y_i, z_i)

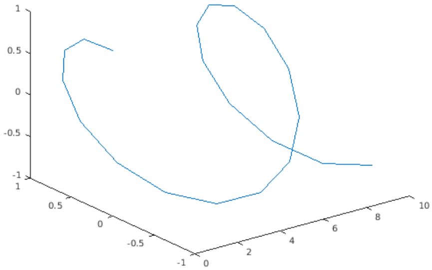

Figure: `plot3(0:0.5:10, sin(0:0.5:10), cos(0:0.5:10))`

Functions

- Let `fv1.m` contain the function $x \mapsto x^2$

```
function y = fv1(x)
    y = x.^2;
end
```

- And let `fv2.m` contain the function $x \mapsto \sin(x)$

```
function y = fv2(x)
    y = sin(x);
end
```

- Then one can plot:

```
>> x = -2:0.1:2;
>> plot(x, fv1(x))
>> plot(x, fv2(x))
```

Function object @

- One can store the function as a variable:

```
>> g = @fv1;
>> g(3)
ans =
     9
>> g = @fv2;
>> g(3)
ans =
     0.1411
```

- then g is a variable, it stores the function

```
>> g = @cos;
```

- One can write a function for plotting functions:

```
function myplot(f, xmin, xmax, step)
    x = xmin:step:xmax;
    plot(x, f(x))
end
>> myplot(g, -2, 2, 0.1)
```